Aus dem Institut für Mathematik
der Universität zu Lübeck

Direktor:
Prof. Dr. Jürgen Prestin

The Nonequispaced Fast SO(3) Fourier Transform, Generalisations and Applications

Inauguraldissertation
zur
Erlangung der Doktorwürde
der Universität zu Lübeck
- Aus der Technisch-Naturwissenschaftlichen Fakultät -

Vorgelegt von

Dipl.-Inf. Antje Vollrath

aus Wolgast

Lübeck, den 24. März 2010

Vorsitzender:	Prof. Dr. Achim Schweikard, Universität zu Lübeck
Gutachter:	Prof. Dr. Jürgen Prestin, Universität zu Lübeck Prof. Dr. Dirk Langemann, Technische Universität Braunschweig
Tag der mündlichen Prüfung:	31. Mai 2010

Contents

1 Introduction

The importance of the Fast Fourier Transform (FFT) can hardly be overestimated. It is the foundation of many signal and image processing procedures, like filtering or recognition; data analysis, like time series encoding or correlation or used to solve partial differential equations. The best known algorithm for the FFT is the one by Cooley and Tukey [20] from 1965, although its origins can be traced back to Gauß in 1805. Today it is an omnipresent algorithm for the analysis and manipulation of all kinds of digital or discrete data.

The FFT permits the fast computation of the discrete Fourier transform giving a decomposition of a periodic function into a linear combination of complex exponentials. Depending of the user's point of view, the FFT either computes an approximation of the classical Fourier transform on the real line, a function of a certain band-limit on the circle $\mathbb{S}^1$, or the expansion of a function defined on a cyclic group. From here generalisations of the FFT scheme split in several branches, two of which are important for the upcoming considerations in this thesis, namely the use of arbitrary, in particular nonequispaced sampling points instead of uniformly distributed ones, and the application of the techniques to other manifolds than the circle or to more general groups, than the cyclic, in particular to non-abelian groups.

The need for using arbitrary samples in a variety of applications, like e.g. medical imaging with MRI led to various works on the so-called nonequispaced fast Fourier transform amongst which we like to point out the work by Potts, Steidl and Tasche on this transform, called the NFFT, that is summarised in [71]. Other approaches include [12, 26]. In these algorithms, the complex exponentials of the function's expansions are efficiently evaluated with a previously chosen target accuracy. They all are using different approximation schemes which trade run time for the precision of the approximation.

The other branch of generalisations is concerned with the restriction to expand functions in terms of complex exponentials. These can be exchanged e.g. for orthogonal polynomials or spherical harmonics, see [23, 65, 70, 77].

Even more generally, as functions defined on certain groups have expansions in terms of suitable basis elements which generalise the complex exponentials in terms of group invariance. Therefore it seems sensible to examine whether one can efficiently evaluated functions given in these generalised basis functions. Indeed, fast algorithms exist for many classes of groups, see [62] for a review of the group theoretic approach to the fast Fourier transform.

Of central interest in this thesis is the fast Fourier transform on the rotation group $\mathrm{SO}(3)$, which is important for a number of applications from various fields. This includes texture analysis [43, 85], protein-protein docking [15, 54, 76], robot workspace generation [17], or spherical image analysis [60], to name just a few.

Motivated by these applications, several different techniques have been proposed for computing Fourier transforms on the rotation group $\mathrm{SO}(3)$ during the past years. The algorithms to compute such transforms are based on evaluating the so-called Wigner-D functions $\mathrm{D}_\ell^{\mathfrak{m}\mathfrak{n}}$ for degree $\ell \in \mathbb{N}$ and orders $\mathfrak{m}, \mathfrak{n} = -\ell, \ldots, \ell$. The Wigner-D functions yield an orthogonal basis of $\mathrm{L}^2(\mathrm{SO}(3))$.

Using these basis functions, we can approximate arbitrary functions $f \in L^2(SO(3))$ by the finite sum

$$f \approx \sum_{\ell=0}^{L} \sum_{m=-\ell}^{\ell} \sum_{n=-\ell}^{\ell} \widehat{f}_\ell^{mn} D_\ell^{mn}. \tag{1.1}$$

The evaluation of this sum is called the discrete $SO(3)$ Fourier transform of maximum degree L. Such a degree-L transform at Q different function samples, or nodes, of f requires $\mathcal{O}(L^3 Q)$ arithmetic operations in a direct calculation, too much computations under real-world conditions. Finding an algorithm computing the same result only with less complexity, is the main result of this thesis. We will introduce two different of them along with a generalisation and a promising application.

There have been previous works that derived fast algorithms for the computation of the above sum. Most of them are based on rewriting (1.1) into a classical Fourier sum over complex exponentials. Risbo [73] describes how the Wigner-D functions themselves are expanded into a Fourier sum and thus evaluates a quadruple sum by means of an FFT leading to an algorithm with $\mathcal{O}(L^4)$ arithmetic operations. The algorithm as described there evaluates the $SO(3)$ Fourier transform at $\mathcal{O}(L^3)$ particular function samples.

Also, Kostelec and Rockmore [53] discuss an $\mathcal{O}(L^4)$ algorithm for $8L^3$ particular nodes. There, the acceleration is achieved by exploiting the tensor product character of the $SO(3)$ basis functions and the particular choice of nodes. They acknowledge that a variation of the Driscoll-Healy algorithm [23] could improve this to $\mathcal{O}(L^3 \log^2 L)$, but note doubts about its performance for realistic amounts of data.

Nevertheless, their idea was pursued in conjuction with this work and combined, for the first time, with the NFFT by Potts, Steidl, and Tasche from [71] to an approximate $\mathcal{O}(L^3 \log^2 L + Q)$ algorithm for Q nodes free of choice. This generalises a similar algorithm for the discrete Fourier transform on the sphere $\mathbb{S}^2$ as presented by Kunis and Potts [57], as well as Keiner and Potts [51].

The comparison of the different approaches by Potts, Prestin, and Vollrath in [68] showed ambivalent results. For relatively large transform sizes (roughly $L > 512$) the asymptotically faster Driscoll-Healy-like algorithm outperforms the method of Kostelec and Rockmore in a synthetic test scenario. But unfortunately, a stabilisation scheme that must be employed for numerical stability interferes with the potential gain in performance under practical conditions. Further considerations, like different memory requirements, their effect on performance, or the achieved accuracy of both methods, seem to make it difficult to decide at this point whether one algorithm should be preferred over the other.

Motivated by these results, a new algorithm emerges here, proposing to replace the Driscoll-Healy-like method with a different one. This approach thereby generalises results established by Rokhlin and Tygert [77] for the discrete Fourier transform on the sphere $\mathbb{S}^2$. The outcome is a new type of algorithm that, as we will demonstrate numerically, has the potential to remove some of the undesired properties mentioned above.

This being the main scope of this work, we nevertheless examine two other interesting topics: generalisations and applications of $SO(3)$ Fourier transforms. For the first time, the algorithms for $SO(3)$ Fourier transforms will be adapted, or rather generalised, to compute Fourier transforms on the group $SU(2)$ of complex rotations of which $SO(3)$ is a subgroup. The newly introduced $SU(2)$ Fourier transform has potential applications in particle physics [82], or in the computation with pseudodifferential operators [78].

The applications that merit our special attention here are on one hand, the combination of the $SO(3)$ Fourier transform with kernel based approximation methods to compute kernel density estimations from electron back scattering diffraction data, a problem relevant in texture analysis.

On the other hand, we will lay out how the nonequispaced $SO(3)$ Fourier transforms can be used to

develop a new algorithm to handle the protein-protein docking problem, an automated procedure that is widely used to predict how proteins might interact with each other. To understand these interactions, it is essential to determine the three-dimensional structure of the participating proteins. Based on the analysis on the known structure of proteins, docking procedures calculate the structure of new formed protein complexes. An essential tool is the Protein Data Bank (PDB) which stores the structure of around 12000 proteins and protein complexes determined by NMR or X-ray techniques, [11]. Provided with this large collection of structural data single proteins, we formulate the protein-protein-docking problem as the computation of atomic coordinates of a protein complex out of the atomic coordinates of the component molecules.
The first automated docking algorithm has been described in 1978 by Wodak and Janin in [92]. Since then many different approaches to tackle the problem have been proposed, see e.g. [27, 40]. The common aspect of these approaches is an optimisation problem. The solution space is the set of motions, i.e. rotations and transformations the molecules can undertake upon formation of new complexes. The according objective function evaluates the quality of the complex.
We will present a strict mathematical description of the problem including protein descriptions to compact the textual approaches found in literature. We will demonstrate the non-convex behaviour of the related objective functions hence motivating the need for a fast and efficient new global search algorithm. Such an algorithm using the nonequispaced $\mathrm{SO}(3)$ Fourier transform and a comparison to existing similar methods, completes this thesis.

Outline We start in Chapter 2 that is intended as a self-contained introduction to the rotation group. We collect basic material about the rotation group $\mathrm{SO}(3)$ including basic notations, important properties concerning rotations and their representations by elements of $\mathrm{SO}(3)$. We compile different parameterisations of rotation and provide a convenient overview how these different parameterisations can be transformed into each other. As an own aspect, we prove basis properties of the parameterisations in a constructive manner using the definition of the $\mathrm{SO}(3)$ as a group of matrices. We also lay the foundation for the following chapter on harmonic analysis on the rotation group by relating its elements to geometric objects and giving a suitable metric on $\mathrm{SO}(3)$.
Then Chapter 3 gives a short summary about harmonic analysis on the rotation group $\mathrm{SO}(3)$ followed by the definition of the Fourier transform on $\mathrm{SO}(3)$ and its adjoint. We motivate the definition of integration of rotation dependent functions by deriving of the Jacobian for different parameterisations of rotations. In Section 3.2, we approach the Fourier transforms on the rotation group from a group theoretic perspective. The key theorem to these considerations, the Peter-Weyl theorem, will be given as well as the definition of the matrix elements of unitary, irreducible representations of $\mathrm{SO}(3)$. We will show that they constitute an orthogonal basis in the space $\mathrm{L}^2(\mathrm{SO}(3))$. By means of these functions, called Wigner-D functions D_ℓ^{mn}, the $\mathrm{SO}(3)$ Fourier transform arises in Definition 3.2.9 and consequently we give the discrete $\mathrm{SO}(3)$ Fourier transform (NDSOFT) in Definition 3.2.11. We will then see that computing convolution and correlation by Fourier transforms follows the same lines on the rotation group as it does in the standard settings.
In Section 3.3, we consider the Wigner-D functions and their decomposition into complex exponentials and Wigner-d functions. Given the relation of the latter to classical orthogonal polynomials, we are able to state important properties, of Wigner-d functions like three-term recurrence relation, Rodrigues formula and symmetries. We then examine another close connection between the rotation group and the two-dimensional sphere by using the $\mathrm{SO}(3)$ Fourier transform to compute convolution and correlation of spherical functions.
A new application, the fast summation of functions on the rotation group is presented in Section 3.5. The main result is the efficient computation of sums of rotation dependent functions by splitting them

as in Equation (3.35). That way, they can be computed using an $\mathrm{SO}(3)$ Fourier transform preceded by an adjoint $\mathrm{SO}(3)$ Fourier transform. In Lemma 3.5.3, we moreover give an error estimate for these computations. This section's algorithms have been applied to a problem from texture analysis and are published in [43]. They have been implemented in C and tested on an Intel Core 2 Duo 2.66 GHz MacBook Pro with 4GB RAM running Mac OS X 10.6.1 in double precision arithmetic.

Chapter 4 provides the central and most important part of the work as, we present new and efficient methods to compute $\mathrm{SO}(3)$ Fourier transforms at arbitrarily sampled rotations, here. The crucial point of the presented algorithms is to replace the Wigner-D functions by a product of complex exponentials such that we can employ the well-analysed nonequispaced fast Fourier transform (NFFT) algorithm for the computations. Its cost, $\mathcal{O}(\mathrm{L}^3 \log \mathrm{L} + \mathrm{Q})$, is that of a classical three-dimensional FFT plus a term linear in the number of nodes Q, see e.g. [12, 26, 56, 71] and the references therein. Moreover, a C subroutine library implementing the NFFT algorithm is available in [49].

In Section 4.1, we will develop two different ways to efficiently compute coefficients of a three-dimensional standard Fourier series out of given $\mathrm{SO}(3)$ Fourier coefficients. Our first approach is described in Section 4.1.1. It describes new algorithm for the fast evaluation of the so-called Wigner-d functions at nonequispaced sampling points. It generalises the fast polynomial transform (FPT) algorithm introduced by [23, 70] which uses a cascade summation based on the three-term recurrence relations of the respective orthogonal polynomials. This approach will be adopted to the Wigner-d functions yielding the extended three-term recurrence relation in Equation (4.5). This algorithm is a first important step to improve the efficiency of the $\mathrm{SO}(3)$ Fourier transform. Its implementation in C is available as a part of the public NFFT subroutine library [49]. We will refer to it as the FWT-C (the fast Wigner transformation based on cascade summation). A more extensive discussion along with numerical results and comparisons of this algorithm have been published in [68].

The second proposed approach is the fast transformation of Wigner-d functions based on semiseparable matrices (FWT-S) covered in Section 4.1.2. Essential to this algorithm is the transformation of sums of arbitrary Wigner-d functions into those of Wigner-d functions of low orders $\mathfrak{m}$ and $\mathfrak{n}$ as stated in Equation (4.6) and the application of the differential operator (4.8) to the Wigner-d functions. There we replace the cascade summation scheme with an approximate technique that is a generalisation of the algorithms proposed by Rokhlin and Tygert in [77] for spherical harmonic expansions on the sphere $\mathbb{S}^2$. We show that the matrix representation of this transformation is related to the eigendecomposition of certain semiseparable matrices in Lemma 4.1.3 and Lemma 4.1.5. This enables us to employ a divide-and-conquer algorithm developed by Chandrasekaran and Gu [16] that we combine with the well-known fast multipole method (FMM) [38] to the desired fast algorithm.

In Section 4.2, we state the central theorem about the nonequispaced fast $\mathrm{SO}(3)$ Fourier transform (NFSOFT), Theorem 4.2.2. A schematic overview of all steps of the NFSOFT with corresponding references within this thesis is provided in Figure 4.8.

Numerical results on the proposed new algorithms can be found in Sections 4.1.3 and 4.2.2. There we also provide numerical results on the fast summation algorithm proposed in Section 3.5. Our conclusion is that both of our methods offer distinct advantages over previous approaches as well as room for further improvements.

The next big issue in this work, is the generalisation of $\mathrm{SO}(3)$ Fourier transforms. Section 5 starts with the novel consideration of nonequispaced Fourier transforms on the complex rotation group $\mathrm{SU}(2)$ which is motivated by the close relation of the groups $\mathrm{SO}(3)$ and $\mathrm{SU}(2)$. Again we determine a set of basis functions for the space $\mathrm{L}^2(\mathrm{SU}(2))$ which are a union of the already known Wigner-D functions and Wigner-D functions for half-integer degrees and orders. We describe the modifications and adaptions of the algorithms described in Section 4. The main results are the fast transform of half-integer Wigner-d functions based on semiseparable matrices that is derived in Lemma 5.1.6 for the

first time and the nonequispaced $\mathrm{SU}(2)$ Fourier transform (NFSUFT) itself given in Theorem 5.1.9. The proposed new algorithm has been implemented in Mathematica.
Another promising generalisation is the Fourier transform on the three-dimensional motion group $\mathrm{SE}(3)$, of which $\mathrm{SO}(3)$ is a subgroup. In contrast to $\mathrm{SO}(3)$ and $\mathrm{SU}(2)$ the motion group is not locally compact bringing about new difficulties in computing Fourier transforms on this group. The $\mathrm{SE}(3)$ Fourier transform is given in Definition 5.2.11. Its computation has an interesting application, which we will discuss extensively in Chapter 6, the protein-protein docking.
Chapter 6 starts with a description of the protein-protein docking problem as the prediction of protein interactions, a central task of structural biology. In this work, we shall focus on the first stage of docking and present two methods that can be categorised as Fourier-based rigid-body docking. This term refers to the search strategy on one hand and to the design of the solution space of the underlying optimisation problem on the other hand. The textual literature on the protein-protein docking problem is condensed to a strict mathematical description. We will introduce two choices of objective functions (6.6) and (6.8). Then, we will demonstrate the difficulties of handling this highly non-convex problem first in a simplified setting, then in the realistic one. The occurrence of numerous local extrema motivates the usage of search algorithm at discrete grid points of $\mathrm{SE}(3)$.
The application of our nonequispaced $\mathrm{SO}(3)$ Fourier transform to the problem is established in Lemma 6.5.4 and the corresponding new algorithm called fast rotational matching is described in Algorithm 2 of Section 6.5. The numerical cost to obtain a solution of the protein docking problem is drastically reduced here. We conclude our considerations of the docking problem by a suggestion for a refinement step.

Acknowledgements First, I thank my advisor Prof. Dr. Jürgen Prestin for his constant guidance and support in writing this thesis. He is not only responsible for involving me in Fourier analysis in the first place, but also for initiating this thesis, for helping me to develop an understanding of the subject and for pointing out new directions and ideas to my research endeavours. I am really grateful for his infinitely patient efforts to helping me sort my occasionally tangled trains of thought to mathematical sound definitions and theorems. He got never tired of talking about my ideas, of proofreading my papers and thesis chapters, or rehearsing my talks.
Moreover, I want to thank Prof. Dr. Dirk Langemann who took a lot load off from me in the last months so that I could finish my thesis and who had a sympathetic ear for my problems whether mathematical or philosophical. The discussions with him improved my work substantially and added new interesting perspectives to it.
I would also like to thank Prof. Dr. Daniel Potts for acquainting me with the concepts of fast algorithms and the NFFT library, which is the important foundation of the algorithms and their implementations presented here. His constructive comments and suggestions during the work on [68] helped me to make necessary improvements.
I would like to thank Dr. Ralf Hielscher with whom I worked on [43] for providing the texture analytical application of the NFSOFT algorithm and Jens Keiner, who has been my co-author in [52] for providing the knowledge on efficient algorithms to compute with semiseparable matrices.
I really enjoyed working at the Institute of Mathematics, University Lübeck, and would like to thank my colleagues for interesting discussions, recreative coffee breaks and being fun to be with.
I highly appreciated the financial support of my doctoral studies by the Graduate School for Computing in Medicine and Life Sciences funded by Germany's Excellence Initiative [DFG GSC 235/1].
Finally, I am heartily thankful to Carsten for his endless patience and encouragement when it was most required and to my family for their constant support and understanding.

2 Rotations and the Rotation Group SO(3)

In the first part of this introductory section, we compile some basic facts and introduce notations concerning the group $\mathrm{SO}(3)$ of rotations in three dimensions. We will give a definition of both, rigid-body rotations and the group consisting of such rotations. Moreover, we will introduce rotation matrices and consider their eigenvalues and eigenvectors. A metric on the rotation group is presented. The second part continues the observation of rotations by presenting different parameterisations of elements of $\mathrm{SO}(3)$ including axis-angle parameterisation, Euler angles and unitary 2×2 matrices. We show how these different parameterisations can be transformed into each other and give an overview on how rotations can be composed and inverted in these different parameterisations. We also briefly consider integration of rotation-dependent functions. If not stated otherwise, most of this section will be based on [17] or [84].

2.1 Three-Dimensional Rotations

Surely, everybody has an intuitive idea what a rotation is. This section aims to provide a mathematical foundation of this intuitive idea that allows us to characterise and to compute with rotations. As stated in the section's title, we are interested in three-dimensional rotations.

Definition 2.1.1 (Rotation)**.** *A rotation of $\mathbb{R}^3$ around the origin $\mathbf{0} \in \mathbb{R}^3$ is a linear map $\rho : \mathbb{R}^3 \to \mathbb{R}^3$ with $\rho(\mathbf{v}) = \mathbf{R}\mathbf{v}$ and an orthogonal matrix $\mathbf{R} \in \mathbb{R}^{3\times 3}$ with $\det(\mathbf{R}) = 1$.*

The composition $\rho = \rho_2 \circ \rho_1$ of two rotations $\rho_1(\mathbf{v}) = \mathbf{R}_1\mathbf{v}$ and $\rho_2(\mathbf{v}) = \mathbf{R}_2\mathbf{v}$ is the map

$$\rho : \mathbf{v} \mapsto \mathbf{R}_2\mathbf{R}_1\mathbf{v}.$$

This can be seen by $\rho(\mathbf{v}) = (\rho_2 \circ \rho_1)(\mathbf{v}) = \rho_2(\rho_1(\mathbf{v})) = \mathbf{R}_2\mathbf{R}_1\mathbf{v}$. The inversion ρ^{-1} of a rotation $\rho(\mathbf{v}) = \mathbf{R}\mathbf{v}$ is the map

$$\rho^{-1} : \mathbf{v} \mapsto \mathbf{R}^{-1}\mathbf{v}$$

as composing ρ^{-1} and ρ gives $\mathbf{v} = \mathrm{id}(\mathbf{v}) = \rho^{-1}(\rho(\mathbf{v})) = \rho^{-1}(\mathbf{R}\mathbf{v})$. This is fulfilled for all $\mathbf{v}$ if $\rho^{-1}(\mathbf{v}) = \mathbf{R}^{-1}\mathbf{v}$.

Lemma 2.1.2. *Given two different orthogonal matrices with determinant one, their corresponding rotations are different as well, i.e. $\mathbf{R}_1 \neq \mathbf{R}_2 \Rightarrow \rho_1 \neq \rho_2$.*

Proof. The two matrices $\mathbf{R}_1, \mathbf{R}_2$ satisfy $\mathbf{R}_2^{-1}\mathbf{R}_1 \neq \mathbf{I}$, hence, there is a vector $\mathbf{v}$ such that

$$\rho_2^{-1}(\rho_1(\mathbf{v})) = \mathbf{R}_2^{-1}\mathbf{R}_1\mathbf{v} \neq \mathbf{v}$$

and therefore $\rho_2^{-1} \circ \rho_1 \neq \mathrm{id}$. ∎

By means of this lemma, we will, from now on, identify a rotation ρ and a matrix $\mathbf{R}$ with each other and refer to $\mathbf{R}$ as a rotation matrix.

Theorem 2.1.3. *The set* $\mathcal{M} = \{\mathbf{R} \in \mathbb{R}^{3\times 3} \mid \det(\mathbf{R}) = 1$ *and* $\mathbf{R}^\mathsf{T}\mathbf{R} = \mathbf{I}\}$ *forms a group with respect to matrix multiplication.*

Proof. G1) Since $\det(\mathbf{R}_1\mathbf{R}_2) = \det(\mathbf{R}_1)\det(\mathbf{R}_2) = 1$ and $(\mathbf{R}_1\mathbf{R}_2)^\mathsf{T}(\mathbf{R}_1\mathbf{R}_2) = \mathbf{R}_2^\mathsf{T}(\mathbf{R}_1^\mathsf{T}\mathbf{R}_1)\mathbf{R}_2) = \mathbf{R}_2^\mathsf{T}\mathbf{R}_2 = \mathbf{I}$, we have $\mathbf{R}_1, \mathbf{R}_2 \in \mathcal{M} \Rightarrow \mathbf{R}_1\mathbf{R}_2 \in \mathcal{M}$, i.e, closure with respect to matrix multiplication.

G2) Seeing that multiplication is associative in general, we have $\mathbf{R}_1(\mathbf{R}_2\mathbf{R}_3) = (\mathbf{R}_1\mathbf{R}_2)\mathbf{R}_3$.

G3) For all $\mathbf{R} \in \mathcal{M}$, $\mathbf{IR} = \mathbf{R}$ holds true. As $\mathbf{I} \in \mathcal{M}$, it is the neutral element of $\mathcal{M}$.

G4) The inverse element of $\mathcal{M}$ is given by $\mathbf{R}^{-1} \in \mathcal{M}$. This is due to $\det(\mathbf{R}^{-1}) = \det(\mathbf{R})^{-1}$ and $(\mathbf{R}^{-1})^\mathsf{T}\mathbf{R}^{-1} = (\mathbf{R}\mathbf{R}^\mathsf{T})^{-1} = \mathbf{I}$. As $\mathbf{R}$ is orthogonal, its inverse is $\mathbf{R}^{-1} = \mathbf{R}^\mathsf{T}$.

■

Definition 2.1.4. *The group* $(\mathcal{M}, \cdot)$ *is called special orthogonal group* $\mathrm{SO}(3)$.

Just like we call the matrices $\mathbf{R}$ rotation matrices, the group $\mathrm{SO}(3)$, which is constituted by rotation matrices, is called the rotation group. Note also that the rotation group $\mathrm{SO}(3)$ is non-abelian. We shall now consider some properties of rotations, from which we will then deduce some more properties of $\mathrm{SO}(3)$.
Every linear map, and in particular a rotation, is completely described by its action on the vectors $\mathbf{v}$ of length $\|\mathbf{v}\| = 1$, with the Euclidean norm $\|\cdot\|$. That is a simple consequence of linearity as $\mathbf{v} = \|\mathbf{v}\|\mathbf{e}$ with $\|\mathbf{e}\| = 1$ with $\rho(\mathbf{v}) = \|\mathbf{v}\|\rho(\mathbf{e})$.

Lemma 2.1.5. *Given three unit vectors* $\mathbf{u}, \mathbf{v}, \mathbf{w} \in \mathbb{R}^3$*, a rotation preserves*

i) the length of a vector $\mathbf{v}$*, i.e.,* $\|\rho(\mathbf{v})\| = \|\mathbf{v}\|$*,*

ii) the angle between two vectors $\mathbf{v}$ *and* $\mathbf{w}$*, i.e.,* $\mathbf{v}\cdot\mathbf{w} = \rho(\mathbf{v})\cdot\rho(\mathbf{w})$*,*

iii) the orientation of three vectors $\mathbf{u}, \mathbf{v}, \mathbf{w}$*, i.e.,* $\det([\mathbf{u}, \mathbf{v}, \mathbf{w}]) = \det([\rho(\mathbf{u}), \rho(\mathbf{v}), \rho(\mathbf{w})])$*,*

where $[\mathbf{u}, \mathbf{v}, \mathbf{w}]$ *denotes a matrix, the columns of which are given by the three vectors,* $\mathbf{u}, \mathbf{v}, \mathbf{w}$.

Proof. We have for

i) $\|\rho(\mathbf{v})\| = \|\mathbf{R}\mathbf{v}\| = \sqrt{(\mathbf{R}\mathbf{v})^\mathsf{T}(\mathbf{R}\mathbf{v})} = \sqrt{\mathbf{v}^\mathsf{T}\mathbf{R}^\mathsf{T}\mathbf{R}\mathbf{v}} = \sqrt{\mathbf{v}^\mathsf{T}\mathbf{v}} = \|\mathbf{v}\|$,

ii) $\rho(\mathbf{v})\cdot\rho(\mathbf{w}) = (\mathbf{R}\mathbf{v})^\mathsf{T}(\mathbf{R}\mathbf{w}) = \mathbf{v}^\mathsf{T}\mathbf{R}^\mathsf{T}\mathbf{R}\mathbf{w} = \mathbf{v}^\mathsf{T}\mathbf{w} = \mathbf{v}\cdot\mathbf{w}$,

iii) and finally

$$\begin{aligned}\det([\rho(\mathbf{u}), \rho(\mathbf{v}), \rho(\mathbf{w})]) &= \det([\mathbf{R}\mathbf{u}, \mathbf{R}\mathbf{v}, \mathbf{R}\mathbf{w}]) = \det(\mathbf{R}[\mathbf{u}, \mathbf{v}, \mathbf{w}]) = \det(\mathbf{R})\det([\mathbf{u}, \mathbf{v}, \mathbf{w}]) \\ &= \det([\mathbf{u}, \mathbf{v}, \mathbf{w}]).\end{aligned}$$

■

The following lemma will show that the properties (i)-(iii) from Lemma 2.1.5 suffice for a linear map ρ to be a rotation. Actually (i) is even an immediate consequence of (ii) when setting $\mathbf{v} = \mathbf{w}$.

Lemma 2.1.6. *A linear map* ρ *fulfilling the properties* $(\mathrm{i}) - (\mathrm{iii})$ *of Lemma 2.1.5 for all unit vectors* $\mathbf{u}, \mathbf{v}, \mathbf{w} \in \mathbb{R}^3$ *is a rotation.*

Proof. Let $\mathbf{R} \in \mathbb{R}^{3\times 3}$ be the matrix of the linear map ρ. An angle preserving transformation described by the matrix $\mathbf{R}$ satisfies

$$\mathbf{v}^T\mathbf{w} = (\mathbf{R}\mathbf{v})^T(\mathbf{R}\mathbf{w}) = \mathbf{v}^T\mathbf{R}^T\mathbf{R}\mathbf{w}$$

for all $\mathbf{w} \in \mathbb{R}^3$. Hence, $\mathbf{v}^T = \mathbf{v}^T\mathbf{R}^T\mathbf{R}$ must hold true for all $\mathbf{v} \in \mathbb{R}^3$ which is only the case if $\mathbf{R}^T\mathbf{R} = \mathbf{I}$, i.e., $\mathbf{R}$ is orthogonal.
If the matrix $\mathbf{R}$, moreover, describes an orientation preserving transformation, it satisfies

$$\det([\mathbf{u},\mathbf{v},\mathbf{w}]) = \det([\mathbf{R}\mathbf{u},\mathbf{R}\mathbf{v},\mathbf{R}\mathbf{w}]) = \det(\mathbf{R})\det([\mathbf{u},\mathbf{v},\mathbf{w}]).$$

Obviously, for arbitrary $\mathbf{u},\mathbf{v},\mathbf{w} \in \mathbb{R}^3$, the equation only holds true for $\det(\mathbf{R}) = 1$. ■

Consider $\mathbf{v}$ to be an eigenvector of the rotation matrix $\mathbf{R}$. By means of the length preservation (Lemma 2.1.5 i)), it fulfils $\|\mathbf{R}\mathbf{v}\| = \|\mathbf{v}\|$. Therefore, we get by $\|\mathbf{R}\mathbf{v}\| = |\lambda|\|\mathbf{v}\| = \|\mathbf{v}\|$ that all eigenvalues of $\mathbf{R}$ have absolute value one, i.e., $|\lambda| = 1$. Since $\mathbf{R}$ is a real-valued 3×3 matrix with $1 = \det(\mathbf{R}) = \lambda_1\lambda_2\lambda_3$, we get one real eigenvalue $\lambda_1 = 1$, a pair of conjugate complex eigenvalues $\lambda_2 = \bar{\lambda}_3$ and hence $\lambda_2\lambda_3 = 1$. This yields $\lambda_2 = 1/\lambda_3 = e^{\pm i\omega}$ with $\omega \in [0,\pi]$.
The eigenvalues of rotation matrices only vary in the argument ω, which can be uniquely determined by the equation

$$\operatorname{tr}(\mathbf{R}) = 1 + e^{i\omega} + e^{-i\omega} = 1 + 2\cos\omega. \tag{2.1}$$

Note that the case $\omega = 0$ leads to $\lambda_1 = \lambda_2 = \lambda_3 = 1$. Taking a closer look on this case, we find that due to the normalisation of its columns, any orthogonal matrix $\mathbf{R} = (g_{ij})_{i,j=1,2,3}$ satisfies $\sum_{j=1}^{3} g_{ij}^2 = 1$ for $i = 1,2,3$, and hence, $|g_{i,j}| \leqslant 1$. If its eigenvalues are equal to one, we conclude from

$$\sum_{i=1}^{3}\sum_{j=1}^{3} g_{ij}^2 = 3 = \sum_{j=1}^{3}\lambda_j = \operatorname{tr}(\mathbf{R}) = \sum_{j=1}^{3} g_{jj}$$

that all non-diagonal elements of $\mathbf{R}$ are zero, while the diagonal elements are one, i.e., $\mathbf{R} = \mathbf{I}$.
Next, we shall use the angle ω to transfer the concept of distance to $\mathrm{SO}(3)$.

Definition 2.1.7 (Distance on $\mathrm{SO}(3)$)**.** *Given $\mathbf{R} \in \mathrm{SO}(3)$, we denote the angle of the rotation $\mathbf{R}$ by $|\mathbf{R}| = \omega$ with*

$$\cos\omega = \frac{\operatorname{tr}(\mathbf{R}) - 1}{2}$$

and $0 \leqslant \omega \leqslant \pi$. The distance between two rotations $\mathbf{R}_1, \mathbf{R}_2 \in \mathrm{SO}(3)$ is defined as the angle of the rotation $\mathbf{R}_2\mathbf{R}_1^{-1}$ that transforms $\mathbf{R}_1$ into $\mathbf{R}_2$, i.e., $\left|\mathbf{R}_2\mathbf{R}_1^{-1}\right|$.

In literature, e.g. [14], we also find the angle of rotation to be called absolute value of the rotation. Later on, in Lemma 2.2.9, we shall see that the just defined distance is indeed a metric on $\mathrm{SO}(3)$.
Using elements of $\mathrm{SO}(3)$ to represent three-dimensional rotations has several computational advantages. Performing a rotation becomes a simple matrix-vector multiplication. By means of a matrix multiplication we cannot only compute a sequence of several rotations, but also combine rotations with other linear transformations like translation or scaling, which can be described by matrix-vector multiplications, too.
But for the applications we have in mind, this representation is not always convenient. Therefore, we introduce different ways to parameterise rotations in the next section. A central thought to this is that any element of $\mathrm{SO}(3)$ can be uniquely described using three parameters which are, in case, the entries

of a rotation matrix.
A rotation matrix $\mathbf{R} = [\mathbf{g}_1, \mathbf{g}_2, \mathbf{g}_3]$ where each vector $\mathbf{g}_i \in \mathbb{R}^3$ for $i = 1, 2, 3$ represents a column of the matrix, has nine entries. But they can not be chosen freely as we impose certain constraints on them. As $\mathbf{R}$ is orthogonal, we have $\mathbf{g}_i \cdot \mathbf{g}_j = \delta_{i,j}$ for $i, j = 1, 2, 3$. As the inner product is commutative we have six constraints for the nine entries of $\mathbf{R}$. This reduced the number of freely eligible entries of $\mathbf{R}$ from nine to three. The condition $\det(\mathbf{R}) = 1$, will reduce the possible choices of the matrix elements but not their total number. We will refer to the three eligible entries as the three degrees of freedom in a rotation.
Just as the rotation matrices, the parameterisations of rotations in the following section also need to have three degrees of freedom; they will just be described in a different way.

2.2 Parameterisations of Rotations

In this section, several parameterisations of rotations are reviewed from [84, Sect. 1.4] and [17, Sect. 5.4] and will be discussed in more detail.

Axis-Angle Parameterisation We start with the maybe most intuitive way to describe a rotation, i.e., by means of a rotation axis around which the rotation takes place and a rotation angle that describes how much the object is rotated.

Definition 2.2.1 (Axis of a Rotation)**.** *Let a rotation be described by the matrix* $\mathbf{R} = (g_{jk})_{j,k=1,\dots,3} \in \mathrm{SO}(3)$ *with* $\mathbf{R} \neq \mathbf{I}$*. We define the axis of the rotation to be the normalised eigenvector* $\mathbf{r}$ *to the eigenvalue* $\lambda = 1$ *of* $\mathbf{R}$*.*

Note that we excluded the case $\mathbf{R} = \mathbf{I}$ in this definition. This is due to the fact that $\mathbf{R} = \mathbf{I}$ has a threefold eigenvalue one and therefore no uniquely determined normalised eigenvector to this eigenvalue. While Definition 2.1.7 gives a formula to retrieve the rotation angle out of a rotation matrix, we give the following lemma for a computation of the rotation axis. Again the case $\mathbf{R} = \mathbf{I}$ is excluded but still, the rotation angle of this matrix can be determined and yields $\omega = |\mathbf{I}| = 0$.

Lemma 2.2.2. *Let* $\mathbf{r}$ *be the axis of the rotation* $\mathbf{R} = (g_{jk})_{j,k=1,2,3} \in \mathrm{SO}(3)$ *with* $\mathbf{R} \neq \mathbf{I}$ *and let* $\mathbf{v} \in \mathbb{R}^3$*. Then* $\mathbf{r}$ *is given by*

$$\mathbf{r} = \frac{1}{||\mathbf{v}||}\mathbf{v} \text{ with } \mathbf{v} = \begin{pmatrix} g_{23} - g_{32} \\ g_{31} - g_{13} \\ g_{12} - g_{21} \end{pmatrix}.$$

Proof. Let $\mathbf{u}$ be an eigenvector of the rotation matrix $\mathbf{R}$ to $\lambda = 1$, we have $\mathbf{Ru} = \mathbf{u}$. Multiplying this with $\mathbf{R}^\mathsf{T}$ we obtain $\mathbf{u} = \mathbf{R}^\mathsf{T}\mathbf{u}$. Combining the two yields $(\mathbf{R} - \mathbf{R}^\mathsf{T})\mathbf{u} = \mathbf{0}$. Hence for $\mathbf{u} = (u_1, u_2, u_3)^\mathsf{T}$, we have

$$\begin{aligned} u_2(g_{12} - g_{21}) - u_3(g_{31} - g_{13}) &= 0, \\ u_3(g_{23} - g_{32}) - u_1(g_{12} - g_{21}) &= 0, \\ u_1(g_{31} - g_{13}) - u_2(g_{23} - g_{32}) &= 0. \end{aligned}$$

This is solved for arbitrary multiples of $\mathbf{u} = (g_{23} - g_{32}, g_{31} - g_{13}, g_{12} - g_{21})^\mathsf{T} = \mathbf{v}$. By Definition 2.2.1 the rotation axis $\mathbf{r}$ is the normalised eigenvector to the eigenvalue $\lambda = 1$ and hence, normalisation of $\mathbf{v}$ proves the lemma. ∎

Now, we investigate some special sets of rotations. We denote the set $\mathcal{Z} = \{\mathbf{Z} \in \mathrm{SO}(3) \mid \mathbf{Z}\mathbf{e}_z = \mathbf{e}_z\}$ of all rotations conserving $\mathbf{e}_z = (0,0,1)^\mathrm{T}$.

Lemma 2.2.3. *Every rotation $\mathbf{Z} \in \mathcal{Z}$ fulfils*

$$\mathbf{Z} = \begin{pmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

for some $\gamma \in [0, 2\pi)$.

Proof. Let $\mathbf{Z} = (z_{\mathrm{j}\mathrm{k}})_{\mathrm{j},\mathrm{k}=1,2,3}$. As $\mathbf{Z}\mathbf{e}_z = (z_{13}, z_{23}, z_{33})^\mathrm{T} = \mathbf{e}_z$, we immediately obtain $z_{13} = z_{23} = 0$ and $z_{33} = 1$. From $\mathbf{e}_z = \mathbf{Z}^\mathrm{T}\mathbf{e}_z$, we additionally get $z_{31} = z_{32} = 0$.
Since now $\mathbf{Z}$ is orthogonal having determinant one, the matrix

$$\begin{pmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \end{pmatrix} \in \mathbb{R}^{2\times 2}$$

formed by the remaining entries of $\mathbf{Z}$ is an orthogonal matrix with determinant one, too. In fact, it is an element of the group $\mathrm{SO}(2)$, defined completely analogous to the $\mathrm{SO}(3)$ as the group of orthogonal 2×2 matrices having determinant one.
In the same manner elements of $\mathrm{SO}(3)$ describe rotations in three dimensions, an element of $\mathrm{SO}(2)$ describes a planar rotation, and we have

$$\begin{pmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \end{pmatrix} = \begin{pmatrix} \cos\gamma & -\sin\gamma \\ \sin\gamma & \cos\gamma \end{pmatrix} \in \mathrm{SO}(2)$$

for some $\gamma \in [0, 2\pi)$. ∎

Note that, in fact, $\mathcal{Z}$ is a subgroup of $\mathrm{SO}(3)$ that is isomorphic to $\mathrm{SO}(2)$.

Remark 2.2.4. *If we compute the rotation axis and angle of an element $\mathbf{Z} \in \mathcal{Z}$, we see that for $\gamma \in [0, \pi)$ we have $\mathbf{r} = \mathbf{e}_z$ and $\omega = \gamma$, while for $\gamma \in [\pi, 2\pi)$ we have $\mathbf{r} = -\mathbf{e}_z$ and $\omega = 2\pi - \gamma$.*

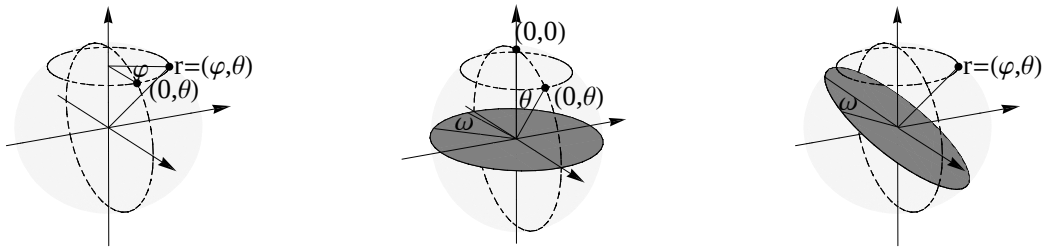
Remark 2.2.5. *Analogously to Lemma 2.2.3, we can show that for $\mathbf{e}_\mathrm{y} = (0,1,0)^\mathrm{T}$ the set $\mathcal{Y} = \{\mathbf{Y} \in \mathrm{SO}(3) \mid \mathbf{Y}\mathbf{e}_\mathrm{y} = \mathbf{e}_\mathrm{y}\}$ contains the rotations*

$$\mathbf{Y} = \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix}$$

for $\beta \in [0, 2\pi)$.

Corollary 2.2.6. *Given a rotation axis $\mathbf{r}$, a fixed rotation matrix $\mathbf{U} \in \mathrm{SO}(3)$ with $\mathbf{U}\mathbf{e}_z = \mathbf{r}$, then every $\mathbf{R} \in \mathrm{SO}(3)$ with $\mathbf{R}\mathbf{e}_z = \mathbf{r}$ can be decomposed into $\mathbf{R} = \mathbf{U}\mathbf{Z}$, for some $\mathbf{Z} \in \mathcal{Z}$.*

Proof. A rotation $\mathbf{R}$ with $\mathbf{R}\mathbf{e}_z = \mathbf{r}$ satisfies $\mathbf{U}^\mathrm{T}\mathbf{R}\mathbf{e}_z = \mathbf{e}_z$ and therefore $\mathbf{Z} = \mathbf{U}^\mathrm{T}\mathbf{R} \in \mathcal{Z}$. Thus, we have $\mathbf{R} = \mathbf{U}\mathbf{Z}$ for some $\mathbf{Z} \in \mathcal{Z}$. ∎



a) b) c)

Figure 2.1: In a) we see an arbitrary rotation axis $\mathbf{r}(\varphi,\theta) \in \mathbb{S}^2$ which is rotated about the angle φ around the z-axis. The resulting new axis $(0,\theta)$ is rotated in b) around the y-axis to point to the north pole of the sphere. The axis $\mathbf{r}$ has now been rotated to the z-axis. Then we perform a rotation about ω around this axis. After this, the two rotations about θ and φ will be reversed, leading, in c), to the actual rotation described by $\mathbf{r}$ and ω. This gives the formula from Lemma 2.2.8.

In fact, $\mathbf{U}\mathcal{Z}$ is a left coset of the subgroup $\mathcal{Z}$ in $\mathrm{SO}(3)$. The set of all such cosets is isomorphic to the two-dimensional unit sphere

$$\mathbb{S}^2 = \{\mathbf{x} \in \mathbb{R}^3 \mid \|\mathbf{x}\| = 1\}.$$

As a rotation axis $\mathbf{r} \in \mathbb{R}^3$ satisfies $\|\mathbf{r}\| = 1$, we have $\mathbf{r} \in \mathbb{S}^2$. Any vector $\mathbf{r} \in \mathbb{S}^2$, and in particular rotation axes, with given Cartesian coordinates $\mathbf{r} = (r_1, r_2, r_3)^{\mathrm{T}}$ can be rewritten in spherical coordinates $\mathbf{r} = (\varphi, \theta)$ where the angles $\theta \in [0, \pi]$ and $\varphi \in [0, 2\pi)$ denote the latitude and the longitude of the point $\mathbf{r}$ on $\mathbb{S}^2$, respectively. For $r_1^2 + r_2^2 > 0$, we have

$$\varphi \quad = \quad \begin{cases} \arccos \frac{r_1}{\sqrt{r_1^2+r_2^2}} & \text{for } r_2 \geqslant 0, \\ 2\pi - \arccos \frac{r_1}{\sqrt{r_1^2+r_2^2}} & \text{for } r_2 < 0, \end{cases} \qquad \text{and } \theta = \arccos r_3.$$

If $r_1 = r_2 = 0$, the longitude φ of the point $\mathbf{r}$ is not uniquely determined. In reverse,

$$\begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix} \quad = \quad \begin{pmatrix} \sin\theta\cos\varphi \\ \sin\theta\sin\varphi \\ \cos\theta \end{pmatrix}$$

holds true.

Definition 2.2.7 (Axis-Angle Parameterisation)**.** *Given a rotation axis* $\mathbf{r} = (\varphi, \theta) \in \mathbb{S}^2$ *and a rotation angle* $\omega \in [0, \pi]$*, we define* $\mathbf{R}_{\mathbf{r}}(\omega) \in \mathrm{SO}(3)$ *by*

$$\mathbf{R}_{\mathbf{r}}(\omega) = \mathbf{R}_{\mathbf{e}_z}(\varphi)\mathbf{R}_{\mathbf{e}_y}(\theta)\mathbf{R}_{\mathbf{e}_z}(\omega)\mathbf{R}^{\mathrm{T}}_{\mathbf{e}_y}(\theta)\mathbf{R}^{\mathrm{T}}_{\mathbf{e}_z}(\varphi).$$

In the notation of Definition 2.2.7, we can denote all elements of the set $\mathcal{Z}$ from Lemma 2.2.3 by $\mathbf{R}_{\mathbf{e}_z}(\omega)$.

Lemma 2.2.8. *Given a rotation axis* $\mathbf{r} = \mathbf{r}(\varphi, \theta)$ *and rotation angle* ω*, a rotation* $\mathbf{R} \in \mathrm{SO}(3)$ *is uniquely determined by* $\mathbf{R} = \mathbf{R}_{\mathbf{r}}(\omega)$.

Proof. By Definition 2.1.7 and (2.1), the rotation angle ω uniquely determines the eigenvalues of $\mathbf{R}$ as $\lambda_1 = 1$ and $\lambda_2 = 1/\lambda_3 = e^{\pm i\omega}$ with $\omega \in [0, \pi]$.
The matrix of a rotation with rotation axis $\mathbf{r}$, satisfies $\mathbf{Rr} = \mathbf{r}$. By Corollary 2.2.6, the rotations $\mathbf{UZ}$ with a fixed $\mathbf{U}$ satisfying $\mathbf{Ue}_z = \mathbf{r}$ and an arbitrary $\mathbf{Z} \in \mathcal{Z}$ are all rotations fulfilling $\mathbf{r} = \mathbf{UZe}_z$. Hence, we have $\mathbf{RUZe}_z = \mathbf{UZe}_z$ and $\mathbf{Z}^T\mathbf{U}^T\mathbf{RUZe}_z = \mathbf{e}_z$.
The matrix $\mathbf{A} = \mathbf{Z}^T\mathbf{U}^T\mathbf{RUZ} \in \mathcal{Z}$ has the same eigenvalues as $\mathbf{R}$. By Lemma 2.2.3, we get

$$\mathbf{A} = \begin{pmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

with $\gamma \in [0, 2\pi)$. Its spectral decomposition is given by $\mathbf{A} = \mathbf{V}\mathbf{D}\overline{\mathbf{V}}^T$. Since the eigenvectors of $\mathbf{A}$ are known, we find with Remark 2.2.4 that

$$\mathbf{V} = \begin{pmatrix} z & z & 0 \\ -iz & iz & 0 \\ 0 & 0 & 1 \end{pmatrix} \text{ and } \mathbf{D} = \begin{pmatrix} e^{i\omega} & 0 & 0 \\ 0 & e^{-i\omega} & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

with $z = \dfrac{1}{\sqrt{2}}$ and the rotation angle ω. Note, that for $\gamma \in [\pi, 2\pi)$, the rotation axis $-\mathbf{e}_z$ is an eigenvector as well as $\mathbf{e}_z$ itself.
Since $\mathbf{A} \in \mathcal{Z}$, we have $\mathbf{A} = \mathbf{ZAZ}^T$ and we can conclude

$$\mathbf{R} = \mathbf{UZAZ}^T\mathbf{U}^T = \mathbf{UAU}^T = \mathbf{U}\left(\mathbf{V}\mathbf{D}\overline{\mathbf{V}}^T\right)\mathbf{U}^T = (\mathbf{UV})\mathbf{D}(\overline{\mathbf{UV}})^T$$

as $\mathbf{U} = \overline{\mathbf{U}}$. We obtain the eigenvectors of $\mathbf{R}$ in $\mathbf{UV}$ independently of $\mathbf{Z} \in \mathcal{Z}$. Having the spectral decomposition of $\mathbf{R}$ it is uniquely determined. ∎

A geometric interpretation of Definiton 2.2.7 and Lemma 2.2.8 specifying the rotation axis is the following. The rotation axis is invariant under rotation about this axis. Converting $\mathbf{r}$ into Cartesian coordinates gives $\mathbf{r} = (\sin\theta\cos\varphi, \sin\theta\sin\varphi, \cos\theta)^T$. This vector satisfies $\mathbf{R}^T_{\mathbf{e}_y}(\theta)\mathbf{R}^T_{\mathbf{e}_z}(\varphi)\mathbf{r} = \mathbf{e}_z$. By $\mathbf{R}_{\mathbf{e}_z}(\omega)\mathbf{e}_z = \mathbf{e}_z$, we can verify

$$\left(\mathbf{R}_{\mathbf{e}_z}(\varphi)\mathbf{R}_{\mathbf{e}_y}(\theta)\right)\mathbf{R}_{\mathbf{e}_z}(\omega)\left(\mathbf{R}^T_{\mathbf{e}_y}(\theta)\mathbf{R}^T_{\mathbf{e}_z}(\varphi)\right)\mathbf{r} = \mathbf{Rr} = \mathbf{r},$$

i.e., that $\mathbf{r}$ is the rotation axis of $\mathbf{R}$. This idea is also depicted in Figure 2.1.
Let us now list a few more useful facts about the axis-angle parameterisation. Given two rotations $\mathbf{R}_{\mathbf{r}_1}(\omega_1), \mathbf{R}_{\mathbf{r}_2}(\omega_2) \in \mathrm{SO}(3)$, the combined rotation $\mathbf{R}_{\mathbf{r}}(\omega) = \mathbf{R}_{\mathbf{r}_1}(\omega_1)\mathbf{R}_{\mathbf{r}_2}(\omega_2)$ is determined by

$$\begin{aligned} \mathbf{r} &= \sin\frac{\omega_1}{2}\cos\frac{\omega_2}{2}\,\mathbf{r}_2 + \sin\frac{\omega_2}{2}\cos\frac{\omega_1}{2}\,\mathbf{r}_1 + \cos\frac{\omega_1}{2}\cos\frac{\omega_2}{2}\,\mathbf{r}_1 \times \mathbf{r}_2, \\ \cos\frac{\omega}{2} &= \cos\frac{\omega_1}{2}\cos\frac{\omega_2}{2} - \sin\frac{\omega_1}{2}\sin\frac{\omega_2}{2}\mathbf{r}_1 \cdot \mathbf{r}_2. \end{aligned} \tag{2.2}$$

This can be verified using Definition 2.2.7. For the identity element $\mathbf{R}_{\mathbf{r}}(\omega) = \mathbf{I}$ of $\mathrm{SO}(3)$, we get by Definition 2.1.7, $\omega = 0$ for the rotation angle while the rotation axis is not uniquely defined. In fact, $\mathbf{R}_{\mathbf{r}_0}(0) = \mathbf{I}$ holds true for any rotation axis $\mathbf{r}_0 \in \mathbb{S}^2$.
The back rotation to $\mathbf{R}_{\mathbf{r}}(\omega)$, i.e., the inverse element, is given by $\mathbf{R}_{-\mathbf{r}}(\omega)$.

The axis-angle parameterisation of rotations helps us to verify the following property of the rotation group $\mathrm{SO}(3)$.

Lemma 2.2.9. *The distance between two rotations, cf. Definition 2.1.7, defines a metric on* SO(3).

Proof. Let $\mathbf{R}_1, \mathbf{R}_2, \mathbf{R}_3 \in \mathrm{SO}(3)$ describe rotations. Let the rotation angle and rotation axis of the combined rotation $\mathbf{R}_j\mathbf{R}_i^{-1}$ for $i, j = 1, 2, 3$ be denoted by $\omega_{i,j}$ and $\mathbf{r}_{i,j}$, respectively.

i) The identity of indiscernibles follows from $\omega_{1,2} = 0 \Leftrightarrow \mathbf{R}_2\mathbf{R}_1^{-1} = \mathbf{I} \Leftrightarrow \mathbf{R}_1 = \mathbf{R}_2$.

ii) As the trace of an orthogonal matrix and its inverse, i.e., its transposed, are equal, symmetry of the distance is shown by $\mathrm{tr}(\mathbf{R}_2\mathbf{R}_1^{-1}) = \mathrm{tr}((\mathbf{R}_2\mathbf{R}_1^{-1})^\mathsf{T}) = \mathrm{tr}(\mathbf{R}_1\mathbf{R}_2^{-1}) \Leftrightarrow \omega_{1,2} = \omega_{2,1}$.

iii) The rotation $\mathbf{R}_3\mathbf{R}_1^{-1}$ can be rewritten as the composition $\mathbf{R}_3\mathbf{R}_1^{-1} = (\mathbf{R}_3\mathbf{R}_2^{-1})(\mathbf{R}_2\mathbf{R}_1^{-1})$. Its rotation angle $\omega_{1,3}$ is determined by

$$\cos\frac{\omega_{1,3}}{2} = \cos\frac{\omega_{1,2}}{2}\cos\frac{\omega_{2,3}}{2} - \sin\frac{\omega_{1,2}}{2}\sin\frac{\omega_{2,3}}{2}\mathbf{r}_{1,2}\cdot\mathbf{r}_{2,3},$$

cf. (2.2). As $|\mathbf{r}_{1,2}\cdot\mathbf{r}_{2,3}| \leqslant 1$, we have

$$\cos\frac{\omega_{1,3}}{2} \geqslant \cos\frac{\omega_{1,2}}{2}\cos\frac{\omega_{2,3}}{2} - \sin\frac{\omega_{1,2}}{2}\sin\frac{\omega_{2,3}}{2} = \cos\frac{\omega_{1,2}+\omega_{2,3}}{2}$$

which proves the triangle inequality $\omega_{1,3} \leqslant \omega_{1,2} + \omega_{2,3}$.

■

Euler Angle Parameterisation In the axis-angle parameterisation, we had three degrees of freedom to describe a rotation uniquely, the angle ω and the two spherical coordinates φ, θ to denote the rotation axis $\mathbf{r}$. Now, we use a different way to parameterise these three degrees of freedom, namely we choose three successive rotations about independent axes and use their absolute values to characterise a rotation. As any triplet of rotations about fixed axes can be used to uniquely describe arbitrary rotations as long as two consecutive rotations have linear independent axes, there exist different conventions for choosing those axes, e.g. [84, 91] use a rotation around $\mathbf{e}_z$ followed by a rotation around $\mathbf{e}_y$ and another rotation around $\mathbf{e}_z$, while [17] use $\mathbf{e}_x$ for the second rotation instead of $\mathbf{e}_y$. Here, we shall define Euler angles as follows.

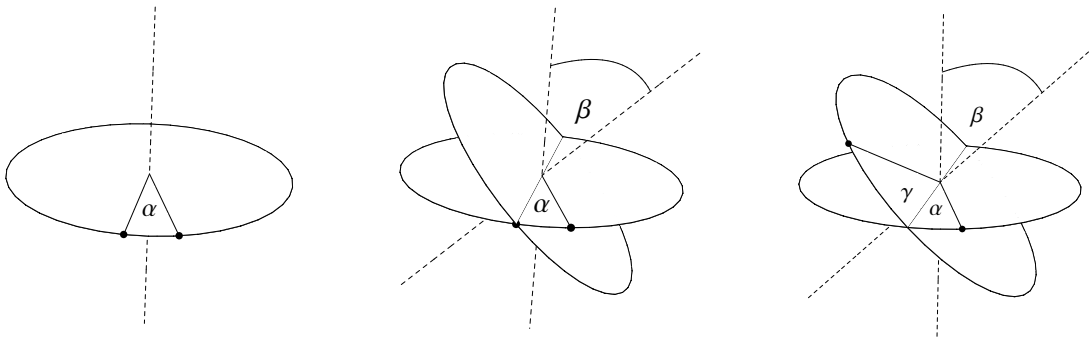
Definition 2.2.10 (Euler Angles)**.** *Given three angles $\alpha, \gamma \in [0, 2\pi)$ and $\beta \in [0, \pi]$, a rotation matrix $\mathbf{R}(\alpha, \beta, \gamma)$ is given by*

$$\mathbf{R}(\alpha, \beta, \gamma) = \mathbf{R}_{\mathbf{e}_z}(\alpha)\mathbf{R}_{\mathbf{e}_y}(\beta)\mathbf{R}_{\mathbf{e}_z}(\gamma).$$

The three angles α, β and γ are called Euler angles of the rotation $\mathbf{R}(\alpha, \beta, \gamma)$, (see Figure 2.2).

Throughout this work, we use this convention for the Euler angles. Whenever occurring, we set $\alpha \to \alpha \mod 2\pi$ and $\gamma \to \gamma \mod 2\pi$. Definition 2.2.10 assigns a rotation matrix to a given set of Euler angles. In the following corollary, we describe a way how to compute the set of Euler angles out of a given a rotation matrix.

Corollary 2.2.11. *A rotation, specified by the rotation matrix $\mathbf{G} = (g_{jk})_{j,k=1,2,3} \in \mathrm{SO}(3)$, is described by the Euler angles α, β and γ for the following choices of Euler angles.*



a) b) c)

Figure 2.2: The Euler angles, described in Definition 2.2.10, are three consecutive rotations around the z-axis (a), the y-axis (b) and again the z-axis (c).

If $|g_{33}| \neq 1$*, then* $\mathbf{R}(\alpha, \beta, \gamma)$ *with*

$$\beta = \arccos g_{33}$$

$$\alpha = \begin{cases} \arccos \frac{g_{13}}{\sqrt{g_{13}^2+g_{23}^2}} & \text{for } g_{23} > 0, \\ 2\pi - \arccos \frac{g_{13}}{\sqrt{g_{13}^2+g_{23}^2}} & \text{for } g_{23} < 0, \end{cases}$$

$$\gamma = \begin{cases} \arccos \frac{-g_{31}}{\sqrt{g_{31}^2+g_{32}^2}} & \text{for } g_{32} > 0, \\ 2\pi - \arccos \frac{-g_{31}}{\sqrt{g_{31}^2+g_{32}^2}} & \text{for } g_{32} < 0. \end{cases}$$

describes the same rotation as $\mathbf{G}$*. If* $g_{33} = 1$ *then* $\mathbf{R}(\alpha, \beta, \gamma)$ *with* $\beta = 0$ *and*

$$\alpha + \gamma = \begin{cases} \arccos g_{11} & \text{for } g_{21} \geqslant 0, \\ 2\pi - \arccos g_{11} & \text{for } g_{21} < 0. \end{cases}$$

describes the same rotation as $\mathbf{G}$*.*
Likewise, for $g_{33} = -1$*, all Euler angels with* $\beta = \pi$ *and*

$$\alpha - \gamma = \begin{cases} \arccos(-g_{11}) & \text{for } g_{21} \leqslant 0, \\ 2\pi - \arccos(-g_{11}) & \text{for } g_{21} > 0. \end{cases}$$

specify the same rotation as $\mathbf{G}$*.*

Proof. Given a matrix $\mathbf{G} = (g_{ij}) \in \mathrm{SO}(3)$ for $i, j = 1, 2, 3$ and matrix

$$\mathbf{R}(\alpha, \beta, \gamma) = \begin{pmatrix} \cos\alpha\cos\beta\cos\gamma - \sin\alpha\sin\gamma & -\cos\gamma\sin\alpha - \cos\alpha\cos\beta\sin\gamma & \cos\alpha\sin\beta \\ \cos\beta\cos\gamma\sin\alpha + \cos\alpha\sin\gamma & \cos\alpha\cos\gamma - \cos\beta\sin\alpha\sin\gamma & \sin\alpha\sin\beta \\ -\cos\gamma\sin\beta & \sin\beta\sin\gamma & \cos\beta \end{pmatrix}$$

given in Euler angles as in Definition 2.2.10. We are now examining how to choose α, β and γ such that $\mathbf{R}(\alpha, \beta, \gamma) = \mathbf{G}$. The comparison shows that β has to be chosen such that $g_{33} = \cos\beta$.

Let us assume for now that $\beta \notin \{0, \pi\}$ and hence $\sin\beta \neq 0$. Considering the remaining entries of the last row of $\mathbf{R}(\alpha, \beta, \gamma)$ we can determine $\gamma \in [0, 2\pi]$ from

$$\cos\gamma = -\frac{r_{31}}{\sin\beta}, \ \sin\gamma = \frac{r_{32}}{\sin\beta}$$

using $\sin\beta = \sqrt{1 - g_{33}^2} = \sqrt{g_{31}^2 + g_{32}^2}$ as

$$\gamma = \begin{cases} \arccos \frac{-g_{31}}{\sqrt{g_{31}^2+g_{32}^2}} & \text{for } g_{32} > 0, \\ 2\pi - \arccos \frac{-g_{31}}{\sqrt{g_{31}^2+g_{32}^2}} & \text{for } g_{32} < 0. \end{cases}$$

In an analogous manner we obtain α using g_{13} and g_{23}.
It remains to examine the cases in which $|g_{33}| = 1$ and hence $\beta \in \{0, \pi\}$. If $g_{33} = 1$, due to the normalisation of the rows and columns of a rotation matrix, we have $g_{j3} = g_{3j} = 0$ for $j = 1, 2$. On the other hand, a rotation matrix in Euler angles with the same third row and column is given by

$$\mathbf{R}(\alpha, 0, \gamma) = \begin{pmatrix} \cos(\alpha+\gamma) & -\sin(\alpha+\gamma) & 0 \\ \sin(\alpha+\gamma) & \cos(\alpha+\gamma) & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

If we now insert

$$\alpha + \gamma = \begin{cases} \arccos g_{11} & \text{for } g_{21} \geqslant 0, \\ 2\pi - \arccos g_{11} & \text{for } g_{21} < 0, \end{cases}$$

into the matrix $\mathbf{R}(\alpha, 0, \gamma) = (r_{ij})_{i,j=1,2,3}$ we find that also $r_{11} = g_{11}$. The equalities $r_{12} = g_{12}$, $r_{21} = g_{21}$, $r_{21} = g_{21}$ and consequently $r_{21} = g_{21}$ follow from row normalisation, up to a sign. The correct sign is established by the given case distinction. Consequently the matrices $\mathbf{G}$ and $\mathbf{R}(\alpha, 0, \gamma)$ are equal, and therefore, describe the same rotation. Note that the proof for $g_{33} = -1$ is completely analogous and will be omitted. ■

Corollary 2.2.11 establishes a rule how to compute Euler angles out of an arbitrary rotation matrix. The following lemma discusses the uniqueness of this assignment.

Lemma 2.2.12. *Every rotation matrix* $\mathbf{R} = (r_{jk})_{j,k=1,2,3} \in \mathrm{SO}(3)$ *with* $|r_{33}| \neq 1$ *has uniquely determined Euler angles.*

Proof. Consider two triples of Euler angles (α, β, γ) and $(\alpha', \beta', \gamma')$ and a rotation $\mathbf{R}$ which can be written in terms of Euler angles using Definition 2.2.10 as

$$\mathbf{R} = \mathbf{R}_{\mathbf{e}_z}(\alpha)\mathbf{R}_{\mathbf{e}_y}(\beta)\mathbf{R}_{\mathbf{e}_z}(\gamma) = \mathbf{R}_{\mathbf{e}_z}(\alpha')\mathbf{R}_{\mathbf{e}_y}(\beta')\mathbf{R}_{\mathbf{e}_z}(\gamma').$$

Rewriting this as

$$\mathbf{R}_{\mathbf{e}_z}(\alpha - \alpha')\mathbf{R}_{\mathbf{e}_y}(\beta) = \mathbf{R}_{\mathbf{e}_y}(\beta')\mathbf{R}_{\mathbf{e}_z}(\gamma' - \gamma) \tag{2.3}$$

and applying $\mathbf{e}_z$ on both sides of the equation yields

$$\mathbf{R}_{\mathbf{e}_z}(\alpha - \alpha')\mathbf{R}_{\mathbf{e}_y}(\beta)\mathbf{e}_z = \mathbf{R}_{\mathbf{e}_y}(\beta')\mathbf{R}_{\mathbf{e}_z}(\gamma' - \gamma)\mathbf{e}_z = \mathbf{R}_{\mathbf{e}_y}(\beta')\mathbf{e}_z.$$

Inserting the formula for a rotation around $\mathbf{e}_z$ from Lemma 2.2.3 and for a rotation around $\mathbf{e}_y$ from Remark 2.2.5, we obtain

$$\begin{pmatrix} \sin\beta\cos(\alpha-\alpha') \\ \sin\beta\sin(\alpha-\alpha') \\ \cos\beta \end{pmatrix} = \begin{pmatrix} \sin\beta' \\ 0 \\ \cos\beta' \end{pmatrix}.$$

From the initial condition $|r_{33}| \neq 1$, we get by Lemma 2.2.3 that $\mathbf{R} \notin \mathcal{Z}$ and $-\mathbf{R} \notin \mathcal{Z}$. This restricts the Euler angle β to the interval $(0,\pi)$ and hence we have $\sin\beta \neq 0$.
Looking at the third component of the above vector, we see $\beta = \beta'$, and as $\sin\beta \neq 0$, the equality for the second component is only satisfied for $\sin(\alpha-\alpha') = 0$, i.e., $\alpha - \alpha' = k\pi$, $k \in \mathbb{N}$.
Inserting this on the first component, we get $\sin\beta\cos(k\pi) = \sin\beta$. This however holds true only for $\cos(k\pi) = 1$, i.e., for even k. Hence, we get $\alpha - \alpha' = 2k\pi$ which, by definition of the Euler angles means that $\alpha = \alpha'$. Now (2.3) gives $\mathbf{R}_{\mathbf{e}_z}(\gamma'-\gamma) = \mathbf{I}$ and hence $\gamma' = \gamma$. ■

Let us now list a few more useful facts about this representation. In contrast to the axis-angle representation, a composed rotation can not conveniently be simplified in Euler angles. The explicit formula is, therefore, omitted here. Instead, we give another useful formula which almost provides the Euler angle representation of a combined rotation. Combining $\mathbf{R}(\alpha_1,\beta_1,\gamma_1)$ and $\mathbf{R}(\alpha_2,\beta_2,\gamma_2)$ using Definition 2.2.10 yields

$$\mathbf{R}(\alpha_1,\beta_1,\gamma_1)\mathbf{R}(\alpha_2,\beta_2,\gamma_2) = \mathbf{R}_{\mathbf{e}_y}(\beta_1)\mathbf{R}(\alpha_2+\gamma_1,\beta_2,\alpha_2+\gamma_1).$$

The identity element of $SO(3)$, $\mathbf{R}(\alpha,\beta,\gamma) = \mathbf{I}$, is not uniquely determined in Euler angles and can be obtained by setting $\beta = 0$ and $\alpha \pm \gamma = 2\pi k$ for $k \in \mathbb{N}$, see Corollary 2.2.11. The back rotation, i.e., the inverse element to $\mathbf{R}(\alpha,\beta,\gamma)$ for $\beta \neq 0$ is given by $\mathbf{R}(2\pi-\gamma,\beta,2\pi-\alpha)$. The back rotation of $\mathbf{R}(\alpha,0,\gamma)$ is given by $\mathbf{R}_{\mathbf{e}_z}(-(\alpha+\gamma))$.

Combining Lemma 2.2.2 and Lemma 2.2.12, it becomes obvious that one can also translate the axis-angle representation into Euler angles and vice versa.
Given three Euler angles $\alpha,\gamma \in [0,2\pi)$ and $\beta \in [0,\pi]$, we have $\mathbf{R}_r(\omega) = \mathbf{R}(\alpha,\beta,\gamma)$ with rotation angle and axis

$$\begin{aligned} \omega &= 2\arccos\left(\cos\frac{\beta}{2}\cos\frac{\alpha+\gamma}{2}\right) \\ \mathbf{r} = (\varphi,\theta) &= \begin{cases} \left(\frac{\pi}{2}+\alpha,\ \frac{\pi}{2}\right), & \text{for } \alpha+\gamma = 0, \\ \left(\frac{\pi}{2}+\frac{\alpha-\gamma}{2},\ \arctan\left(\frac{\tan\frac{\beta}{2}}{\sin\frac{\alpha+\gamma}{2}}\right)\right), & \text{otherwise.} \end{cases} \end{aligned} \tag{2.4}$$

This can be verified using Corollary 2.2.11 to compute a rotation matrix out of the Euler angles followed by calculating the eigenvalues and eigenvectors of that matrix to obtain rotation axis and angle, see also [84, p.26]. In case we are given $\mathbf{R}_r(\omega)$, with $\omega \in (0,\pi)$ the Euler angles can be computed by

$$\begin{aligned} \alpha &= \arctan\left(\cos\theta\tan\frac{\omega}{2}\right) + \varphi - \frac{\pi}{2}, \\ \beta &= 2\arcsin\left(\sin\theta\sin\frac{\omega}{2}\right), \\ \gamma &= \arctan\left(\cos\theta\tan\frac{\omega}{2}\right) - \varphi + \frac{\pi}{2}. \end{aligned}$$

This can be seen by using the Definitions 2.2.7 and 2.2.10. Note, taht we excluded $\omega \in \{0, \pi\}$. In that case, we do not find unique Euler angles, but can only determine the sum $\alpha + \gamma$ and β.
These transitions between Euler angles and axis-angle parameterisation will be especially useful in Section 3.5.
The next parameterisation we will discuss gives a connection between the rotation group $\mathrm{SO}(3)$ and another matrix group which is in some sense a generalisation of $\mathrm{SO}(3)$ and which we are going to consider in Section 5.1.

Special unitary 2×2 matrices In the following, we present a way to characterise rotations using special complex-valued 2×2 matrices, rather than real-valued orthogonal 3×3 matrices.

Theorem 2.2.13. *The set* $\mathcal{N} = \{\mathbf{U} \in \mathbb{C}^{2\times 2} \mid \det(\mathbf{U}) = 1 \text{ and } \overline{\mathbf{U}}^{\mathsf{T}}\mathbf{U} = \mathbf{I}\}$ *forms a group with respect to matrix multiplication.*

Proof. G1) Let $\mathbf{U}_1, \mathbf{U}_2 \in \mathcal{N}$, since $\det(\mathbf{U}_1\mathbf{U}_2) = \det(\mathbf{U}_1)\det(\mathbf{U}_2) = 1$ and $(\overline{\mathbf{U}_1\mathbf{U}_2})^{\mathsf{T}}(\mathbf{U}_1\mathbf{U}_2) = \overline{\mathbf{U}_2}^{\mathsf{T}}(\overline{\mathbf{U}_1}^{\mathsf{T}}\mathbf{U}_1)\mathbf{U}_2) = \overline{\mathbf{U}_2}^{\mathsf{T}}\mathbf{U}_2 = \mathbf{I}$, we see that $\mathbf{U}_1\mathbf{U}_2 \in \mathcal{N}$, i.e, $\mathcal{N}$ is closed with respect to matrix multiplication.

G2) As multiplication is associative, we have $\mathbf{U}_1(\mathbf{U}_2\mathbf{U}_3) = (\mathbf{U}_1\mathbf{U}_2)\mathbf{U}_3$ for $\mathbf{U}_1, \mathbf{U}_2, \mathbf{U}_3 \in \mathcal{N}$.

G3) For all $\mathbf{U} \in \mathcal{N}$, $\mathbf{I}\mathbf{U} = \mathbf{U}$ holds true. As $\mathbf{I} \in \mathcal{N}$, it is the neutral element of $\mathcal{N}$.

G4) The inverse element $\mathbf{U}^{-1} \in \mathcal{N}$ is given by $\mathbf{U}^{-1} = \overline{\mathbf{U}}^{\mathsf{T}}$.

■

Definition 2.2.14. *The group* $(\mathcal{N}, \cdot)$ *is called special unitary group* $\mathrm{SU}(2)$.

In correspondence to the rotation group $\mathrm{SO}(3)$, the special unitary group $\mathrm{SU}(2)$ is also called the complex rotation group. It also shares the properties of length, angle and orientation preservation (cf. Lemma 2.1.5) and that the eigenvalues of the matrix elements have absolute value one. Like elements of $\mathrm{SO}(3)$, elements of $\mathrm{SU}(2)$ offer three real degrees of freedom, too. Due to their unitarity they can be written as

$$\mathbf{U} = \begin{pmatrix} a & b \\ -\overline{b} & \overline{a} \end{pmatrix}$$

with $a, b \in \mathbb{C}$ and $a\overline{a} + b\overline{b} = 1$ (see e.g. [64, Chap. 5-16]). Of the two components $a, b \in \mathbb{C}$, we may choose real and imaginary part. But the constraint $\det(\mathbf{U}) = a\bar{a} + b\bar{b} = 1$ reduces these four choices to three.
We, so far, dealt with arbitrary vectors $\mathbf{u} \in \mathbb{R}^3$ which were to be rotated by applying an element of $\mathrm{SO}(3)$ to them. To rotate three-dimensional objects by using elements of $\mathrm{SU}(2)$, we need to modify their representation.

Lemma 2.2.15. *Let* $\mathbf{x} = (x_1, x_2, x_3)^{\mathsf{T}} \in \mathbb{R}^3$ *be given in Cartesian coordinates and let*

$$\mathbf{X} = \begin{pmatrix} -x_3 & x_1 + ix_2 \\ x_1 - ix_2 & x_3 \end{pmatrix}$$

be a corresponding matrix. Computing $\mathbf{Y} = \mathbf{U}\mathbf{X}\overline{\mathbf{U}}^{\mathsf{T}}$ *for an arbitrary* $\mathbf{U} \in \mathrm{SU}(2)$ *with* $\mathbf{U} = \begin{pmatrix} a & b \\ -\bar{b} & \bar{a} \end{pmatrix}$ *yields* $\mathbf{Y} = \begin{pmatrix} -y_3 & y_1 + iy_2 \\ y_1 - iy_2 & y_3 \end{pmatrix}$ *with* $(y_1, y_2, y_3)^{\mathsf{T}} = \mathbf{R}\mathbf{x}$ *for*

$$\mathbf{R} = \mathbf{R}(a,b) = \frac{1}{2}\begin{pmatrix} a^2 - b^2 + \bar{a}^2 - \bar{b}^2 & i(a^2 + b^2 - \bar{a}^2 - \bar{b}^2) & 2(ab + \bar{a}\bar{b}) \\ -i(a^2 - b^2 - \bar{a}^2 + \bar{b}^2) & a^2 + b^2 + \bar{a}^2 + \bar{b}^2 & 2i(\bar{a}\bar{b} - ab) \\ -2(\bar{a}b + a\bar{b}) & 2i(\bar{a}b - a\bar{b}) & 2(a\bar{a} - b\bar{b}) \end{pmatrix} \in \mathrm{SO}(3).$$

Proof. The matrix $\mathbf{Y} = \mathbf{U}\mathbf{X}\overline{\mathbf{U}}^{\mathsf{T}}$ takes the form

$$\mathbf{Y} = \begin{pmatrix} \overline{a}b(x_1 - ix_2) + a\overline{b}(x_1 + ix_2) + (b\overline{b} - a\overline{a})x_3 & -b^2(x_1 - ix_2) + a^2(x_1 + ix_2) + 2abx_3 \\ \overline{a}^2(x_1 - ix_2) - \overline{b}^2(x_1 + ix_2) + 2\overline{a}\overline{b}x_3 & -\overline{a}b(x_1 - ix_2) - a\overline{b}(x_1 + ix_2) + (a\overline{a} - b\overline{b})x_3 \end{pmatrix}.$$

We can now directly assign

$$\begin{aligned} y_1 &= \frac{1}{2}\left(x_1(a^2 - b^2 + \overline{a}^2 - \overline{b}^2) + x_2 i(a^2 + b^2 - \overline{a}^2 - \overline{b}^2) + x_3(2ab + 2\overline{a}\overline{b})\right) \\ y_2 &= \frac{1}{2}\left(x_1 i(-a^2 + b^2 + \overline{a}^2 - \overline{b}^2) + x_2(a^2 + b^2 + \overline{a}^2 + \overline{b}^2) + x_3 i(2\overline{a}\overline{b} - 2ab)\right) \\ y_3 &= -(\overline{a}b + a\overline{b})x_1 + i(\overline{a}b - a\overline{b})x_2 + (a\overline{a} - b\overline{b})x_3 \end{aligned}$$

which proves that $(y_1, y_2, y_3)^{\mathsf{T}} = \mathbf{R}\mathbf{x}$ with $\mathbf{R}$ given as in the lemma. One can also see by straightforward computations that this matrix is indeed a real-valued matrix. Exemplarily, we show that $ab + \overline{a}\overline{b} \in \mathbb{R}$. We have

$$\mathrm{Im}(ab + \overline{a}\overline{b}) = \mathrm{Im}(ab + \overline{a}\overline{b} + 1) = \mathrm{Im}(ab + \overline{a}\overline{b} + a\overline{a} + b\overline{b}) = \mathrm{Im}((\overline{a} + b)(a + \overline{b}) = 0.$$

We can also see that $\mathbf{R}$ is orthogonal with determinant one. ∎

Obviously we could also derive rotation axis and angle as well as Euler angles of a unitary rotation matrix $\mathbf{U} \in \mathrm{SU}(2)$.
Conversely, we can get the corresponding unitary matrix $\mathbf{U}$ from any of the previously mentioned parameterisations of a rotation. Exemplarily, we state the relation between a rotation given in Euler angles and $\mathbf{U} \in \mathrm{SU}(2)$ as it is the most convenient. There are two unitary matrices $\mathbf{U}$ performing the same rotation as a given rotation $\mathbf{R}(\alpha, \beta, \gamma)$. They are

$$\mathbf{U}(\alpha, \beta, \gamma) = \pm \begin{pmatrix} e^{\frac{i(\alpha+\gamma)}{2}} \cos\frac{\beta}{2} & e^{-\frac{i(\alpha+\gamma)}{2}} \sin\frac{\beta}{2} \\ -e^{\frac{i(\alpha-\gamma)}{2}} \sin\frac{\beta}{2} & e^{\frac{i(\alpha-\gamma)}{2}} \cos\frac{\beta}{2} \end{pmatrix}. \tag{2.5}$$

A more detailed description and proof of this representation can be found e.g. in [82].
Finally, we consider a category of parameterisations that will be very useful in the next Section 3 as it gives a connection between the rotation groups $\mathrm{SO}(3), \mathrm{SU}(2)$ and certain geometric objects.

Parameterisation of SO(3) Related to Geometric Objects Consider the set

$$\mathcal{B} = \{\mathbf{x} \in \mathbb{R}^3 \mid \mathbf{x} = \omega\mathbf{r}, \text{ for } \omega \in [0, \pi], \mathbf{r} \in \mathbb{S}^2\}.$$

As $\|\mathbf{r}\| = 1$, we see that all $\mathbf{x} \in \mathcal{B}$ satisfy $\|\mathbf{x}\| = \omega \leqslant \pi$. We can identify ω and $\mathbf{r}$ with the rotation angle and rotation axis of an element $\mathbf{R}_{\mathbf{r}}(\omega) \in \mathrm{SO}(3)$. By doing so, we can uniquely identify rotations

by elements from $\mathcal{B}$. As for two points $\mathbf{x}, \mathbf{x}' \in \mathcal{B}$ and two rotations $\mathbf{R}_{\mathbf{r}}(\omega), \mathbf{R}_{\mathbf{r}'}(\omega') \in \mathrm{SO}(3)$, we have $\mathbf{x} = \mathbf{x}' \Leftrightarrow \mathbf{R}_{\mathbf{r}}(\omega) = \mathbf{R}_{\mathbf{r}'}(\omega')$, cf. Lemma 2.2.8.
At this point, we introduce the three-dimensional unit sphere $\mathbb{S}^3 = \{\mathbf{x} \in \mathbb{R}^4 \mid \|\mathbf{x}\| = 1\}$ embedded in $\mathbb{R}^4$. Points $\mathbf{x} \in \mathbb{S}^3$ are given in spherical coordinates by

$$\mathbf{x} = \begin{pmatrix} \sin\omega \sin\theta \cos\varphi \\ \sin\omega \sin\theta \sin\varphi \\ \sin\omega \cos\theta \\ \cos\omega \end{pmatrix} \tag{2.6}$$

with $\theta, \omega \in [0, \pi]$ and $\varphi \in [0, 2\pi)$. Note that for any fixed value $\omega \in (0, \pi)$, this parameterises a two-dimensional sphere of radius $|\sin\omega|$. If we restrict ω to the interval $[0, \frac{\pi}{2}]$, this mapping of $\mathbf{x} \in \mathbb{S}^3$ to a sphere with radius $|\sin\omega|$ will be unique. Otherwise ω and $\pi - \omega$ will yield the same two-dimensional sphere. In correspondence to this, we define the upper hemisphere $\mathbb{S}^3_+ = \mathbb{S}^3 \cap \{\mathbf{x} \in \mathbb{R}^4 \mid \mathrm{x}_4 > 0\}$ of $\mathbb{S}^3$. It is the set of all points $\mathbf{x} \in \mathbb{R}^4$ satisfying

$$\mathbf{x} = \begin{pmatrix} \sin\frac{\omega}{2} \sin\theta \cos\varphi \\ \sin\frac{\omega}{2} \sin\theta \sin\varphi \\ \sin\frac{\omega}{2} \cos\theta \\ \cos\frac{\omega}{2} \end{pmatrix}. \tag{2.7}$$

Considering only $\mathbf{x} \in \mathbb{S}^3_+$, we have a bijective projection onto elements $\mathbf{x}' \in \mathcal{B}$, and hence, $\mathbf{x}$ uniquely determines a rotation, as well.
The conversion (2.7) allows us to see a connection between the metric on $\mathbb{S}^3_+$ and the metric on $\mathrm{SO}(3)$ from Lemma 2.2.9. If we compute the angle between two points $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{S}^3_+$ in spherical coordinates (2.7) and compare the result with the combination of two rotations in axis-angle parameterisation from (2.2), we see the equivalence of the two metrics.
Concerning the group of complex rotations, a point $\mathbf{x} \in \mathbb{S}^3$ uniquely determines a complex rotation $\mathbf{U} \in \mathrm{SU}(2)$. This can be seen by converting (2.5) into its axis-angle parameterisation, which yields (2.6).
We could consider even more parameterisations of rotations, like Rodrigues or Euler parameters, skew-symmetric 3×3 matrices, or quaternions, to name just a few. But as we will not use these parameterisations, we refer the reader to [17] for further information on this subject.

3 Harmonic Analysis on the Rotation Group

This chapter gives a short summary about harmonic analysis on the rotation group $\mathrm{SO}(3)$. In the first two sections we collect the ingredients that are needed to consider harmonic analysis on $\mathrm{SO}(3)$ amongst which we find the definition of integration over the group $\mathrm{SO}(3)$, in Section 3.1. To actually define the Fourier transform on $\mathrm{SO}(3)$ and its inverse in Section 3.2, we shall consider the space of square integrable functions on $\mathrm{SO}(3)$ and its orthogonal basis functions, the Wigner-D functions. The latter arise from group representations of $\mathrm{SO}(3)$, or to be more exact, from unitary, irreducible representations of $\mathrm{SO}(3)$. We will find them to be matrix-valued functions, the matrix elements of which constitute orthogonal basis functions on the rotation group, the so-called Wigner-D functions. The key theorem to these considerations, the Peter-Weyl theorem, will be given as well as the definition of Wigner-D functions resulting from it.
As the central aspect, the Fourier transform and its inverse will be defined. Moreover, we show how the discrete versions of both transforms can be defined by sampling, on one hand, and by using quadrature rules, on the other. We will then see that computing convolution and correlation follows the same lines on the rotation group as it does in the standard settings.
To conclude the section, we get back to the various parameterisations of the rotation group and exemplarily show in Section 3.3 for the Euler angles how explicit formulae for the Wigner-D functions can be found, i.e., we look at homogeneous polynomials and the Laplace operator on $\mathrm{SO}(3)$. The Wigner-D functions will arise as eigenfunctions of the latter.
Subsequently in Section 3.4, we will point out the relations between the $\mathrm{SO}(3)$ and the sphere $\mathbb{S}^2$ with their respective orthogonal basis functions, the Wigner-D functions and the spherical harmonics; as well as other connections between these two manifolds.
In the final Section 3.5, we consider an application of Fourier transforms on the rotation group, namely, the fast summation of functions, and especially, radial basis functions on $\mathrm{SO}(3)$.

3.1 Integration of Rotation-Dependent Functions

In the previous section, we saw that the rotation group $\mathrm{SO}(3)$ corresponds to certain geometric objects, e.g. elements of $\mathrm{SO}(3)$ can be identified with points $\mathbf{x} \in \mathbb{S}^3_+$ on the upper hemisphere of the three-sphere. The standard metric on $\mathbb{S}^3_+$ provides a topology also for the rotation group $\mathrm{SO}(3)$, and therefore, we can deal with $\mathrm{SO}(3)$ as a geometric object itself.
In fact, the rotation group is a locally, compact group which allows us to perform analysis on $\mathrm{SO}(3)$, cf. [87]. Note that the same argument applies to $\mathrm{SU}(2)$.

Definition 3.1.1. *Integration of a function* $\mathrm{f} : \mathrm{SO}(3) \to \mathbb{R}$ *defined on rotations* $\mathbf{R} \in \mathrm{SO}(3)$*, which are parameterised by rotation axis and rotation angle, reads as*

$$\int_{\mathrm{SO}(3)} \mathrm{f}(\mathbf{R})\,\mathrm{d}\mathbf{R} = \frac{1}{4\pi^2}\int_0^{2\pi}\int_0^{\pi}\int_0^{\pi} \mathrm{f}(\mathbf{R}_{\mathbf{r}(\varphi,\theta)}(\omega))(\cos\omega - 1)\sin\theta\;\mathrm{d}\omega\,\mathrm{d}\theta\,\mathrm{d}\varphi$$

with the normalised volume element $\mathrm{d}\mathbf{R}$ *on* $\mathrm{SO}(3)$ *defined in terms of the axis-angle parameterisation as* $\mathrm{d}\mathbf{R} = \frac{1}{4\pi^2}(\cos\omega - 1)\sin\theta\;\mathrm{d}\omega\,\mathrm{d}\theta\,\mathrm{d}\varphi$.

This definition originates from the parameterisation of the $\mathrm{SO}(3)$ by elements of $\mathbb{S}^3_+$. The surface area of $\mathbb{S}^3_+$ is

$$\int_{\mathbb{S}^3_+} \mathrm{d}\mathbf{x} = \int_0^{2\pi}\int_0^{\pi}\int_0^{\pi} \sqrt{\det(\mathbf{J}\,\mathbf{J}^{\mathsf{T}})}\,\mathrm{d}\omega\,\mathrm{d}\theta\,\mathrm{d}\varphi, \tag{3.1}$$

by the transformation formula, with the Jacobian $\mathbf{J} \in \mathbb{R}^{4\times 3}$ being the Jacobian of the coordinate transform (2.7),

$$\mathbf{J} = \frac{\partial \mathbf{x}}{\partial(\omega,\theta,\varphi)} = \begin{pmatrix} \frac{1}{2}\cos\frac{\omega}{2}\cos\varphi\sin\theta & \frac{1}{2}\cos\frac{\omega}{2}\sin\theta\sin\varphi & \frac{1}{2}\cos\frac{\omega}{2}\cos\theta & -\frac{1}{2}\sin\frac{\omega}{2} \\ \cos\theta\cos\varphi\sin\frac{\omega}{2} & \cos\theta\sin\frac{\omega}{2}\sin\varphi & -\sin\frac{\omega}{2}\sin\theta & 0 \\ -\sin\frac{\omega}{2}\sin\theta\sin\varphi & \cos\varphi\sin\frac{\omega}{2}\sin\theta & 0 & 0 \end{pmatrix}. \tag{3.2}$$

As $\det(\mathbf{J}\,\mathbf{J}^{\mathsf{T}}) = \frac{1}{4}\sin^4\frac{\omega}{2}\sin^2\theta$, we get

$$\int_{\mathbb{S}^3_+} \mathrm{d}\mathbf{x} = \frac{1}{4}\int_0^{2\pi}\int_0^{\pi}\int_0^{\pi} (\cos\omega - 1)\sin\theta\;\mathrm{d}\omega\,\mathrm{d}\theta\,\mathrm{d}\varphi = \pi^2,$$

the surface area of one hemisphere of the $\mathbb{S}^3$. Hence, integration of $\mathrm{f} = 1$ gives

$$\int_{\mathrm{SO}(3)} \mathrm{d}\mathbf{R} = \frac{1}{4\pi^2}\int_0^{2\pi}\int_0^{\pi}\int_0^{\pi} (\cos\omega - 1)\sin\theta\,\mathrm{d}\omega\,\mathrm{d}\theta\,\mathrm{d}\varphi = \frac{1}{\pi^2}\int_{\mathbb{S}^3_+} \mathrm{d}\mathbf{x} = 1.$$

Remark 3.1.2. *The volume element* $\mathrm{d}\mathbf{R}$ *from Definition 3.1.1 gives the Haar measure* μ *of* $\mathrm{SO}(3)$ *by* $\mathrm{d}\mathbf{R} = \mathrm{d}\mu(\mathbf{R})$ *. For a more extended overview on Haar measures, see e.g. [41] or [44].*

Corollary 3.1.3. *Integration of a function* $\mathrm{f} : \mathrm{SO}(3) \to \mathbb{R}$ *depending on rotations* $\mathbf{R} \in \mathrm{SO}(3)$ *parameterised in Euler angles reads as*

$$\int_{\mathrm{SO}(3)} \mathrm{f}(\mathbf{R})\,\mathrm{d}\mathbf{R} = \frac{1}{8\pi^2}\int_0^{2\pi}\int_0^{\pi}\int_0^{2\pi} \mathrm{f}(\mathbf{R}(\alpha,\beta,\gamma))\sin\beta\,\mathrm{d}\alpha\,\mathrm{d}\beta\,\mathrm{d}\gamma.$$

Proof. To determine the volume element of $\mathrm{SO}(3)$ in terms of its Euler angle representation, we again use the coordinate transformation formula (3.1) and conclude

$$\int_{\mathbb{S}^3} \mathrm{d}\mathbf{x} = \int_0^{2\pi}\int_0^{\pi}\int_0^{2\pi} \sqrt{\det(\mathbf{J}\,\mathbf{J}^{\mathsf{T}})}\,\mathrm{d}\alpha\,\mathrm{d}\beta\,\mathrm{d}\gamma$$

with the Jacobian $\mathbf{J} = \dfrac{\partial \mathbf{x}}{\partial(\alpha,\beta,\gamma)}$ obtained by inserting the conversion formulae (2.4) into the Jacobian (3.2). As $\det(\mathbf{J}\mathbf{J}^{\mathsf{T}}) = \dfrac{1}{64}\sin^2\beta$ holds true, we get $\mathrm{d}\mathbf{R} = \dfrac{1}{8}\sin\beta\,\mathrm{d}\alpha\,\mathrm{d}\beta\,\mathrm{d}\gamma$ by Definition 3.1.1. This proves the corollary. ∎

In an analogous way, we can also define the volume element on $\mathrm{SU}(2)$.

Definition 3.1.4. *The normalised volume element* $\mathrm{d}\mathbf{U}$ *on* $\mathrm{SU}(2)$ *is defined in terms of the Euler angle parameterisation as* $\mathrm{d}\mathbf{U} = \dfrac{1}{16\pi^2}\sin\beta\;\mathrm{d}\alpha\,\mathrm{d}\beta\,\mathrm{d}\gamma$.

Again this definition is consistent with the geometric interpretation of the $\mathrm{SU}(2)$ as points on $\mathbb{S}^3$. Its surface area is double the area of the upper hemisphere used in the $\mathrm{SO}(3)$ case. The Euler angles for complex rotations are from the interval $[0, 2\pi) \times [0, \pi) \times [-2\pi, 2\pi)$, (cf. [17, p.294]). Hence, by Corollary 3.1.3, we obtain

$$\int_{\mathrm{SU}(2)} f(\mathbf{U})\, d\mathbf{U} = \frac{1}{16\pi^2} \int_{-2\pi}^{2\pi} \int_0^{\pi} \int_0^{2\pi} f(\mathbf{U}(\alpha, \beta, \gamma)) \sin\beta \, d\alpha \, d\beta \, d\gamma.$$

If we are able to integrate functions on $\mathrm{SO}(3)$ and $\mathrm{SU}(2)$, it may be natural to ask whether we might also convolute functions on these groups. Indeed, this can be done if we restrict ourselves to functions $f \in L^2(\mathrm{SO}(3))$ which is defined completely analogous to the standard consisting of equivalence classes of functions $f : \mathrm{SO}(3) \to \mathbb{C}$ satisfying $\int_{\mathrm{SO}(3)} |f(\mathbf{R})|^2 \, d\mathbf{R} < \infty$ and equipped with an inner product of two functions $f, g \in L^2(\mathrm{SO}(3))$ given by

$$\langle f, g \rangle \;=\; \int_{\mathrm{SO}(3)} f(\mathbf{R}) \overline{g(\mathbf{R})} \, d\mathbf{R}. \tag{3.3}$$

The convolution of two such functions $f, g \in L^2(\mathrm{SO}(3))$ is written as

$$(f * g)(\mathbf{Q}) = \int_{\mathrm{SO}(3)} f(\mathbf{R}) g(\mathbf{R}^{\mathrm{T}}\mathbf{Q}) \, d\mathbf{R}. \tag{3.4}$$

Completely analogously, we define $L^2(\mathrm{SU}(2))$, with an inner product

$$\langle h, k \rangle \;=\; \int_{\mathrm{SU}(2)} h(\mathbf{U}) \overline{k(\mathbf{U})} \, d\mathbf{U} \tag{3.5}$$

for two functions $h, k \in L^2(\mathrm{SU}(2))$. The convolution of these two functions reads as

$$(h * k)(\mathbf{V}) = \int_{\mathrm{SU}(2)} h(\mathbf{U}) k(\mathbf{U}^{\mathrm{T}}\mathbf{V}) \, d\mathbf{U}. \tag{3.6}$$

3.2 Fourier Transforms and Convolution on SO(3)

In the following, we provide an analogue of the well-known Fourier transform and the Fourier series for the rotation group $\mathrm{SO}(3)$. Most of the results presented in this section can be applied, not only to the rotation group, but also in a more general context, to locally compact groups. The reader interested in the general theory is referred to the books [44] or [87]. We will however provide a few basic definitions.

Group Representations We start by giving a brief review on representation theory on groups. Most of these results occur in the context of representation theory of finite groups. However, they can be directly transferred to infinite groups, on the condition that a Haar measure can be defined on the group. This is the case for the rotation group $\mathrm{SO}(3)$, cf. Remark 3.1.2 and Definition 3.1.1, and for the complex rotation group $\mathrm{SU}(2)$, see Definition 3.1.4. Actually this is the case for all locally compact groups, as e.g. [44] lay out in great detail. We begin with the central concept in harmonic analysis on groups: representations.

Definition 3.2.1 (Representations)**.** *Let* $N \in \mathbb{N}$ *and* V *be an* N*-dimensional vector space, and let* $\{\mathbf{v}_i \in V \mid i = 1, \dots, N\}$ *be some basis in* V*. Moreover, let* G *be a locally compact matrix group, the elements of which act on vectors* $\mathbf{v} \in V$ *by the operation of composition, and let* $GL(V)$ *denote the set of all linear transformations over* V*. Then the group homomorphism* $D : G \to GL(V)$ *is called a representation of* G *on* V*.*

Corollary 3.2.2. *Consider the function* $f : V \to V$ *and* $D : G \to GL(V)$ *with* $(D(\mathbf{G})f)(\mathbf{v}) = f(\mathbf{G}^{-1}\mathbf{v})$ *where* $\mathbf{G} \in G$ *and* $\mathbf{v} \in V$*. Then* D *is a representation of* G *on* V*.*

Proof. For $\mathbf{G}_1, \mathbf{G}_2 \in G$, we have

$$D(\mathbf{G}_1\mathbf{G}_2)f(\mathbf{v}) = f((\mathbf{G}_1\mathbf{G}_2)^{-1}\mathbf{v}) = f(\mathbf{G}_2^{-1}\mathbf{G}_1^{-1}\mathbf{v}) = (D(\mathbf{G}_2)D(\mathbf{G}_1)f)(\mathbf{v}).$$

Therefore $D : G \to GL(V)$ is indeed a homomorphism as specified in Definition 3.2.1. ∎

Let V and W be n-dimensional vector spaces. Two representations $D_V : G \to GL(V)$ and $D_W : G \to GL(W)$ are denoted as *equivalent*, or $D_V \cong D_W$, if there is an isomorphism of the vector spaces $A : V \to W$ such that we obtain $A(D_V(\mathbf{G})\mathbf{v}) = D_W(\mathbf{G})A(\mathbf{v})$, or equivalently

$$D_V(\mathbf{G})\mathbf{v} = A^{-1}(D_W(\mathbf{G})A(\mathbf{v})),$$

for any $\mathbf{G} \in G$ and $\mathbf{v} \in V$.
This can be phrased differently. The representations $D_V(\mathbf{G})$ and $D_W(\mathbf{G})$ as well as the isomorphism A can be identified with $N \times N$ matrices and the above equation becomes $D_V(\mathbf{G}) = A^{-1}D_W(\mathbf{G})A$. That means, the matrices of two equivalent representations are similar. Two equivalent representations D_V and D_W give the same linear transformation but for different basis functions.

A subspace $V_1 \subseteq V$ satisfying $D(\mathbf{G})V_1 \subseteq V_1$ for all $\mathbf{G} \in G$ and a representation $D : G \to GL(V)$ is called G-*invariant*. If the only G-invariant subspaces of V are $\{\mathbf{0}\}$ and V itself, the representation $D(\mathbf{G})$ is called *irreducible*. On the other hand, a representation $D(\mathbf{G})$ is called *reducible* if there is a nontrivial G-invariant subspace V_1 of V.
A reducible representation $D(\mathbf{G})$ satisfies

$$D(\mathbf{G}) \cong \begin{pmatrix} D_1(\mathbf{G}) & \mathbf{0} \\ \mathbf{0} & D_2(\mathbf{G}) \end{pmatrix}$$

for two representations $D_1(\mathbf{G})$ and $D_2(\mathbf{G})$, cf. [91, p.85]. This has two main consequences. On one hand, any finite-dimensional reducible representation matrix can be decomposed by similarity transforms until it is no longer block-diagonal, rendering the resulting representations irreducible.
On the other hand, if we can decompose V into a direct sum of G-invariant subspaces $V = V_1 \oplus \dots \oplus V_n$ with $n \in \mathbb{N}$, then the representation $D(\mathbf{G})$ fulfils

$$D(\mathbf{G}) \cong \begin{pmatrix} D_1(\mathbf{G}) & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & D_i(\mathbf{G}) \end{pmatrix}.$$

Whenever the representations $D_i(\mathbf{G})$ for $i = 1, \dots, n$ are irreducible, $D(\mathbf{G})$ will be called completely reducible.

We state the following important lemma without proof (for a proof, see e.g. [44, pp.7-8]). It tells us, among others, that finite-dimensional irreducible representations result in invertible matrices.

Lemma 3.2.3 (Schur's Lemma)**.** *Let* $D_V : G \to GL(V)$ *and* $D_W : G \to GL(W)$ *be two irreducible representations of a group* G*. Suppose there is a linear transformation* $A : V \to W$ *such that* $(D_W(\mathbf{G})A)(\mathbf{v}) = A(D_V(\mathbf{G})\mathbf{v})$ *for any* $\mathbf{G} \in G$ *and* $\mathbf{v} \in V$*. Then* A *is either zero or an isomorphism. If* $V = W$ *then* $A = \lambda\mathbf{I}$ *for* $\lambda \in \mathbb{C}$*.*

The representation $D : G \to GL(V)$ is called *unitary* if there is a G-invariant positive definite Hermitian form $\langle \cdot\ ,\ \cdot \rangle$ on V, i.e., $\langle \mathbf{v}, \mathbf{w} \rangle = \langle D(\mathbf{G})\mathbf{v}, D(\mathbf{G})\mathbf{w} \rangle = \langle \mathbf{G}\mathbf{v}, \mathbf{G}\mathbf{w} \rangle$, $\forall \mathbf{G} \in G$, $\forall \mathbf{v}, \mathbf{w} \in V$. A group equipped with a Haar measure possesses such a Hermitian form as

$$\langle \mathbf{v}, \mathbf{w} \rangle = \int_G (D(\mathbf{G})\mathbf{v}, D(\mathbf{G})\mathbf{w})\, d\mu(\mathbf{G})$$

fulfils the required properties for any inner product $(\cdot\ ,\ \cdot)$ on V.
One now shows that every unitary representation D of the compact group G on a finite-dimensional vector space V is completely reducible. This can be seen by considering a nontrivial G-invariant subspace V_1 of V and its orthogonal complement $V_1^\perp$. Using the above Hermitian form we can conclude by

$$\langle D(\mathbf{G})\mathbf{v}^\perp, \mathbf{v} \rangle = \langle \mathbf{v}^\perp, D(\mathbf{G})^{-1}\mathbf{v} \rangle = 0 \qquad \mathbf{v} \in V_1, \mathbf{v}^\perp \in V_1^\perp$$

that $V_1^\perp$ is G-invariant as well. Hence, we decomposed V into two G-invariant subspaces. Repeated application of the same arguments will result in the sought decomposition.
Given a unitary representation $D(\mathbf{G})$, a complex-valued function $D_{1,2}(\mathbf{G}) = \langle D(\mathbf{G})\mathbf{v}_1, \mathbf{v}_2 \rangle$ for $\mathbf{v}_1, \mathbf{v}_2 \in V$ and $\mathbf{G} \in G$ is called representative function on the group G. We state the following lemma about representative functions. Note that the representative functions are actually the matrix elements of a represention.

Lemma 3.2.4. *Let* $\{\mathbf{v}_i \mid i = 1, \ldots, N\}$ *be an orthogonal basis of the* N*-dimensional vector space* V *with respect to* $\langle \cdot, \cdot \rangle$*. Then the representative functions* $D_{ij}(\mathbf{G}) = \langle D(\mathbf{G})\mathbf{v}_i, \mathbf{v}_j \rangle, i, j = 1, ..., N, \mathbf{G} \in G$ *of the group* G *with respect to the irreducible representation* $D(\mathbf{G})$ *form a set of orthogonal functions.*

Proof. On a locally compact group G, we can define an inner product by

$$\langle f, h \rangle = \int_G f(\mathbf{G})\overline{h(\mathbf{G})}\, d\mathbf{G}$$

with respect to the integration over the group, cf. Definition 3.1.1 and (3.3), for the $SO(3)$ case as well as Remark 3.1.2.
Let $A : V \to V$ be a linear transformation. By Corollary 3.2.2, we have $(D(\mathbf{G})A)(\mathbf{v}) = A(\mathbf{G}^{-1}\mathbf{v})$ and thus $(D(\mathbf{G}^{-1})A)(\mathbf{G}^{-1}\mathbf{v}) = A(\mathbf{v})$. By Lemma 3.2.3, the linear transformation A is described by a multiple of the identity matrix $A = \lambda\mathbf{I}$. We obtain $(D(\mathbf{G}^{-1})A)(\mathbf{G}^{-1}\mathbf{v}) = \dfrac{\operatorname{tr}(A)}{N}\mathbf{v}$.
Taking the inner product with respect to $\mathbf{v}_k$ on both sides of the equation and replacing $\mathbf{v}$ with $\mathbf{v}_\ell$, we get

$$\int_G \langle D(\mathbf{G})A(\mathbf{G}^{-1}\mathbf{v}_\ell), \mathbf{v}_j \rangle\, d\mathbf{G} = \frac{\operatorname{tr}(A)\langle \mathbf{v}_\ell, \mathbf{v}_j \rangle}{N}. \tag{3.7}$$

Now let $A(\mathbf{v}) = \langle \mathbf{v}, \mathbf{v}_k \rangle \mathbf{v}_i$. The trace of the corresponding matrix satisfies $\operatorname{tr}(A) = \langle \mathbf{v}_i, \mathbf{v}_k \rangle$.
We can use this in the integral (3.7) to obtain

$$\begin{aligned} \frac{\langle \mathbf{v}_i, \mathbf{v}_k \rangle \langle \mathbf{v}_\ell, \mathbf{v}_j \rangle}{N} &= \int_G \langle \langle D^{-1}(\mathbf{G})\mathbf{v}_\ell, \mathbf{v}_k \rangle D(\mathbf{G})\mathbf{v}_i, \mathbf{v}_j \rangle\, d\mathbf{G} \\ &= \int_G \langle D(\mathbf{G})\mathbf{v}_i, \mathbf{v}_j \rangle \langle D^{-1}(\mathbf{G})\mathbf{v}_\ell, \mathbf{v}_k \rangle\, d\mathbf{G}. \end{aligned}$$

As $D(\mathbf{G})$ is unitary, we have

$$\frac{\langle \mathbf{v}_i, \mathbf{v}_k\rangle\langle \mathbf{v}_\ell, \mathbf{v}_j\rangle}{N} = \int_G \langle D(\mathbf{G})\mathbf{v}_i, \mathbf{v}_j\rangle\overline{\langle D(\mathbf{G})\mathbf{v}_k, \mathbf{v}_\ell\rangle}\, d\mathbf{G} = \langle D_{i,j}, D_{k,\ell}\rangle.$$

As $\mathbf{v}_k$ for $k = 1, \ldots, N$ are orthogonal basis vectors of V, we get

$$\langle \mathbf{v}_i, \mathbf{v}_k\rangle\langle \mathbf{v}_\ell, \mathbf{v}_j\rangle = \begin{cases} \langle \mathbf{v}_i, \mathbf{v}_i\rangle\langle \mathbf{v}_j, \mathbf{v}_j\rangle & \text{for } i = k \text{ and } j = \ell \\ 0 & \text{otherwise.} \end{cases}$$

This proves the orthogonality of the representative functions $D_{i,j}$. ■

From the previous lemma it is clear that the representative functions depend not only on the representation itself but also on the choice of the basis in the space V. It would be, therefore, convenient to study functions of representations that remain invariant under a change of basis. Such a function will be called the character of a representation.
The character $\chi : G \to \mathbb{C}$ of a representation D of the group G on an N-dimensional vector space is defined by

$$\chi_D(\mathbf{G}) = \sum_{i=1}^{N} D_{ii}(\mathbf{G}), \qquad \mathbf{G} \in G. \tag{3.8}$$

It is conjugate invariant, i.e. $\chi_D(\mathbf{G}\mathbf{G}'\mathbf{G}^{-1}) = \chi_D(\mathbf{G}')$. If D and D' are irreducible, then we have

$$\langle \chi_D, \chi_{D'}\rangle = \begin{cases} 0 & \text{if } D \ncong D', \\ 1 & \text{if } D \cong D'. \end{cases} \tag{3.9}$$

We will examine characters of representations of $SO(3)$ more closely in Section 3.5.

Now we have all ingredients to formulate the following theorem, which provides an analogue to Fourier's theorem for a locally compact group as it specifies the orthogonality relations of representative functions on a locally compact group G. For a proof and extended information of the theorem see [90, pp. 157 ff.].

Theorem 3.2.5 (Peter-Weyl Theorem)**.** *The representative functions $D_{i,j}$ of all representations D of the locally compact group G form a complete, orthogonal system in $L^2(G)$.*

The following Lemma states that we do not need all representations of G to construct an orthogonal basis.

Lemma 3.2.6. *Let $\Lambda = \{D_\ell\}$ denote the set of all inequivalent, finite-dimensional, unitary, irreducible representations of a compact group G and let D_ℓ^{mn} be the representative functions on G with respect to each representation D_ℓ. Then the set of functions*

$$\{D_\ell^{mn}(\mathbf{G}) \mid D_\ell \in \Lambda;\ m, n = 1, \ldots\}$$

is an orthogonal basis of $L^2(G)$.

Proof. This follows by putting together Lemma 3.2.4 and the Peter-Weyl theorem. ■

An Orthogonal Basis of $L^2(SO(3))$ Now we apply these general results to the rotation group $SO(3)$ to characterise an orthogonal basis for it. The rotation group $SO(3)$ is a locally compact group which fits into the setting from the previous subsection. Recall that $SO(3)$ possesses an integration invariant Haar measure, cf. Section 3.1.
At first, we consider the 2-sphere $\mathbb{S}^2$ as it is a suitable vector space on which the elements of $SO(3)$ act transitively, i.e., for all $\xi_1, \xi_2 \in \mathbb{S}^2$, we can find an element $\mathbf{R} \in SO(3)$ such that $\xi_2 = \mathbf{R}\xi_1$. Let $\xi \in \mathbb{S}^2$ and let $(\varphi, \theta) \in [0, 2\pi) \times [0, \pi]$ be its coordinates. For any $\ell \in \mathbb{N}_0$ and $m = -\ell, \dots \ell$ the spherical harmonics of degree ℓ are defined as

$$Y_\ell^m(\xi) = \sqrt{\frac{2\ell+1}{4\pi}}\sqrt{\frac{(\ell-m)!}{(\ell+m)!}}P_\ell^m(\cos\theta)e^{im\phi}$$

where $P_\ell^m : [-1,1] \to \mathbb{R}$ are *associated Legendre polynomials*, cf. [81], that arise as the derivatives of ordinary Legendre polynomials $P_\ell(x)$ and are given by

$$P_\ell^m(x) = (-1)^m(1-x^2)^{\frac{m}{2}}\frac{d^m}{dx^m}P_\ell(x) = \frac{(-1)^m}{2^\ell \ell!}(1-x^2)^{\frac{m}{2}}\frac{d^{\ell+m}}{dx^{\ell+m}}(x^2-1)^\ell. \tag{3.10}$$

Moreover, the spherical harmonics satisfy the orthogonality relation

$$\int_{\mathbb{S}^2} Y_\ell^m(\xi)Y_{\ell'}^{m'}(\xi)\,d\xi = \delta_{\ell\ell'}\delta_{mm'}. \tag{3.11}$$

The subspace $\mathrm{Harm}_\ell(\mathbb{S}^2) = \mathrm{span}\{Y_\ell^m \mid m = -\ell, \dots, \ell\}$ spanned by spherical harmonics with a fixed degree $\ell \in \mathbb{N}$ is called harmonic space of degree ℓ. The harmonic spaces $\mathrm{Harm}_\ell(\mathbb{S}^2)$ provide a complete system $SO(3)$-invariant subspaces of $L^2(\mathbb{S}^2)$, i.e.,

$$L^2(\mathbb{S}^2) = \mathrm{clos}_{L^2}\bigoplus_{\ell=0}^{\infty}\mathrm{Harm}_\ell(\mathbb{S}^2). \tag{3.12}$$

This has two main consequences. Firstly, as also (3.11) is fulfilled the set $\{Y_\ell^m \mid \ell \in \mathbb{N}_0, |m| \leqslant \ell\}$ forms an orthonormal basis of $L^2(\mathbb{S}^2)$.
And secondly, the representations $D(\mathbf{R})$ for $\mathbf{R} \in SO(3)$ acting on functions $f \in L^2(\mathbb{S}^2)$ are completely reducible. They decompose into irreducible representations $D_\ell(\mathbf{G})$ corresponding to the harmonic subspaces $\mathrm{Harm}_\ell(\mathbb{S}^2)$ in the sense that any function $f \in \mathrm{Harm}_\ell(\mathbb{S}^2)$ satisfies

$$D_\ell(\mathbf{G})f(\xi) = f(\mathbf{G}^{-1}\xi).$$

Having considered this, we take a closer look to the representative function resulting from the irreducible representations $D_\ell(\mathbf{R})$.

Definition 3.2.7 (Wigner-D functions)**.** *Let $D_\ell(\mathbf{R})$ for any $\ell \in \mathbb{N}$ and $\mathbf{R} \in SO(3)$ be a unitary, irreducible representation of $SO(3)$ on $\mathrm{Harm}_\ell(\mathbb{S}^2)$. Then the representative functions on $SO(3)$ given by $D_\ell^{mn}(\mathbf{R}) = \langle D_\ell(\mathbf{R})Y_\ell^m, Y_\ell^n\rangle$ for $m, n = -\ell, \dots, \ell$ are called Wigner-D functions of degree ℓ and orders m and n.*

Using Wigner-D functions of fixed degree $\ell \in \mathbb{N}$, we define the subspaces

$$\mathrm{Harm}_\ell(SO(3)) = \mathrm{span}\{D_\ell^{mn} : m, n = -\ell, \dots, \ell\}. \tag{3.13}$$

Lemma 3.2.8. *The set of Wigner-D functions*

$$\{D_\ell^{mn}(\mathbf{R}) \mid \ell \in \mathbb{N}_0\,,\ m,n = -\ell,\dots,\ell\}$$

forms an orthogonal basis of $L^2(SO(3))$.

Proof. Employing the set of spherical harmonics $\{Y_\ell^m \mid m = -\ell,\dots,\ell\}$ as an orthonormal basis of $\mathrm{Harm}_\ell(\mathbb{S}^2)$, the proof follows directly from Lemma 3.2.6 and (3.12). ∎

The Wigner-D functions D_ℓ^{mn} are not normalised with respect to the inner product (3.3), but they satisfy the orthogonality conditions

$$\int_{SO(3)} D_\ell^{mm'}(\mathbf{R})\overline{D_{\ell'}^{nn'}(\mathbf{R})}\ d\mathbf{R} = \frac{8\pi^2}{2\ell+1}\delta_{mn}\delta_{m'n'}\delta_{\ell\ell'}. \tag{3.14}$$

We can conclude from Lemma 3.2.8 and (3.12) that $L^2(SO(3))$ decomposes into the direct sum

$$L^2(SO(3)) = \mathrm{clos}_{L^2}\bigoplus_{\ell=0}^{\infty}\mathrm{Harm}_\ell(SO(3)).$$

Motivated by Lemma 3.2.8, we give the following Definition.

Definition 3.2.9 (SO(3) Fourier Expansion)**.** *The series expansion*

$$f(\mathbf{R}) = \sum_{\ell=0}^{\infty}\sum_{m=-\ell}^{\ell}\sum_{n=-\ell}^{\ell}\widehat{f}_\ell^{mn}D_\ell^{mn}(\mathbf{R}),$$

of a function $f \in L^2(SO(3))$ *in terms of the Wigner-D functions* D_ℓ^{mn}*, for any* $\mathbf{R} \in SO(3)$ *is called the* SO(3) Fourier expansion *with Fourier coefficients* $\hat{f}_\ell^{mn}$ *given by the inner product*

$$\widehat{f}_\ell^{mn} \quad = \quad \frac{2\ell+1}{8\pi^2}\langle f, D_\ell^{mn}\rangle. \tag{3.15}$$

The Fourier expansion of functions $f, g \in L^2(SO(3))$ allows convenient computation of their convolution.

Lemma 3.2.10. *Given the* SO(3) *Fourier expansions of* f *and* g*,*

$$f(\mathbf{R}) = \sum_{\ell=0}^{\infty}\sum_{m=-\ell}^{\ell}\sum_{n=-\ell}^{\ell}\hat{f}_\ell^{mn}D_\ell^{mn}(\mathbf{R}), \quad g(\mathbf{R}) = \sum_{\ell=0}^{\infty}\sum_{m=-\ell}^{\ell}\sum_{n=-\ell}^{\ell}\hat{g}_\ell^{mn}D_\ell^{mn}(\mathbf{R}),$$

we find the Fourier coefficients $\hat{h}_\ell^{mn}$ *of their convolution* $h(\mathbf{R}) = (f * g)(\mathbf{R})$ *to be*

$$\hat{h}_\ell^{mn} = \frac{2\ell+1}{8\pi^2}\sum_{k=-\ell}^{\ell}\hat{f}_\ell^{mk}\hat{g}_\ell^{kn}.$$

Proof. The Fourier coefficients $\hat{h}_\ell^{mn}$ of the convolution $(f * g)(\mathbf{R})$ are given by

$$\hat{h}_\ell^{mn} = \frac{2\ell+1}{8\pi^2}\langle h(\mathbf{R}), D_\ell^{mn}\rangle = \frac{2\ell+1}{8\pi^2}\int_{SO(3)} h(\mathbf{R})\overline{D_\ell^{mn}(\mathbf{R})}\,d\mathbf{R}.$$

Inserting the definition of convolution in SO(3) from (3.4), and rewriting g in terms of its Fourier expansion, we get

$$\begin{aligned}\frac{8\pi^2}{2\ell+1}\hat{h}_\ell^{mn} &= \int_{SO(3)}\int_{SO(3)} f(\mathbf{Q})\, g(\mathbf{Q}^T\mathbf{R})\,\overline{D_\ell^{mn}(\mathbf{R})}\,d\mathbf{Q}\,d\mathbf{R} \\ &= \int_{SO(3)}\int_{SO(3)} f(\mathbf{Q}) \sum_{\ell'=0}^{\infty}\sum_{m'=-\ell'}^{\ell'}\sum_{n'=-\ell'}^{\ell'} \hat{g}_{\ell'}^{m'n'} D_{\ell'}^{m'n'}(\mathbf{Q}^T\mathbf{R})\overline{D_\ell^{mn}(\mathbf{R})}\,d\mathbf{Q}\,d\mathbf{R}.\end{aligned}$$

By the addition theorem of Wigner-D functions,

$$D_\ell^{mn}(\mathbf{Q}^T\mathbf{R}) = \sum_{k=-\ell}^{\ell} D_\ell^{mk}(\mathbf{Q}^T)D_\ell^{kn}(\mathbf{R}), \quad |m|,|n| \leqslant \ell,$$

cf. [84], we obtain

$$\frac{8\pi^2}{2\ell+1}\hat{h}_\ell^{mn} = \int_{SO(3)}\int_{SO(3)} f(\mathbf{Q}) \sum_{\ell'=0}^{\infty}\sum_{m',n',k'=-\ell'}^{\ell'} \hat{g}_{\ell'}^{m'n'} D_{\ell'}^{m'k}(\mathbf{Q}^T)D_{\ell'}^{kn'}(\mathbf{R})\overline{D_\ell^{mn}(\mathbf{R})}\,d\mathbf{R}\,d\mathbf{Q}.$$

As the Wigner-D functions satisfy the orthogonality relation (3.14), the expression simplifies to

$$\hat{h}_\ell^{mn} = \int_{SO(3)} \sum_{m'=-\ell}^{\ell} f(\mathbf{Q})\hat{g}_\ell^{m'n} D_\ell^{m'm}(\mathbf{Q}^T)\,d\mathbf{Q} = \int_{SO(3)} \sum_{m'=-\ell}^{\ell} f(\mathbf{Q})\hat{g}_\ell^{m'n}\overline{D_\ell^{mm'}(\mathbf{Q})}\,d\mathbf{Q}.$$

Now, we rewrite f in terms of its Fourier expansion and use the orthogonality of Wigner-D functions once more,

$$\begin{aligned}\hat{h}_\ell^{mn} &= \int_{SO(3)} \sum_{m'=-\ell}^{\ell}\sum_{\ell''=0}^{\infty}\sum_{m'',n''=-\ell''}^{\ell''} \hat{f}_{\ell''}^{m''n''} D_{\ell''}^{m''n''}(\mathbf{Q})\hat{g}_\ell^{m'n}\overline{D_\ell^{mm'}(\mathbf{Q})}\,d\mathbf{Q} \\ &= \frac{2\ell+1}{8\pi^2}\sum_{m'=-\ell}^{\ell} \hat{f}_\ell^{mm'}\hat{g}_\ell^{m'n}\end{aligned}$$

which proves the lemma. ∎

Note that the Wigner-D functions also allow us to conveniently compute convolutions of functions on $L^2(\mathbb{S}^2)$. We will consider this in Section 3.4.

Discrete SO(3) Fourier Transforms In the preceding paragraphs, we saw how Fourier expansions are defined on the rotation group SO(3). It is, therefore, quite natural to ask whether we can also transfer the formulae for its well known variations, the Fourier partial sum and the discrete Fourier transform, to SO(3). Especially the latter will be needed in order to construct algorithms for fast

Fourier transforms on SO(3) in Section 4.

For $L \in \mathbb{N}$ consider functions $f \in L^2(SO(3))$ the Fourier coefficients of which fulfil $f_\ell^{mn} = 0$ for $\ell > L$. We call these functions L-*band-limited*.
Moreover, we define the function spaces

$$\mathbb{D}_L = \bigoplus_{\ell=0}^{L} \mathrm{Harm}_\ell(SO(3))$$

for arbitrary $L \in \mathbb{N}$ the elements of which are the above mentioned band-limited functions. An orthogonal basis of these spaces is given, due to (3.13), by

$$\{D_\ell^{mn}(\mathbf{R}) \mid \ell = 0, \dots, L\,;\ m, n = -\ell, \dots, \ell\}.$$

For convenience, we define an ordered set of indices

$$\mathcal{I}_L = \{(\ell, m, n) :\ \ell = 0, \dots, L\,;\ m, n = -\ell, \dots, \ell\} \tag{3.16}$$

corresponding to all sets of admissible indices (ℓ, m, n) within the space $\mathbb{D}_L$. Throughout this work, we keep the particular order of the indices fixed.
The dimension of the spaces $\mathbb{D}_L$ is given by

$$\dim(\mathbb{D}_L) = |\mathcal{I}_L| = \sum_{\ell=0}^{L} (2\ell+1)^2 = \frac{1}{3}(L+1)(2L+1)(2L+3).$$

Now, any function $f \in \mathbb{D}_L$ can be written as its unique finite *Fourier partial sum*

$$f(\mathbf{R}) = \sum_{(\ell,m,n)\in\mathcal{I}_L} \widehat{f}_\ell^{mn} D_\ell^{mn}(\mathbf{R}). \tag{3.17}$$

Evaluating the Fourier partial sum (3.17) not for all $\mathbf{R} \in SO(3)$, but only for a finite choice of rotations, i.e, a sampling set on SO(3) leads to the following definition.

Definition 3.2.11 (Discrete Fourier Transform on SO(3))**.** *Let $\mathcal{R}_Q = \{\mathbf{R}_j \in SO(3) \mid j = 1, \dots, Q\}$, be an arbitrary finite set of rotations. Then*

$$f(\mathbf{R}_q) = \sum_{(\ell,m,n)\in\mathcal{I}_L} \widehat{f}_\ell^{mn} D_\ell^{mn}(\mathbf{R}_q), \quad q = 1, \dots, Q,$$

with given Fourier coefficients $\hat{\mathbf{f}} = (\hat{f}_\ell^{mn})_{(\ell,m,n)\in\mathcal{I}_L}$ evaluates a function $f \in \mathbb{D}_L$ at the nodes $\mathbf{R}_1, \dots, \mathbf{R}_Q$. The corresponding operator $\mathbf{D}_{L,\mathcal{R}_Q} : \mathbb{C}^{|\mathcal{I}_L|} \to \mathbb{C}^Q$ with $f(\mathbf{R}_q) = \left[\mathbf{D}_{L,\mathcal{R}_Q}\hat{\mathbf{f}}\right]_q$ is called nonequispaced discrete SO(3) *Fourier transform* (NDSOFT).

As usual, the operator from Definition 3.2.11 will be identified with its corresponding matrix $\mathbf{D}_{L,\mathcal{R}_Q} \in \mathbb{C}^{|\mathcal{I}_L|\times Q}$. Hence, the NDSOFT can be thought of as the matrix vector multiplication $\mathbf{f} = \mathbf{D}_{L,\mathcal{R}_Q}\widehat{\mathbf{f}}$ with $\mathbf{f} = (f(\mathbf{R}_q))_{q=1,\dots,Q} \in \mathbb{C}^Q$ and $\hat{\mathbf{f}}$ as in Definition 3.2.11.
In general, the matrix $\mathbf{D}_{L,\mathcal{R}_Q}$ is not squared. Thus, the NDSOFT is not readily inverted. But we can immediately define the adjoint NDSOFT using the adjoint operator $\mathbf{D}_{L,\mathcal{R}_Q}^H$ by

$$\left[\mathbf{D}_{L,\mathcal{R}_Q}^H \mathbf{f}\right]_{(\ell,m,n)} = \tilde{f}_\ell^{mn} = \sum_{q=1}^{Q} f(\mathbf{R}_q)\, D_\ell^{mn}(\mathbf{R}_q), \tag{3.18}$$

for all admissible $(\ell, m, n) \in \mathcal{I}_L$. The coefficients $\tilde{f}_\ell^{mn}$ we obtain will generally not be equal to the SO(3) Fourier coefficients $\widehat{f}_\ell^{mn}$.
But since the Fourier coefficients $\hat{f}_\ell^{mn}$ are given by the inner product (3.15), i.e., by known integrals, they can be recovered from the function values $f(\mathbf{R}_q)$ if a quadrature rule with weights w_q, $q = 1, \ldots, Q$, and a sufficient order of exactness is available. In that case, we can compute the *inverse NDSOFT* by

$$\hat{f}_\ell^{mn} = \sum_{q=1}^{Q} w_q \, f(\mathbf{R}_q) \, D_\ell^{mn}(\mathbf{R}_q), \qquad (\ell, m, n) \in \mathcal{I}_L.$$

For a special set of nodes, we will consider the inverse NDSOFT in Section 4.2. But apart from that, we are not going into more detail about it. For more information on how one could analyse the SO(3) Fourier coefficients out of given samples by quadrature formulae and their corresponding sampling sets, the reader is referred to [36]. For the cases where we do not have a matching quadrature rule, e.g. if there are less function samples than Fourier coefficients, we refer to [35], where the authors describe how to use the adjoint NDSOFT for interpolation.

Complexity of the SO(3) Fourier Transform We conclude this section with a few remarks on the computational complexity of the NDSOFT, its adjoint and inverse. The application of the operators $\mathbf{D}_{L,\mathcal{R}_Q}$ and $\mathbf{D}_{L,\mathcal{R}_Q}^H$ takes $\mathcal{O}(QL^3)$ operations, owing to the size of the matrix $\mathbf{D}_{L,\mathcal{R}_Q}$. The discrete Fourier transform of a function $f \in \mathbb{D}_L$ needs $\mathcal{O}(L^3)$ Fourier coefficients as given input along with Q rotations on which the function will be evaluated. This adds up to $\mathcal{O}(Q + L^3)$ input values, which gives a lower bound for the computational complexity any algorithm computing the NDSOFT can have. One of the main aspects of this work is to find an algorithm for computing the NDSOFT that has a lower complexity as the direct $\mathcal{O}(QL^3)$ and which is as close as possible to the lower bound of $\mathcal{O}(Q + L^3)$. In Section 4.2, we will consider *fast* SO(3) *Fourier transforms* (NFSOFT) which are fast algorithms to compute the same result as the NDSOFT. Their complexity will not reach the lower bound. But we will approach it with two approximate algorithms, one of $\mathcal{O}(L^3 \log L + Q)$ and one of $\mathcal{O}(L^3 \log^2 L + Q)$ flops, in Sections 4.1.1 and 4.1.2, respectively. We shall also discuss the merits and drawbacks of both algorithms. But before, we need to discuss some important properties of the Wigner-D functions which are key to develop the NFSOFT algorithms.

3.3 Wigner-D and Wigner-d Functions

Definition 3.2.7 characterised the Wigner-D functions D_ℓ^{mn} as the matrix elements of the unitary irreducible representations of the group SO(3) on $\mathbb{S}^2$ (cf. also Lemma 3.2.6). Due to this, they satisfy the representation property

$$D_\ell^{mn}(\mathbf{RS}) = \sum_{k=-\ell}^{\ell} D_\ell^{mk}(\mathbf{R}) D_\ell^{kn}(\mathbf{S}), \qquad \mathbf{R}, \mathbf{S} \in SO(3) \tag{3.19}$$

as well as $\overline{D_\ell^{mn}(\mathbf{R})} = D_\ell^{mn}(\mathbf{R}^{-1})$ as a consequence of unitarity, cf. Lemma 3.2.6.

Recall from Section 2.1 that the elements of SO(3) have three degrees of freedom. It has proven to be a powerful idea to split up the Wigner–D functions according to these degrees of freedom. More precisely, in the following, we will use the Euler angle parameterisation to give explicit expressions

for Wigner-D functions. It is, of course, also possible to give the Wigner-D functions in any other parameterisation mentioned in Section 2.2 but the Euler angles are especially helpful as they will provide a direct connection between the spherical harmonics and the Wigner-D functions.
To get a parameterisation of Wigner-D functions in Euler angles, we consider the Laplace-Beltrami operator.

Definition 3.3.1 (Laplace-Beltrami operator)**.** *Let* $\mathbf{R}(q_1, q_2, q_3) \in \mathrm{SO}(3)$ *be given in a parameterisation of* $\mathrm{SO}(3)$*. The Laplace-Beltrami operator on* $\mathrm{SO}(3)$ *acting on a function* $f \in C^2(\mathrm{SO}(3))$ *in the given parameterisation is defined as*

$$\Delta_{\mathrm{SO}(3)} = \frac{1}{\sqrt{det(\mathbf{J}\mathbf{J}^\mathsf{T})}} \sum_{i,j=1}^{3} \frac{\partial f}{\partial q_i} \sqrt{det(\mathbf{J}\mathbf{J}^\mathsf{T})}\, (\mathbf{J}\mathbf{J}^\mathsf{T})^{-1}_{i,j} \frac{\partial f}{\partial q_j}$$

with the Jacobian $\mathbf{J} = \frac{\partial \mathbf{x}}{\partial(q_1,q_2,q_3)}$ *of the coordinate transform of* $\mathbf{x} \in \mathbb{S}^3_+$ *with respect to the given parameterisation.*

For the Jacobian of the coordinate transform in Euler angles $\mathbf{J} = \dfrac{\partial \mathbf{x}}{\partial(\alpha,\beta,\gamma)}$, we get

$$\mathbf{J}\mathbf{J}^\mathsf{T} = \begin{pmatrix} 1 & 0 & \cos\beta \\ 0 & 1 & 0 \\ \cos\beta & 0 & 1 \end{pmatrix} \text{ and } (\mathbf{J}\mathbf{J}^\mathsf{T})^{-1} = \frac{1}{\sin^2\beta} \begin{pmatrix} 1 & 0 & -\cos\beta \\ 0 & \sin^2\beta & 0 \\ -\cos\beta & 0 & 1 \end{pmatrix}.$$

Applying this to Definition 3.3.1 and using $\sqrt{\det(\mathbf{J}\mathbf{J}^\mathsf{T})} = \sin\beta$, we find

$$\Delta_{\mathrm{SO}(3)} = \frac{1}{\sin^2\beta}\left(\frac{\partial^2}{\partial\alpha^2} - 2\cos\beta\frac{\partial^2}{\partial\alpha\partial\gamma} + \frac{\partial^2}{\partial\gamma^2}\right) + \frac{\partial^2}{\partial\beta^2} + \cot\beta\frac{\partial}{\partial\beta}.$$

Note that the definition of the Laplace–Beltrami operator $\Delta_{\mathrm{SO}(3)}$ does not depend on a particular choice of coordinate system which implies rotational invariance of $\Delta_{\mathrm{SO}(3)}$, cf. [41, Sect. 2.4.2.]. Due to the rotational invariance, we can see from $\Delta_{\mathbb{S}^2} Y_\ell^m(\xi) = \ell(\ell+1)Y_\ell^m(\xi)$, (cf, [31, Sect. 3.5]) that $\Delta_{\mathrm{SO}(3)} Y_\ell^m(\mathbf{R}^{-1}\xi) = \ell(\ell+1)Y_\ell^m(\mathbf{R}^{-1}\xi)$ is satisfied. Employing Definition 3.2.7 of the Wigner-D functions, we conclude that the Wigner-D functions are eigenfunctions of the Laplace-Beltrami operator fulfilling

$$\Delta_{\mathrm{SO}(3)} D_\ell^{mn} = \ell(\ell+1) D_\ell^{mn}, \tag{3.20}$$

for $\ell \in \mathbb{N}$.
This allows us to find an explicit expression for Wigner-D functions in Euler angles using a separation of variables approach with $D_\ell^{mn}(\mathbf{R}(\alpha,\beta,\gamma)) = d_1(\alpha)d_2(\gamma)d_3(\beta)$. This yields three ordinary differential equations with periodic boundary conditions;

$$\begin{aligned} d_1'' + m^2 d_1 = 0, \quad d_1(0) = d_1(2\pi), \quad d_1'(0) = d_1'(2\pi), \\ d_2'' + n^2 d_1 = 0, \quad d_2(0) = d_2(2\pi), \quad d_2'(0) = d_2'(2\pi), \end{aligned}$$

$$\begin{aligned} (\sin\beta\, d_3')' + \left(\ell(\ell+1)\sin\beta - \frac{n^2 - 2mn\cos\beta + m^2}{\sin\beta}\right) d_3 &= 0, \\ d_3(0) = d_3(\pi) \qquad d_3'(0) &= d_3'(\pi). \end{aligned}$$

The solutions for d_1 and d_2 are given by $d_1(\alpha) = e^{-im\alpha}$ and $d_2(\gamma) = e^{-in\gamma}$ for $m, n \in \mathbb{Z}$. To solve the equation for d_3 we set $x = \cos\beta$ and obtain the following differential equation, [86, p. 138],

$$(1-x^2)d_3'' - 2xd_3' + \left(\ell(\ell+1) - \frac{(n-m)^2}{2(1-x)} - \frac{(n+m)^2}{2(1+x)}\right) d_3 = 0. \tag{3.21}$$

The solution of the differential equation for d_3 is given by the Rodrigues formula

$$d_3(x) = \frac{(-1)^{\ell-n}}{2^\ell}\sqrt{\frac{(\ell+m)!}{(\ell-n)!(\ell+n)!(\ell-m)!}}\sqrt{\frac{(1-x)^{n-m}}{(1+x)^{m+n}}}\frac{d^{\ell-m}}{dx^{\ell-m}}\frac{(1-x)^{n+\ell}}{(1+x)^{n-\ell}}. \tag{3.22}$$

This solution will be referred to as the Wigner-d function $d_\ell^{mn}(x) = d_3(x)$.
Putting $D_\ell^{mn}(\mathbf{R}(\alpha,\beta,\gamma)) = d_1(\alpha)d_2(\gamma)d_3(\beta)$ together again, we get an explicit expression for the Wigner-D functions in terms of Euler angles. For $|m|, |n| \leqslant \ell \in \mathbb{N}$ the Wigner-D functions are given by

$$D_\ell^{mn}(\alpha,\beta,\gamma) = e^{-im\alpha}e^{-in\gamma}d_\ell^{mn}(\cos\beta). \tag{3.23}$$

Hence, the Wigner-D functions factorise into an exponential function depending on α and γ and a Wigner-d function depending on β only. We continue by laying out important properties of the latter.

Properties of Wigner-d Functions The Wigner-d functions d_ℓ^{mn} are related to Jacobi polynomials cf. [84, p. 78] and [81, pp. 58] as

$$d_\ell^{mn}(x) = \varepsilon\sqrt{\frac{\left(\ell-\frac{\mu+\nu}{2}\right)!\left(\ell+\frac{\mu+\nu}{2}\right)!}{\left(\ell-\frac{\mu-\nu}{2}\right)!\left(\ell+\frac{\mu-\nu}{2}\right)!}}2^{-\frac{\mu+\nu}{2}}(1-x)^{\frac{\mu}{2}}(1+x)^{\frac{\nu}{2}}P_{\ell-\frac{\mu+\nu}{2}}^{(\mu,\nu)}(x), \tag{3.24}$$

where $\mu = |n-m|$, $\nu = |n+m|$ and

$$\varepsilon = \begin{cases} 1 & \text{if } n \geqslant m, \\ (-1)^{n-m} & \text{if } n < m. \end{cases}$$

From Equation (3.24) it can be seen that the Wigner-d functions d_ℓ^{mn} are polynomials of degree ℓ if $m+n$, and therefore, also $n-m$ are even. If $n \pm m$ are odd, $\sqrt{1-x^2}d_\ell^{mn}(x)$ will be a polynomial of degree ℓ. As distinguished from the degree ℓ, we denote m and n as the orders of the Wigner-d functions d_ℓ^{mn}.
A variety of properties of Wigner-d functions is a consequence of the fact that they are matrix elements of unitary, irreducible representations of SO(3) for special elements $\mathbf{R} \in \mathrm{SO}(3)$, i.e., $d_\ell^{mn}(\cos\beta) = D_\ell^{mn}(\mathbf{R}(0,\beta,0))$. At first, Wigner-d functions form a complete system of orthogonal functions for fixed m and n, with respect to the inner product

$$\langle d_\ell^{mn}, d_{\ell'}^{mn}\rangle = \int_{-1}^{1} d_\ell^{mn}(x)\, d_{\ell'}^{mn}(x)\, dx = \frac{2}{2\ell+1}\delta_{\ell,\ell'}. \tag{3.25}$$

Obviously, the Wigner-d functions are not normalised, therefore we introduce the normalised Wigner-d functions which are given by $\tilde{d}_\ell^{mn} = \sqrt{\frac{2\ell+1}{2}}d_\ell^{mn}$.

Lemma 3.3.2. *Wigner-d functions fulfil certain symmetry relations including*

$$d_\ell^{mn}(-x) = (-1)^{\ell+n}d_\ell^{-m+n}(x)$$

as well as

$$d_\ell^{mn}(x) = (-1)^{m+n}d_\ell^{nm}(x) = (-1)^{m+n}d_\ell^{-m-n}(x) = d_\ell^{-n-m}(x).$$

Proof. We set $x = \cos\beta$. By using the representation property (3.19) on $d_\ell^{mn}(\cos\beta) = D(\mathbf{R}(0,\beta,0))$ and the fact that $d_\ell^{mn}(-1) = (-1)^{\ell+n}\delta_{-m+n}$, we can write

$$\begin{aligned} d_\ell^{mn}(-\cos\theta) = d_\ell^{mn}(\cos(\pi-\theta)) &= \sum_{k=-\ell}^{\ell} d_\ell^{mk}(\cos\pi) d_\ell^{kn}(\cos\theta) \\ &= \sum_{k=-\ell}^{\ell} (-1)^{\ell+k}\delta_{-mk} d_\ell^{kn}(\cos\theta) = (-1)^{\ell+n} d_\ell^{-mn}(\cos\theta). \end{aligned}$$

Considering that the D_ℓ^{mn} are elements of a unitary matrix, we have

$$D_\ell^{mn}(\mathbf{R}^{-1}) = \overline{D_\ell^{nm}(\mathbf{R})}.$$

Setting $\mathbf{R} = \mathbf{R}(0,\beta,0)$, we get the back rotation $\mathbf{R}^{-1} = \mathbf{R}(\pi,\beta,\pi)$. For the separated Wigner-D functions, we have

$$D_\ell^{mn}(\mathbf{R}^{-1}) = e^{-im\pi} d_\ell^{mn}(\cos\beta) e^{-in\pi} = (-1)^{m+n} d_\ell^{mn}(\cos\theta).$$

As the Wigner-d functions are real-valued functions, we get

$$\overline{D_\ell^{nm}(\mathbf{R}(0,\beta,0))} = d_\ell^{nm}(\cos\beta).$$

Putting this together yields

$$(-1)^{m+n} d_\ell^{mn}(\cos\beta) = d_\ell^{nm}(\cos\beta).$$

By repeated application of these two symmetries, the other symmetries from the lemma follow immediately. ∎

Wigner-d Functions of Different Orders and Degrees We will now examine a fundamental property of the Wigner-d functions that are used for the algorithms in Section 4. That is, the possibility to connect Wigner-d functions d_ℓ^{mn} for given orders m and n, on one hand, to Wigner-d functions $d_{\ell'}^{mn}$ of lower degrees $\ell' < \ell$, and on the other hand, to Wigner-d functions $d_\ell^{m'n'}$ of lower orders $m' < m$ and $n' < n$.
The first connection can be realised by a three-term recurrence relation that reads for $|m|, |n| \leqslant \ell$ as

$$d_{\ell+1}^{mn}(x) = (u_\ell^{mn} x + v_\ell^{mn}) d_\ell^{mn}(x) + w_\ell^{mn} d_{\ell-1}^{mn}(x), \qquad x = \cos\theta, \tag{3.26}$$

with the recurrence coefficients given by

$$\begin{aligned} u_\ell^{mn} &= \frac{(\ell+1)(2\ell+1)}{\sqrt{(\ell+m+1)(\ell-m+1)(\ell+n+1)(\ell-n+1)}}, \\ v_\ell^{mn} &= \frac{-mn(2\ell+1)}{\ell\sqrt{(\ell+m+1)(\ell-m+1)(\ell+n+1)(\ell-n+1)}}, \\ w_\ell^{mn} &= \frac{-(\ell+1)\sqrt{(\ell^2-m^2)(\ell^2-n^2)}}{\ell\sqrt{(\ell+m+1)(\ell-m+1)(\ell+n+1)(\ell-n+1)}} \end{aligned}$$

where we set $d_\ell^{mn}(x) = 0$ for all $\ell < \max(|m|,|n|)$. This three-term recurrence relation results from the fact that the Wigner-d functions d_ℓ^{mn} are special cases of Jacobi polynomials (cf. (3.24)).

To formulate the second connection, we take into account that a finite linear combination of Wigner-d functions d_ℓ^{mn}, $\ell = \max(|m|,|n|), \ldots, L$, can be expressed in terms of Wigner-d functions $d_\ell^{m'n'}$, $\ell = \max(|m'|,|n'|), \ldots, L$, of certain lower orders m' and n' given by the following lemma.

Lemma 3.3.3. *Let* $L \in \mathbb{N}$*, and* $m, n \in \mathbb{Z}$ *with* $|m|, |n| \leqslant L$*. By* $\mathbb{D}_L^{mn}$ *we denote the space spanned by the functions* d_ℓ^{mn} *for* $\ell = max(|m|, |n|), \ldots, L$*. If we define*

$$m' = \begin{cases} 1 & \text{if } m = \pm n \neq 0, \\ 0 & \text{else,} \end{cases} \qquad n' = \begin{cases} 0 & \text{if } m = n = 0, \\ \pm 1 & \text{if } m = \pm n \neq 0, \\ 2 & \text{if } m + n \text{ even, } |m| \neq |n|, \\ 1 & \text{if } m + n \text{ odd, } |m| \neq |n|, \end{cases}$$

then we have $\mathbb{D}_L^{mn} \subseteq \mathbb{D}_L^{m'n'}$.

Proof. The lemma provides five distinctions. These are

i) $m = n = 0$: In this case, we simply get $\mathbb{D}_L^{00} = \mathbb{D}_L^{00}$.

ii) $m = n \neq 0$: Recalling Equation (3.24), the Wigner-d functions $d_\ell^{mn}(x)$ are related to Jacobi polynomials $P_{\ell-\max(|m|,|m|)}^{(|n-m|,|m+n|)}(x)$. Using this, we find the space $\mathbb{D}_L^{mm}$ to be spanned by functions $(1+x)^{|m|} P_\ell^{(0,2|m|)}(x)$ for $\ell = 0, \ldots, L - |m|$ multiplied with a scaling factor.
On the other hand, the space $\mathbb{D}_L^{11}$ is spanned by the functions $(1+x)P_\ell^{(0,2)}$ with $\ell = 0, \ldots, L-1$ multiplied by a factor. It follows that we have $\mathbb{D}_L^{mm} \subseteq \mathbb{D}_L^{11}$, for $m \neq 0$.

iii) $m = -n \neq 0$: Analogously to the previous case, the space $\mathbb{D}_L^{m-m}$ is spanned by functions $(1-x)^{|m|} P_\ell^{(2|m|,0)}(x)$ for $\ell = 0, \ldots, L-|m|$ multiplied with a scaling factor while the corresponding space $\mathbb{D}_L^{1-1}$ is spanned by the functions $(1-x)P_\ell^{(2,0)}$ with $\ell = 0, \ldots, L-1$ multiplied by a factor. Hence, it contains $\mathbb{D}_L^{m-m}$ for $m \neq 0$.

iv) $m + n$ odd: We set $|m + n| = 2M + 1$ and $|n - m| = 2N + 1$. In this notation, the space $\mathbb{D}_L^{mn}$ is spanned by functions $(1+x)^M (1-x)^N \sqrt{1-x^2} P_\ell^{(N,M)}(x)$ for $\ell = 0, \ldots, L - M - N - 1$ multiplied with a scaling factor.
The corresponding space $\mathbb{D}_L^{01}$ is spanned by the functions $\sqrt{1+x^2} P_\ell^{(1,1)}$ with $\ell = 0, \ldots, L - 1$ multiplied by a factor. Hence, we get $\mathbb{D}_L^{mn} \subseteq \mathbb{D}_L^{01}$.

v) $m + n$ even, $|m| \neq |n|$: Finally, we set $|m + n| = 2M$ and $|n - m| = 2N$. In this notation, the space $\mathbb{D}_L^{mn}$ is spanned by functions $(1+x)^M (1-x)^N P_\ell^{(N,M)}(x)$ for $\ell = 0, \ldots, L - M - N$ multiplied with a scaling factor.
The corresponding space $\mathbb{D}_L^{02}$ is spanned by the functions $P_\ell^{(2,2)}$ with $\ell = 0, \ldots, L-2$ multiplied by a factor. So indeed, we get $\mathbb{D}_L^{mn} \subseteq \mathbb{D}_L^{02}$.

■

Remark 3.3.4. *The quantities* m' *and* n' *will always depend on given orders* m *and* n *of Wigner-d functions, i.e.,* $m' = m'(m, n)$ *and* $n' = n'(m, n)$ *. But in order to avoid these lengthy notations, we will just write* m' *and* n'*. Also, we introduce the notation* $L_0 = \max(|m|, |n|)$ *and* $L_0' = \max(|m'|, |n'|)$.

Now let m and n be fixed and let

$$f = \sum_{\ell=L_0}^{L} \hat{f}_\ell^{mn} d_\ell^{mn}$$

be a finite expansion in Wigner-d functions d_ℓ^{mn}, $\ell = L_0, \dots, L$. Due to Lemma 3.3.3, f can be expressed in Wigner-d functions $d_\ell^{m'n'}$ of low orders,

$$f = \sum_{\ell=L_0'}^{L} \bar{f}_\ell^{mn} d_\ell^{m'n'} \tag{3.27}$$

where the sought transformation from coefficients $\hat{f}_\ell^{mn}$ to coefficients $\bar{f}_\ell^{mn}$ is linear. Therefore, there exists a matrix $\mathbf{A}^{mn} = (a_{\ell,k}) \in \mathbb{R}^{(L-L_0'+1)\times(L-L_0+1)}$ such that the vectors $\hat{\mathbf{f}}^{mn} = (\hat{f}_\ell) \in \mathbb{C}^{L-L_0+1}$ and $\bar{\mathbf{f}}^{mn} = (\bar{f}_\ell) \in \mathbb{C}^{L-L_0'+1}$ are related by the equation

$$\bar{\mathbf{f}}^{mn} = \mathbf{A}^{mn} \hat{\mathbf{f}}^{mn}. \tag{3.28}$$

Due to the orthogonality of the Wigner-d functions, the entries of the matrix $\mathbf{A}^{mn}$ are given by $a_{\ell,k} = \frac{\sqrt{(2\ell+1)(2k+1)}}{2} \langle d_\ell^{m',n'}, d_k^{m,n} \rangle$.

Corollary 3.3.5. *The matrices $\mathbf{A}^{mn}$ obey the following symmetries for $|m| + |n| > 2$;*

$$\mathbf{A}^{mn} = (-1)^{m-n} \mathbf{A}^{-m-n}, \quad \mathbf{A}^{mn} = (-1)^{m-n} \mathbf{A}^{nm},$$
$$\mathbf{A}^{mn} = \operatorname{diag}\left((-1)^{\ell+m+\delta_{|m|,|n|}}\right)_{\ell=L_0'}^{L} \mathbf{A}^{-mn} \operatorname{diag}\left((-1)^{\ell}\right)_{\ell=L_0}^{L}.$$

Moreover, for $|m| + |n| \leqslant 2$ we have

$$\mathbf{A}^{mn} = \begin{cases} -\mathbf{I} & \text{for } (m,n) = (0,-1) \text{ or } (m,n) = (1,0), \\ \mathbf{I} & \text{else.} \end{cases}$$

Proof. These are direct consequences of Lemma 3.3.2 and Lemma 3.3.3. ∎

The efficient application of the matrix $\mathbf{A}^{mn}$ to a vector will be discussed at length in Section 4.1. To show how this can be done, we will derive properties of $\mathbf{A}^{mn}$ that will enable us to apply some well-known fast algorithms.

3.4 Rotations on the 2-Sphere $\mathbb{S}^2$

In Section 2.2, we already saw that there is a relation between the rotation group $SO(3)$ and the sphere $\mathbb{S}^2$. The connection was reflected e.g. in the axis-angle parameterisation which characterises a rotation uniquely by an angle $\omega \in [0, 2\pi)$ and a unit vector $\mathbf{r} \in \mathbb{S}^2$. Also the Euler angle parameterisation gives a connection. Consider any point $\mathbf{p} \in \mathbb{S}^2$ with spherical coordinates $\mathbf{p} = (\alpha, \beta)$. It can be written as $\mathbf{p} = \mathbf{R}_{\mathbf{e}_z}(\alpha)\mathbf{R}_{\mathbf{e}_y}(\beta)\mathbf{e}_z$, cf. Lemma 2.2.8 and its proof. This is consistent with the Euler angles parameterisation of a rotation established in Definition 2.2.10 as $\mathbf{p} = \mathbf{R}_{\mathbf{e}_z}(\alpha)\mathbf{R}_{\mathbf{e}_y}(\beta)\mathbf{R}_{\mathbf{e}_z}(\gamma) = \mathbf{R}(\alpha, \beta, \gamma)$ is fulfilled for arbitrary $\gamma \in [0, 2\pi)$. Yet another link was given in Definition 3.2.7 where we defined the Wigner-D functions by means of spherical harmonics Y_ℓ^m, an orthonormal basis of $L^2(\mathbb{S}^2)$. This gives reason to collect some more properties that connect the sphere $\mathbb{S}^2$ and the rotation group $SO(3)$. By a reformulation of Definition 3.2.7, we get the representation property

$$Y_\ell^n(\mathbf{R}^{-1}\mathbf{p}) = \sum_{m=-\ell}^{\ell} Y_\ell^m(\mathbf{p}) D_\ell^{mn}(\mathbf{R}), \qquad \text{for } |m| \leqslant \ell, \mathbf{p} \in \mathbb{S}^2, \mathbf{R} \in SO(3), \tag{3.29}$$

reflecting the rotational invariance of the spherical harmonics.
Now, it comes as no surprise that Wigner-d as well as Wigner-D functions generalise functions that are known from harmonic analysis on $\mathbb{S}^2$. In fact, by setting one order of the Wigner-d functions to zero we get associated Legendre functions by

$$d_\ell^{m0}(x) = d_\ell^{0-m}(x) = \sqrt{\frac{(\ell-m)!}{(\ell+m)!}} P_\ell^m(x), \tag{3.30}$$

cf. Equations (3.10) and (3.24). Due to (3.30), the Wigner-d functions are also called *generalised associated Legendre functions*. We now obtain immediately another relation between Wigner-D functions D_ℓ^{mn} and spherical harmonics Y_ℓ^m as

$$\begin{aligned} Y_\ell^m(\alpha,\beta) &= \sqrt{\frac{2\ell-1}{4\pi}} e^{im\alpha} d_\ell^{m0}(\cos\beta) = \sqrt{\frac{2\ell-1}{4\pi}} \overline{D_\ell^{m0}(\mathbf{R}(\alpha,\beta,\gamma))} \\ &= \sqrt{\frac{2\ell-1}{4\pi}} D_\ell^{0-m}(\mathbf{R}(\gamma,\beta,\alpha)) \end{aligned}$$

where the Euler angle $\gamma \in [0, 2\pi)$ can be chosen freely. Owing to this relationship, Wigner-D functions are sometimes also called *generalised spherical harmonics* (e.g. [14]).
As a nice consequence of the above properties, one obtains the correlation between functions on $L^2(\mathbb{S}^2)$ using SO(3)- Fourier transforms. Computing a rotation-dependent correlation $C(\mathbf{R})$ of two functions $f, g \in L^2(\mathbb{S}^2)$ leads to evaluate the integral

$$C(\mathbf{R}) = \int_{\mathbb{S}^2} f(\mathbf{p}) \overline{g(\mathbf{R}^{-1}\mathbf{p})}\, d\mathbf{p}, \qquad \mathbf{R} \in SO(3). \tag{3.31}$$

Being a basis of the $L^2(\mathbb{S}^2)$, we can use the spherical harmonics Y_ℓ^m to expand the functions f and g into

$$f(\mathbf{p}) = \sum_{\ell=0}^{\infty} \sum_{m=-\ell}^{\ell} a_\ell^m Y_\ell^m(\mathbf{p}), \quad g(\mathbf{p}) = \sum_{\ell=0}^{\infty} \sum_{m=-\ell}^{\ell} b_\ell^m Y_\ell^m(\mathbf{p})$$

with Fourier coefficients a_ℓ^m and b_ℓ^m.
Inserting these expansions into (3.31) using the representation property (3.29) yields

$$\begin{aligned} C(\mathbf{R}) &= \int_{\mathbb{S}^2} \left(\sum_{\ell=0}^{\infty} \sum_{m=-\ell}^{\ell} a_\ell^m Y_\ell^m(\mathbf{p}) \right) \overline{\left(\sum_{\ell'=0}^{\infty} \sum_{n=-\ell'}^{\ell'} b_{\ell'}^n Y_{\ell'}^n(\mathbf{R}^{-1}\mathbf{p}) \right)}\, d\mathbf{p} \\ &= \sum_{\ell=0}^{\infty} \sum_{\ell'=0}^{\infty} \sum_{m=-\ell}^{\ell} \sum_{k=-\ell'}^{\ell'} \sum_{n=-\ell'}^{\ell'} \overline{D_{\ell'}^{kn}(\mathbf{R})} a_\ell^m \overline{b_{\ell'}^n} \int_{\mathbb{S}^2} Y_\ell^m(\mathbf{p}) \overline{Y_{\ell'}^k(\mathbf{p})}\, d\mathbf{p}. \end{aligned}$$

Knowing that the Y_ℓ^m constitute an orthonormal basis in $L^2(\mathbb{S}^2)$ for different degrees ℓ and orders m and considering the symmetries of Wigner-d functions from Lemma 3.3.2, we obtain the following equation

$$\begin{aligned} C(\mathbf{R}) &= \sum_{\ell=0}^{\infty} \sum_{m=-\ell}^{\ell} \sum_{n=-\ell}^{\ell} (-1)^{m+n} a_\ell^m \overline{b_\ell^n} D_\ell^{-m-n}(\mathbf{R}) \\ &= \sum_{\ell=0}^{\infty} \sum_{m=-\ell}^{\ell} \sum_{n=-\ell}^{\ell} (-1)^{m+n} a_\ell^{-m} \overline{b_\ell^{-n}} D_\ell^{mn}(\mathbf{R}). \end{aligned}$$

This perfectly fits into the setting of the NDSOFT from Definition 3.2.11.

3.5 Summation of Functions on the Rotation Group

To conclude this chapter on harmonic analysis on the rotation group, we present in some detail an interesting application of Fourier transforms on $\mathrm{SO}(3)$. That is, the fast summation of functions where we especially focus on the summation of radial basis functions to be established in Definition 3.5.1. The summation of radial basis functions is a key task in kernel based approximation methods, which have proved to be a suitable tool for solving a large class of problems on the rotation group, e.g. interpolation, least squares approximation, clustering or principle component analysis [29, 39, 85, 93]. Fast radial basis function algorithms that utilise fast Fourier techniques to find an approximation of the function f, which then can be evaluated at arbitrary nodes, are well established in the Euclidean and the spherical case. They have been discussed in [50, 69]. There, they come off well compared to other algorithms like Moving Squares, Thin Plate Splines, partition of unity, or fast multipole (see e.g. [28]). All this gave reason to establish a fast radial basis function algorithm for the rotation group as well. This was initially done in [43] and shall be reviewed here.
We will show how to evaluate linear combinations of radial functions on the rotation group based on the NDSOFT. This approach takes $\mathcal{O}(J + K)$ arithmetic operations for J and K arbitrarily distributed source and target nodes, respectively. This is a significant improvement over a naive algorithm computing the same result with complexity $\mathcal{O}(JK)$. We will also investigate a selection of radial functions and give explicit error bounds.
For more extensive information on these issues, we refer the reader to [43] which also provides numerical tests of runtime and errors of the presented algorithm along with more examples of kernel functions and an application of the method, namely the kernel density estimation from electron back scattering diffraction data, a problem relevant in texture analysis, see [2].

Radial Functions on SO(3) Using the distance measure from Definition 2.1.7, we define radial functions on $\mathrm{SO}(3)$.

Definition 3.5.1. *A function* $f\colon \mathrm{SO}(3) \to \mathbb{C}$ *is called a* radial function *with centre* $\mathbf{R}_0 \in \mathrm{SO}(3)$ *if it depends only on the rotational distance to* $\mathbf{R}_0 \in \mathrm{SO}(3)$. *That means for all* $\mathbf{R}_1, \mathbf{R}_2 \in \mathrm{SO}(3)$ *that fulfil* $|\mathbf{R}_1\mathbf{R}_0^{-1}| = |\mathbf{R}_2\mathbf{R}_0^{-1}|$ *with respect to the metric on* $\mathrm{SO}(3)$ *(see Lemma 2.2.9), the function* f *satisfies* $f(\mathbf{R}_1) = f(\mathbf{R}_2)$.

Note that a conjugate invariant function on $\mathrm{SO}(3)$ (cf. (3.8)) is automatically radial with centre $\mathbf{R}_0 = \mathbf{I}$. We already saw an example of such conjugate invariant function; the character $\chi_D(\mathbf{R})$ of a representation D (again, cf. (3.8)). Recall that it was defined to be the trace of a matrix representation of an element $\mathbf{R} \in \mathrm{SO}(3)$. So in case of the Wigner-D functions, we have

$$\chi_\ell(\mathbf{R}) = \sum_{m=-\ell}^{\ell} D_\ell^{mm}(\mathbf{R}).$$

Definition 2.1.7 established a connection between the rotational distance $|\mathbf{R}|$ and the axis-angle parameterisation, i.e., $|\mathbf{R}_\mathbf{r}(\omega)| = \omega$. Having this in mind, we formulate the following lemma.

Lemma 3.5.2. *The character* $\chi_\ell(\mathbf{R}_\mathbf{r}(\omega))$ *of a rotation* $\mathbf{R} \in \mathrm{SO}(3)$, *with* $\ell \in \mathbb{N}_0$ *given in terms of axis* $\mathbf{r} \in \mathbb{S}^2$ *and angle* $\omega \in [0, \pi]$, *can be calculated by*

$$\chi_\ell(\mathbf{R}_\mathbf{r}(\omega)) = U_{2\ell}\left(\cos\frac{\omega}{2}\right)$$

where

$$U_\ell(\cos\omega) = \frac{\sin(\ell+1)\omega}{\sin\omega},$$

for $\omega \in (0,\pi)$ *and* $U_\ell(1) = \ell+1$ *and* $U_\ell(-1) = (-1)^\ell(\ell+1)$ *denote the Chebyshev polynomials of second kind.*

Proof. As $\chi_\ell(\mathbf{R}_\mathbf{r}(\omega))$ is conjugate invariant, it satisfies $\chi_\ell(\mathbf{R}_\mathbf{r}(\omega)) = \chi_\ell(\mathbf{S}^{-1}\mathbf{R}_\mathbf{r}(\omega)\,\mathbf{S})$ for $\mathbf{S} \in \mathrm{SO}(3)$. By means of Definition 2.2.8, we can indeed find a rotation $\mathbf{S}$ such that $\mathbf{S}^{-1}\mathbf{R}_\mathbf{r}(\omega)\mathbf{S} = \mathbf{R}_{\mathbf{e}_z}(\omega)$. In Euler angles this rotation is denoted by $\mathbf{R}_{\mathbf{e}_z}(\omega) = \mathbf{R}(\omega,0,0)$. Applying formula (3.23) for Wigner-D functions separated in Euler angles, we get for the character

$$\chi_\ell(\mathbf{R}_\mathbf{r}(\omega)) \;=\; \sum_{m=-\ell}^{\ell} D_\ell^{mm}(\omega,0,0) = \sum_{m=-\ell}^{\ell} e^{-im\omega} = 1 + 2\sum_{m=1}^{\ell}\cos m\omega = U_{2\ell}\left(\cos\frac{\omega}{2}\right).$$

■

Using the conversion (2.4) from axis-angle to Euler angle parameterisation, we obtain the character for rotations given in arbitrary Euler angles by

$$\chi_\ell(\mathbf{R}(\alpha,\beta,\gamma)) = U_{2\ell}\left(\cos\frac{\beta}{2}\cos\frac{\alpha+\gamma}{2}\right).$$

Lemma 3.5.2 motivates an alternate definition of radial functions, that is a definition via a Fourier expansion in terms of Chebyshev polynomials of even degree. More precisely, we say, a function $f \in L^2(\mathrm{SO}(3))$ is a radial function with centre $\mathbf{R}_0 \in \mathrm{SO}(3)$ if and only if its Fourier coefficients $\hat{f}_\ell^{mn}$ fulfil

$$\hat{f}_\ell^{mn} = \hat{f}_\ell D_\ell^{mn}(\mathbf{R}_0), \quad \ell \in \mathbb{N}_0,\ m,n = -\ell,\dots,\ell,$$

for certain coefficients $\hat{f}_\ell$. Specifically, we then get

$$f(\mathbf{R}) = \sum_{\ell=0}^{\infty}\hat{f}_\ell\sum_{m=-\ell}^{\ell}\sum_{n=-\ell}^{\ell} D_\ell^{mn}(\mathbf{R})\overline{D_\ell^{mn}(\mathbf{R}_0)} = \sum_{\ell=0}^{\infty}\hat{f}_\ell\, U_{2\ell}\Big(\cos\frac{|\mathbf{R}\mathbf{R}_0^{-1}|}{2}\Big). \tag{3.32}$$

We will use this formula to construct some examples of radial functions.
As a consequence of the orthogonality of the characters (3.9) and of the Peter-Weyl-Theorem, we observe that the characters χ_ℓ for $\ell \in \mathbb{N}$ are a basis of the subspace of conjugate invariant functions on $\mathrm{SO}(3)$, cf. [44], that is why they are also called radial basis functions.
Let us now collect some radial functions on $\mathrm{SO}(3)$, which we denote by $\psi(\mathbf{R})$ with $\mathbf{R} \in \mathrm{SO}(3)$.

1. **The Generating Function's kernel:** The generating function of the Chebyshev polynomials of second kind is given by (cf. [5, Sec. 7])

$$\sum_{\ell=0}^{\infty}\kappa^\ell U_\ell(t) = \frac{1}{1-2\kappa t+\kappa^2}, \quad t \in [-1,1],\kappa \in (0,1). \tag{3.33}$$

Employing $(-1)^\ell U_\ell(-t) = U_\ell(t)$, we construct a radial function $\psi\colon \mathrm{SO}(3) \to \mathbb{R}$ based on the generating function for any $\kappa \in (0,1)$ by

$$\psi(\mathbf{R}) = \sum_{\ell=0}^{\infty}\kappa^{2\ell}U_{2\ell}\left(\cos\frac{|\mathbf{R}|}{2}\right) = \frac{\frac{1}{2}}{1-2\kappa\cos\frac{|\mathbf{R}|}{2}+\kappa^2} + \frac{\frac{1}{2}}{1+2\kappa\cos\frac{|\mathbf{R}|}{2}+\kappa^2},$$

which is in correspondence to (3.32).

2. **The Abel–Poisson Kernel:** By differentiating the generating function (3.33) with respect to κ, a second summation formula for the Chebyshev polynomials is obtained by

$$\sum_{\ell=0}^{\infty}(2\ell+1)\kappa^{\ell}U_{\ell}(t)=\frac{1-\kappa^2}{(1-2\kappa t+\kappa^2)^2},\quad t\in[-1,1],\kappa\in(0,1).$$

Again after symmetrisation, we get the Abel–Poisson kernel on $\mathrm{SO}(3)$ ([63, Sec. 17]),

$$\begin{aligned}\psi(\mathbf{R})&=\sum_{\ell=0}^{\infty}(2\ell+1)\kappa^{2\ell}U_{2\ell}\left(\cos\frac{|\mathbf{R}|}{2}\right)\\&=\frac{1}{2}\left(\frac{1-\kappa^2}{(1-2\kappa\cos\frac{|\mathbf{R}|}{2}+\kappa^2)^2}+\frac{1-\kappa^2}{(1+2\kappa\cos\frac{|\mathbf{R}|}{2}+\kappa^2)^2}\right).\end{aligned}$$

3. **The von Mises–Fisher Kernel:** For any $\kappa>0$, the von Mises–Fisher kernel on $\mathrm{SO}(3)$ is defined as (cf. [42])

$$\psi(\mathbf{R})=\sum_{\ell=0}^{\infty}\frac{\mathcal{I}_{\ell}(\kappa)-\mathcal{I}_{\ell+1}(\kappa)}{\mathcal{I}_0(\kappa)-\mathcal{I}_1(\kappa)}U_{2\ell}\left(\cos\frac{|\mathbf{R}|}{2}\right)=\frac{1}{\mathcal{I}_0(\kappa)-\mathcal{I}_1(\kappa)}e^{\kappa\cos|\mathbf{R}|},$$

where $\mathcal{I}_n$, $n\in\mathbb{N}_0$ denote the modified Bessel functions of first kind

$$\mathcal{I}_n(\kappa)=\frac{1}{\pi}\int_0^{\pi}e^{\kappa\cos\omega}\cos n\omega\,d\omega.$$

4. **The Gauss-Weierstrass Kernel:** For $\kappa>0$ the Gauss-Weierstrass kernel on $\mathrm{SO}(3)$ is defined by its Fourier series

$$\psi(\mathbf{R})=\sum_{\ell=0}^{\infty}(2\ell+1)e^{-\ell(\ell+1)\kappa}U_{2\ell}\left(\cos\frac{|\mathbf{R}|}{2}\right).$$

5. **The de la Vallée Poussin Kernel:** This kernel differs from the others given previously as it has a finite Fourier-Chebyshev expansion

$$\psi(\mathbf{R})=\frac{\binom{2\kappa+1}{\kappa}}{(2\kappa+1)2^{2\kappa}}\cos^{2\kappa}\frac{|\mathbf{R}|}{2}=\frac{1}{(2\kappa+1)2^{2\kappa}}\sum_{\ell=0}^{\kappa}(2\ell+1)\binom{2\kappa+1}{\kappa-\ell}U_{2\ell}\left(\cos\frac{|\mathbf{R}|}{2}\right),$$

for any $\kappa\in\mathbb{N}$. More details are provided in [10].

All these kernels ψ have in common that they define positive and monotonously decreasing radial functions on $\mathrm{SO}(3)$. The parameter κ determines the sharpness of the peak of ψ, see Figure 3.1. Furthermore, we have $\psi(\mathbf{I})\to\infty$ as $\kappa\to1$.

Fast Summation of Radial Functions Let $J,K\in\mathbb{N}$ and let $\mathcal{S}_j=(\mathbf{S}_1,\dots,\mathbf{S}_J)$, $\mathbf{S}_j\in\mathrm{SO}(3)$ and $\mathcal{T}_k=(\mathbf{T}_1,\dots,\mathbf{T}_K)$, $\mathbf{T}_k\in\mathrm{SO}(3)$ be lists of rotations. Given a radial function $\psi\colon\mathrm{SO}(3)\to\mathbb{C}$, and a coefficient vector $\mathbf{c}=(c_1,\dots,c_J)\in\mathbb{C}^J$, we are concerned with evaluating the sum

$$f(\mathbf{T}_k)=\sum_{j=1}^{J}c_j\psi(\mathbf{T}_k\mathbf{S}_j^{-1}),\quad k=1,\dots,K,\tag{3.34}$$

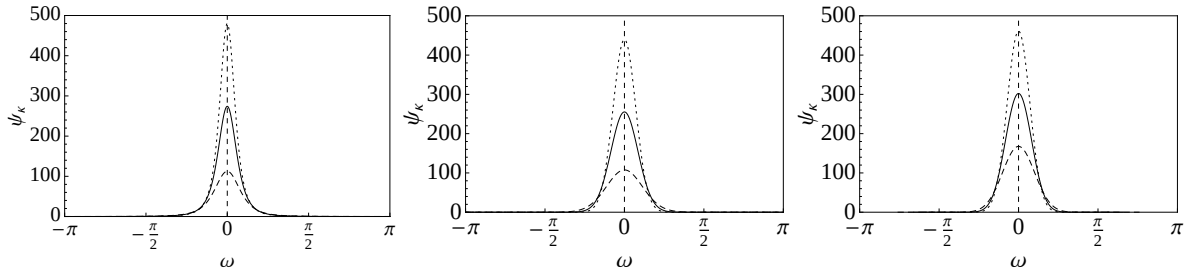


Figure 3.1: From left to right, we see examples of the following kernels: the Abel-Poisson Kernel for $\kappa = 0.8, 0.85, 0.875$, the von Mises-Fisher Kernel for $\kappa = 8, 14, 20$ and the de la Vallée Poussin Kernel for $\kappa = 20, 30, 40$ plotted with dashed, solid and dotted lines, respectively.

for all rotations $\mathbf{T}_k \in \mathcal{T}_K$. We will call the rotations $\mathbf{S}_j \in \mathcal{S}_J$ source nodes and the rotations $\mathbf{T}_k \in \mathcal{T}_K$ target nodes.
The function ψ can be approximated by its truncated Fourier series expansion as in (3.32);

$$\psi(\mathbf{T}\mathbf{S}^{-1}) \approx \psi_L(\mathbf{T}\mathbf{S}^{-1}) = \sum_{(\ell,m,n)\in\mathcal{I}_L} \hat{\psi}_\ell D_\ell^{mn}(\mathbf{T})\overline{D_\ell^{mn}(\mathbf{S})}, \quad \mathbf{S},\mathbf{T} \in \mathrm{SO}(3),$$

with a fixed cut-off degree $L \in \mathbb{N}_0$. Inserting this into the sum (3.34) leads to a separation of the source nodes $\mathbf{S}_j$, $j = 1,\dots,J$, and the target nodes $\mathbf{T}_k$, $k = 1,\dots,K$, as

$$f(\mathbf{T}_k) \approx f_L(\mathbf{T}_k) = \sum_{(\ell,m,n)\in\mathcal{I}_L} \hat{\psi}_\ell \left(\sum_{j=1}^{J} c_j \overline{D_\ell^{mn}(\mathbf{S}_j)} \right) D_\ell^{mn}(\mathbf{T}_k) \tag{3.35}$$

is satisfied.
Based on this representation, our fast summation algorithm splits into three steps:

1. the calculation of the innermost sum which is an adjoint nonequispaced Fourier transform as in (3.18),
2. the multiplication with the Fourier coefficients $\hat{\psi}_\ell$, and
3. the computation of the outer sums which is essentially a nonequispaced discrete $\mathrm{SO}(3)$ Fourier transform evaluated at the target nodes $\mathbf{T}_k$, $k = 1,\dots,K$ (cf. Definition 3.2.11).

By this separation, we achieved the reduction of the computational complexity as the first step has $\mathcal{O}(J)$ arithmetic operations, while the third step is of complexity $\mathcal{O}(K)$. The second step does not depend on the input nodes at all, but on the cut-off degree L of the Fourier sum only. Hence, it does not contribute to the complexity in terms of nodes. In total, we get a $\mathcal{O}(J+K)$ algorithm compared to the naive $\mathcal{O}(JK)$ one.
Of course, the cut-off degree L contributes to the complexity in all three steps but is neglected here. We will get back to this aspect in Section 4.2 after introducing the algorithms for the fast computation of the NDSOFT and its adjoint.
Expressing the original summation problem (3.34) as the matrix vector product $\mathbf{f} = \boldsymbol{\Psi}\mathbf{c}$ with

$$\mathbf{f} = (f(\mathbf{T}_k))_{k=1,\dots,K} \in \mathbb{C}^K,\ \boldsymbol{\Psi} \in \mathbb{C}^{K\times J},\ \Psi_{kj} = \psi(\mathbf{T}_k\mathbf{S}_j^{-1}) \text{ and } \mathbf{c} = (c_1,\dots,c_J) \in \mathbb{C}^J,$$

the fast summation algorithm corresponds to a rank $|\mathcal{I}_L|$ approximation $\boldsymbol{\Psi}_L \in \mathbb{C}^{K\times J}$ of the matrix $\boldsymbol{\Psi}$. We obtain a reformulation of (3.35) in terms of a matrix-vector multiplication by $\mathbf{f}_L = \boldsymbol{\Psi}_L \mathbf{c}$, with

$$\mathbf{f}_L \approx f(\mathbf{T}_k)_{k=1,\dots,K} \in \mathbb{C}^K, \text{ and } \boldsymbol{\Psi}_L \in \mathbb{C}^{K\times J}, \quad [\boldsymbol{\Psi}_L]_{kj} \approx \psi(\mathbf{T}_k \mathbf{S}_j^{-1}).$$

In particular, (3.35) allows a factorisation of the matrix $\boldsymbol{\Psi}_L$ into

$$\boldsymbol{\Psi}_L = \mathbf{D}_{L,\mathcal{T}_K} \hat{\boldsymbol{\Psi}}_L \mathbf{D}^H_{L,\mathcal{S}_J},$$

where $\mathbf{D}_{L,\mathcal{T}_K}$, $\mathbf{D}_{L,\mathcal{S}_J}$ are the Fourier matrices as defined in Definition 3.2.11, and $\hat{\boldsymbol{\Psi}}_L \in \mathbb{C}^{|\mathcal{I}_L|\times|\mathcal{I}_L|}$ is given by

$$\hat{\boldsymbol{\Psi}}_L = \operatorname{diag}(\hat{\boldsymbol{\psi}}_L), \ [\hat{\boldsymbol{\psi}}_L]_{\ell,m,n} = \hat{\psi}_\ell.$$

Fast Summation of Arbitrary Functions on $SO(3)$ In case that the function ψ is not radial, its truncated Fourier series expansion may be written using the representation property (3.19) of the Wigner–D functions as

$$\psi(\mathbf{T}\mathbf{S}^{-1}) \approx \psi_L(\mathbf{T}\mathbf{S}^{-1}) = \sum_{(\ell,m,n)\in\mathcal{I}_L} \hat{\psi}_\ell^{mn} \sum_{h=-\ell}^{\ell} D_\ell^{mh}(\mathbf{T})\overline{D_\ell^{nh}(\mathbf{S})}, \quad \mathbf{S},\mathbf{T} \in SO(3).$$

Substitution in (3.34) and rearranging of the sums again yields a separation of source and target nodes

$$f(\mathbf{T}_k) \approx f_L(\mathbf{T}_k) = \sum_{(\ell,m,n)\in\mathcal{I}_L} D_\ell^{mn}(\mathbf{T}_k) \sum_{h=-\ell}^{\ell} \hat{\psi}_\ell^{mh} \sum_{j=1}^{J} c_j \overline{D_\ell^{nh}(\mathbf{S}_j)}, \qquad k = 1,\dots,K,$$

where the innermost sum is an adjoint Fourier transform and the outer most sum is a direct Fourier transform. In contrast to the radial case, Step 2 now consists of $L+1$ matrix-matrix multiplications

$$\hat{\mathbf{f}}_\ell = \hat{\boldsymbol{\psi}}_\ell \hat{\mathbf{c}}_\ell, \quad \ell = 0,\dots,L,$$

where matrices of Fourier coefficients $\hat{\mathbf{f}}_\ell, \hat{\boldsymbol{\psi}}_\ell, \hat{\mathbf{c}}_\ell \in \mathbb{C}^{(2\ell+1)\times(2\ell+1)}$ are defined as

$$[\hat{\boldsymbol{\psi}}_\ell]_{mn} = \hat{\psi}_\ell^{mn}, \quad [\hat{\mathbf{c}}_\ell]_{mn} = [\mathbf{D}_{L,\mathcal{S}_J}\mathbf{c}]_{\ell,m,n} \text{ and } [\hat{\mathbf{f}}_\ell]_{mn} = \hat{f}_\ell^{mn}, \quad m,n = -\ell,\dots,\ell.$$

In terms of nodes, again the complexity is reduced from $\mathcal{O}(JK)$ to $\mathcal{O}(J+K)$.

Error Estimates Next we discuss the error introduced by the approximation of $\boldsymbol{\Psi}$ by $\boldsymbol{\Psi}_L$. Obviously, this error depends on the decay rate of the Fourier coefficients $\hat{\psi}_\ell^{mn}$.

Lemma 3.5.3. *For a radial function $\psi \in L^\infty(SO(3))$, we have the error estimate*

$$\|f - f_L\|_\infty \leqslant \|\mathbf{c}\|_1 \sum_{l>L} (2\ell+1)\left|\hat{\psi}_\ell\right|$$

while for a general $\psi \in L^\infty(SO(3))$, the error is bounded by

$$\|f - f_L\|_\infty \leqslant \|\mathbf{c}\|_1 \sum_{\ell>L} \sqrt{2\ell+1}\,\|\hat{\boldsymbol{\psi}}_\ell\|_F,$$

where $\|\hat{\boldsymbol{\psi}}_\ell\|_F$ denotes the Frobenius norm of the matrix $\hat{\boldsymbol{\psi}}_\ell \in \mathbb{C}^{(2\ell+1)\times(2\ell+1)}$.

Proof. The first assertion follows from

$$\|f - f_L\|_\infty \leqslant \sum_{j=1}^{J} \left|c_j\right| \max_{\mathbf{T}\in SO(3)} \left| \sum_{\ell>L} \left|\hat{\psi}_\ell\right| U_{2\ell}\left(\cos\frac{\left|\mathbf{T}\mathbf{S}_j^{-1}\right|}{2}\right)\right|.$$

For the second assertion, we rearrange the values of the Wigner-D functions at $\mathbf{R} \in SO(3)$ as matrices $\mathbf{D}_\ell(\mathbf{R}) \in \mathbb{C}^{(2\ell+1)\times(2\ell+1)}$ with

$$[\mathbf{D}_\ell(\mathbf{R})]_{mn} = D_\ell^{mn}(\mathbf{R}).$$

Hence, we can write

$$\begin{aligned}\|f - f_L\|_\infty &\leqslant \sum_{j=1}^{J} \left|c_j\right| \max_{\mathbf{T}\in SO(3)} \left| \sum_{\ell>L} \sum_{m=-\ell}^{\ell} \sum_{n=-\ell}^{\ell} \hat{\psi}_\ell^{mn} D_\ell^{mn}(\mathbf{T})\right| \\ &= \|\mathbf{c}\|_1 \max_{\mathbf{T}\in SO(3)} \left| \sum_{\ell>L} \operatorname{tr}(\hat{\boldsymbol{\psi}}_\ell^H \mathbf{D}_\ell(\mathbf{T}))\right|.\end{aligned}$$

Applying the Cauchy-Schwarz inequality to the Frobenius inner product, we obtain

$$\begin{aligned}\|f - f_L\|_\infty &\leqslant \|\mathbf{c}\|_1 \max_{\mathbf{T}\in SO(3)} \sum_{\ell>L} \|\hat{\boldsymbol{\psi}}_\ell\|_F \left(\operatorname{tr}(\mathbf{D}_\ell(\mathbf{T})\mathbf{D}_\ell^H(\mathbf{T}))\right)^{1/2} \\ &= \|\mathbf{c}\|_1 \max_{\mathbf{T}\in SO(3)} \sum_{\ell>L} \|\hat{\boldsymbol{\psi}}_\ell\|_F U_{2\ell}(1)^{1/2} \\ &= \|\mathbf{c}\|_1 \sum_{\ell>L} \sqrt{2\ell+1}\, \|\hat{\boldsymbol{\psi}}_\ell\|_F.\end{aligned}$$

■

Analogously to [50], one obtains immediately the following approximation error between the matrices $\boldsymbol{\Psi}$ and $\boldsymbol{\Psi}_L$ with respect to the p-matrix norm.
For a radial function $\psi \in L^\infty(SO(3))$ and $1 \leqslant p \leqslant \infty$, we have the following condition on the p-matrix norm

$$\|\boldsymbol{\Psi} - \boldsymbol{\Psi}_L\|_p \leqslant J^{1-\frac{1}{p}} K^{\frac{1}{p}} \sum_{\ell>L} (2\ell+1)\left|\hat{\psi}_\ell\right|.$$

For a general function $\psi \in L^\infty(SO(3))$ one has

$$\|\boldsymbol{\Psi} - \boldsymbol{\Psi}_L\|_p \leqslant J^{1-\frac{1}{p}} K^{\frac{1}{p}} \sum_{\ell>L} \|\hat{\boldsymbol{\psi}}_\ell\|_F.$$

Let us now apply the error estimates for $\frac{\|f-f_L\|_\infty}{\|\mathbf{c}\|_1}$ to the particular, previously defined kernels by considering the sum

$$\sum_{\ell=L+1}^{\infty} (2\ell+1)\left|\hat{\psi}_\ell\right|.$$

We computed for the radial function derived from the generating function an error of

$$\begin{aligned}\sum_{\ell=L+1}^{\infty} (2\ell+1)\kappa^{2\ell} &= \kappa^{2L+2}\left(\frac{2L}{1-\kappa^2} + \frac{3-\kappa^2}{(1-\kappa^2)^2}\right) \\ &= \mathcal{O}(L\kappa^{2L}) \qquad \text{for fixed } 0 < \kappa < 1.\end{aligned}$$

Analogously, for the Abel–Poisson kernel, we have

$$\begin{aligned}\sum_{\ell=L+1}^{\infty}(2\ell+1)^2\kappa^{2\ell} &= \kappa^{2L+2}\left(\frac{4L(L+1)}{1-\kappa^2}+\frac{8L+9+\kappa^4-2\kappa^2}{(1-\kappa^2)^3}\right)\\ &= \mathcal{O}(L^2\kappa^{2L}) \qquad \text{for fixed } 0<\kappa<1.\end{aligned}$$

For the Gauss-Weierstrass kernel, we have

$$\begin{aligned}\sum_{\ell=L+1}^{\infty}(2\ell+1)^2 e^{-\ell(\ell+1)\kappa} &< \sum_{\ell=L+1}^{\infty}(2\ell+1)^2 e^{-(L+1)(\ell+1)\kappa}\\ &= e^{-(L+1)^2\kappa}\left(\frac{4L(L+1)}{e^{(L+1)\kappa}-1}+\frac{(8L+9)e^{2(L+1)\kappa}+1-2e^{(L+1)\kappa}}{(e^{(L+1)\kappa}-1)^3}\right)\\ &= \mathcal{O}(L^2 e^{-(L+1)(L+2)\kappa}) \qquad \text{for fixed } \kappa>0.\end{aligned}$$

For the von Mises-Fisher kernel, we use orthogonality of the cosine system. Under the condition that we chose $\ell > \kappa+2$, the resulting sum can be approximated by the error estimate in the Leibniz criterion. This yields

$$\begin{aligned}&\left|\sum_{r=0}^{\infty}\frac{\kappa^r}{\pi r!}\int_0^{\pi}\cos^r\omega(\cos\ell\omega-\cos(\ell+1)\omega)\,d\omega\right|\\ &= \left|\sum_{r=\ell}^{\infty}\frac{\kappa^r}{\pi r!}\int_0^{\pi}\cos^r\omega(\cos\ell\omega-\cos(\ell+1)\omega)\,d\omega\right|\\ &= \left|\sum_{r=\ell}^{\infty}\frac{\kappa^r}{\pi r!}\left(\frac{(-1)^r 2^{-\ell}\pi^{\frac{3}{2}}\cos\frac{\pi}{2}(\ell+r)\Gamma(1+r)}{\Gamma(\frac{1+\ell-r}{2})\Gamma(1-\ell+r)\Gamma(\frac{2+\ell+r}{2})}-\frac{(-1)^r 2^{-\ell}\pi^{\frac{3}{2}}\cos\frac{\pi}{2}(1+\ell+r)\Gamma(1+r)}{2\Gamma(\frac{2+\ell-r}{2})\Gamma(-\ell+r)\Gamma(\frac{3+\ell+r}{2})}\right)\right|\\ &< \frac{\kappa^\ell}{2^\ell\ell!}.\end{aligned}$$

We first reckon

$$\begin{aligned}&\sum_{\ell=L+1}^{\infty}(2\ell+1)\left|\frac{\mathcal{I}_\ell(\kappa)-\mathcal{I}_{\ell+1}(\kappa)}{\mathcal{I}_0(\kappa)-\mathcal{I}_1(\kappa)}\right|\\ &= \frac{1}{\mathcal{I}_0(\kappa)-\mathcal{I}_1(\kappa)}\sum_{\ell=L+1}^{\infty}(2\ell+1)\left|\sum_{r=0}^{\infty}\frac{\kappa^r}{\pi r!}\int_0^{\pi}\cos^r\omega(\cos\ell\omega-\cos(\ell+1)\omega)\,d\omega\right|\\ &< \frac{1}{\mathcal{I}_0(\kappa)-\mathcal{I}_1(\kappa)}\sum_{\ell=L+1}^{\infty}\frac{(2\ell+1)\kappa^\ell}{2^\ell\ell!}.\end{aligned}$$

This sum can be estimated by two geometric series. Thus, one obtains

$$\begin{aligned}
&\frac{1}{\mathcal{I}_0(\kappa)-\mathcal{I}_1(\kappa)}\sum_{\ell=L+1}^{\infty}\frac{(2\ell+1)\kappa^\ell}{2^\ell \ell!}\\
&< \frac{1}{\mathcal{I}_0(\kappa)-\mathcal{I}_1(\kappa)}\left(\frac{\kappa^{L+1}}{2^L L!}\sum_{s=0}^{\infty}\left(\frac{\kappa}{2(L+1)}\right)^s+\frac{\kappa^{L+1}}{2^{L+1}(L+1)!}\sum_{s=0}^{\infty}\left(\frac{\kappa}{2(L+2)}\right)^s\right)\\
&= \frac{\kappa^{L+1}}{\mathcal{I}_0(\kappa)-\mathcal{I}_1(\kappa)}\left(\frac{(L+1)}{2^{L-1}L!(2(L+1)-\kappa)}+\frac{(L+2)}{2^{L+1}(L+1)!(2(L+2)-\kappa)}\right)\\
&= \mathcal{O}\left(\frac{\kappa^L}{L!}\right) \qquad \text{for fixed } \kappa>0.
\end{aligned}$$

Since the de la Vallée Poussin kernel has a finite Fourier expansion, the approximation error becomes exactly zero when choosing the cutoff degree $L \geqslant \kappa$. However, for very large κ truncating the Fourier expansion at a cutoff degree $L < \kappa$ might be desirable. Details on this can be found in [43].
For the sake of completeness, we also state the asymptotic estimate for the approximation error of the de la Vallée Poussin kernel for $\lambda \to \infty$ and $L = \lambda\sqrt{\kappa+\frac{1}{2}} \in \mathbb{N}$. It is given by

$$\begin{aligned}
\frac{\|f-f_L\|_\infty}{\|\mathbf{c}\|_1} &= \frac{\binom{2\kappa+1}{\kappa}}{(2\kappa+1)2^{2\kappa}}\sum_{\ell=L}^{\kappa}\frac{(2\ell+1)^2\kappa!(\kappa+1)!}{(\kappa-\ell)!(\kappa+\ell+1)!}\\
&= \frac{4}{2\kappa+1}2^{-2\kappa}\sum_{\ell=0}^{\kappa-L}(\kappa+\frac{1}{2}-\ell)^2\binom{2\kappa+1}{\ell} \to 4\int_\lambda^\infty \ell^2\,\mathrm{d}N_{0,1}(\ell).
\end{aligned}$$

where $N_{0,1}(\ell)$ is the normal distribution

$$N_{\mu,\sigma}(x) = \frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}}$$

with mean $\mu = 0$ and standard deviation $\sigma = 1$. There is an upper bound for this approximation error that does not depend on κ and decays exponentially in λ. It satisfies for $\kappa \in \mathbb{N}$,

$$\frac{\|f-f_L\|_\infty}{\|\mathbf{c}\|_1} \leqslant \frac{C}{(\kappa+\frac{1}{2})^{3/2}}\int_{-\infty}^{-\lambda\sqrt{\kappa+\frac{1}{2}}} l^2 e^{-\frac{1}{2}\frac{\ell^2}{(\kappa+\frac{1}{2})/2}}\,\mathrm{d}\ell \leqslant C\int_{-\infty}^{-\lambda}(\ell+1)^2 e^{-\frac{1}{8}(\ell-1)^2}\,\mathrm{d}\ell$$

for some constant $C > 0$ that does neither depend on n nor on ℓ. The proof of this error bound, can be found in [43].

If we seek to compute the sum of radial basis functions as in (3.35), one type of error that will occur is the one which depends on the polynomial cut-off of the kernel functions ψ. We just gave theoretic error bounds for this kind of error for some kernel functions. Having in mind that we want to find fast algorithms to perform the computation of (3.35), we will be dealing with another type of error, namely the approximation error of the nonequispaced fast $SO(3)$ Fourier transform algorithm.
We will provide more information on this type of error in Section 4.2. There we will also perform numerical experiments on the runtime of the fast summation algorithms. But before this can be done, we will introduce, describe and analyse the algorithms we need to compute the $SO(3)$ Fourier transforms in a fast manner.

4 Algorithms for SO(3) Fourier Transforms

In this section, we present the most important part of the whole work, namely the fast algorithms to compute $SO(3)$ Fourier transforms at arbitrarily sampled rotations. Recall that we have already introduced the foundations of harmonic analysis on $SO(3)$ and in particular defined the discrete $SO(3)$ Fourier transform (NDSOFT) in Section 3.2. In this paragraph, we will outline the general strategy to efficiently compute the NDSOFT by our nonequispaced fast $SO(3)$ Fourier transform algorithm (NFSOFT) before it will be described in detail in the upcoming subsections.
This algorithm is based on evaluating the Wigner-D functions D_ℓ^{mn}, which yield an orthogonal basis of $L^2(SO(3))$ as we stated in Lemma 3.2.8.
Let $L \in \mathbb{N}$, by means of Definition 3.2.11 every function $f \in \mathbb{D}_L$ has a unique series expansion in terms of Wigner-D functions,

$$f = \sum_{(\ell,m,n)\in\mathcal{I}_L} \hat{f}_\ell^{mn} D_\ell^{mn},$$

with $SO(3)$ Fourier coefficients $\hat{f}_\ell^{mn} = \frac{2\ell+1}{8\pi^2}\langle f, D_\ell^{mn}\rangle$. Moreover, we have a set $\mathcal{R}_Q$ of Q arbitrary rotations $\mathbf{R}$ given in their Euler angles

$$\mathcal{R}_Q = \{\mathbf{R}(\alpha_q, \beta_q, \gamma_q) : q = 1, \ldots, Q\}, \qquad \text{with } Q \in \mathbb{N}.$$

If parameterised in Euler angles, we can rewrite their series expansion using Equation (3.23) as

$$f(\mathbf{R}(\alpha_q, \beta_q, \gamma_q)) = \sum_{(\ell,mn)\in\mathcal{I}_L} \hat{f}_\ell^{mn} e^{-im\alpha_q} d_\ell^{mn}(\cos\beta_q) e^{-in\gamma_q} \tag{4.1}$$

where the $SO(3)$ Fourier coefficients $\hat{f}_\ell^{mn} \in \mathbb{C}$ are given by the integrals

$$\hat{f}_\ell^{mn} = \int_0^{2\pi}\int_0^{\pi}\int_0^{2\pi} f(\mathbf{R}(\alpha,\beta,\gamma)) e^{im\alpha} d_\ell^{mn}(\cos\beta) e^{in\gamma} \sin\beta \, d\gamma \, d\beta \, d\alpha.$$

If we rearrange the triple sums of Equation (4.1) to

$$f(\mathbf{R}(\alpha_q, \beta_q, \gamma_q)) = \sum_{m=-L}^{L}\sum_{n=-L}^{L} e^{-im\alpha_q} e^{-in\gamma_q} \sum_{\ell=\max(|m|,|n|)}^{L} \hat{f}_\ell^{mn} d_\ell^{mn}(\cos\beta_q), \tag{4.2}$$

we will see the merit of the Euler angle representation. The sum (4.2) almost resembles

$$f(\mathbf{R}(\alpha_q, \beta_q, \gamma_q)) = \sum_{\ell=-L}^{L}\sum_{m=-L}^{L}\sum_{n=-L}^{L} \hat{h}_\ell^{mn} e^{-im\alpha_q} e^{-i\ell\beta_q} e^{-in\gamma_q}, \tag{4.3}$$

a standard three-dimensional Fourier sum, except for the sum over the degree ℓ. But for us, the main difference between the first and the second sum is that for the second one we know an algorithm that can evaluate the sum fast, namely the NFFT algorithm. Its cost is that of a classical three-dimensional FFT plus a term linear in the number of nodes Q, $\mathcal{O}(L^3 \log L + Q)$; see eg.; [12, 26, 56, 71]

and the references therein. Moreover, a C subroutine library implementing the NFFT algorithm is available; [49]. Of course, this is only helpful if we can efficiently compute the coefficients $\hat{h}_l^{mn}$ out of the $SO(3)$ Fourier coefficients $\hat{f}_l^{mn}$ (see Figure 4.1 for a schematic representation of the coefficient transform).
Hence, the crucial point to replace the triple sum of Wigner-D functions by a triple sum with complex exponentials is the modification of the sum over ℓ to

$$\sum_{\ell=\max(|m|,|n|)}^{L} \hat{f}_\ell^{mn} d_\ell^{mn}(\cos\beta) = \sum_{\ell=-L}^{L} \hat{h}_\ell^{mn} e^{-i\ell\beta}.$$

Note that this replacement will not depend on the given rotation. Instead, it is a rotation-independent transformation of coefficients. We take two main steps to do this. At first, we exchange expansions in Wigner-d functions with expansions in Chebyshev polynomials of first kind

$$\sum_{\ell=\max(|m|,|n|)}^{L} \hat{f}_\ell^{mn} d_\ell^{mn}(\cos\beta) = \sum_{\ell=0}^{L} \hat{g}_\ell^{mn} T_\ell(\cos\beta).$$

And secondly, in a far easier step, we obtain the sought expansion in complex exponential functions. The transformation from Wigner-d function expansions to Chebyshev expansions will be considered in Section 4.1 and we will refer to it as the discrete transformation of Wigner-d functions (DWT).
We will actually present two different ways to efficiently compute the Chebyshev coefficients $\hat{g}_\ell^{mn}$ out of the $SO(3)$ Fourier coefficients $\hat{f}_\ell^{mn}$: the first approach is based on the fast polynomial transform (FPT) algorithm introduced by [23, 70] which uses a cascade summation based on the three-term recurrence relations of the respective orthogonal polynomials. This approach has been adopted to the Wigner-d functions in [68, 89] where the previously mentioned three-term recurrence relation (3.26) is used. The implementation of this approach is part of the NFFT subroutine library [49]. We will refer to it as the FWT-C (the fast Wigner transformation based on cascade summation) and discuss it in more detail in Section 4.1.1.
The second approach is the fast transformation of Wigner-d functions based on semiseparable matrices (FWT-S). Essential to this approach are the transformation of arbitrary Wigner-d functions into those of low orders m and n (cf. (3.27)) and the application of a differential operator to the Wigner-d functions. It will be covered in Section 4.1.2. There we replace the cascade summation scheme with an approximate technique that is a generalisation of the algorithms proposed in [77] for spherical harmonic expansions on the sphere $\mathbb{S}^2$. This technique is based on an efficient algorithm to compute matrix-vector multiplications with semiseparable matrices. An application of this technique to Gegenbauer polynomials has been analysed in [48]. Here it will be used for Wigner-d functions for the first time. We will complete the consideration on transformations of Wigner-d functions by comparing the two approaches in terms of runtime and accuracy in Section 4.1.3.
After the central step of computing with Wigner-d functions, Section 4.2 returns to the bigger scope of the $SO(3)$ Fourier transform as we briefly explain how to replace Chebyshev polynomials with complex exponentials to obtain

$$\sum_{\ell=0}^{L} \hat{g}_\ell^{mn} T_\ell(\cos\beta_q) = \sum_{\ell=-L}^{L} \hat{h}_\ell^{mn} e^{-i\ell\beta_q}.$$

Having finally obtained the three-dimensional Fourier sum (4.3) we were looking for, we can invoke the NFFT algorithm. We are now able to compute fast $SO(3)$ Fourier transforms (NFSOFT). We

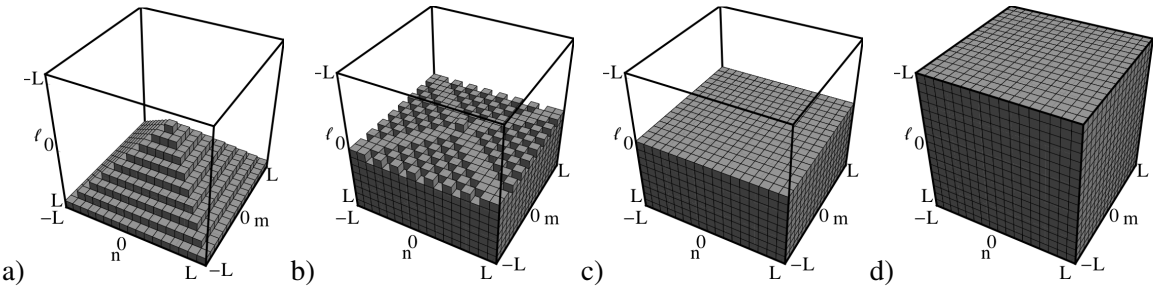


Figure 4.1: This figure shows an overview of the four transitions of coefficients we perform to get from Equation (4.2) to (4.3). The coordinates (ℓ, m, n) of each cube represent the orders m and n, and the degree ℓ of the corresponding coefficient.
In a) we see a schematic arrangement of the $SO(3)$ Fourier coefficients, with which we start our computations. When using the FWT-S, these coefficients will be transformed into the coefficients seen in b), and then into the coefficients of c). The FWT-C omits this step directly producing the Chebyshev coefficients depicted in c). Finally in d) we see the coefficients from the standard three-dimensional Fourier sum from (4.3), on which we perform the NFFT algorithm. We will explain these transforms in the following sections, for a comprehensive overview, please see Figure 4.8.

will summarise the algorithm, derive its adjoint version and consider its complexity. Depending on whether we use the FWT-C or the FWT-S we will get algorithms of $\mathcal{O}(L^4 + Q)$, $\mathcal{O}(L^3 \log^2 L + Q)$ or $\mathcal{O}(L^3 \log L + Q)$ arithmetic operations where Q is the number of sampling nodes and L the cutoff degree of the Fourier sum. This is a huge improvement over the $\mathcal{O}(L^3 Q)$ operations for the naive computation of the NDSOFT.
In Section 4.2.2 we provide numerical tests on accuracy and runtime. In the course of that, we will test the summation algorithms for radial and arbitrary functions on $SO(3)$ that have been introduced in Section 3.5. We will also test a quadrature rule in the adjoint transform for a special sampling set on $SO(3)$. This gives a brief example on how the evaluation of $SO(3)$ Fourier coefficients from function samples can be accomplished, although this is no direct focus of this work. However, readers interested in the subject of the Fourier analysis of functions on $SO(3)$ are recommended to consider [35] and [79] where two different types of algorithms for the $SO(3)$ Fourier analysis, i.e., the inverse NDSOFT, in our terminology, are described. These Fourier analysis algorithms are based on our NFSOFT algorithm as well as on its adjoint.
Also, there are other approaches to compute samples of functions on $SO(3)$ at special, i.e., equispaced sampling sets the NDSOFT, e.g. [53] and [73]. The authors of [53] use a similar approach to split the Wigner-D functions but they divide their transform differently into a two-dimensional FFT and a direct recursive evaluation of Wigner-d functions which depends on the chosen sampling set and in contrast to our NFSOFT is no node-dependent but a coefficient transformation. In [73] it is described how to expand the Wigner-d functions directly into a Fourier sum and thus evaluating a quadruple sum by means of an FFT. Both works describe $\mathcal{O}(L^4)$ algorithms although the authors of [53] point out that their algorithm has the potential to be a $\mathcal{O}(L^3 \log^2 L)$ one, as well. Note that the number of source rotations Q does not occur in the total complexity of these equispaced transforms as it depends on L and is of order $Q = \mathcal{O}(L^3)$. We see that our NFSOFT algorithm has favourable complexity over these two and can moreover be applied to arbitrary sampling sets on $SO(3)$.

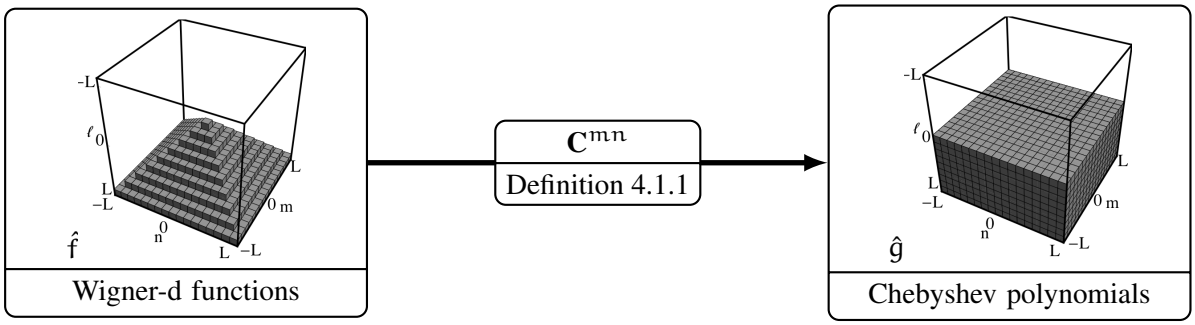


Figure 4.2: The aim of the transform of Wigner-d functions is to turn expansions of these functions with maximum degree L into expansions of Chebyshev polynomials of first kind. The figure shows the input and output coefficients of the transform, in the sense that each cube with coordinates (ℓ, m, n) represents a coefficient of orders m and n, and degree ℓ. The transformation between these coefficients is a linear one and can, hence, be expressed as a matrix. In this transform, the matrices $\mathbf{C}^{mn}$ for $m, n = -L, \dots, L$ realise the change of basis.

4.1 Fast Transforms of Wigner-d Functions

The aim of this section is to find a transformation which turns linear combinations of Wigner-d functions d_ℓ^{mn} for a fixed set of orders $m, n \in \mathbb{N}$, with $|m|, |n| \leqslant \ell$, into linear combinations of Chebyshev polynomials of first kind $T_\ell(x) = \cos(\ell \arccos x)$.
Recall Equation (3.24) which gave an expression of Wigner-d functions in terms of Jacobi polynomials. From this equation, we saw that the Wigner-d functions d_ℓ^{mn} are polynomials of degree ℓ if $m+n$ is even. If $n+m$ is odd, the Wigner-d function $d_\ell^{mn}(x)$ is almost a polynomial but multiplied with the factor $\sqrt{1-x^2}$. In that case, it is a polynomial of degree $\ell - 1$.
Consequently, the task for this section is to find coefficients $\hat{g}_\ell^{mn}$ such that for any given set of orders m and n and known coefficients $\hat{f}_\ell^{mn}$, we get the series expansion

$$\sum_{\ell=L_0}^{L} \hat{f}_\ell^{mn} d_\ell^{mn}(x) = \begin{cases} \sum_{\ell=0}^{L} \hat{g}_\ell^{mn} T_\ell(x) & \text{if } m+n \text{ even,} \\ \sum_{\ell=0}^{L-1} \hat{g}_\ell^{mn} \sqrt{1-x^2}\, T_\ell(x) & \text{if } m+n \text{ odd.} \end{cases} \tag{4.4}$$

for $x \in [-1, 1]$ and with $L_0 = \max(|m|, |n|)$, see also Figure 4.2.

Definition 4.1.1. *For given $L \in \mathbb{N}$ and fixed orders m and n, the change of basis, given in Equation (4.4), from* SO(3) *Fourier coefficients $\hat{\mathbf{f}}^{mn} = \left(\hat{f}_{L_0}^{mn}, \dots, \hat{f}_L^{mn}\right)^T$ to the vector of Chebyshev coefficients $\hat{\mathbf{g}}^{mn} = (\hat{g}_0^{mn}, \dots, \hat{g}_L^{mn})^T$ is described by the matrix $\mathbf{C}^{mn} \in \mathbb{C}^{(L+1)\times(L-L_0)}$ through*

$$\hat{\mathbf{g}}^{mn} = \mathbf{C}^{mn} \hat{\mathbf{f}}^{mn}.$$

We will refer to the application of the matrix $\mathbf{C}^{mn}$ to a suitably sized vector as the discrete Wigner-d functions transformation *(DWT).*

Let us take a closer look at the DWT. It is well known that polynomials of a certain degree ℓ can be represented by their values in Chebyshev nodes $\tau_{\ell,k} = \cos\left(\frac{2k+1}{2\ell+2}\pi\right)$. We will use these nodes for the

computation of the DWT as we want to have the transition between the two types of basis function in (4.4) independent of the actual source nodes x. We, hence, replace d_ℓ^{mn} by $\left(d_\ell^{mn}(\tau_{\ell,k})\right)_{k=0}^{\ell}$. In terms of matrix-vector notation the basis transformation (4.4) is rewritten as

$$\mathbf{D}^{mn}\hat{\mathbf{f}}^{mn} = \begin{cases} \mathbf{T}\hat{\mathbf{g}}^{mn} & \text{for } m+n \text{ even}, \\ \mathbf{T}\mathbf{S}\hat{\mathbf{g}}^{mn} & \text{for } m+n \text{ odd}, \end{cases}$$

where the vectors $\hat{\mathbf{f}}^{mn} \in \mathbb{C}^{L-L_0+1}$, $\hat{\mathbf{g}}^{mn} \in \mathbb{C}^{L+1}$ contain the SO(3) Fourier coefficients and Chebyshev coefficients, respectively; and we evaluate the occurring polynomials at their Chebyshev nodes by

$$\begin{aligned} \mathbf{T} &= \left(\cos\frac{(2k+1)\ell\pi}{2(L+1)}\right)_{k,\ell=0,\ldots,L}, \quad \mathbf{S} = \operatorname{diag}\left(\sin\frac{(k+1)\pi}{L+2}\right)_{k=0,\ldots,L} \\ \mathbf{D}^{mn} &= \left(d_\ell^{mn}\left(\cos\frac{(2k+1)\pi}{2(L+1)}\right)\right)_{k=0,\ldots,L;\,\ell=L_0,\ldots,L}. \end{aligned}$$

Thus, the whole transforms, i.e., the matrix $\mathbf{C}^{mn}$ given in Definition 4.1.1 can be separated into

$$\mathbf{C}^{mn} = \begin{cases} \mathbf{T}^{-1}\mathbf{D}^{mn} & \text{for } m+n \text{ even}, \\ \mathbf{S}^{-1}\mathbf{T}^{-1}\mathbf{D}^{mn} & \text{for } m+n \text{ odd} \end{cases}$$

where we get by simple computations

$$\begin{aligned} \mathbf{T}^{-1} &= \left(\frac{2-\delta_{0k}}{L+1}\cos\frac{(2\ell+1)k\pi}{2(L+1)}\right)_{k,\ell=0,\ldots,L}, \\ \mathbf{S}^{-1} &= \operatorname{diag}\left(\left(\sin\frac{(k+1)\pi}{L+2}\right)^{-1}\right)_{k=0,\ldots,L}. \end{aligned}$$

The total complexity is determined by the matrix $\mathbf{D}^{mn}$ the application of which to the vector $\hat{\mathbf{f}}^{mn}$ costs $\mathcal{O}(L^2)$ operations. Neither the multiplication by $\mathbf{T}^{-1}$ which is realised by the discrete cosine transform (DCT) from [8] in $\mathcal{O}(L\log L)$ nor the multiplication by the diagonal matrix $\mathbf{S}^{-1}$ in $\mathcal{O}(L)$ steps will increase the asymptotic complexity. In the forthcoming Section 4.1.1, we present an approach that manages to split the matrix $\mathbf{C}^{mn}$ such that the multiplication with the new matrices can be done in as little as $\mathcal{O}(L\log^2 L)$ steps. In Section 4.1.2, we will present another approach with a even more favourable complexity of $\mathcal{O}(L\log L)$.

4.1.1 The Fast Transform of Wigner-d Functions Based on Cascade Summation

Our aim is the efficient computation of matrix-vector multiplications using the matrices $\mathbf{C}^{mn}$ for all $m,n = -L,\ldots,L$. A more detailed description of the algorithm and its implementation, along with extensive numerical tests, can be found in [68] but is summarised here.

The fast polynomial transform, such as the one to be outlined here, has been described in [57] and [23] for the sphere $\mathbb{S}^2$. Our algorithm is a generalisation of the algorithm presented in [57] for the associated Legendre functions on the two-dimensional sphere, in the sense that Wigner-d functions are actually a generalisation of the associated Legendre functions (cf. Section 3.4).

The essential property of Wigner-d functions we are exploiting here is their three-term recurrence

relation given in (3.26). Based on this relation the Clenshaw algorithm [18] can evaluate a linear combination of Wigner-d functions $\sum_{\ell=L_0}^{L} \hat{f}_\ell^{mn} d_\ell^{mn}(x)$ at one node x at a time. This is not favourable in our case as we seek to evaluate this sum for general x. For this, we first need a more general form of the three-term recurrence (3.26). This generalisation is two-fold, it comprises an extension of the three-term recurrence relation to Wigner-d functions for orders $|m|, |n| > \ell$ that have been previously undefined, as well as the possibility to shift the degree ℓ of d_ℓ^{mn} by an arbitrary $c \in \mathbb{N}$, instead of only one step. These steps have been described for the Wigner-d functions in [68]. At first, we modify the three-term recurrence relation by defining the Wigner-d functions for $|m|, |n| > \ell$ as well. Note that this is the important step which allows us to realise the fast Wigner-d transform in a fast and stable way. We obtain for arbitrary $L \in \mathbb{N}$, $m, n = -L, \dots, L$ and $\ell = 0, \dots, L$ the extended three-term recurrence formula

$$d_{\ell+1}^{mn}(x) = (\alpha_\ell^{mn} x + \beta_\ell^{mn}) d_\ell^{mn}(x) + \gamma_\ell^{mn} d_{\ell-1}^{mn}(x), \tag{4.5}$$

where the recurrence coefficients read as

$$\alpha_0^{mn} = \begin{cases} 1 & \text{for } m = n, \\ 0 & \text{for } m + n \text{ odd}, m \neq n, \\ -1 & \text{otherwise}, \end{cases} \quad \alpha_\ell^{mn} = \begin{cases} (-1)^{m+n+1} & \text{for } 0 < \ell \leqslant L_0 - \min(|m|, |n|), \\ \frac{mn}{|mn|} & \text{for } L_0 - \min(|m|, |n|) < \ell < L_0, \\ u_\ell^{mn} & \text{for } L_0 \leqslant \ell, \end{cases}$$

$$\beta_\ell^{mn} = \begin{cases} 1 & \text{for } 0 \leqslant \ell < L_0, \\ 0 & \text{for } m = n = 0, \\ v_\ell^{mn} & \text{otherwise} \end{cases} \quad \text{and} \quad \gamma_\ell^{mn} = \begin{cases} 0 & \text{for } \ell \leqslant L_0, \\ w_\ell^{mn} & \text{otherwise}, \end{cases}$$

using u_ℓ^{mn}, v_ℓ^{mn} and w_ℓ^{mn}, the recurrence coefficients from (3.26). To start the recurrence for all $m, n = -L, \dots, L$, we set $d_{-1}^{mn} = 0$; and with $\lambda_{mn} = \frac{\sqrt{(2\min(|m|,|n|))!}}{2^{\min(|m|,|n|)} \min(|m|,|n|)!}$, we obtain

$$d_0^{mn}(x) = \begin{cases} \lambda_{mn} & \text{for } m + n \text{ even}, \\ \lambda_{mn}\sqrt{1 - x^2} & \text{for } m + n \text{ odd}. \end{cases}$$

For the second modification of the three-term recurrence relation, we introduce associated Wigner-d functions $d_\ell^{mn}(\cdot, c)$ with a shift parameter $c \in \mathbb{N}$ by

$$\begin{aligned} d_{-1}^{mn}(x, c) &= 0, \quad d_0^{mn}(x, c) = 1, \\ d_{\ell+1}^{mn}(x, c) &= (\alpha_{\ell+c}^{mn} x + \beta_{\ell+c}^{mn}) d_\ell^{mn}(x, c) + \gamma_{\ell+c}^{mn} d_{\ell-1}^{mn}(x, c). \end{aligned}$$

One can prove that the sequence $\left\{ d_\ell^{mn}(x, c) \right\}_{\ell \in \mathbb{N}}$ is again orthogonal, cf. [88, Sect. 2].
Instead of only one step as in the modified recurrence (4.5), we will now shift the degree ℓ of d_ℓ^{mn} by $c \in \mathbb{N}$ steps by

$$d_{\ell+c}^{mn}(x) = d_c^{mn}(x, \ell) d_\ell^{mn}(x) + \gamma_\ell^{mn} d_{c-1}^{mn}(x, \ell + 1) d_{\ell-1}^{mn}(x).$$

This alone does not yet yield a faster algorithm. A serious drawback is that we still can only compute the recurrence for one node at a time. Instead, we would like to make the computations node-independent, i.e., compute with functions instead of numbers.
We know that Wigner-d functions d_ℓ^{mn} are polynomials of either degree ℓ or $\ell - 1$ if we neglect

the occasionally occurring term $\sqrt{1-x^2}$. In fact we will not treat this term in the transformation of Wigner-d functions but will include it later on the transformation from Chebyshev coefficients to standard Fourier coefficients. It is well known that such polynomials can be represented by their Chebyshev expansion

$$d_\ell^{mn} = \sum_{k=0}^{\ell} \alpha_k T_k, \qquad\qquad d_\ell^{mn} \sim (\alpha_k)_{k=0}^{\ell}$$

or, as previously mentioned, by their values at Chebyshev nodes $\tau_{\ell,k}$ with

$$d_\ell^{mn}(\tau_{\ell,k}) = d_\ell^{mn}\Big(\cos\Big(\tfrac{2k+1}{2\ell+2}\pi\Big)\Big), \qquad d_\ell^{mn} \sim \big(d_\ell^{mn}(\tau_{\ell,k})\big)_{k=0}^{\ell}.$$

Switching between both representations allows us to find an efficient algorithm to compute the Chebyshev representation of the linear combination $\sum_{\ell=L_0}^{L} \hat{f}_\ell^{mn} d_\ell^{mn}(x) = \sum_{\ell=0}^{L} \hat{g}_\ell^{mn} T_\ell(x)$.

A basic step includes the treatment of the coefficients $\hat{f}_\ell^{mn}$, not as numbers, but as polynomials of degree zero. On them, we can employ the generalised Clenshaw procedure using the new three-term recurrence.

The sum over the Wigner-d functions is split into consecutive blocks of four summands, of which the respective Chebyshev expansion is known. To combine the four summands in each block, we make use of the discrete cosine transforms, DCT-III and DCT-II, which are an efficient way to convert functions in Chebyshev representation to their representation in Chebyshev nodes and back, see [8]. The exact details on the theoretical background of this step can be found in [89]. After this, we get new blocks, this time containing polynomials of degree at most 3, of which we just computed the Chebyshev expansion. This step is repeated until we are left with one single block of one single polynomial of degree at most ℓ. A graphic interpretation of this would lead to a cascade which becomes smaller towards the end. The original algorithm [23] described the transposed version of this transform, i.e., the transition from Chebyshev expansions to an expansion in other polynomials. Hence, it had a "classical", growing cascade which initially led to the name *cascade summation*.

The described cascade has a depth of $\mathcal{O}(\log L)$ layers. On each layer, except for the first and last, we perform a DCT which takes $\mathcal{O}(L \log L)$ operations, yielding a total $\mathcal{O}(L \log^2 L)$ for the whole computation.

This complexity is already an improvement over the one of $\mathcal{O}(L^2)$ the DWT possesses. This gives reason to call the described algorithm the *fast transformation of Wigner-d functions* (FWT). We specify FWT-C (for cascade summation), not to confound this methods with the one from Section 4.1.2. But we encounter a new problem here: numerical instabilities. When using exact arithmetic the FWT-C is exact. But when computing in finite precision, small errors, introduced by the DCT algorithms in the cascade, cause numerical instabilities (cf. [72]) when they are multiplied with large function values of the associated Wigner-d functions $d_c^{mn}(x,\ell)$. Those occur for function values $|x| \approx 1$ at certain admissible triples $(c,m,n) \in \mathcal{I}_L$. This already happens for comparatively small sizes of $\ell > 12$.

An effective approach to improve the stability of this summation has been developed in [24, 72]. The authors identify unstable multiplication steps in the cascade and replace them by a stabilisation step whenever the functions to be transformed exceed a certain threshold κ at $|x| \approx 1$. This corresponds to removing the critical polynomial from the cascade, dealing with it separately and inserting it again at the very end, on the last layer of the cascade. In a scenario where every single polynomial is removed from the cascade, we would get back exactly the slow DWT algorithm.

The algorithmic details on this stabilisation method, as well as its implementation for associated Legendre functions, can be found in [57] and [49], respectively. This method has been directly applied to and implemented for the Wigner-d functions in [68].
Although we now have a method to improve the stability of our computation, its application will increase the runtime of the algorithm. Each stabilisation step will cost $\mathcal{O}(L \log L)$ operations. That means, if we need for than $\mathcal{O}(\log L)$ of them, the FWT-C will no longer be a $\mathcal{O}(L \log^2 L)$ algorithm.
So far, we do not know an upper bound for the number of stabilisation steps with respect to the bandwidth L for a given threshold κ and thus the true asymptotical complexity of the stabilised FWT-C. However, the numerical experiments conducted in [68] support the conjecture that, although, the stabilised FWT is slower than the unstabilised version with $\mathcal{O}(L \log^2 L)$ flops, it is still asymptotically faster than the DWT with $O(L^2)$ flops.
Apart from this, the numerical stability of this method has neither been proven. It is just more stable than the unstabilised version. These drawbacks of the FWT-C lead to a new algorithm which will be presented in the next section. It will overcome the problems of the stabilisation and yield a more favourable complexity of $\mathcal{O}(L \log L)$ instead of $\mathcal{O}(L \log^2 L)$.

4.1.2 The Fast Transform of Wigner-d Functions Based on Semiseparable Matrices

The algorithm we present in this section is, like the cascade summation in Section 4.1.1 an approximate technique the complexity of which increases with increase of accuracy. Furthermore, it is a generalisation of the algorithm proposed in [77] for spherical harmonic expansions on the two-sphere. Their technique is based on an efficient algorithm to compute matrix-vector multiplications of semiseparable matrices. An application of this technique to Gegenbauer polynomials has been analysed in [48] already.
Again, we will show how to exchange expansions in Wigner-d functions with expansions in Chebyshev polynomials of first kind. Essential to this are the transformation of arbitrary Wigner-d functions into those of low orders and the application of a differential operator to the Wigner-d functions. This will yield specially structured matrices $\mathbf{G}^{m,n}$ that are semiseparable. Applying a divide-and-conquer-type algorithm from [16] with a minor extension to our matrices of semiseparability rank one, we compute the eigendecomposition of these matrices. The algorithm, called the *fast transformation of Wigner-d functions based on semiseparable matrices* (FWT-S), has complexity $\mathcal{O}(L \log L)$ if applied to a $L \times L$ matrix.
Again, we are given a set of orders m and n and coefficients $\hat{f}_\ell^{mn}$, and we seek to find the coefficients $\hat{g}_\ell^{mn}$ in

$$\sum_{\ell=L_0}^{L} \hat{f}_\ell^{mn} d_\ell^{mn}(x) = \begin{cases} \sum_{\ell=0}^{L} \hat{g}_\ell^{mn} T_\ell(x) & \text{if } m+n \text{ even}, \\ \sum_{\ell=0}^{L-1} \hat{g}_\ell^{mn} \sqrt{1-x^2}\, T_\ell(x) & \text{if } m+n \text{ odd}. \end{cases}$$

But this time we introduce an intermediate step in this transform:

1. Starting from the $SO(3)$ Fourier coefficients $\hat{f}_\ell^{mn}$ for $|m|, |n| > \ell$, we consider the transform

$$\sum_{\ell=L_0}^{L} \hat{f}_\ell^{mn} d_\ell^{mn}(x) = \sum_{\ell=L_0'}^{L} \bar{f}_\ell^{mn} d_\ell^{m'n'}(x), \tag{4.6}$$

where m', n' are those low orders that have been defined in Lemma 3.3.3. Recall also Equation (3.28) which defined the matrix $\mathbf{A}^{mn}$ that realised the change of basis between Wigner-d functions and Wigner-d functions of low orders. That is, $\mathbf{A}^{mn} \in \mathbb{R}^{(L-L_0'+1)\times(L-L_0+1)}$ satisfied

$$\bar{\mathbf{f}}^{mn} = \mathbf{A}^{mn}\hat{\mathbf{f}}^{mn}$$

for $\hat{\mathbf{f}}^{mn} = (\hat{f}_\ell) \in \mathbb{C}^{L-L_0+1}$ and $\bar{\mathbf{f}}^{mn} = (\bar{f}_\ell) \in \mathbb{C}^{L-L_0'+1}$.

2. The next step is to replace the Wigner-d functions $d_\ell^{m'n'}$ by Chebyshev polynomials of first kind T_ℓ, in order to get

$$\sum_{\ell=L_0'}^{L} \bar{f}_\ell^{mn} d_\ell^{m'n'}(x) = \begin{cases} \sum_{\ell=0}^{L} \hat{g}_\ell^{mn} T_\ell(x) & \text{if } m+n \text{ even,} \\ \sum_{\ell=0}^{L-1} \hat{g}_\ell^{mn} \sqrt{1-x^2}\, T_\ell(x) & \text{if } m+n \text{ odd.} \end{cases} \tag{4.7}$$

For the matrix-vector notation, we introduce the matrices $\mathbf{B}^{mn} \in \mathbb{R}^{(L+1)\times(L-L_0'+1)}$, for fixed $m, n = -L, \dots, L$. They realise the change of basis in (4.7), i.e., they satisfy

$$\hat{\mathbf{g}}^{mn} = \mathbf{B}^{mn}\bar{\mathbf{f}}^{mn},$$

with $\hat{\mathbf{g}}^{mn} = (\hat{g}_\ell^{mn}) \in \mathbb{C}^{L+1}$. An explicit formula for the matrices $\mathbf{B}^{mn}$ will be given in Lemma 4.1.6.

Step 1: Computing the Matrices $\mathbf{A}^{mn}$: Let $L \in \mathbb{N}$ and the two orders m, n with $|m|, |n| \leqslant L$ be given. On account of Corollary 3.3.5, we can restrict the cases of orders we need to consider. In fact, it is enough to assume $0 \leqslant m \leqslant L$, and $0 \leqslant n \leqslant m$. All remaining cases can be reduced to these cases by using one or more of the respective symmetries from Corollary 3.3.5. Figure 4.3 illustrates this fact.

Moreover, we will always compute with normalised Wigner-d functions $\tilde{d}_\ell^{mn} = \sqrt{\frac{2\ell+1}{2}} d_\ell^{mn}$ in this section. While for the FWT-C the key property was the three-term recurrence relation of the Wigner-d functions, we now revert to the differential equation from which the Wigner-d function arose. Putting together (3.20) and (3.21), the Wigner-d functions are eigenfunctions of the differential operator

$$\mathcal{D}^{mn}(y) = -(1-x^2)y'' + 2xy' + \left(\frac{|n-m|^2}{2(1-x)} + \frac{|m+n|^2}{2(1+x)}\right) y \tag{4.8}$$

to the eigenvalues $\lambda_\ell = \ell(\ell+1)$.

If we compare the differential operator $\mathcal{D}^{mn}$ for arbitrary orders and the differential operator $\mathcal{D}^{m'n'}$ for the low orders from Lemma 3.3.3, we encounter a similarity between them. To be more precise, we have

$$\mathcal{D}^{mn} - \mathcal{D}^{m'n'} = \mathcal{D}^- + \mathcal{D}^+,$$

with the operators $\mathcal{D}^-$ and $\mathcal{D}^+$ defined by

$$\mathcal{D}^-(y) = \frac{|n-m|^2 - |n'-m'|^2}{2(1-x)} y, \qquad \mathcal{D}^+(y) = \frac{|n+m|^2 - |n'+m'|^2}{2(1+x)} y.$$

Both operators are of much simpler structure than $\mathcal{D}^{m,n}$. Note that although $\mathcal{D}^-$ and $\mathcal{D}^+$ depend on m and n we omit their indices here to improve readability.

Using the differential operator $\mathcal{D}^{mn}$, we define a certain matrix.

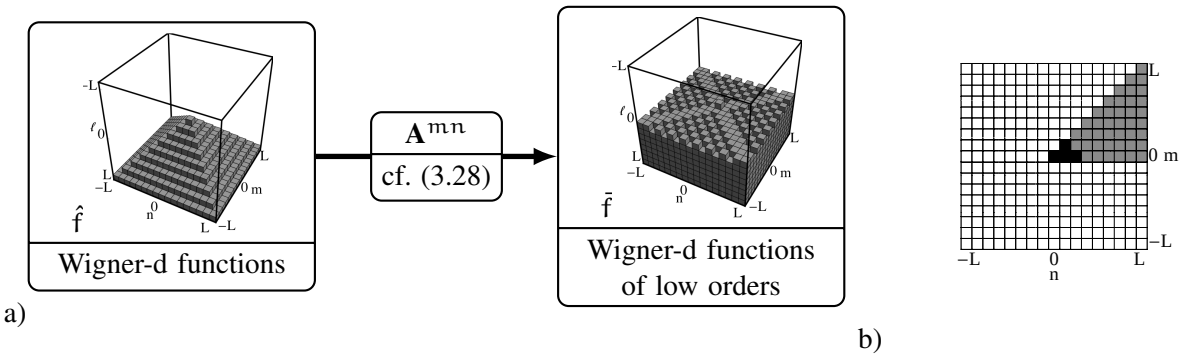


Figure 4.3: The figure depicts the transform between expansions in Wigner-d functions with maximum degree L and Wigner-d functions of low orders. Again, input and output coefficients of orders m and n, and degree ℓ of the transform are represented by the cubes with coordinates (ℓ, m, n). In this step of the transform, the matrices $\mathbf{A}^{mn}$ for $m, n = -L, \dots, L$ realise the change of basis.
In b), we see the reduction of relevant matrices $\mathbf{A}^{mn}$ (grey squares) due to the symmetries from Corollary 3.3.5. If moreover $m + n \leqslant 2$ holds true, the matrix $\mathbf{A}^{mn}$ is the identity matrix (black squares).

Definition 4.1.2. *The matrix* $\mathbf{G}^{mn} = (g_{\ell,k}) \in \mathbb{R}^{(L-L_0'+1)\times(L-L_{L_0'}+1)}$ *is defined by*

$$g_{\ell,k} = \langle \tilde{d}_\ell^{m'n'}, \mathcal{D}^{mn}(\tilde{d}_k^{m'n'}) \rangle.$$

This will be very helpful in the development of our fast algorithm in the sense that, on one hand, the matrix $\mathbf{A}^{mn}$ comprises the eigenvectors of the matrix $\mathbf{G}^{mn}$, and on the other hand, $\mathbf{G}^{mn}$ is a specially structured matrix on which we can apply some well-known fast algorithms.

Lemma 4.1.3. *Let* $\mathbf{a}_k = (a_{L_0',k}, \dots, a_{L,k})^T \in \mathbb{R}^{L-L_0'+1}$, $L_0' \leqslant k \leqslant L$, *be the* $(k - L_0 + 1)$*st column of the matrix* $\mathbf{A}^{mn}$. *Then* $\mathbf{a}_k$ *is a normalised eigenvector of the matrix* $\mathbf{G}^{mn}$ *to the eigenvalue* $\lambda_k = k(k+1)$.

Proof. Recall that

$$\tilde{d}_k^{mn} = \sum_{\ell=L_0'}^{L} \langle \tilde{d}_\ell^{m'n'}, \tilde{d}_k^{mn} \rangle \tilde{d}_\ell^{m'n'} = \sum_{\ell=L_0'}^{L} a_{\ell,k} \tilde{d}_\ell^{m'n'}.$$

Consider the $(\ell - L_0' + 1)$st row of the matrix-vector product $\mathbf{G}^{mn}\,\mathbf{a}_k$ for $L_0' \leqslant \ell \leqslant L$ which satisfies

$$\begin{aligned}
\sum_{j=L_0'}^{L} g_{\ell,j} a_{j,k} &= \sum_{j=L_0'}^{L} \langle \tilde{d}_\ell^{m'n'}, \mathcal{D}^{mn}(\tilde{d}_j^{m'n'}) \rangle a_{j,k} = \langle \tilde{d}_\ell^{m'n'}, \mathcal{D}^{mn} \left(\sum_{j=L_0'}^{L} a_{j,k} \tilde{d}_j^{m'n'} \right) \rangle \\
&= \langle \tilde{d}_\ell^{m'n'}, \mathcal{D}^{mn}(\tilde{d}_k^{mn}) \rangle = \langle \tilde{d}_\ell^{m'n'}, \lambda_k \tilde{d}_k^{mn} \rangle = \langle \tilde{d}_\ell^{m'n'}, \lambda_k \sum_{j=L_0'}^{L} a_{j,k} \tilde{d}_j^{m'n'} \rangle \\
&= \lambda_k \sum_{j=L_0'}^{L} \langle \tilde{d}_\ell^{m'n'}, \tilde{d}_j^{m'n'} \rangle a_{j,k} = \lambda_k a_{\ell,k}.
\end{aligned}$$

The vectors $\mathbf{a}_k$ are automatically normalised due to the use of the normalised Wigner-d functions. ∎

Now that we know the connection between the matrices $\mathbf{A}^{mn}$ and $\mathbf{G}^{mn}$, we move on examining the structure of $\mathbf{G}^{mn}$. The matrix $\mathbf{G}^{m,n}$ can be split into a sum of three matrices, each of which we can analyse separately. This is achieved by using the operators $\mathcal{D}^-$ and $\mathcal{D}^+$ to rewrite the entries $g_{\ell,k}$ of $\mathbf{G}^{m,n}$ as

$$g_{\ell,k} = \langle \tilde{d}_\ell^{m'n'}, \mathcal{D}^{m'n'}(\tilde{d}_k^{m'n'})\rangle + \langle \tilde{d}_\ell^{m'n'}, \mathcal{D}^{-}(\tilde{d}_k^{m'n'})\rangle + \langle \tilde{d}_\ell^{m'n'}, \mathcal{D}^{+}(\tilde{d}_k^{m'n'})\rangle.$$

This leads to a representation of the matrix $\mathbf{G}^{m,n}$ in terms of a sum of three $(L-L_0'+1)\times(L-L_0'+1)$ matrices, $\mathbf{G}^{mn} = \mathbf{D} + \mathbf{S}^- + \mathbf{S}^+$ with

$$\begin{aligned} \mathbf{D} &= (d_{\ell,k}), \quad d_{\ell,k} = \langle \mathcal{D}^{m'n'}(\tilde{d}_k^{m'n'}), \tilde{d}_\ell^{m'n'}\rangle, \\ \mathbf{S}^- &= (s^-_{\ell,k}), \quad s^-_{\ell,k} = \langle \mathcal{D}^{-}(\tilde{d}_k^{m'n'}), \tilde{d}_\ell^{m'n'}\rangle \\ \mathbf{S}^+ &= (s^+_{\ell,k}), \quad s^+_{\ell,k} = \langle \mathcal{D}^{+}(\tilde{d}_k^{m'n'}), \tilde{d}_\ell^{m'n'}\rangle. \end{aligned} \tag{4.9}$$

The matrix $\mathbf{D}$ is actually a diagonal one as $\tilde{d}_k^{m'n'}$ is an eigenfunction of the differential operator $\mathcal{D}^{m'n'}$ to the eigenvalue $\lambda_k = k(k+1)$, and we, therefore, obtain

$$d_{\ell,k} = \langle \mathcal{D}^{m'n'}(\tilde{d}_k^{m'n'}), \tilde{d}_\ell^{m'n'}\rangle = \langle \lambda_k \tilde{d}_k^{m'n'}, \tilde{d}_\ell^{m'n'}\rangle = \lambda_k \delta_{\ell,k}.$$

While the matrices $\mathbf{S}^-$ and $\mathbf{S}^+$ do not have such a nice structure as $\mathbf{D}$, their entries can also be given explicitly. The quite extensive derivation of these formulae can be found in [52, Lemma 4.7]. We omit this here, mainly due to the fact that Section 5.1 features a new proof of the corresponding formulae for Wigner-d functions in half-integer orders (cf. Lemma 5.1.6) which follows a similar procedure as the one in this case, but can not be found elsewhere. The entries of the matrices $\mathbf{S}^-$ and $\mathbf{S}^+$ are given by

$$\begin{aligned} s^-_{\ell,k} &= \int_{-1}^{1} \tilde{d}_k^{m'n'}(x)\frac{|n-m|^2-|n'-m'|^2}{2(1-x)}\tilde{d}_\ell^{m'n'}(x)\,dx &&= (|n-m|^2-|n'-m'|^2)h_{\tilde{\ell},\tilde{k}} \\ s^+_{\ell,k} &= \int_{-1}^{1} \tilde{d}_k^{m'n'}(x)\frac{|n+m|^2-|m'+n'|^2}{2(1+x)}\tilde{d}_\ell^{m'n'}(x)\,dx &&= (-1)^{\ell+k}(|n+m|^2-|n'+m'|^2)h_{\tilde{\ell},\tilde{k}} \end{aligned}$$

with $\tilde{\ell} = \min\{\ell,k\}$, $\tilde{k} = \max\{\ell,k\}$ and

$$h_{\ell,k} = \frac{\sqrt{(\ell+\frac{1}{2})(k+\frac{1}{2})}}{4}\begin{cases} \dfrac{\ell(\ell+1)}{k(k+1)} & \text{if } m = \pm n, \\ \sqrt{\dfrac{(\ell-1)\ell(\ell+1)(\ell+2)}{(k-1)k(k+1)(k+2)}} & \text{if } m+n \text{ even}, \\ 2\sqrt{\dfrac{\ell(\ell+1)}{k(k+1)}} & \text{if } m+n \text{ odd}. \end{cases}$$

Now we are ready to examine the special structure of the matrix $\mathbf{G}^{m,n}$ a bit closer.

Definition 4.1.4 (Semiseparable Matrix)**.** *An $n\times n$ matrix $\mathbf{G}$ is called* semiseparable of rank R*, if there exist vectors $\mathbf{d}, \mathbf{u}_r, \mathbf{v}_r \in \mathbb{R}^n$, $r = 1,\dots,R$, such that*

$$\mathbf{G} = \operatorname{diag}(\mathbf{d}) + \sum_{r=1}^{R} \left(\operatorname{triu}\left(\mathbf{u}_r\,\mathbf{v}_r^T\right) + \operatorname{tril}\left(\mathbf{v}_r\,\mathbf{u}_r^T\right)\right).$$

Here, $\operatorname{diag}(\mathbf{d})$ *denotes the diagonal matrix with the entries of the vector $\mathbf{d}$ on its diagonal. Furthermore,* $\operatorname{triu}\left(\mathbf{u}_r\,\mathbf{v}_r^T\right)$ *and* $\operatorname{tril}\left(\mathbf{v}_r\,\mathbf{u}_r^T\right)$ *are the strictly upper and lower triangular parts of the rank-one matrices $\mathbf{u}_r\,\mathbf{v}_r^T$ and $\mathbf{v}_r\,\mathbf{u}_r^T$, respectively; see [83].*

Lemma 4.1.5. *The matrix* $\mathbf{G}^{mn}$ *from Definition 4.1.2 is semiseparable with semiseparability rank* $R = 2$*, i.e.,*

$$\mathbf{G}^{mn} = \operatorname{diag}(\mathbf{d}) + \operatorname{triu}\left(\mathbf{u}_1 \mathbf{v}_1^T\right) + \operatorname{tril}\left(\mathbf{v}_1 \mathbf{u}_1^T\right) + \operatorname{triu}\left(\mathbf{u}_2 \mathbf{v}_2^T\right) + \operatorname{tril}\left(\mathbf{v}_2 \mathbf{u}_2^T\right), \tag{4.10}$$

with $\mathbf{d} = (d_\ell)$, $\mathbf{u}_1 = \left(u_\ell^1\right)$, $\mathbf{v}_1 = \left(v_\ell^1\right)$, $\mathbf{u}_2 = \left(u_\ell^2\right)$, $\mathbf{v}_2 = \left(v_\ell^2\right) \in \mathbb{R}^{L-L_0'+1}$*, for* $\ell = L_0', \ldots, L$*, where*

$$d_\ell = \ell(\ell+1) + \left(|n-m|^2 - |n'-m'|^2 + |n+m|^2 - |n'+m'|^2\right)\frac{\gamma^2(2\ell+1)}{8},$$

$$u_\ell^1 = w_\ell^1 w_\ell, \quad v_\ell^1 = w_\ell^1 w_\ell^{-1}, \quad u_\ell^2 = w_\ell^2 w_\ell, \quad v_\ell^2 = w_\ell^2 w_\ell^{-1},$$

$$w_\ell^1 = \frac{\gamma}{2}\sqrt{(|n-m|^2 - |n'-m'|^2)(\ell+\tfrac{1}{2})}, \quad w_\ell^2 = (-1)^\ell \frac{\gamma}{2}\sqrt{(|n+m|^2 - |n'+m'|^2)(\ell+\tfrac{1}{2})},$$

$$w_\ell = \begin{cases} \ell(\ell+1) & \text{if } m = \pm n, \\ \sqrt{(\ell-1)\ell(\ell+1)(\ell+2)} & \text{if } m+n \text{ even}, \\ \sqrt{\ell(\ell+1)} & \text{if } m+n \text{ odd}, \end{cases} \quad \gamma = \begin{cases} \sqrt{2} & \text{if } m+n \text{ odd}, \\ 1 & \text{else}. \end{cases}$$

Proof. Using the decomposition of matrices $\mathbf{G}^{mn}$ from (4.9), we get

$$\mathbf{S}^- = +\operatorname{triu}\left(\mathbf{u}_1 \mathbf{v}_1^T\right) + \operatorname{tril}\left(\mathbf{v}_1 \mathbf{u}_1^T\right) \text{ and } \mathbf{S}^+ = \operatorname{triu}\left(\mathbf{u}_2 \mathbf{v}_2^T\right) + \operatorname{tril}\left(\mathbf{v}_2 \mathbf{u}_2^T\right).$$

By separating the coefficients $h_{\ell,k}$ into a product of a term depending only on ℓ and a term depending only on k, we obtain the proposed coefficients u_ℓ^1 and v_ℓ^1 for the matrix entries $s_{\ell,k}^-$ and u_ℓ^2 and v_ℓ^2 for the matrix entries $s_{\ell,k}^+$. The proof follows by straightforward computations. ∎

Let us now consider some special cases of the matrix $\mathbf{G}^{mn}$. If $|m-n| \leqslant 2$ or $|m+n| \leqslant 2$, $\mathbf{G}^{mn}$ becomes semiseparable with rank one as either $|m+n| - |m'+n'| = 0$ or $|m-n| - |m'-n'| = 0$ is fulfilled. This will eliminate one semiseparable part from the matrix $\mathbf{G}^{mn}$ in Lemma 4.1.5.
If $m = 0$ or $n = 0$, we have $|m-n| = |m+n|$, and $|m'-n'| = |m'+n'|$, and therefore, $w_\ell^1 = (-1)^\ell w_\ell^2$. As a consequence, each element $g_{\ell,k}$ from $\mathbf{G}$ where $\ell+k$ is odd is eliminated. This checker board structured matrix $\mathbf{G}^{mn}$ can be rearranged such that the resulting matrix is block-diagonal with two blocks, $\mathbf{G}_e$, and $\mathbf{G}_o$, where each is a symmetric semiseparable matrix of semiseparability rank one. Note that these are the transforms that are applied to the Wigner-d function $\tilde{d}_\ell^{m0}$ for $m \leqslant \ell$ which are by (3.30) the normalised associated Legendre functions. An algorithm for the fast evaluation of associated Legendre functions, based on semiseparable matrices of rank one, has been described in [77].

Fast Algorithms for Semiseparable Matrices Now we know that the matrices $\mathbf{G}^{mn}$ are semiseparable either of rank two or one and that the matrices $\mathbf{A}^{mn}$ from (3.28) contain the eigenvectors of the respective $\mathbf{G}^{mn}$. But so far we did not discuss how such eigenvector matrix of a semiseparable matrix can be efficiently applied to a vector. A method to do this for matrices with semiseparability rank one is described in [16]. With a minor extension the same method can be applied to our matrices with semiseparability rank two, too. We give a very brief outline of this method and refer the reader to [16] for more specific details on the implementation.
The main idea for the divide-and-conquer approach of the algorithm is the fact that the matrix $\mathbf{G}^{mn}$ can be written as

$$\mathbf{G}^{mn} = \begin{pmatrix} \mathbf{G}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{G}_2 \end{pmatrix} + \rho_1 \mathbf{w}_1 \mathbf{w}_1^T + \rho_2 \mathbf{w}_2 \mathbf{w}_2^T,$$

where $\rho_1, \rho_2 = \pm 1$ are freely chosen scalars, $\mathbf{w}_1, \mathbf{w}_2 \in \mathbb{R}^n$, and $\mathbf{G}_1$ and $\mathbf{G}_2$ are matrices of the same type as $\mathbf{G}^{mn}$ and may, therefore, be decomposed recursively in the same manner as $\mathbf{G}^{mn}$. This is repeated until we can directly compute the eigendecomposition of the matrices. A proof of this can be found in [52, Sec. 4.1.2].
After the divide phase, the eigendecompositions of those smaller matrices are combined, in the conquer phase, to eigendecompositions of the next bigger matrices, following the decomposition tree, until the sought eigendecomposition of the original matrix is obtained. So, suppose that we have computed the eigendecompositions $\mathbf{G}_j = \mathbf{A}_j \Lambda_j \mathbf{A}_j^T$, with diagonal eigenvalue matrices Λ_j, and orthogonal eigenvector matrices $\mathbf{A}_j$ for $j = 1, 2$. For the matrix $\mathbf{G}^{mn}$, this implies the representation

$$\mathbf{G}^{mn} = \begin{pmatrix} \mathbf{A}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_2 \end{pmatrix} \left(\Lambda + \rho_1 \mathbf{z}_1 \mathbf{z}_1^T + \rho_2 \mathbf{z}_2 \mathbf{z}_2^T \right) \begin{pmatrix} \mathbf{A}_1^T & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_2^T \end{pmatrix},$$

with

$$\Lambda = \begin{pmatrix} \Lambda_1 & \mathbf{0} \\ \mathbf{0} & \Lambda_2 \end{pmatrix}, \qquad \mathbf{z}_j = \begin{pmatrix} \mathbf{A}_1^T & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_2^T \end{pmatrix} \mathbf{w}_j, \quad j = 1, 2.$$

From this representation, it takes two steps to obtain the eigendecomposition of the matrix $\mathbf{G}^{mn}$; the solution of two rank-one modified diagonal eigenproblems. In the original algorithm from [16] only one step is necessary. In this sense, our strategy is a generalisation of the original algorithm, but a minor one as all its properties are preserved.
This way, we find

$$\left(\Lambda + \rho_1 \mathbf{z}_1 \mathbf{z}_1^T + \rho_2 \mathbf{z}_2 \mathbf{z}_2^T \right) = \tilde{\mathbf{A}}_1 \left(\tilde{\Lambda}_1 + \rho_2 \tilde{\mathbf{z}}_2 \tilde{\mathbf{z}}_2^T \right) \tilde{\mathbf{A}}_1^T = \tilde{\mathbf{A}}_1 \left(\tilde{\mathbf{A}}_2 \tilde{\Lambda}_2 \tilde{\mathbf{A}}_2^T \right) \tilde{\mathbf{A}}_1^T$$

with $\tilde{\mathbf{z}_2} = \tilde{\mathbf{A}}_2^T \mathbf{z_2}$. The eigenvector matrix $\mathbf{A}^{mn}$ of $\mathbf{G}^{mn}$ is, therefore, given by

$$\mathbf{A}^{mn} = \begin{pmatrix} \mathbf{A}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_2 \end{pmatrix} \tilde{\mathbf{A}}_1 \tilde{\mathbf{A}}_2.$$

It is important to note that we do not compute the $\mathbf{A}^{mn}$ explicitly, but only its effect on a vector, as the corresponding eigenvector matrices $\tilde{\mathbf{A}}_1$ and $\tilde{\mathbf{A}}_2$ can be efficiently applied with an approximate algorithm at linear cost.
An analysis of the methods, found in [16], shows that if $\mathbf{A}^{mn}$ is an $L \times L$ matrix, it can be applied to any vector at cost $\mathcal{O}(L \log L)$. It is important to note that the resulting algorithm also includes a pre-computation part which computes $\mathcal{O}(L \log L)$ values in advance.

Step 2: Computing the Matrices $\mathbf{B}^{mn}$: The next step is to replace the Wigner-d functions $d_\ell^{m'n'}$ by Chebyshev polynomials of first kind T_ℓ (cf. Figure 4.4). Here the main idea is to exploit the relation between Wigner-d functions and Jacobi polynomials (3.24). As we reduced the occurring orders of Wigner-d functions to the low coefficients m' and n' from Lemma 3.3.3, we only need to consider five cases, namely,

$$\tilde{d}_\ell^{0,0} = \sqrt{\frac{2\ell+1}{2}} P_\ell^{(0,0)}, \qquad \tilde{d}_\ell^{0,2} = b_{\ell,\ell}^2 P_\ell^{(1,1)} + b_{\ell-2,\ell}^2 P_{\ell-2}^{(1,1)},$$

$$\tilde{d}_\ell^{0,1} = \sqrt{1-x^2} - \frac{1}{2}\sqrt{\frac{2\ell+1}{2}}\sqrt{\frac{\ell+1}{\ell}} P_{\ell-1}^{(1,1)}, \quad \tilde{d}_\ell^{1,1} = b_{\ell,\ell}^+ P_\ell^{(0,1)} + b_{\ell-1,\ell}^+ P_{\ell-1}^{(0,1)},$$

$$\tilde{d}_\ell^{1,-1} = b_{\ell,\ell}^- P_\ell^{(1,0)} + b_{\ell-1,\ell}^- P_{\ell-1}^{(1,0)}$$

with

$$b^{+}_{k,\ell} = \frac{1}{\sqrt{2(2\ell+1)}} \begin{cases} \ell & \text{if } k = \ell, \\ \ell+1 & \text{if } k = \ell-1, \end{cases}$$

$$b^{-}_{k,\ell} = \frac{1}{\sqrt{2(2\ell+1)}} \begin{cases} -\ell & \text{if } k = \ell, \\ \ell+1 & \text{if } k = \ell-1, \end{cases}$$

$$b^{2}_{k,\ell} = \frac{1}{2\sqrt{2(2\ell+1)}} \sqrt{\frac{(\ell+1)(\ell+2)}{(\ell-1)\ell}} \begin{cases} -\dfrac{\ell(\ell-1)}{(\ell+1)} & \text{if } k = \ell, \\ \ell & \text{if } k = \ell-2. \end{cases}$$

These relations are a direct consequence of [1, p.782]. Note that only four different types of the Jacobi polynomials occur that, moreover, have only two different choices for the parameters. Our next aim is to transform them into a sum of Legendre polynomials P_j which satisfy $P_j(x) = P_j^{(0,0)}(x)$. We use the following identities

$$P_\ell^{(0,1)} = \sum_{j=0}^{\ell} c_{j,\ell}^{01} P_j^{(0,0)}, \qquad c_{\ell,j}^{01} = (-1)^{j+\ell} \frac{2j+1}{\ell+1},$$

$$P_\ell^{(1,0)} = \sum_{j=0}^{\ell} c_{j,\ell}^{10} P_j^{(0,0)}, \qquad c_{\ell,j}^{10} = \frac{2j+1}{\ell+1},$$

$$P_\ell^{(1,1)} = \sum_{j=0}^{\ell} c_{j,\ell}^{11} P_j^{(0,0)}, \qquad c_{j,\ell}^{11} = \begin{cases} 0 & \text{if } j+\ell \text{ odd}, \\ \frac{2(2j+1)}{\ell+2} & \text{else}, \end{cases}$$

from [61] and [48]. Having arrived at a representation of the Wigner-d functions in terms of Legendre functions one step remains to be done; the transition to Chebyshev polynomials which is now described by

$$P_\ell^{(0,0)} = \sum_{j=0}^{\ell} d_{j,\ell}^{0} T_j, \qquad d_{j,\ell}^{0} = \frac{2-\delta_{j,0}}{\pi} \frac{\Gamma\left(\frac{\ell-j}{2}+\frac{1}{2}\right)}{\Gamma\left(\frac{\ell-j}{2}+1\right)} \frac{\Gamma\left(\frac{\ell+j}{2}+\frac{1}{2}\right)}{\Gamma\left(\frac{\ell+j}{2}+1\right)}.$$

For a representation of the matrices $\mathbf{B}^{m,n}$, $m,n = -L,\dots,L$, we can now state the following lemma.

Lemma 4.1.6. *The matrices $\mathbf{B}^{mn}$, realising the change of basis from* (4.7)*, are given by*

$$\mathbf{B}^{m,n} = \begin{cases} \mathbf{D}^0\,\mathbf{B}^0 & \text{if } m' = n' = 0, \\ \mathbf{D}^0\,\mathbf{C}^{11}\,\mathbf{B}^1 & \text{if } m' = 0,\ n' = 1, \\ \mathbf{D}^0\,\mathbf{C}^{11}\,\mathbf{B}^2 & \text{if } m' = 0,\ n' = 2, \\ \mathbf{D}^0\,\mathbf{C}^{01}\,\mathbf{B}^+ & \text{if } m' = 1,\ n' = 1, \\ \mathbf{D}^0\,\mathbf{C}^{10}\,\mathbf{B}^- & \text{if } m' = 1,\ n' = -1, \end{cases}$$

where the corresponding matrices are defined as

$$\mathbf{B}^0 \in \mathbb{R}^{(L+1)\times(L+1)}, \quad \mathbf{B}^1 \in \mathbb{R}^{L\times L}, \quad \mathbf{B}^2 \in \mathbb{R}^{(L+1)\times(L-1)}, \quad \mathbf{B}^{\pm} \in \mathbb{R}^{(L+1)\times L},$$
$$\mathbf{C}^{01} \in \mathbb{R}^{(L+1)\times(L+1)}, \quad \mathbf{C}^{10} \in \mathbb{R}^{(L+1)\times(L+1)}, \quad \mathbf{C}^{11} \in \mathbb{R}^{(L+1)\times(L+1)}, \quad \mathbf{D}^1 \in \mathbb{R}^{(L+1)\times(L+1)},$$

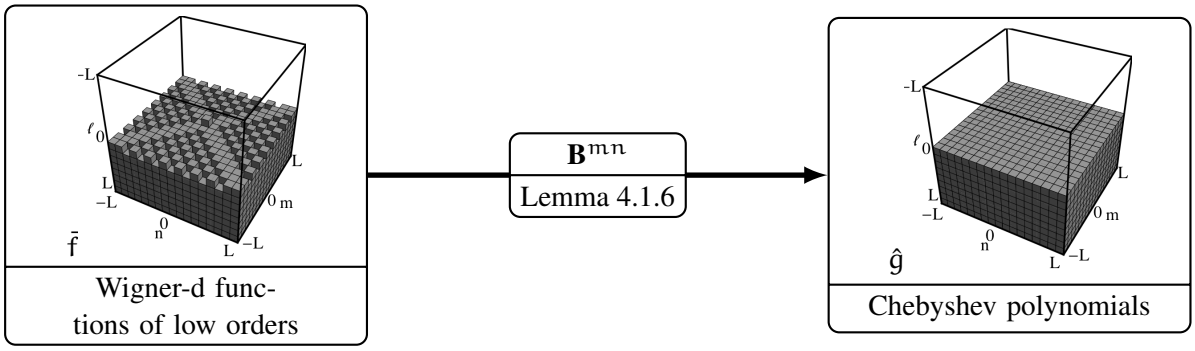


Figure 4.4: The aim of the second step of the FWT-S is to turn expansions of Wigner-d functions of low orders with maximum degree L into expansions of Chebyshev polynomials of first kind. The figure shows the input and output coefficients of the transform, in the sense that each cube with coordinates (ℓ, m, n) represents a coefficient of orders m and n, and degree ℓ. In this transform, the matrices $\mathbf{B}^{mn}$ for $m, n = -L, \dots, L$ realise the change of basis.

with

$$\mathbf{B}^0 = \operatorname{diag}\left(\sqrt{\frac{2\ell+1}{2}}\right)_{\ell=0,\dots,L}, \qquad \mathbf{B}^1 = \operatorname{diag}\left(-\frac{1}{2}\sqrt{\frac{2\ell+1}{2}}\sqrt{\frac{\ell+1}{\ell}}\right)_{\ell=0,\dots,L},$$

$$\mathbf{B}^2 = \begin{pmatrix} b^2_{0,2} & & & & \\ 0 & b^2_{1,3} & & & \\ b^2_{2,2} & 0 & b^2_{2,4} & & \\ & b^2_{3,3} & 0 & \ddots & \\ & & b^2_{4,4} & \ddots & \\ & & & \ddots & \end{pmatrix}, \qquad \mathbf{B}^{\pm} = \begin{pmatrix} b^{\pm}_{0,1} & & & \\ b^{\pm}_{1,1} & b^{\pm}_{1,2} & & \\ & b^{\pm}_{2,2} & \ddots & \\ & & \ddots & \end{pmatrix},$$

$$\mathbf{C}^{01} = \begin{pmatrix} c^{01}_{0,0} & \cdots & c^{01}_{0,L} \\ & \ddots & \vdots \\ & & c^{01}_{L,L} \end{pmatrix}, \qquad \mathbf{C}^{10} = \begin{pmatrix} c^{10}_{0,0} & \cdots & c^{10}_{0,L} \\ & \ddots & \vdots \\ & & c^{10}_{L,L} \end{pmatrix},$$

$$\mathbf{C}^{11} = \begin{pmatrix} c^{11}_{0,0} & \cdots & c^{11}_{0,L} \\ & \ddots & \vdots \\ & & c^{11}_{L,L} \end{pmatrix}, \qquad \mathbf{D}^0 = \begin{pmatrix} d^0_{0,0} & \cdots & d^0_{0,L} \\ & \ddots & \vdots \\ & & d^0_{L,L} \end{pmatrix}.$$

Proof. The lemma follows by straightforward computations using the recurrence relations of Jacobi polynomials found in [1, p.782]. ∎

Applying the diagonal and bi-diagonal matrices $\mathbf{B}^0$, $\mathbf{B}^1$, $\mathbf{B}^2$, $\mathbf{B}^{\pm}$ is clearly an $\mathcal{O}(L)$ process. Due to the separability of the entries $c^{+}_{\ell,j}$, $c^{-}_{\ell,j}$ and $d^{1}_{\ell,j}$, with respect to the indices ℓ and j, the matrices $\mathbf{C}^{+}$, $\mathbf{C}^{-}$, and $\mathbf{D}^1$ can be applied at cost $\mathcal{O}(L)$, too. The remaining step, the transition from Legendre polynomials to Chebyshev polynomials represented by $\mathbf{D}^0$ can also be computed efficiently. An $\mathcal{O}(L)$ method has been described in [4] and [48] to apply this matrix to a suitable vector.

A Short Summary on the FWT- S The matrix $\mathbf{C}^{mn} \in \mathbb{C}^{(L+1)\times(L-L_0+1)}$ established in Definition 4.1.1 for all $L, m, n \in \mathbb{N}$ with $|m|, |n| \leqslant L$ can be applied to a vector in $\mathcal{O}(L \log L)$ steps by splitting it into the product $\mathbf{C}^{mn} = \mathbf{A}^{mn}\mathbf{B}^{mn}$. The matrices $\mathbf{A}^{mn}, \mathbf{B}^{mn}$ are defined in (3.28) and Lemma 4.1.6, respectively. We saw that the application of $\mathbf{A}^{mn}$ demands $\mathcal{O}(L \log L)$ operations and is based on the efficient computation with semiseparable matrices while the application of $\mathbf{B}^{mn}$ is realised in $\mathcal{O}(L)$ steps, only. In contemplation of the step which takes the most effort, we call the application of $\mathbf{A}^{mn}\mathbf{B}^{mn}$ to a suitable vector, the fast transformation of Wigner-d functions based on semiseparable matrices, FWT-S.
Compared to the naive $\mathcal{O}(L^2)$ approach of the DWT and even the $\mathcal{O}(L \log^2 L)$ approach of the (unstabilised) FWT-C, this is an improvement in terms of runtime behaviour in any case. The following section will analyse the bearings of the FWT-S a bit further as we take a look at stability and actual runtime and compare it to the DWT and FWT-C.

4.1.3 Numerical Results

We have implemented and tested the fast transforms of Wigner-d functions in C on an Intel Core 2 Duo 2.66 GHz MacBook Pro with 4GB RAM running Mac OS X 10.6.1 in double precision arithmetic. There we have used Apple's `gcc-4.2` compiler with the optimisation options `-O3 -fomit-frame-pointer -malign-double -ffast-math -mtune=core2 -march=core2`. In addition to our code, we used the FFTW 3.2.1 [32] and the NFFT 3.1.2 [49] libraries that we compiled with the same compiler settings.
To test accuracy and time requirements of the fast transformation of Wigner-d functions, we computed the sums

$$f(x_i) = \sum_{\ell=0}^{L} \sum_{\ell=L_0}^{L} \hat{f}_\ell^{mn} d_\ell^{mn}(x_i) \tag{4.11}$$

for given SO(3) Fourier coefficients $\hat{f}_\ell^{mn}$ at 1000 randomly chosen samples $x_i \in [-1, 1]$ and fixed orders m and n. On one hand, we computed this sum directly using the three-term recurrence relation (4.5) of the Wigner-d functions together with the Clenshaw algorithm [18].
On the other hand we computed the sum (4.11) by the transforms described in the previous sections. That means, we first computed Chebyshev coefficients g^{mn} that satisfy

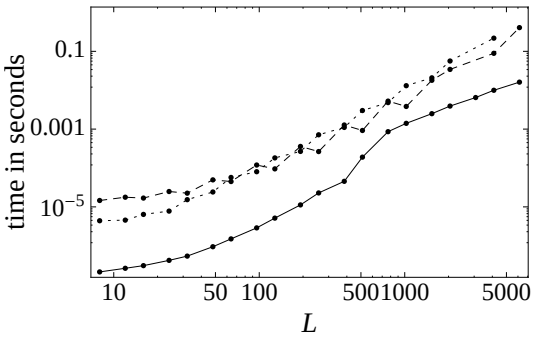
$$\sum_{\ell=L_0}^{L} \hat{f}_\ell^{mn} d_\ell^{mn}(x) = \sum_{\ell=0}^{L} \hat{g}_\ell^{mn} T_\ell(x)$$

and only then applied the Clenshaw algorithm to evaluate the Chebyshev expansion

$$f(x_i) = \sum_{\ell=0}^{L} \hat{g}_\ell^{mn} T_\ell(x_i)$$

at the sampling nodes x_i.

The following numerical tests examine the time requirements and accuracy of the transform of Wigner-d function. For fixed orders m and n, we computed the vector of Chebyshev coefficients $\mathbf{g}^{mn} = (g_0^{mn}, \dots, g_L^{mn})^T$ from given SO(3) Fourier coefficients $\widehat{\mathbf{f}}^{mn} = \left(\widehat{f}^{mn}_{\max(|m|,|n|)}, \dots, \widehat{f}^{mn}_L\right)^T$ by evaluating $\mathbf{g}^{mn} = \mathbf{C}^{mn}\widehat{\mathbf{f}}^{mn}$ using a matrix $\mathbf{C}^{mn} \in \mathbb{C}^{(L+1)\times(L-\max(|m|,|n|))}$. We will consider three different variations of the transform described by the matrix $\mathbf{C}^{mn}$:



L	FWT-S	FWT-C	DWT
64	1.5e-6	4.4e-6	5.7e-6
128	5.1e-6	9.4e-6	1.8e-5
256	2.2e-6	2.6e-5	7.1e-5
512	1.9e-5	9.2e-5	3.0e-4
1024	1.4e-4	3.8e-4	1.3e-3
2048	3.9e-4	3.4e-3	5.6e-3
4096	9.9e-4	8.8e-3	2.1e-2

Figure 4.5: The plot shows the average times for evaluating a function $f(x_i)$ as in (4.11) out of randomly chosen SO(3) Fourier coefficients $\widehat{f}_\ell^{mn} \in \left[-\frac{1}{2}, \frac{1}{2}\right]$ at randomly chosen sampling points $x_i \in [-1, 1]$ for $i = 1, \dots, 1000$ as a function of the degree L for $L = 16, \dots, 2048$. Depicted are the time requirements of a single DWT (dotted line), a single FWT-C (dashed line) and FWT-S (solid line). To emphasise the more favourable runtime of the FWT-S, we added a table of absolute times for selected values of L. Especially for larger L the improvement is apparent.

1. DWT: the transform corresponding to a direct multiplication with the matrix $\mathbf{C}^{mn}$ as described in Definition 4.1.1

2. FWT-C: the same transform as above but using the concepts described in Section 4.1.1 including cascade summation and stabilisation scheme.

3. FWT-S: the transform described in Section 4.1.2 that computes the multiplication with the matrix $\mathbf{C}^{mn}$ in two steps, by rewriting $\mathbf{C}^{mn} = \mathbf{B}^{mn}\mathbf{A}^{mn}$. This transform is based on a fast algorithm to compute with semi-seperable matrices.

Due to the symmetry properties of Wigner-d functions from Lemma 3.3.2, we consider only transforms of orders $m = 0, \dots, L$ and $n = 0, \dots, m$ for a given maximum degree L. For degrees $L \geqslant 16$, we computed the Wigner-d function transform for 120 different pairs of orders per degree L. Figure 4.5 shows the average time for one transform as a function of the degree L.
It shows that the fastest of the three presented algorithms is the FWT-S while the DWT is the slowest. This is in accordance with the asymptotic complexities of $\mathcal{O}(L \log L)$ for the FWT-S and $\mathcal{O}(L^2)$ for the DWT, we discussed. Concerning the FWT-C, we should observe a $\mathcal{O}(L \log^2 L)$ behaviour. But note that so far, we are not able to determine in advance the number of stabilisation steps needed in the FWT-C and hence we can neither prove the asymptotic complexity of the stabilized FWT nor predict at which orders this problem occurs. However, when looking at the absolute times we see that the stabilised FWT is still a large improvement over the DWT especially for larger degrees. More information on the stabilisation issue can be found in [68].
Another thing that can be observed in the FWT-C is the irregular behaviour. This is as well a result of the stabilisation scheme discussed in Section 4.1.1. When the stabilisation scheme is used in a transform for a certain set of orders the duration of a transform rises and from the plot in Figure 4.5 it appears that the FWT-C becomes almost as slow as the DWT. This however is not the case as we see in the right table of the figure and also in the results of [68].

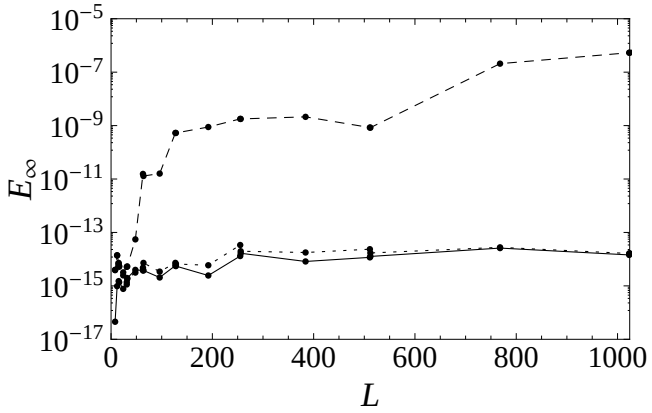


Figure 4.6: The figure shows the error E_∞ that occurs at evaluating a function $f(x_i)$ out of randomly chosen SO(3) Fourier coefficients $\widehat{f}_\ell^{mn} \in \left[-\frac{1}{2}, \frac{1}{2}\right]$ at randomly chosen sampling points $x_i \in [-1, 1]$ for $i = 1, \ldots, 1000$ by means of the DWT (dotted line), FWT-C (dashed line) or FWT-S (solid line). The error is plotted as a function of the degree L for $L = 16, \ldots, 1024$.

The next test should shed light on the errors produced during the transform of Wigner-d functions. Figure 4.6 shows the error

$$E_\infty = \frac{\|\mathbf{f} - \mathbf{f}_{\mathrm{FWT}}\|_\infty}{\|\mathbf{f}\|_1}$$

between the function samples $\mathbf{f} = (f(x_i))_{i=1,\ldots,1000}$ computed directly and the function samples computed by one of the three variations of the Wigner-d transform. The norms $\|\cdot\|_1$ and $\|\cdot\|_\infty$ are the l^p-norms of vectors.
We see that the FWT-S and DWT show basically the same error. Although the FWT-C has a slightly higher error it will not exceed 10^{-7} for the degrees, we examined here. The Chebyshev coefficients computed by the FWT-C algorithm show an error of about 10^{-12} compared to the exact coefficients.

4.2 Fast SO(3) Fourier Transforms

In this section, we return to the bigger scope of the SO(3) Fourier transform, i.e., from sums of Wigner-d functions we return to sums of Wigner-D functions. Using one of the algorithms presented in the last two sections to efficiently compute with expansions in Wigner-d functions, we are now left with Chebyshev expansions that can be easily transformed into sums of complex exponentials. This transition, just like the transform of Wigner-d functions, will be a coefficient transform which is done for fixed orders m and n.
But after this, we will, at last, incorporate the actual nonequispaced sampled rotations into our computations. This completes the nonequispaced fast SO(3) Fourier transform, NFSOFT, which will be finally described here. Subsequently, the adjoint NFSOFT will be given. We will obtain for both variants algorithms of complexity, of at least, $\mathcal{O}(L^3 \log L + Q)$, where Q is the number of sampled rotations and L the cutoff degree of the Fourier sum.

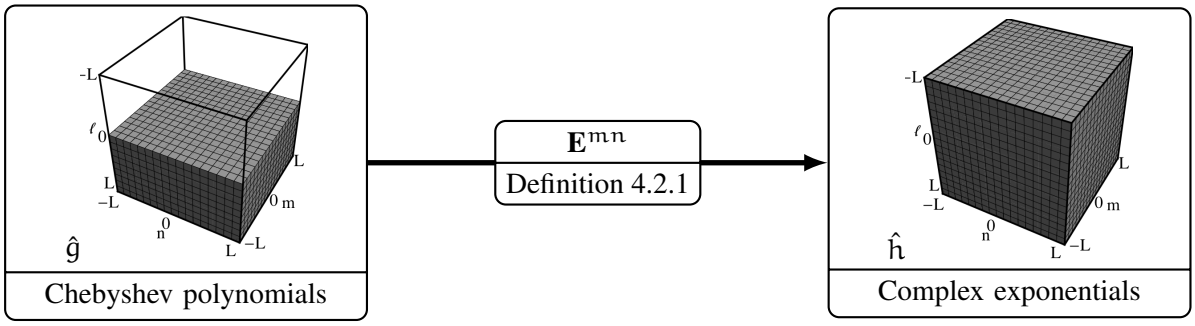


Figure 4.7: In the last step of the coefficient transform, we turn expansions of Chebyshev polynomials with maximum degree L into expansions of complex exponentials. The figure shows the input and output coefficients of the transform, in the sense that each cube with coordinates (ℓ, m, n) represents a coefficient of orders m and n, and degree ℓ. The transformation between these coefficients is expressed by the matrices $\mathbf{E}^{mn}$ for $m, n = -L, \ldots, L$.

4.2.1 The Nonequispaced Fast SO(3) Fourier Transform (NFSOFT)

After we generated Chebyshev coefficients $\hat{g}_\ell^{mn}$ from the $SO(3)$ Fourier coefficients $\hat{f}_\ell^{mn}$, one more coefficient transform needs to be done. That is the transform that turns a Chebyshev expansion into a Fourier sum. More specific, we seek the coefficients $\hat{h}_\ell^{mn}$ in

$$\sum_{\ell=-L}^{L} \hat{h}_\ell^{mn} e^{-i\ell\beta} = \begin{cases} \sum_{\ell=0}^{L} \hat{g}_\ell^{mn} T_\ell(\cos\beta) & \text{if } m+n \text{ even}, \\ \sum_{\ell=0}^{L-1} \hat{g}_\ell^{mn} \sqrt{1-\cos^2\beta}\, T_\ell(\cos\beta) & \text{if } m+n \text{ odd} \end{cases}$$

for $\beta \in [0, \pi]$. Again, we find a schematic figure of this transform (cf. Figure 4.7).
Note that in contrast to Sections 4.1.1 and 4.1.2, we return to the Euler angle representation, in the sense that, we replace $x = \cos\beta$. This also allows us a convenient representation of the Chebyshev polynomials of first kind as

$$T_\ell(x) = T_\ell(\cos\beta) = \cos(\ell\beta) = \frac{1}{2}\left(e^{i\ell\beta} + e^{-i\ell\beta}\right)$$

and the non-polynomial part as

$$\sqrt{1-x^2} = \sqrt{1-\cos^2\beta} = \sin\beta = -\frac{i}{2}\left(e^{i\beta} - e^{-i\beta}\right).$$

The following definition now gives the sought transform.

Definition 4.2.1. *For given $L \in \mathbb{N}$ and fixed orders m and n with $|m|, |n| \leqslant L$, the change of basis, given in Equation (4.2.1) from Chebyshev coefficients $\hat{\mathbf{g}}^{mn} = \left(\hat{g}_0^{mn}, \ldots, \hat{g}_L^{mn}\right)^T$ to the vector of Fourier coefficients $\hat{\mathbf{h}}^{mn} = (\hat{h}_{-L}^{mn}, \ldots, \hat{h}_L^{mn})^T$, is described by the matrix $\mathbf{E}^{mn} \in \mathbb{C}^{(2L+1)\times(L+1)}$ through*

$$\hat{\mathbf{h}}^{mn} = \mathbf{E}^{mn} \hat{\mathbf{g}}^{mn}.$$

Depending on whether $m + n$ is even or odd we have

$$\mathbf{E}^{mn} = \begin{cases} \mathbf{E}_1^{mn} & \text{if } m+n \text{ even,} \\ \mathbf{E}_2^{mn}\mathbf{E}_1^{mn} & \text{if } m+n \text{ odd.} \end{cases}$$

By means of the complex representation of the Chebyshev polynomials $T_\ell(\cos\beta)$, the matrix $\mathbf{E}_1^{mn}$ is written as

$$\mathbf{E}_1^{mn} = \begin{pmatrix} & & & \frac{1}{2} \\ & & \iddots & \\ & \frac{1}{2} & & \\ 1 & & & \\ & \frac{1}{2} & & \\ & & \ddots & \\ & & & \frac{1}{2} \end{pmatrix} \in \mathbb{C}^{(2L+1)\times(L+1)}$$

whereas due to

$$\sin\beta\, T_\ell(\cos\beta) = \tfrac{1}{2\pi}\sin(\beta)\, T_\ell(\cos\beta) = -\tfrac{i}{4}\left(e^{i\beta} - e^{-i\beta}\right)\left(e^{i\ell\beta} + e^{-i\ell\beta}\right)$$

the matrix $\mathbf{E}_2^{mn}$ reads as

$$\mathbf{E}_2^{mn} = \frac{i}{2}\begin{pmatrix} 0 & 1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & 1 \\ & & -1 & 0 \end{pmatrix} \in \mathbb{C}^{(2L+1)\times(L+1)}.$$

Clearly, the application of $\mathbf{E}^{mn}$ has a total cost of $\mathcal{O}(L)$ operations.
Now we have discussed all necessary transforms to turn the SO(3) Fourier coefficients into standard Fourier coefficients.

Finalising the Nonequispaced Fast SO(3) Fourier Transform (NFSOFT) In Section 3.2, we defined the SO(3) Fourier transform for nonequispaced sampled rotations, cf. Defintion 3.2.11. We introduced the matrix $\mathbf{D}_{L,\mathcal{R}_Q} = (D_\ell^{mn}(\mathbf{R}_q))_{(\ell,m,n)\in\mathcal{I}_L, q=1,\dots,Q}$ that, applied to a vector of SO(3) Fourier coefficients $\hat{\mathbf{f}} = (\hat{f}_\ell^{mn})_{(\ell,m,n)\in\mathcal{I}_L}$, computed function samples $\mathbf{f} = (f(\mathbf{R}_q))_{q=1,\dots,Q} \in \mathbb{C}^Q$ of a function $f \in \mathbb{D}_L$ at a sampling set $\mathcal{R}_Q$. The naive evaluation of

$$\mathbf{f} = \mathbf{D}_{L,\mathcal{R}_Q}\widehat{\mathbf{f}}$$

takes $\mathcal{O}(L^3Q)$ operations, due to the size of $\mathbf{D}_{L,\mathcal{R}_Q}$. But owing to the previously considered coefficient transformations, a factorisation of the matrix $\mathbf{D}_{L,\mathcal{R}_Q}$ which allows its efficient application is now tangible. The fast multiplication with the factorised matrix $\mathbf{D}_{L,\mathcal{R}_Q}$ is called the NFSOFT. The following theorem specifies the according algorithm.

Theorem 4.2.2. *The matrix $\mathbf{D}_{L,\mathcal{R}_Q}$, representing the NDSOFT, can be split into the matrix product*

$$\mathbf{D}_{L,\mathcal{R}_Q} = \mathbf{F}_{L,\mathcal{R}_Q}\,\mathbf{C}$$

where

$$\mathbf{F}_{L,\mathcal{R}_Q} = \left(e^{-im\alpha_q}e^{-i\ell\beta_q}e^{-in\gamma_q}\right)_{q=1,\dots,Q;(\ell,m,n)\in\mathcal{I}_L}$$

is a rotation-dependent trivariate Fourier matrix and

$$\mathbf{C} = \operatorname{diag}\left(\mathbf{E}^{mn}\mathbf{C}^{mn}\right)_{m,n=-L,\dots,L}$$

is the diagonal block matrix containing the rotation-independent coefficient transforms from Definitions 4.2.1 and 4.1.1*, though evaluated by either the FWT-C or FWT-S. The application of the matrix product* $\mathbf{F}_{L,\mathcal{R}_Q}\,\mathbf{C}$ *to a suitably sized vector represents the NFSOFT and takes at least* $\mathcal{O}(L^3 \log L + Q)$ *operations.*

Proof. Considering their respective definitions, the subsequent multiplication of the matrices $\mathbf{C}^{mn}$ and $\mathbf{E}^{mn}$, for fixed orders m and n, with the vector of SO(3) Fourier coefficients of the same orders costs $\mathcal{O}(L \log L)$ and $\mathcal{O}(L)$ operations (cf. Section 4.1.2 and Definition 4.2.1). If we seek to evaluate the SO(3) Fourier sum

$$f(\mathbf{R}(\alpha_q,\beta_q,\gamma_q)) \;=\; \sum_{\ell=0}^{L}\sum_{m=-\ell}^{\ell}\sum_{n=-\ell}^{\ell}\hat{f}_\ell^{mn}D_\ell^{mn}(\mathbf{R}(\alpha_q,\beta_q,\gamma_q))$$

of a function $f \in \mathbb{D}_L$, for $q = 1,\dots,Q$, we have $(2L+1)^2 = \mathcal{O}(L^2)$ different pairs of orders m and n.
First, we use formula (3.2.7) to split up the Wigner-D functions according to the Euler angles of $\mathbf{R}(\alpha_q,\beta_q,\gamma_q)$. Then, we may rewrite $f(\mathbf{R}(\alpha_q,\beta_q,\gamma_q))$ as

$$f(\mathbf{R}(\alpha_q,\beta_q,\gamma_q)) \;=\; \sum_{m=-L}^{L} e^{-im\alpha_q}\sum_{n=-L}^{L} e^{-in\gamma_q}\sum_{\ell=L_0}^{L}\hat{f}_\ell^{mn}d_\ell^{mn}(\cos\beta_q).$$

The matrices $\mathbf{C}^{mn}$ and $\mathbf{E}^{mn}$ effect the computation of the coefficients $\hat{h}_\ell^{mn}$ from the coefficients $\hat{f}_\ell^{mn}$; and we obtain

$$f(\mathbf{R}(\alpha_q,\beta_q,\gamma_q)) \;=\; \sum_{m=-L}^{L} e^{-im\alpha_q}\sum_{n=-L}^{L} e^{-in\gamma_q}\sum_{\ell=-L}^{L}\hat{h}_\ell^{mn}e^{-i\ell\beta_q}$$

in $\mathcal{O}(L^3 \log L)$ operations.
Thus, we obtain a three-dimensional Fourier sum which can be represented by the trivariate Fourier matrix $\mathbf{F}_{L,\mathcal{R}_Q}$ and which can be computed by means of the NFFT algorithm in $\mathcal{O}(L^3 + Q)$ steps (cf. [71]). Putting this together yields the proposed complexity of $\mathcal{O}(L^3 \log L + Q)$. ■

There is a tabular overview in Figure 4.8 that gives a summary of the transforms performed during an NFSOFT, along with references to their matrix representations and complexities.

Corollary 4.2.3. *The adjoint NFSOFT, i.e., the matrix-vector multiplication with* $\mathbf{D}_{L,\mathcal{R}_Q}^H$ *as in (3.18) reads in matrix-vector notation as*

$$\hat{\mathbf{f}} = \mathbf{D}_{L,\mathcal{R}_Q}^H\mathbf{f}.$$

Corresponding to the matrix $\mathbf{D}_{L,\mathcal{R}_Q}$*, we split up its adjoint in a similar way into*

$$\hat{\mathbf{f}} = \mathbf{C}^H\mathbf{F}_{L,\mathcal{R}_Q}^H\mathbf{f}.$$

Hence, it has the same complexity as the NFSOFT.

We shall now present some numerical examples to demonstrate performance and accuracy of the NFSOFT algorithm and its adjoint.

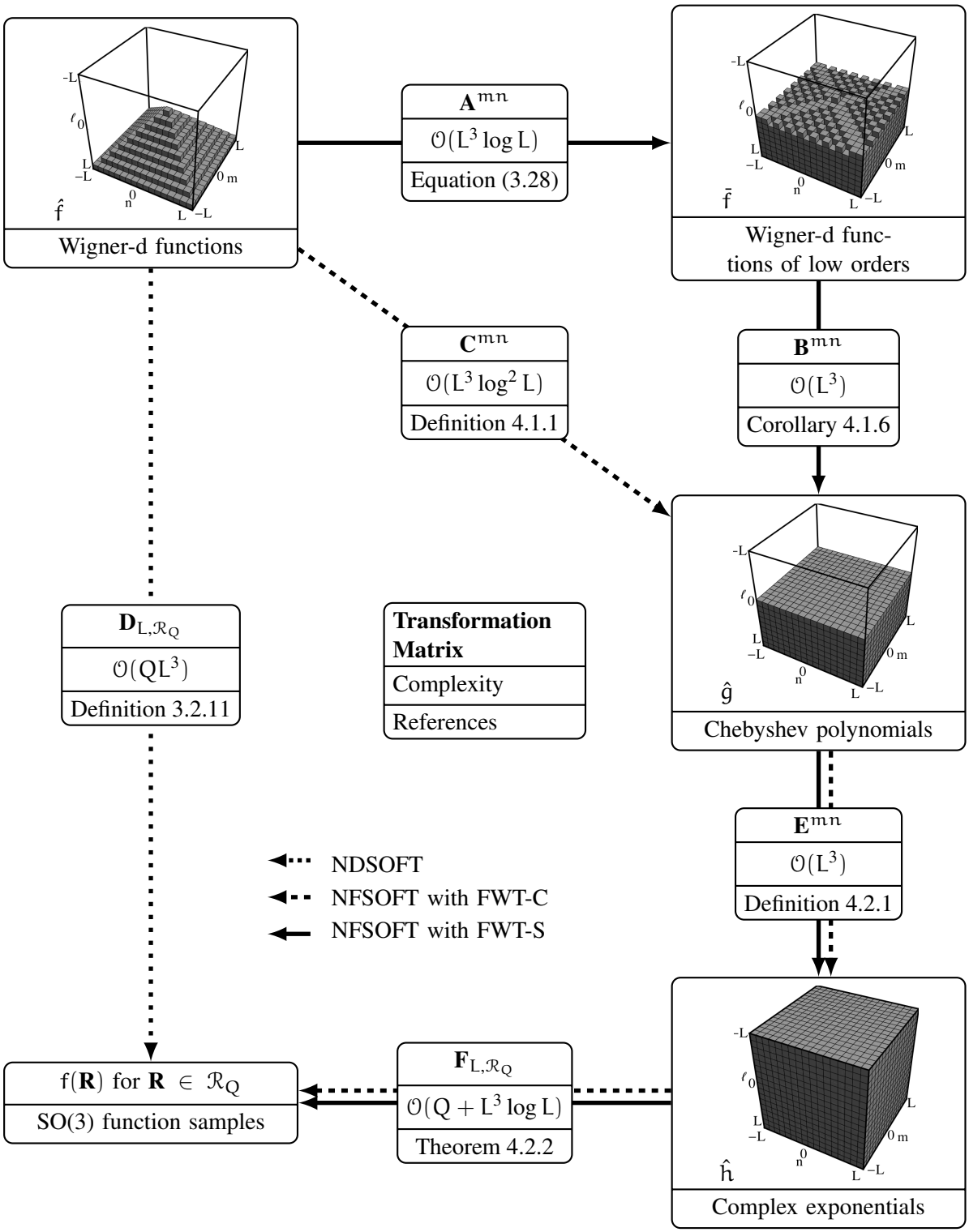


Figure 4.8: The NFSOFT in a nutshell. This chart depicts the sequence of matrix-vector multiplications necessary to compute SO(3) Fourier transforms of a function $f \in \mathbb{D}_L$ at Q nonequispaced input rotations. Note that $\mathbf{A}^{mn}$ actually means that all matrices $\mathbf{A}^{mn}$ for $m, n = -L, \ldots, L$ need to be applied to suitably ordered coefficients.

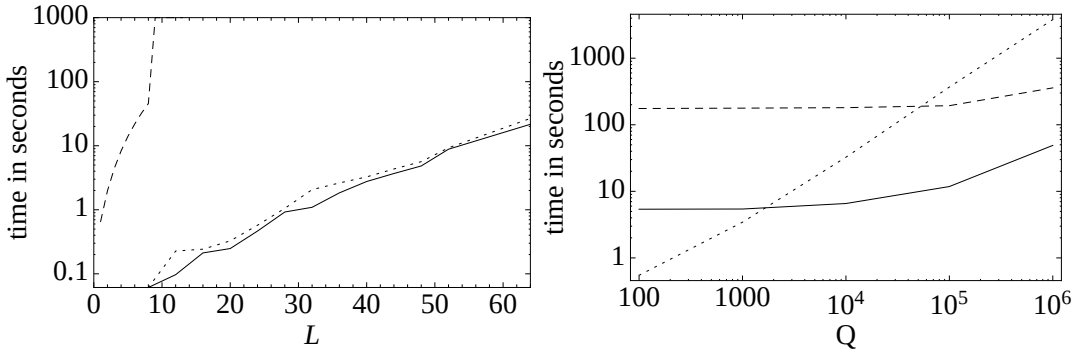


Figure 4.9: The left graph shows the runtime of an SO(3) Fourier transform as a function of the degree L for $L = 16, \ldots, 128$ for the three nonequispaced variations NDSOFT (dotted), NFSOFT(FWT-C) (dashed) and NFSOFT(FWT-S) (solid).
In the right graph we see the runtime of the NFSOFT(FWT-S) as a function of the number of input nodes $Q = 10^n$ for $n = 2, 3, \ldots, 6$ and different bandwidths $L = 32$ (solid) and $L = 64$ (dashed) as well as the runtime of the NDSOFT for bandwidth $L = 32$ with Q nodes (dotted).

4.2.2 Numerical Results

We test our NFSOFT algorithm, and compare it to the NDSOFT, i.e., the naive evaluation of (3.2.11). We chose the following three variations of our algorithm:

1. NFSOFT (FWT-C): the transform from Theorem 4.2.2 using FWT-C (see Section 4.1.1) with stabilization and the NFFT,
2. NFSOFT (FWT-S): the transform as above but with FWT-S (see Section 4.1.2),
3. NDSOFT: the transform as above using the DWT and nonequispaced discrete Fourier transform (NDFT) and thus directly evaluating Equation (3.2.11).

Like in Section 4.1.3 we tested the C-routines of the NFSOFT and NDSOFT on an Intel Core 2 Duo 2.66 GHz MacBook Pro with 4 GB RAM running Mac OS X 10.6.1 in double precision arithmetic. There we have used Apple's `gcc-4.2` compiler with the optimisation options from Section 4.1.3. In addition to our code, we used the FFTW 3.2.1 [32] and the NFFT 3.1.2 [49] libraries that we compiled with the same compiler settings.

The first test examines the time requirements of the various nonequispaced variations of the SO(3) Fourier transform mentioned above. From the vector $\widehat{\mathbf{f}} = (\widehat{f}_\ell^{mn})_{(\ell,m,n)\in\mathcal{I}_L}$ of randomly generated SO(3) Fourier coefficients $\widehat{f}_\ell^{mn} \in \left[-\frac{1}{2}, \frac{1}{2}\right]$, we computed the vector of function samples $\mathbf{f} = (f(\mathbf{R}_q))_{\mathbf{R}_q\in\mathcal{R}_Q}$ for Q randomly chosen rotations $\mathbf{R}_q \in SO(3)$.
This was done by evaluating the matrix-vector-product $\mathbf{f} = \mathbf{D}_{L,\mathcal{R}_Q}\widehat{\mathbf{f}}$ with the nonequispaced SO(3) Fourier matrix $\mathbf{D}_{L,\mathcal{R}_Q} = (D_\ell^{mn}(\mathbf{R}_q))_{\mathbf{R}_q\in\mathcal{R}_Q;(\ell,m,n)\in\mathcal{I}_L}$.
In Figure 4.9 we show the time requirements for the NDSOFT, NFSOFT (FWT-C) and the NFSOFT (FWT-S), respectively. The number of nodes is set to $Q = 1000$ while we test the algorithms for different bandwidths. We see that the two NFSOFT algorithms outperform the NDSOFT for all bandwidths.

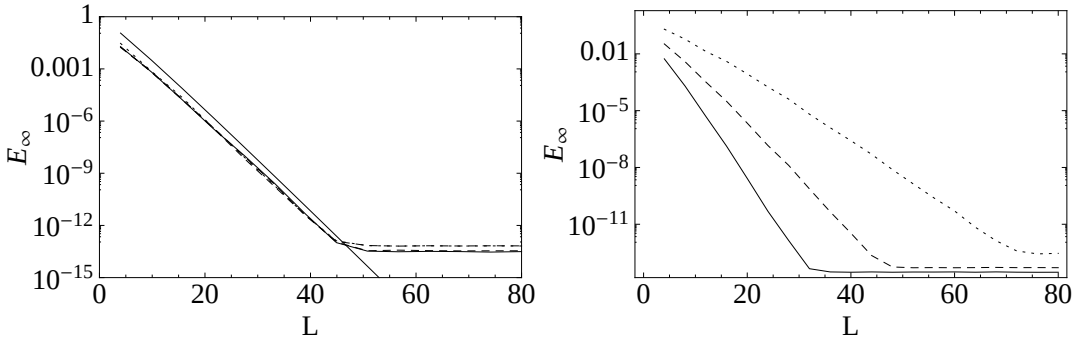


Figure 4.10: This figure shows the accuracy as a function of the cut-off degree L where we used the Abel–Poisson kernel with $\kappa = 0.6$. In the left graph we fixed the number of source nodes $M = 10^3$ and plotted the accuracy for different choices of Q target nodes ($Q = 10^3$ (dashed), $Q = 10^4$ (solid), $Q = 10^5$ (dotted), $Q = 10^6$ (dot-dashed)). The solid, bold line in the graph shows the theoretical error bound from Equation (3.36).
To give an idea about the localisation property of the used kernel functions the right graphs give the error E_∞ depending on the degree L of the Fourier expansion (3.35) for the Abel–Poisson Kernel ψ_κ for $\kappa = 0.7$ (solid), $\kappa = 0.8$ (dashed) and $\kappa = 0.85$ (dotted), as depicted in Figure 3.1.

Comparing the two NFSOFT versions, we see that they do not show any significant differences. This is not surprising considering that the NFFT algorithm dominates the runtime behavior completely. In other words, after the improvement from $\mathcal{O}(L^3Q)$ to $\mathcal{O}(L^3 \log L + Q)$ following the exchange of the NDFT algorithm with the NFFT, the Q-independent FWT algorithms with complexities of $\mathcal{O}(L^3 \log^2 L)$ compared to $\mathcal{O}(L^3 \log L)$ effect only a small change considering the size of the degrees L we used. The results from Section 4.1.3 suggest that for larger bandwidths an improved runtime behaviour of the NFSOFT using the FWT-S should be seen. However, due to memory limitations we are only able to compute the NFSOFT for Fourier expansions of maximum degree $L \leqslant 90$. Surely, we could try computing the NFSOFT on machines with more RAM, but as the memory requirements depend cubicly on the degree L as well. Consider only the memory needed to store all Fourier coefficients for the final NFFT, doubling the maximum degree L means we need to store eight-times more coefficients and hence at least eightfold memory.
The right plot of Figure 4.9 shows the runtime of the NFSOFT(FWT-S) as a function of the number of input nodes for $L = 32$ (solid) and $L = 64$ (dashed) compared to the runtime of the NDSOFT at $L = 32$. We see that up to $Q = L^3$ nodes the runtime is almost constant, i.e., the bandwidth controls the runtime of the NFSOFT. For larger number of nodes they become the dominant factor over the bandwidth. We then see linear growth of runtime which verifies that the nodes only add linear to the asymptotic complexity. If we computed these figures also for the NFSOFT (FWT-C), we would not spot a difference in time between the NFSOFT (FWT-S) here. That is due to the fact that both Wigner transforms, FWT-C and FWT-S, are independent of the input nodes (see Section 4.1).

To conclude our numerical tests, we like to show an example for the fast summation of radial functions on $SO(3)$ from Section 3.5. Following Lemma 3.5.3 we calculated the error

$$E_\infty = \frac{\|\mathbf{f} - \mathbf{f}_L\|_\infty}{\|\mathbf{c}\|_1}$$

and compared it to the theoretical estimate from the lemma. Here, the vector $\mathbf{f} = (f(\mathbf{R}_q))_{q=1,\ldots,Q} \in \mathbb{C}^Q$ contains the directly computed values of f at the target nodes $\mathbf{R}_q \in SO(3)$, while we compute $\mathbf{f}_L$ from (3.35). Again the rotations, $\mathbf{R}_q$ for the Q target nodes, and $\mathbf{S}_m \in SO(3)$ with $m = 1, \ldots, M$, for the M source nodes, were chosen randomly. Furthermore, the vector $\mathbf{c} = (c_m)_{m=1,\ldots,M} \in \mathbb{C}^M$ also contains random values. Figure 4.10 shows two error calculations for the Abel–Poisson kernel, one comparing the theoretical error bound with the numerical determined for varying number of target nodes, the other one showing the error for differently localised kernel fucntions. It can be seen that the number of target nodes has no effect on the accuracy, and that the theoretical error bound is met until it drops below the accuracy encountered by the NFSOFT algorithm at around 10^{-12}. We also see that for sharper, i.e. better localised kernels a higher maximum degree L of the Fourier expansion is required to achieve the sought accuracy, as expected.

5 Generalisations of SO(3) Fourier Transforms

In the previous chapters we considered the group $\mathrm{SO}(3)$ from many perspectives. We reviewed representation theory on the group, defined the Fourier transform of functions on $\mathrm{SO}(3)$ and developed fast algorithms to compute these Fourier transforms and also sums of functions on $\mathrm{SO}(3)$.
Now we like to give two generalisations of Fourier transforms on the rotation group:

- Fourier Transforms on the complex rotation group
- Fourier Transforms on the motion group.

Recalling Section 2.2, we saw that there is another matrix group by which we can describe rotations, namely, the special unitary group $\mathrm{SU}(2)$. The $\mathrm{SU}(2)$ is sometimes also called complex rotation group. In fact, there is a two-to-one homomorphism from $\mathrm{SU}(2)$ to $\mathrm{SO}(3)$ (cf. [91, pp. 157–161]) and so $\mathrm{SU}(2)$ and $\mathrm{SO}(3)$ share some nice properties, in particular, the unitary irreducible representations of $\mathrm{SO}(3)$ are a subset of those from $\mathrm{SU}(2)$ (cf. e.g. [17]). We saw that unitary irreducible representations are the key ingredient to define Fourier transforms on a group. The aim of the first section in this chapter is to transfer the concept of the $\mathrm{SO}(3)$ Fourier transform to the $\mathrm{SU}(2)$ to obtain a fast algorithm for its computation. Such a fast algorithm could be used in applications from particle physics [82], or to compute with pseudodifferential operators on $\mathrm{SU}(2)$ [78]. While $\mathrm{SO}(3)$ is diffeomorphic to the upper hemisphere of the 3-sphere $\mathbb{S}^3$, $\mathrm{SU}(2)$ is diffeomorphic to the $\mathbb{S}^3$. This means that by a simple coordinate transform a fast algorithm for the computation of $\mathbb{S}^3$ Fourier transforms based on the Fourier transformation on $\mathrm{SU}(2)$ would be at hand.
After that we would like to consider Fourier transforms on the three-dimensional motion group. Rigid-body motion in three-dimensional space consists of translation and rotation. This leads to another direction of generalising $\mathrm{SO}(3)$ Fourier transforms, though a more difficult one. The group of three-dimensional rigid-body motions $\mathrm{SE}(3)$ does not share as much properties with $\mathrm{SO}(3)$ as $\mathrm{SU}(2)$ does. In particular, $\mathrm{SE}(3)$ is not compact which leads to infinite-dimensional representation matrices of the group. Hence, computing Fourier transforms on the group will become more challenging. On the other hand the computation of $\mathrm{SE}(3)$ Fourier transforms has a vast field of applications, one of which we will discuss extensively in Chapter 6.

5.1 SU(2) Fourier Transforms

The relation between the elements of the rotation group $\mathrm{SO}(3)$ (cf. Definition 2.1.4) and the special unitary group $\mathrm{SU}(2)$ (cf. Definition 2.2.14) was established by (2.2.15) and (2.5) describing the two-to-one homomorphism between the two groups.
A different way to express elements in $\mathrm{SU}(2)$ in Euler angles as in (2.5) is by enlarging the domain of the third Euler angle γ from $\gamma \in [0, 2\pi)$ to $\gamma \in [-2\pi, 2\pi)$. Note, that we could also take the first Euler angle, α. Therefore, Euler angles in in this section will always be $\alpha \in [0, 2\pi)$, $\beta \in [0, \pi]$ and $\gamma \in [-2\pi, 2\pi)$.

Let us now collect the main ingredients for defining Fourier transforms on $SU(2)$. We have considered the integration of $SU(2)$ functions with a suitable integration invariant volume element in Definition 3.1.4. If using arguments parameterised in terms of Euler angles, the inner product of two functions $f, g \in L^2(SU(2))$ reads as

$$\langle f, g\rangle_{SU(2)} = \int_{SU(2)} f(\mathbf{U})\overline{g(\mathbf{U})}\, d\mathbf{U} = \int_{-2\pi}^{2\pi}\int_{0}^{\pi}\int_{0}^{2\pi} f(\mathbf{U}(\alpha,\beta,\gamma))\overline{g(\mathbf{U}(\alpha,\beta,\gamma))}\, \sin\beta\, d\alpha\, d\beta\, d\gamma.$$

Now we still need a complete set of irreducible unitary representations of $SU(2)$ to obtain an orthogonal basis on $L^2(SU(2))$. But we are already half-way there since the representations of $SO(3)$ are a subset of those in $SU(2)$. The matrix elements of irreducible unitary representations of $SU(2)$ arise in the same manner as they did on $SO(3)$, as eigenfunctions of the Laplace-Beltrami operator (cf. Definition 3.3.1). Indeed, the sought orthogonal basis functions are the already known Wigner-D functions D_ℓ^{mn} but defined for a different set of indices (ℓ, m, n). The group $SU(2)$ is a double cover of $SO(3)$, i.e., there is a two-to-one homomorphism from $SU(2)$ to $SO(3)$, which motivates the extention of the Wigner-D functions to elements of $SU(2)$. So in this chapter, Wigner-D functions will be functions $D_\ell^{mn} : SU(2) \to \mathbb{C}$.

Definition 5.1.1 (Half-integer index set)**.** *Let* $\mathbb{N}_{\frac{1}{2}} = \{\frac{2k+1}{2} \mid k \in \mathbb{N}\}$ *denote the set of all half-integer numbers. Based on this, we define a set of indices* $\mathcal{J}_L = \{(\ell, m, n) \mid \ell \in \mathbb{N}_{\frac{1}{2}}; \ell < L;\ m, n = -\ell, \ldots, \ell\}$ *for* $L \in \mathbb{N}$.

We state the following lemma without proof.

Lemma 5.1.2. *The set of Wigner-D functions*

$$\{D_\ell^{mn}(\mathbf{U}) \mid \ell \in \mathbb{N} \cup \mathbb{N}_{\frac{1}{2}},\ m, n = -\ell, \ldots, \ell\}$$

forms an orthogonal basis of $L^2(SU(2))$. *The* $L^2(SU(2))$ *decomposes into the direct sum*

$$L^2(SU(2)) = \mathit{clos}_{L^2} \bigoplus_{\ell=0,\frac{1}{2},1,\ldots} \mathit{Harm}_\ell(SU(2)),$$

where

$$\mathit{Harm}_\ell(SU(2)) = \mathit{span}\{D_\ell^{mn} :\ m, n = -\ell, \ldots, \ell\}.$$

A detailed derivation of the Wigner-D functions as matrix elements of the irreducible unitary representations of $SU(2)$ can be found in [17, pp. 281-296]. We will omit this and continue giving $SU(2)$ Fourier transforms.
Any function $f \in L^2(SU(2))$ has a unique series expansion in terms of the Wigner-D functions, the $SU(2)$ *Fourier expansion*

$$f(\mathbf{U}) = \sum_{\ell \in \mathbb{N}\cup\mathbb{N}_{\frac{1}{2}}} \sum_{m=-\ell}^{\ell} \sum_{n=-\ell}^{\ell} \widehat{f}_\ell^{mn} D_\ell^{mn}(\mathbf{U}),$$

for any $\mathbf{U} \in SU(2)$ and with Fourier coefficients $\hat{f}_\ell^{mn}$ given by the inner product

$$\widehat{f}_\ell^{mn} \quad = \quad \frac{2\ell+1}{16\pi^2}\langle f, D_\ell^{mn}\rangle_{L^2(SU(2))}. \tag{5.1}$$

Note the factor $\frac{2\ell+1}{16\pi^2}$ which is due to the normalisation of Wigner-D functions with respect to the inner product on SU(2). Using the seperation of Wigner-D functions into components with only one Euler angle, (3.23) and considering the normalisation of Wigner-d functions d_ℓ^{mn} from (3.25) we get

$$\begin{aligned}\langle D_\ell^{mn}, D_{\ell'}^{m'n'}\rangle_{SU(2)} &= \int_{-2\pi}^{2\pi}\int_0^{\pi}\int_0^{2\pi} e^{-i(m\alpha+n\gamma)} e^{i(m'\alpha+n'\gamma)} d_\ell^{mn}(\beta) d_{\ell'}^{m'n'}(\beta)\, \sin\beta \, d\alpha \, d\beta \, d\gamma \\ &= \frac{16\pi^2}{2\ell+1}\delta_{\ell,\ell'}\delta_{m,m'}\delta_{n,n'}.\end{aligned}$$

For $L \in \mathbb{N}$ consider functions $f \in L^2(SU(2))$ the Fourier coefficients of which fulfil $f_\ell^{mn} = 0$ for $\ell > L$. In correspondence to the SO(3) case, we define the function spaces

$$\mathbb{D}_L^{SU(2)} = \bigoplus_{\ell=0,\frac{1}{2},1,\ldots,L} \mathrm{Harm}_\ell(SU(2))$$

for arbitrary $L \in \mathbb{N}$ the elements of which are the above mentioned band-limited functions. An orthogonal basis of these spaces is given by

$$\{D_\ell^{mn}(\mathbf{U}) \mid (\ell, m, n) \in \mathcal{I}_L \cup \mathcal{J}_L\}$$

where $\mathbf{U} \in SU(2)$ and $\mathcal{I}_L$ is the index set defined in (3.16).
The dimension of the spaces $\mathbb{D}_L^{SU(2)}$ is given by

$$\dim(\mathbb{D}_L^{SU(2)}) = |\mathcal{I}_L| + |\mathcal{J}_L| = \sum_{\ell=0}^{L}(2\ell+1)^2 + \sum_{\ell=0}^{L}(2\ell)^2 = \sum_{\ell=0}^{2L+1}\ell^2 = \frac{1}{3}(L+1)(2L+1)(4L+3).$$

For functions on these spaces we now define the discrete Fourier transform.

Definition 5.1.3 (Discrete Fourier Transform on SU(2))**.** *Let* $\mathcal{U}_Q = (\mathbf{U}_1, \ldots, \mathbf{U}_Q)$, $\mathbf{U}_q \in SU(2)$ *define an arbitrary sampling set on* SU(2)*. Then*

$$f(\mathbf{U}_q) = \sum_{(\ell,m,n)\in\mathcal{I}_L\cup\mathcal{J}_L} \widehat{f}_\ell^{mn} D_\ell^{mn}(\mathbf{U}_q), \quad q = 1, \ldots, Q,$$

with given Fourier coefficients $\hat{\mathbf{f}} = (\hat{f}_\ell^{mn})_{(\ell,m,n)\in\mathcal{I}_L\cup\mathcal{J}_L}$ *evaluates a function* $f \in \mathbb{D}_L^{SU(2)}$ *at a set of nodes* $\mathcal{U}_Q = \{\mathbf{U}_q \in SU(2) \mid q = 1, \ldots, Q\}$*. The corresponding operator* $\mathbf{D}_{L,\mathcal{U}_Q} \colon \mathbb{C}^{|\mathcal{I}_L\cup\mathcal{J}_L|} \to \mathbb{C}^Q$ *with* $f(\mathbf{U}_q) = \left[\mathbf{D}_{L,\mathcal{R}_Q}\hat{\mathbf{f}}\right]_q$ *is called nonequispaced discrete* SU(2) *Fourier transform* (NDSUFT)*.*

The operator $\mathbf{D}_{L,\mathcal{U}_Q}$ corresponds to a matrix $\mathbf{D}_{L,\mathcal{U}_Q} \in \mathbb{C}^{|\mathcal{I}_L\cup\mathcal{J}_L|\times Q}$. Hence, the NDSUFT can be thought of as the matrix vector multiplication $\mathbf{f} = \mathbf{D}_{L,\mathcal{U}_Q}\widehat{\mathbf{f}}$ with $\mathbf{f} = (f(\mathbf{U}_q))_{q=1,\ldots,Q} \in \mathbb{C}^Q$ and $\hat{\mathbf{f}}$ as in Definition 5.1.3.
Most conveniently, for the efficient computation of the NDSUFT, we can reuse the NFSOFT algorithms by splitting the sum from Definition 5.1.3 into

$$f(\mathbf{U}_q) = \sum_{(\ell,m,n)\in\mathcal{I}_L} \widehat{f}_\ell^{mn} D_\ell^{mn}(\mathbf{U}_q) + \sum_{(\ell,m,n)\in\mathcal{J}_L} \widehat{f}_\ell^{mn} D_\ell^{mn}(\mathbf{U}_q), \quad q = 1, \ldots, Q.$$

The sum over the index set $\mathcal{I}_L$ can be computed by the NFSOFT algorithm. In the following, it remains to show how the sum over the half-integer Wigner-D functions

$$h(\mathbf{U}_q) = \sum_{(\ell,m,n)\in\mathcal{I}_L} \widehat{f}_\ell^{mn} D_\ell^{mn}(\mathbf{U}_q), \quad q = 1,\dots,Q; \tag{5.2}$$

can be efficiently computed using various modifications of the NFSOFT.
Again, we will try to convert the sum (5.2) into a three-dimensional standard Fourier sum to employ the NFFT algorithm for its computation. Let us start by separating the sums according to the Euler angles as we did in the SO(3) case

$$h(\mathbf{U}_q) = h(\mathbf{U}(\alpha_q,\beta_q,\gamma_q) = \sum_{(\ell,m,n)\in\mathcal{I}_L} \widehat{f}_\ell^{mn} e^{-i(m\alpha+n\gamma)} d_\ell^{mn}(\beta_q).$$

While it is quite clear how the complex exponentials behave for $\ell \in \mathbb{N}_{\frac{1}{2}}$ and $m,n = -\ell,\dots,\ell$, respectively, the behaviour of half-integer Wigner-d functions needs further examination; and we also need to show that we can efficiently compute the coefficients $\widehat{h}_\ell^{mn}$ from $\widehat{f}_\ell^{mn}$.

The Fast Transformation of Wigner-d Functions of Half-Integer Orders and Degree
The aim of the following paragraphs is to convert Wigner-d functions of half-integer orders and degree into Chebyshev polynomials. Recall the definition of Wigner-d functions in terms of Jacobi polynomials from (3.24) as

$$d_\ell^{mn}(x) = \varepsilon\sqrt{\frac{(\ell-\frac{\mu+\nu}{2})!\,(\ell+\frac{\mu+\nu}{2})!}{(\ell-\frac{\mu-\nu}{2})!\,(\ell+\frac{\mu-\nu}{2})!}}\,2^{-\frac{\mu+\nu}{2}}(1-x)^{\frac{\mu}{2}}(1+x)^{\frac{\nu}{2}}P_{\ell-L_0}^{(\mu,\nu)}(x), \tag{5.3}$$

where $\mu = |n-m|$, $\nu = |n+m|$, $L_0 = \max(|m|,|n|)$ and

$$\varepsilon = \begin{cases} 1 & \text{if } n \geqslant m, \\ (-1)^{n-m} & \text{if } n < m. \end{cases}$$

If we insert orders and degree from the half-integer index set $(\ell,m,n) \in \mathcal{I}_L$, we find $\mu,\nu \in \mathbb{N}$. Moreover, if μ is odd then ν will be even, and the other way around. This conveniently yields $\ell \pm \frac{\mu+\nu}{2} \in \mathbb{N}$ to define the x-independent factorials and by $\ell - L_0 \in \mathbb{N}$ we obtain an integer degree of the Jacobi polynomials.
The Wigner-d functions d_ℓ^{mn} for $(\ell,m,n) \in \mathcal{I}_L$ are no polynomials. But by dividing them by either $\sqrt{1-x}$ or $\sqrt{1+x}$, depending on whether μ or ν is odd, they become polynomials of degree $\ell - \frac{1}{2}$, for $\ell \in \mathbb{N}_{\frac{1}{2}}$.
Therefore, it is possible to perform a change of basis to Chebyshev polynomials such that

$$\sum_{\ell=L_0}^{L-\frac{1}{2}} \hat{f}_\ell^{mn} d_\ell^{mn}(x) = \sum_{\ell=\frac{1}{2}}^{L-\frac{1}{2}} \hat{g}_\ell^{mn} T_{\ell-\frac{1}{2}}(x) \begin{cases} \sqrt{1+x} & \text{if } m+n \text{ even}, \\ \sqrt{1-x} & \text{if } m+n \text{ odd}, \end{cases} \tag{5.4}$$

for $x \in [-1,1]$ and for fixed m and n holds true. The coefficients $\hat{g}_\ell^{mn}$ can, again, be obtained in different ways. E.g., we could exploit the three-term recurrence relation 3.26 to modify the FWT-C algorithm to the Wigner-d functions of half integer orders. We shall not examine this here but instead consider the necessary modifications to the FWT-S to compute the fast transformation of Wigner-d functions of half-integer orders and degree. We formulate a lemma similar to Lemma 3.3.3 to identify the necessary low orders into which the Wigner-d functions will be transformed first.

Lemma 5.1.4. *Let* $L, |m|, |n| \in \mathbb{N}_{\frac{1}{2}}$, $|m|, |n| \leqslant L$, *and denote by* $\mathbb{D}_L^{mn}$ *the space spanned by the functions* d_ℓ^{mn}, $\ell = L_0, \dots, L$. *Moreover, define*

$$m' := \begin{cases} \frac{1}{2} & \text{if } |m| = |n| = \frac{1}{2}, \\ \frac{3}{2} & \text{if } |m| > \frac{1}{2}. \end{cases} \quad n' := \begin{cases} \pm\frac{1}{2} & \text{if } m = \pm n = \frac{1}{2}, \\ \pm\frac{3}{2} & \text{if } m = \pm n,\ |m| > \frac{1}{2}, \\ \frac{1}{2} & \text{if } m + n \text{ even},\ m \neq n, \\ -\frac{1}{2} & \text{if } m + n \text{ odd},\ m \neq n. \end{cases} \tag{5.5}$$

Then we have $\mathbb{D}_L^{mn} \subseteq \mathbb{D}_L^{m'n'}$.

Proof. The proof follows the same lines as the proof from Lemma 3.3.3. While for $m' = \frac{1}{2}$, and hence $m' = |n'|$ the proof is immediate, we examine the cases for $m' = \frac{3}{2}$. The space $\mathbb{D}_L^{mn}$ for even $m + n$ is spanned by the functions $d_{L_0}^{mn}, \dots, d_L^{mn}$ which can be expressed as Jacobi polynomials $P_{\ell-\frac{3}{2}}^{(1,2)}, \dots, P_L^{(1,2)}$ multiplied by $\sqrt{1-x}(1+x)$ and a normalising factor. These functions span $\mathbb{D}_L^{\frac{3}{2},\frac{1}{2}}$. Analogously, the $P_{\ell-\frac{3}{2}}^{(2,1)}, \dots, P_L^{(2,1)}$ multiplied with $\sqrt{1+x}(1-x)$ span $\mathbb{D}_L^{\frac{3}{2},-\frac{1}{2}}$, the space of Wigner-d functions with odd $m+n$; while the $P_{\ell-\frac{3}{2}}^{(0,3)}, \dots, P_L^{(0,3)}$ or $P_{\ell-\frac{3}{2}}^{(3,0)}, \dots, P_L^{(3,0)}$ multiplied with $\sqrt{(1 \pm x)^3}$ span $\mathbb{D}_L^{\frac{3}{2},\pm\frac{3}{2}}$, the spaces of Wigner-d functions with $m = \pm n$. ∎

Let $L \in \mathbb{N}$ and m, n with $|m|, |n| \in \mathbb{N}_{\frac{1}{2}}$ be fixed and let f be a finite expansion in Wigner-d functions d_ℓ^{mn}, $\ell = L_0, \dots, L - \frac{1}{2}$. By Lemma 5.1.4 we see that, like in the integer order case, f can be expressed in Wigner-d functions $d_\ell^{m'n'}$ of low orders,

$$f = \sum_{\ell=L_0}^{L-\frac{1}{2}} \hat{f}_\ell^{mn} d_\ell^{mn} = \sum_{\ell=L_0'}^{L-\frac{1}{2}} \bar{f}_\ell^{mn} d_\ell^{m'n'}$$

where the sought transformation from coefficients $\hat{f}_\ell^{mn}$ to coefficients $\bar{f}_\ell^{mn}$ is linear. Therefore, there exists a matrix $\mathbf{A}^{mn} = (a_{\ell,k}) \in \mathbb{R}^{(L-L_0'+\frac{1}{2})\times(L-L_0+\frac{1}{2})}$ such that the vectors $\hat{\mathbf{f}}^{mn} = (\hat{f}_\ell) \in \mathbb{C}^{L-L_0+\frac{1}{2}}$ and $\bar{\mathbf{f}}^{mn} = (\bar{f}_\ell) \in \mathbb{C}^{L-L_0'+\frac{1}{2}}$ are related by the equation

$$\bar{\mathbf{f}}^{mn} = \mathbf{A}^{mn} \hat{\mathbf{f}}^{mn}. \tag{5.6}$$

The entries of the matrix $\mathbf{A}^{mn}$ are given by $a_{\ell,k} = \frac{\sqrt{(2\ell+1)(2k+1)}}{2} \langle d_\ell^{m'n'}, d_k^{mn} \rangle$.
Again for Wigner-d functions d_ℓ^{mn} of half-integer orders $|m|, |n| \in \mathbb{N}_{\frac{1}{2}}$ and degree, $L \in \mathbb{N}_{\frac{1}{2}}$ we can formulate some special cases and symmetries.

Lemma 5.1.5. *For* $|m|, |n| \in \mathbb{N}_{\frac{1}{2}}$ *it holds true that* $\mathbf{A}^{mn} = \mathbf{I}$ *if* (m, n) *is one of* $(\frac{1}{2}, \pm\frac{1}{2})$, $(\frac{3}{2}, \pm\frac{1}{2})$ *or* $(-\frac{3}{2}, \frac{1}{2})$, *and* $\mathbf{A}^{mn} = -\mathbf{I}$ *if* (m, n) *is either* $(-\frac{3}{2}, -\frac{3}{2})$ *or* $(-\frac{1}{2}, \frac{1}{2})$. *Also, the known symmetries*

$$\mathbf{A}^{mn} = (-1)^{m-n} \mathbf{A}^{-m-n}, \quad \mathbf{A}^{mn} = (-1)^{m-n} \mathbf{A}^{nm}$$

hold true. Additionally, we have

$$\mathbf{A}^{mn} = \operatorname{diag}\left((-1)^{\ell+m}\right)_{\ell=L_0'}^{L-\frac{1}{2}} \qquad \mathbf{A}^{-mn} \operatorname{diag}\left((-1)^{\ell-\frac{(-1)^{m+n}}{2}}\right)_{\ell=L_0}^{L-\frac{1}{2}}$$

which slightly differs from the case of integer orders and degree cf. Corollary 3.3.5.

Proof. These are direct consequences of Lemma 3.3.2 and Lemma 5.1.4. ■

The symmetries from this lemma allow us to restrict our considerations of the matrices $\mathbf{A}^{mn}$ to $m, n \in \mathbb{N}_{\frac{1}{2}}$ with $n \leqslant m \leqslant L - \frac{1}{2}$ as the remaining cases can be deduced from them. Note that from now we use the normalised Wigner-d functions, denoted by $\tilde{d}_\ell^{mn}$, again. Recall the approach of the FWT-S in Section 4.1.2. The fast algorithm developed there was based on the realisation that the matrices $\mathbf{A}^{mn}$ are composed of the eigenvectors of the matrices $\mathbf{G}^{mn}$, cf. Definition 4.1.2 and Lemma 4.1.3. After showing that the matrices $\mathbf{G}^{mn}$ are semiseparable of rank at most two, we employed a fast algorithm that applies eigenvectors of semiseparable matrices to other suitably sized, arbitrary vectors.
Consider the matrices $\mathbf{G}^{mn}$, from Definition 4.1.2 for $|m|, |n| \in \mathbb{N}_{\frac{1}{2}}$. The explicit entries of these matrices for half integer orders and degree vary from the integer cases. We shall derive them in the following lemma.

Lemma 5.1.6. *Let $L \in \mathbb{N}$ and $|m|, |n| \in \mathbb{N}_{\frac{1}{2}}$ be given such that the pair (m, n) does not belong to the set of special cases from Lemma 5.1.5. The matrix $\mathbf{G}^{mn}$, from Definition 4.1.2 can be split into $\mathbf{G}^{mn} = \mathbf{D} + \mathbf{S}^- + \mathbf{S}^+$ with $\mathbf{D}, \mathbf{S}^-, \mathbf{S}^+$ as defined in* (4.9)*. For $\ell, k = \frac{3}{2}, \ldots, L - \frac{1}{2}$ the entries of the matrix $\mathbf{D} = (d_{\ell,k})$ are given by $d_{\ell,k} = \ell(\ell+1)\delta_{\ell,k}$, while $\mathbf{S}^- = (s^-_{\ell,k})$ and $\mathbf{S}^+ = (s^+_{\ell,k})$ satisfy*

$$s^-_{\ell,k} = (\mu^2 - \mu'^2)\frac{\sqrt{(\ell+\frac{1}{2})(k+\frac{1}{2})}}{8} \times \begin{cases} 0 & \text{for } m = n, \\ \frac{(\ell+\frac{1}{2})}{3(k+\frac{1}{2})}\frac{(\ell-\frac{1}{2})(\ell+\frac{3}{2})}{(k-\frac{1}{2})(k+\frac{3}{2})} & \text{for } m = -n, \\ 2\sqrt{\frac{(\ell-\frac{1}{2})(\ell+\frac{3}{2})}{(k-\frac{1}{2})(k+\frac{3}{2})}} & \text{for } m+n \text{ even}, \\ \frac{(\ell+\frac{1}{2})}{(k+\frac{1}{2})}\sqrt{\frac{(\ell-\frac{1}{2})(\ell+\frac{3}{2})}{(k-\frac{1}{2})(k+\frac{3}{2})}} & \text{for } m+n \text{ odd}, \end{cases}$$

$$s^+_{\ell,k} = (-1)^{\ell+k+1} \quad (\nu^2 - \nu'^2)\frac{\sqrt{(\ell+\frac{1}{2})(k+\frac{1}{2})}}{8} \times \begin{cases} \frac{(\ell+\frac{1}{2})}{3(k+\frac{1}{2})}\frac{(\ell-\frac{1}{2})(\ell+\frac{3}{2})}{(k-\frac{1}{2})(k+\frac{3}{2})} & \text{for } m = n, \\ 0 & \text{for } m = -n, \\ \frac{(\ell+\frac{1}{2})}{(k+\frac{1}{2})}\sqrt{\frac{(\ell-\frac{1}{2})(\ell+\frac{3}{2})}{(k-\frac{1}{2})(k+\frac{3}{2})}} & \text{for } m+n \text{ even}, \\ 2\sqrt{\frac{(\ell-\frac{1}{2})(\ell+\frac{3}{2})}{(k-\frac{1}{2})(k+\frac{3}{2})}} & \text{for } m+n \text{ odd}, \end{cases}$$

where $\tilde{\ell} := \min\{\ell, k\}$, $\tilde{k} := \max\{\ell, k\}$.

Proof. We will sketch the proof only for the entries $s^-_{\ell,k}$. The procedure for $s^+_{\ell,k}$ is completely analogous; and the entries $d_{\ell,k}$ are the same as in the integer order case. Note that owing to the symmetry $s^-_{\ell,k} = s^-_{k,\ell}$ we assume, without loss of generality, $k > \ell$. First we expand the expressions for $s^-_{\ell,k}$ (cf. (4.9)) to

$$s^-_{\ell,k} = \frac{\mu^2 - \mu'^2}{2}\sqrt{(\ell+\tfrac{1}{2})(k+\tfrac{1}{2})}\int_{-1}^{1} \frac{1}{1-x} d_k^{m'n'}(x) d_\ell^{m'n'}(x) dx.$$

Expressing the Wigner-d functions in terms of Jacobi polynomials as in (3.24), and using $L_0' = m'$, $\mu' = m' - n'$ and $\nu' = m' + n'$, we get

$$s^-_{\ell,k} = \frac{\mu^2 - \mu'^2}{16}\sqrt{\frac{(k-m')!(k+m')!}{(k-n')!(k+n')!}\frac{(\ell-m')!(\ell+m')!}{(\ell-n')!(\ell+n')!}}\sqrt{(\ell+\tfrac{1}{2})(k+\tfrac{1}{2})}$$
$$\times \int_{-1}^{1}(1-x)^{m'-n'-1}(1+x)^{m'+n'}P^{(m'-n',m'+n')}_{k-m'}(x)P^{(m'-n',m'+n')}_{\ell-m'}(x)\,dx.$$

Let us examine the four occurring cases of m and n.

i) For $m+n$ even, we have $m' = \frac{3}{2}$, $n' = \frac{1}{2}$. The coefficients now read as

$$s^-_{\ell,k} = \frac{\mu^2 - \mu'^2}{32}\sqrt{\frac{(k+\frac{3}{2})}{(k-\frac{1}{2})}\frac{(\ell+\frac{3}{2})}{(\ell-\frac{1}{2})}}\sqrt{(\ell+\tfrac{1}{2})(k+\tfrac{1}{2})}\int_{-1}^{1}(1+x)^2P^{(1,2)}_{k-\frac{3}{2}}(x)P^{(1,2)}_{\ell-\frac{3}{2}}(x)\,dx.$$

By [1, 22.7.16]

$$(1+x)P^{(1,2)}_n(x) = P^{(1,1)}_n(x) + \frac{n+1}{n+2}P^{(1,1)}_{n+1}(x)$$

holds true which we apply twice on our above expression. This leads to evaluating the integral

$$\int_{-1}^{1}P^{(1,1)}_m(x)P^{(1,1)}_n(x)\,dx = \frac{2}{m+2}[P^{(1,1)}_n(x)P^{(0,0)}_{m+1}(x)]^1_{-1} - \frac{n+3}{n+2}\int_{-1}^{1}P^{(0,0)}_{m+1}(x)P^{(2,2)}_{n-1}\,dx,$$

by partial integration. The integral over the interval $[-1,1]$ of a Legendre polynomial $P^{(0,0)}_{m+1}$ times any polynomials of smaller degree evaluates to zero. As we consider only orders with $n \leqslant m$ the integral on the right side of the equation vanishes and we have

$$\int_{-1}^{1}P^{(1,1)}_m(x)P^{(1,1)}_n(x)\,dx = 4\frac{m+1}{n+2}.$$

Inserting this in the formula for $s^-_{\ell,k}$, we obtain the $s^-_{\ell,k}$ as stated in the lemma by

$$\int_{-1}^{1}(1+x)^2P^{(1,2)}_{k-\frac{3}{2}}(x)P^{(1,2)}_{\ell-\frac{3}{2}}(x)\,dx = 8\frac{\ell-\frac{1}{2}}{k+\frac{3}{2}}.$$

ii) For $m+n$ odd, we have $m' = \frac{3}{2}$, $n' = -\frac{1}{2}$. The coefficients satisfy

$$s^-_{\ell,k} = \frac{\mu^2 - \mu'^2}{32}\sqrt{\frac{(k+\frac{3}{2})}{(k-\frac{1}{2})}\frac{(\ell+\frac{3}{2})}{(\ell-\frac{1}{2})}}\sqrt{(\ell+\tfrac{1}{2})(k+\tfrac{1}{2})}\int_{-1}^{1}(1-x)(1+x)P^{(2,1)}_{k-\frac{3}{2}}(x)P^{(2,1)}_{\ell-\frac{3}{2}}(x)\,dx.$$

Using almost the same idea as in the $m+n$ even case, here we are applying [1, 22.7.15] only once to get

$$(1-x)P^{(2,1)}_n(x) = P^{(1,1)}_n(x) - \frac{n+1}{n+2}P^{(1,1)}_n(x)$$

but also use partial integration by which we obtain

$$\int_{-1}^{1}(1-x)(1+x)P^{(2,1)}_{k-\frac{3}{2}}(x)P^{(2,1)}_{\ell-\frac{3}{2}}(x)\,dx = 4\frac{(\ell-\frac{1}{2})(\ell+\frac{1}{2})}{(k+\frac{1}{2})(k+\frac{3}{2})}$$

in an analogous manner as in i). From this, the assertion in the lemma follows.

iii) For $m = n$ we get $m' = n' = \frac{3}{2}$; and as $\mu^2 - \mu'^2 = 0$ we obtain immediately $s^-_{\ell,k} = 0$.

iv) For $m = -n$, and hence, $m' = \frac{3}{2}$, $n' = -\frac{3}{2}$ the coefficients simplify to

$$s^-_{\ell,k} = \frac{\mu^2 - \mu'^2}{32}\sqrt{(\ell+\tfrac{1}{2})(k+\tfrac{1}{2})}\int_{-1}^{1}(1-x)^2 P^{(3,0)}_{k-\frac{3}{2}}(x)P^{(3,0)}_{\ell-\frac{3}{2}}(x)\,dx.$$

The proof of this part is the most lengthy one and can be done by induction over ℓ and k to show that

$$I_{\ell-\frac{2}{3},k-\frac{2}{3}} = \int_{-1}^{1}(1-x)^2 P^{(3,0)}_{k-\frac{3}{2}}(x)P^{(3,0)}_{\ell-\frac{3}{2}}(x)\,dx = \frac{4}{3}\frac{(\ell-\frac{1}{2})(\ell+\frac{1}{2})(\ell+\frac{3}{2})}{(k-\frac{1}{2})(k+\frac{1}{2})(k+\frac{3}{2})} \tag{5.7}$$

is satisfied and we will get the coefficients from the lemma.

If $\ell = L'_0 = \frac{3}{2}$, then formula [34, p. 228, 7.391, 4.] gives the induction base by

$$I_{\frac{3}{2},k-\frac{3}{2}} = \int_{-1}^{1}(1-x)^2 P^{(3,0)}_{k-\frac{3}{2}}(x)\,dx = \frac{16}{(k-\frac{1}{2})(k+\frac{1}{2})(k+\frac{3}{2})}, \qquad k = \frac{3}{2}, \frac{5}{2}, \dots .$$

For the inductive step, we fix $\ell \geqslant \frac{5}{2}$ and assume that (5.7) holds true for ℓ replaced by $\ell - 1$ and all $k \geqslant \ell - 1$. We now calculate the integral $I_{\ell,k}$ by induction over k. The corresponding induction bases are obtained by setting $k = \ell$ on one hand. Then formula [34, p. 228, 7.391, 5.] asserts that

$$I_{\ell-\frac{3}{2},\ell-\frac{3}{2}} = \int_{-1}^{1}(1-x)^2\left(P^{(3,0)}_{\ell-\frac{3}{2}}(x)\right)^2\,dx = \frac{4}{3}.$$

On the other hand by setting $k = \ell + 1$, we obtain the second base by using the three-term recurrence for Jacobi polynomials [81, p. 71] which eventually gives

$$I_{\ell-\frac{3}{2},\ell-\frac{1}{2}} = \int_{-1}^{1}(1-x)^2 P^{(3,0)}_{\ell-\frac{3}{2}}(x)P^{(3,0)}_{\ell-\frac{1}{2}}(x)\,dx = \frac{4}{3}\frac{(\ell-\frac{1}{2})}{(\ell+\frac{5}{2})}.$$

Let us examine why. By the three-term recurrence formula [81, p. 71], we obtain

$$\begin{aligned} P^{(3,0)}_{\ell-\frac{1}{2}}(x) \;=\; & \left(\frac{9(\ell+\frac{1}{2})}{2\ell(\ell-\frac{1}{2})(\ell+\frac{5}{2})} + \frac{2(\ell+1)(\ell+\frac{1}{2})}{(\ell-\frac{1}{2})(\ell+\frac{5}{2})}x\right)P^{(3,0)}_{\ell-\frac{3}{2}}(x) \\ & -\frac{(\ell-\frac{3}{2})(\ell+1)(\ell+\frac{3}{2})}{(\ell-\frac{1}{2})(\ell+\frac{5}{2})\ell}P^{(3,0)}_{\ell-\frac{5}{2}}(x), \end{aligned}$$

which we insert into $I_{\ell-\frac{3}{2},\ell-\frac{1}{2}}$ yielding

$$\begin{aligned} I_{\ell-\frac{3}{2},\ell-\frac{1}{2}} =& \frac{1}{(\ell-\frac{1}{2})(\ell+\frac{5}{2})}\Bigg(\frac{9(\ell+\frac{1}{2})}{2\ell}\int_{-1}^{1}(1-x)^2\left(P^{(3,0)}_{\ell-\frac{3}{2}}(x)\right)^2\,dx \\ & + 2(\ell+1)(\ell+\frac{1}{2})\int_{-1}^{1}x(1-x)^2\left(P^{(3,0)}_{\ell-\frac{3}{2}}(x)\right)^2\,dx \\ & - \frac{(\ell-\frac{3}{2})(\ell+1)(\ell+\frac{3}{2})}{\ell}\int_{-1}^{1}(1-x)^2 P^{(3,0)}_{\ell-\frac{3}{2}}(x)P^{(3,0)}_{\ell-\frac{5}{2}}(x)\,dx\Bigg). \end{aligned}$$

The first and the third integral are in fact equal to $I_{\ell-\frac{3}{2},\ell-\frac{3}{2}}$ and $I_{\ell-\frac{5}{2},\ell-\frac{3}{2}}$, respectively, so we can apply the induction base and assumption. Using $x(1-x)^2 = (1-x)^2 - (1-x)^3$ and the orthogonality relation of the Jacobi polynomials, the second integral may be rearranged to

$$\begin{aligned}\int_{-1}^{1} x(1-x)^2 \left(P^{(3,0)}_{\ell-\frac{3}{2}}(x)\right)^2 \,dx &= \int_{-1}^{1} (1-x)^2 \left(P^{(3,0)}_{\ell-\frac{3}{2}}(x)\right)^2 \,dx - \int_{-1}^{1} (1-x)^3 \left(P^{(3,0)}_{\ell-\frac{3}{2}}(x)\right)^2 \,dx. \\ &= I_{\ell-\frac{3}{2},\ell-\frac{3}{2}} - \frac{4}{2\ell+1}\end{aligned}$$

Now, all quantities for $I_{\ell-\frac{3}{2},\ell-\frac{1}{2}}$ are known and we get by straight forward computation the sought expression for $I_{\ell-\frac{3}{2},\ell-\frac{1}{2}}$. To finalise the induction we now need to consider the term $I_{\ell-\frac{3}{2},k-\frac{3}{2}}$ for $k \geqslant \ell+2$ and use once more the three-term recurrence formula of Jacobi polynomials to obtain

$$\begin{aligned}I_{\ell-\frac{3}{2},k-\frac{3}{2}} =& \frac{1}{(k-\frac{3}{2})(k+\frac{3}{2})}\Bigg(\frac{9(k-\frac{1}{2})}{2(k-1)}\int_{-1}^{1}(1-x)^2 P^{(3,0)}_{\ell-\frac{3}{2}}(x)P^{(3,0)}_{k-\frac{5}{2}}(x)\,dx \\ &+ 2k(k-\frac{1}{2})\int_{-1}^{1} x(1-x)^2 P^{(3,0)}_{\ell-\frac{3}{2}}(x)P^{(3,0)}_{k-\frac{5}{2}}(x)\,dx \\ &- \frac{(k-\frac{5}{2})(k+1)(k+\frac{1}{2})}{k}\int_{-1}^{1}(1-x)^2 P^{(3,0)}_{\ell-\frac{3}{2}}(x)P^{(3,0)}_{k-\frac{7}{2}}(x)\,dx\Bigg).\end{aligned}$$

Assuming Equation (5.7) is satisfied for $I_{\ell-\frac{3}{2},\ell-\frac{5}{2}}$ and $I_{\ell-\frac{3}{2},\ell-\frac{7}{2}}$, we get

$$I_{\ell-\frac{2}{3},k-\frac{2}{3}} = \frac{4}{3}\frac{(\ell-\frac{1}{2})(\ell+\frac{1}{2})(\ell+\frac{3}{2})}{(k-\frac{1}{2})(k+\frac{1}{2})(k+\frac{3}{2})}$$

for k and have thus proven the lemma. Note that we omitted writing down the inductive step, as it is rather lengthy and was conveniently done in Mathematica.

■

Corollary 5.1.7. *The matrix* $\mathbf{G}^{mn}$ *from Definition 4.1.2 is symmetric semiseparable for* $m, n \in \mathbb{N}_{\frac{1}{2}}$ *with semiseparability rank* $R = 2$*, i.e.,*

$$\mathbf{G}^{mn} = \operatorname{diag}(\mathbf{d}) + \operatorname{triu}\left(\mathbf{u}_- \mathbf{v}_-^T\right) + \operatorname{tril}\left(\mathbf{v}_- \mathbf{u}_-^T\right) + \operatorname{triu}\left(\mathbf{u}_+ \mathbf{v}_+^T\right) + \operatorname{tril}\left(\mathbf{v}_+ \mathbf{u}_+^T\right), \tag{5.8}$$

with $\mathbf{d} = (d_\ell)$, $\mathbf{u}_- = \left(u_\ell^-\right)$, $\mathbf{v}_- = \left(v_\ell^-\right)$, $\mathbf{u}_+ = \left(u_\ell^+\right)$, $\mathbf{v}_+ = \left(v_\ell^+\right) \in \mathbb{R}^{L-L'_*+1}$, *for* $\ell = L'_*, \ldots, L$, *where*

$$d_\ell = \ell(\ell+1) + \frac{\ell+\frac{1}{2}}{4} \cdot \begin{cases} 2\left(\mu^2-\mu'^2\right) + \left(\nu^2-\nu'^2\right) & \text{for } m+n \text{ even,} \\ \left(\mu^2-\mu'^2\right) + 2\left(\nu^2-\nu'^2\right) & \text{for } m+n \text{ odd,} \\ \frac{2}{3}\left(\mu^2-\mu'^2\right) + \frac{2}{3}\left(\nu^2-\nu'^2\right) & \text{for } |m| = |n| \end{cases}$$

and

$$u_\ell^- := w_\ell^- W_\ell^-, \quad v_\ell^- := w_\ell^- / W_\ell^-, \quad u_\ell^+ := w_\ell^+ W_\ell^+, \quad v_\ell^+ := w_\ell^+ / W_\ell^+,$$

with

$$w_\ell^- := \frac{\gamma^-}{2}\sqrt{(\mu^2-\mu'^2)(\ell+\tfrac{1}{2})}, \quad w_\ell^+ := (-1)^{\ell+\frac{1}{2}}\frac{\gamma^+}{2}\sqrt{(\nu^2-\nu'^2)(\ell+\tfrac{1}{2})},$$

for

$$\gamma^{+} = \begin{cases} \sqrt{2} & \text{if } m+n \text{ odd,} \\ 1 & \text{if } m+n \text{ even,} \\ \sqrt{\frac{2}{3}} & \text{if } m=n, \\ 0 & \text{if } m=-n, \end{cases} \qquad \gamma^{-} = \begin{cases} \sqrt{2} & \text{if } m+n \text{ even,} \\ 1 & \text{if } m+n \text{ odd,} \\ 0 & \text{if } m=n, \\ \sqrt{\frac{2}{3}} & \text{if } m=-n, \end{cases}$$

and

$$W_{\ell}^{-} = \sqrt{(\ell-\tfrac{1}{2})(\ell+\tfrac{3}{2})} \times \begin{cases} 1 & \text{if } m+n \text{ even,} \\ (\ell+\frac{1}{2}) & \text{if } m+n \text{ odd,} \\ (\ell+\frac{1}{2})\sqrt{(l-\frac{1}{2})(l+\frac{3}{2})} & \text{if } |m|=|n|\,, \end{cases}$$

$$W_{\ell}^{+} = \sqrt{(\ell-\tfrac{1}{2})(\ell+\tfrac{3}{2})} \times \begin{cases} (\ell+\frac{1}{2}) & \text{if } m+n \text{ even,} \\ 1 & \text{if } m+n \text{ odd,} \\ (\ell+\frac{1}{2})\sqrt{(l-\frac{1}{2})(l+\frac{3}{2})} & \text{if } |m|=|n|\,. \end{cases}$$

Now we see that the same divide and conquer approach used for the FWT-S in Section 4.1.2 can be applied here and we obtain the coefficients $\bar{f}_{\ell}^{mn}$ for $(\ell, m, n) \in \mathcal{J}_L$ of the expansion in low order Wigner-d functions,

$$\sum_{\ell=L_0}^{L-\frac{1}{2}} \hat{f}_{\ell}^{mn} d_{\ell}^{mn} = \sum_{\ell=L_0'}^{L-\frac{1}{2}} \bar{f}_{\ell}^{mn} d_{\ell}^{m'n'}.$$

The next step is now to convert the low order Wigner-d functions into Chebyshev polynomials,

$$\sum_{\ell=L_0'}^{L-\frac{1}{2}} \bar{f}_{\ell}^{mn} d_{\ell}^{m'n'}(x) = \sum_{\ell=0}^{L-\frac{1}{2}} \hat{g}_{\ell}^{mn} T_{\ell-\frac{1}{2}}(x) \times \begin{cases} \sqrt{1+x} & \text{if } m+n \text{ even,} \\ \sqrt{1-x} & \text{if } m+n \text{ odd.} \end{cases} \tag{5.9}$$

We need to consider the following identities which are consequences of (3.24) and [1, 22.7.15-22.7.17]. We have

$$d_{\ell}^{\frac{1}{2},\frac{1}{2}}(x) = \sqrt{1+x}\frac{\sqrt{2l+1}}{2} P_{\ell-\frac{1}{2}}^{(0,1)}(x), \quad d_{\ell}^{\frac{1}{2},-\frac{1}{2}}(x) = \sqrt{1-x}\frac{\sqrt{2l+1}}{2} P_{\ell-\frac{1}{2}}^{(1,0)}(x),$$

$$d_{\ell}^{\frac{3}{2},\pm\frac{1}{2}}(x) = \sqrt{1\mp x}\left(b_{\ell,\ell}^{21} P_{\ell-\frac{3}{2}}^{(1,1)}(x) \pm b_{\ell+1,\ell}^{21} P_{\ell-\frac{1}{2}}^{(1,1)}(x) \right),$$

$$d_{\ell}^{\frac{3}{2},\frac{3}{2}}(x) = \sqrt{1+x}\left(b_{\ell,\ell}^{30} P_{\ell-\frac{3}{2}}^{(0,2)}(x) + b_{\ell+1,\ell}^{30} P_{\ell-\frac{1}{2}}^{(0,2)}(x) \right),$$

$$d_{\ell}^{\frac{3}{2},-\frac{3}{2}}(x) = \sqrt{1-x}\left(b_{\ell,\ell}^{30} P_{\ell-\frac{3}{2}}^{(2,0)}(x) - b_{\ell+1,\ell}^{30} P_{\ell-\frac{1}{2}}^{(2,0)}(x) \right),$$

with

$$b_{k,\ell}^{21} = \sqrt{\frac{\ell+\frac{3}{2}}{8}} \times \begin{cases} \sqrt{\frac{\ell+\frac{1}{2}}{\ell-\frac{1}{2}}} & \text{if } k=\ell, \\ \sqrt{\frac{\ell-\frac{1}{2}}{\ell+\frac{1}{2}}} & \text{if } k=\ell+1, \end{cases} \qquad b_{k,\ell}^{03} = \frac{1}{2\sqrt{2\ell+1}} \times \begin{cases} (\ell+\frac{3}{2}) & \text{if } k=\ell, \\ (\ell-\frac{1}{2}) & \text{if } k=\ell+1, \end{cases}$$

and

$$\mathrm{P}_\ell^{(2,0)} = \sum_{\mathrm{j}=0}^{\ell} \mathrm{b}_{\ell,\mathrm{j}}^{20} \mathrm{P}_\mathrm{j}^{(1,0)} \qquad \text{with} \qquad \mathrm{b}_{\ell,\mathrm{j}}^{20} = \frac{(\mathrm{j}+1)(2\mathrm{j}+2)}{(\ell+1)(\ell+2)},$$

$$\mathrm{P}_\ell^{(0,2)} = \sum_{\mathrm{j}=0}^{\ell} \mathrm{b}_{\ell,\mathrm{j}}^{02} \mathrm{P}_\mathrm{j}^{(0,1)} \qquad \text{with} \qquad \mathrm{b}_{\ell,\mathrm{j}}^{02} = (-1)^{\mathrm{j}+1} \frac{(\mathrm{j}+1)(2\mathrm{j}+2)}{(\ell+1)(\ell+2)},$$

to facilitate the change of basis (5.9). To express (5.9) in matrix-vector notation, we introduce the matrices $\mathbf{B}^{\mathrm{m}\mathrm{n}} \in \mathbb{R}^{\mathrm{L}\times(\mathrm{L}-\mathrm{L}_0'+\frac{1}{2})}$, for $\mathrm{m},\mathrm{n} = -\mathrm{L}+\frac{1}{2},\dots,\mathrm{L}-\frac{1}{2}$. They satisfy

$$\hat{\mathbf{g}}^{\mathrm{m}\mathrm{n}} = \mathbf{B}^{\mathrm{m}\mathrm{n}}\, \bar{\mathbf{f}}^{\mathrm{m}\mathrm{n}},$$

with $\hat{\mathbf{g}}^{\mathrm{m}\mathrm{n}} = (\hat{\mathrm{g}}_\ell^{\mathrm{m}\mathrm{n}}) \in \mathbb{C}^{\mathrm{L}}$ and $\bar{\mathbf{f}}^{\mathrm{m}\mathrm{n}} = (\bar{\mathrm{f}}_\ell^{\mathrm{m}\mathrm{n}}) \in \mathbb{C}^{\mathrm{L}-\mathrm{L}_0'+\frac{1}{2}}$.

Corollary 5.1.8. *The above mentioned matrices* $\mathbf{B}^{\mathrm{m}\mathrm{n}}$ *are given by*

$$\mathbf{B}^{\mathrm{m},\mathrm{n}} = \begin{cases} \mathbf{D}^0\,\mathbf{C}^{01}\,\mathbf{B}^{01} & \text{if } \mathrm{m}' = \mathrm{n}' = \frac{1}{2}, \\ \mathbf{D}^0\,\mathbf{C}^{10}\,\mathbf{B}^{01} & \text{if } \mathrm{m}' = -\mathrm{n}' = \frac{1}{2}, \\ \mathbf{D}^0\,\mathbf{C}^{01}\,\mathbf{B}^{02}\,\mathbf{B}^{03} & \text{if } \mathrm{m}' = \mathrm{n}' = \frac{3}{2}, \\ \mathbf{D}^0\,\mathbf{C}^{10}\,\mathbf{B}^{20}\,\mathbf{B}^{03} & \text{if } \mathrm{m}' = -\mathrm{n}' = \frac{3}{2}, \\ \mathbf{D}^0\,\mathbf{C}^{11}\,\mathbf{B}^{12} & \text{if } \mathrm{m}' = \frac{3}{2},\ \mathrm{n}' = \frac{1}{2}, \\ \mathbf{D}^0\,\mathbf{C}^{11}\,\mathbf{B}^{21} & \text{if } \mathrm{m}' = \frac{3}{2},\ \mathrm{n}' = -\frac{1}{2}, \end{cases}$$

where $\mathbf{C}^{10}$, $\mathbf{C}^{01}$, $\mathbf{C}^{11}$ *and* $\mathbf{D}^0$ *are given as in Corollary* 4.1.6 *and*

$$\mathbf{B}^{12} = \begin{pmatrix} \mathrm{b}_{1,0}^{21} & & \\ \mathrm{b}_{1,1}^{21} & \mathrm{b}_{2,1}^{21} & \\ & \mathrm{b}_{2,2}^{\pm} & \ddots \\ & & \ddots \end{pmatrix}, \qquad \mathbf{B}^{21} = \begin{pmatrix} -\mathrm{b}_{1,0}^{21} & & \\ \mathrm{b}_{1,1}^{21} & -\mathrm{b}_{2,1}^{21} & \\ & \mathrm{b}_{2,2}^{\pm} & \ddots \\ & & \ddots \end{pmatrix},$$

$$\mathbf{B}^{03} = \begin{pmatrix} \mathrm{b}_{1,0}^{03} & & \\ \mathrm{b}_{1,1}^{03} & \mathrm{b}_{2,1}^{03} & \\ & \mathrm{b}_{2,2}^{03} & \ddots \\ & & \ddots \end{pmatrix}, \qquad \mathbf{B}^{30} = \begin{pmatrix} -\mathrm{b}_{1,0}^{03} & & \\ \mathrm{b}_{1,1}^{03} & -\mathrm{b}_{2,1}^{03} & \\ & \mathrm{b}_{2,2}^{03} & \ddots \\ & & \ddots \end{pmatrix},$$

$$\mathbf{B}^{20} = \begin{pmatrix} \mathrm{b}_{0,0}^{20} & \dots & \mathrm{b}_{0,\mathrm{L}}^{20} \\ & \ddots & \vdots \\ & & \mathrm{b}_{\mathrm{L},\mathrm{L}}^{20} \end{pmatrix}, \qquad \mathbf{B}^{02} = \begin{pmatrix} \mathrm{b}_{0,0}^{02} & \dots & \mathrm{b}_{0,\mathrm{L}}^{02} \\ & \ddots & \vdots \\ & & \mathrm{b}_{\mathrm{L},\mathrm{L}}^{02} \end{pmatrix}.$$

The NFSUFT Setting $x = \cos\beta$, recall that the change of basis

$$\sum_{\ell=0}^{L} \hat{g}_\ell^{mn} T_\ell(\cos\beta) = \sum_{\ell=-L}^{L} \hat{h}_\ell^{mn} e^{-i\ell\beta}$$

was described by the matrix $\mathbf{E}^{mn}$ (cf. Definition 4.2.1) which in this case is given by $\mathbf{E}^{mn} = \mathbf{E}_1^{mn}$. By employing these matrices, we finalise the node-independent coefficient transforms for the half-integer Wigner-d functions, i.e., we can compute the coefficients $\hat{h}_\ell^{mn}$ in

$$\sum_{\ell=L_0}^{L} \hat{f}_\ell^{mn} d_\ell^{mn}(\cos\beta_q) \quad = \quad \sum_{\ell=-L+\frac{1}{2}}^{L-\frac{1}{2}} \hat{h}_\ell^{mn} r(\beta_q) e^{-i(\ell-\frac{1}{2})\beta_q} \tag{5.10}$$

where

$$r(\beta_q) = \begin{cases} \sqrt{1+\cos\beta_q} & \text{if } m+n \text{ even,} \\ \sqrt{1-\cos\beta_q} & \text{if } m+n \text{ odd.} \end{cases}$$

Inserting this into (5.2), we get

$$h(\mathbf{U}_q) = \sum_{\mathcal{J}_L} \hat{f}_\ell^{mn} D_\ell^{mn}(\mathbf{U}(\alpha_q,\beta_q,\gamma_q)) = \sum_{m,n=-L+\frac{1}{2}}^{L-\frac{1}{2}} \sum_{\ell=-L}^{L-1} \hat{h}_{\ell+\frac{1}{2}}^{mn} r(\beta_q) e^{-im\alpha_q} e^{-i\ell\beta_q} e^{-in\gamma_q}.$$

Note that we did not get rid of the factors $\sqrt{1\pm\cos\beta_q}$. We can handle this by splitting the Fourier transform into two transforms of half size, one for the coefficients where $m+n$ even, and one for the coefficients with $m+n$ odd. This will not change the total asymptotic complexity. We shall now resume the algorithm to compute the NFSUFT and consider the complexity of its steps.

The Nonequispaced Fast SU(2) Fourier Transform (NFSUFT) We have defined the SU(2) Fourier transform for nonequispaced sampled rotations in Defintion 5.1.3 and introduced the matrix $\mathbf{D}_{L,\mathcal{U}_Q} = (D_\ell^{mn}(\mathbf{U}_q))_{(\ell,m,n)\in\mathcal{I}_L\cup\mathcal{J}_L, q=1,\dots,Q}$ that, applied to a vector of SU(2) Fourier coefficients $\hat{\mathbf{f}} = (\hat{f}_\ell^{mn})_{(\ell,m,n)\in\mathcal{I}_L\cup\mathcal{J}_L}$, computed function samples $\mathbf{f} = (f(\mathbf{U}_q))_{q=1,\dots,Q} \in \mathbb{C}^Q$ of a function $f \in \mathbb{D}_L^{\frac{1}{2}}$ at a sampling set $\mathcal{U}_Q$. The naive evaluation of

$$\mathbf{f} = \mathbf{D}_{L,\mathcal{U}_Q}\widehat{\mathbf{f}}$$

takes $\mathcal{O}(L^3Q)$ operations, due to the size of the matrix $\mathbf{D}_{L,\mathcal{U}_Q}$. But owing to the previously considered coefficient transformations, a factorisation of the matrix $\mathbf{D}_{L,\mathcal{U}_Q}$ which allows its efficient application is tangible. The fast multiplication with the factorised matrix $\mathbf{D}_{L,\mathcal{U}_Q}$ is called the NFSUFT. The following theorem specifies the according algorithm which is based on the NFSOFT algorithm (cf. Theorem 4.2.2).

Theorem 5.1.9. *The matrix $\mathbf{D}_{L,\mathcal{U}_Q}$, representing the NDSUFT, can be rearranged and split into a block diagonal matrix consisting of three blocks of matrix products,*

$$\mathbf{D}_{L,\mathcal{U}_Q} = \begin{pmatrix} \mathbf{F}^e_{L,\mathcal{U}_Q}\mathbf{C}^e & & \mathbf{0} \\ & \mathbf{F}^o_{L,\mathcal{U}_Q}\mathbf{C}^o & \\ \mathbf{0} & & \mathbf{F}^i_{L,\mathcal{U}_Q}\mathbf{C} \end{pmatrix}$$

with

$$\begin{aligned}
\mathbf{F}^{e}_{L,\mathcal{U}_Q} &= \sqrt{1+\cos\beta_q}\left(e^{-im\alpha_q}e^{-i\ell\beta_q}e^{-in\gamma_q}\right)_{q=1,\dots,Q;(\ell,m,n)\in\mathcal{J}_L\wedge m+n \text{ even}},\\
\mathbf{F}^{e}_{L,\mathcal{U}_Q} &= \sqrt{1-\cos\beta_q}\left(e^{-im\alpha_q}e^{-i\ell\beta_q}e^{-in\gamma_q}\right)_{q=1,\dots,Q;(\ell,m,n)\in\mathcal{J}_L\wedge m+n \text{ odd}},\\
\mathbf{F}^{i}_{L,\mathcal{U}_Q} &= \left(e^{-im\alpha_q}e^{-i\ell\beta_q}e^{-in\gamma_q}\right)_{q=1,\dots,Q;(\ell,m,n)\in\mathcal{I}_L}
\end{aligned}$$

and

$$\begin{aligned}
\mathbf{C}^{e} &= \operatorname{diag}\left(\mathbf{E}_1^{mn}\mathbf{B}^{mn}\mathbf{A}^{mn}\right)_{-L+\frac{1}{2}\leqslant m,n\leqslant L-\frac{1}{2}\wedge m+n \text{ even}},\\
\mathbf{C}^{e} &= \operatorname{diag}\left(\mathbf{E}_1^{mn}\mathbf{B}^{mn}\mathbf{A}^{mn}\right)_{-L+\frac{1}{2}\leqslant m,n\leqslant L-\frac{1}{2}\wedge m+n \text{ odd}},\\
\mathbf{C}^{i} &= \operatorname{diag}\left(\mathbf{E}^{mn}\mathbf{B}^{mn}\mathbf{A}^{mn}\right)_{-L\leqslant m,n\leqslant L}.
\end{aligned}$$

The matrices $\mathbf{E}^{mn}$, $\mathbf{E}_1^{mn}$ *are given in Definition 4.2.1, while* $\mathbf{B}^{mn}$ *is given in the Corollaries 4.1.6 and 5.1.8 and* $\mathbf{A}^{mn}$ *is defined in* (3.28) *and* (5.6).
The application of the rearranged matrix $\mathbf{D}_{L,\mathcal{U}_Q}$ *to a suitably sized vector represents the NFSUFT and takes at least* $\mathcal{O}(L^3\log L+Q)$ *operations.*

Proof. If we seek to evaluate the SU(2) Fourier sum

$$f(U_q)=\sum_{(\ell,m,n)\in\mathcal{I}_L\cup\mathcal{J}_L}\widehat{f}_\ell^{mn}D_\ell^{mn}(U_q),\quad q=1,\dots,Q,$$

of a function $f\in\mathbb{D}_L$, for $q=1,\dots,Q$, we encounter $8L^2+4L+1=\mathcal{O}(L^2)$ different pairs of orders m and n.
Applying coefficients $\widehat{f}_\ell^{mn}$ for fixed m,n, sorted in vectors, to the suitable matrices $\mathbf{A}^{mn}$ costs $\mathcal{O}(L\log L)$ operations per set of orders. The subsequent application of $\mathbf{B}^{mn}$ as well as $\mathbf{E}^{mn}$ or $\mathbf{E}_1^{mn}$ costs $\mathcal{O}(L)$. Putting this together it takes $\mathcal{O}(L^3\log L)$ operations to perform the change of basis,

$$\begin{aligned}
\sum_{(\ell,m,n)\in\mathcal{I}_L\cup\mathcal{J}_L}\widehat{f}_\ell^{mn}D_\ell^{mn}(U_q) &= \sum_{\ell,m,n=-L}^{L}\hat{h}_\ell^{mn}e^{-im\alpha_q}e^{-i\ell\beta_q}e^{-in\gamma_q}\\
&\quad+\sum_{\ell=-L}^{L-1}\sum_{\substack{m,n=-L+\frac{1}{2}\\ m+n \text{ even}}}^{L-\frac{1}{2}}\hat{h}_{\ell+\frac{1}{2}}^{mn}\sqrt{1+\beta_q}e^{-im\alpha_q}e^{-i\ell\beta_q}e^{-in\gamma_q}\\
&\quad+\sum_{\ell=-L}^{L-1}\sum_{\substack{m,n=-L+\frac{1}{2}\\ m+n \text{ odd}}}^{L-\frac{1}{2}}\hat{h}_{\ell+\frac{1}{2}}^{mn}\sqrt{1-\beta_q}e^{-im\alpha_q}e^{-i\ell\beta_q}e^{-in\gamma_q}.
\end{aligned}$$

We obtained three three-dimensional Fourier sums which can be represented by a matrix vector product of the trivariate Fourier matrices $\mathbf{F}^{i}_{L,\mathcal{U}_Q}$, $\mathbf{F}^{e}_{L,\mathcal{U}_Q}$ and $\mathbf{F}^{o}_{L,\mathcal{U}_Q}$ with the vectors $\hat{\mathbf{h}}^{i}\in\mathbb{C}^{(2L+1)^3}$, $\hat{\mathbf{h}}^{e}\in\mathbb{C}^{4L^3}$ and $\hat{\mathbf{h}}^{o}\in\mathbb{C}^{4L^3}$. Each of these three evaluations takes $\mathcal{O}(L^3\log L+Q)$ operations and as they are independent from each other, this is also the total complexity of the algorithm. ∎

5.2 SE(3) Fourier Transforms

Like in case of the rotations surely everyone has an intuitive idea what a motion is. We shall start this section by briefly giving a mathematical description of this idea. We will define the group of rigid-body motions in three dimensions and consider integration of functions that take motions as arguments. Like on SO(3) we will use unitary irreducible representations of the group to define a set of orthogonal basis functions by means of which we define the Fourier transform on the group.

Definition 5.2.1 (Rigid-body Motion)**.** *A rigid-body motion in* $\mathbb{R}^3$ *with respect to the origin* $\mathbf{0} \in \mathbb{R}^3$ *is a linear map* $\rho : \mathbb{R}^3 \to \mathbb{R}^3$ *with* $\rho(\mathbf{v}) = \mathbf{R}\mathbf{v} + \mathbf{t}$ *where* $\mathbf{t} \in \mathbb{R}^3$ *and* $\mathbf{R} \in \mathrm{SO}(3)$.

The composition $\rho = \rho_2 \circ \rho_1$ of two rigid-body motions $\rho_1(\mathbf{v}) = \mathbf{R}_1\mathbf{v} + \mathbf{t}_1$ and $\rho_2(\mathbf{v}) = \mathbf{R}_2\mathbf{v} + \mathbf{t}_2$ is the map

$$\rho : \mathbf{v} \mapsto \mathbf{R}_2\mathbf{R}_1\mathbf{v} + \mathbf{R}_2\mathbf{t}_1 + \mathbf{t}_2. \tag{5.11}$$

This can be seen by $\rho(\mathbf{v}) = (\rho_2 \circ \rho_1)(\mathbf{v}) = \rho_2(\rho_1(\mathbf{v})) = \mathbf{R}_2\rho_1(\mathbf{v}) + \mathbf{t}_2$. The inversion ρ^{-1} of a rigid-body motion $\rho(\mathbf{v}) = \mathbf{R}\mathbf{v} + \mathbf{t}$ is the map

$$\rho^{-1} : \mathbf{v} \mapsto \mathbf{R}^{-\mathsf{T}}\mathbf{v} - \mathbf{R}^{\mathsf{T}}\mathbf{t}$$

as composing ρ^{-1} and ρ gives $\mathbf{v} = \mathrm{id}(\mathbf{v}) = \rho^{-1}(\rho(\mathbf{v})) = \rho^{-1}(\mathbf{R}\mathbf{v} + \mathbf{t})$. This is fulfilled for all $\mathbf{v}$ if $\rho^{-1}(\mathbf{v}) = \mathbf{R}^{\mathsf{T}}\mathbf{v} - \mathbf{R}^{\mathsf{T}}\mathbf{t}$.

Lemma 5.2.2. *Given two pairs* $(\mathbf{R}_1, \mathbf{t}_1)$ *and* $(\mathbf{R}_2, \mathbf{t}_2)$ *where* $\mathbf{t}_1, \mathbf{t}_2 \in \mathbb{R}^3$ *and* $\mathbf{R}_1, \mathbf{R}_2 \in \mathrm{SO}(3)$*, their corresponding rigid-body motions are different as well, i.e.* $(\mathbf{R}_1, \mathbf{t}_1) \neq (\mathbf{R}_2, \mathbf{t}_2) \Rightarrow \rho_1 \neq \rho_2$.

Proof. Let the two matrices $\mathbf{R}_1, \mathbf{R}_2$ satisfy $\mathbf{R}_2^{-1}\mathbf{R}_1 \neq \mathbf{I}$ hence, there is a vector $\mathbf{v}$ such that

$$\rho_2^{-1}(\rho_1(\mathbf{v})) = \mathbf{R}_2^{\mathsf{T}}\mathbf{R}_1\mathbf{v} + \mathbf{R}_2^{\mathsf{T}}(\mathbf{t}_1 - \mathbf{t}_2) \neq \mathbf{v}$$

for any $\mathbf{t}_1, \mathbf{t}_2 \in \mathbb{R}^3$. On the other hand there is also a vector $\mathbf{v}$ such that for $\mathbf{t}_1 \neq \mathbf{t}_2$ and arbitrary $\mathbf{R}_1, \mathbf{R}_2 \in \mathrm{SO}(3)$, we have

$$\rho_2^{-1}(\rho_1(\mathbf{v})) = \mathbf{R}_2^{\mathsf{T}}\mathbf{R}_1\mathbf{v} + \mathbf{R}_2^{\mathsf{T}}(\mathbf{t}_1 - \mathbf{t}_2) \neq \mathbf{v}$$

and therefore $\rho_2^{-1} \circ \rho_1 \neq \mathrm{id}$. ■

By means of this lemma, we will, from now on, identify a rigid-body motion ρ and a tupel $(\mathbf{R}, \mathbf{t})$ with each other.
It also follows that every rigid-body motion can be decomposed into a rotation followed by a translation with $(\mathbf{R}, \mathbf{t}) = (\mathbf{I}, \mathbf{t}) \circ (\mathbf{R}, \mathbf{0})$. Note, that this decomposition is not commutative as we have $(\mathbf{R}, \mathbf{0}) \circ (\mathbf{I}, \mathbf{t}) = (\mathbf{R}, \mathbf{R}\mathbf{t})$.

Theorem 5.2.3. *The set* $\mathcal{M} = \{(\mathbf{R}, \mathbf{t}) \in (\mathrm{SO}(3), \mathbb{R}^3)\}$ *forms a group with respect to the composition* $\circ$ *defined in* (5.11).

Proof. G1) We immediately see that the composition $(\mathbf{R}_1, \mathbf{t}_1) \circ (\mathbf{R}_2, \mathbf{t}_2) = (\mathbf{R}_1\mathbf{R}_2, \mathbf{R}_1\mathbf{t}_2 + \mathbf{t}_1)$ is again an element of $\mathcal{M}$.

G2) Associativity $((\mathbf{R}_1, \mathbf{t}_1) \circ (\mathbf{R}_2, \mathbf{t}_2)) \circ (\mathbf{R}_3, \mathbf{t}_3) = (\mathbf{R}_1, \mathbf{t}_1) \circ ((\mathbf{R}_2, \mathbf{t}_2) \circ (\mathbf{R}_3, \mathbf{t}_3))$ follows as

$$(\mathbf{R}_1\mathbf{R}_2, \mathbf{R}_1\mathbf{t}_2 + \mathbf{t}_1) \circ (\mathbf{R}_3, \mathbf{t}_3) = (\mathbf{R}_1\mathbf{R}_2\mathbf{R}_3, \mathbf{R}_1\mathbf{R}_2\mathbf{t}_3 + \mathbf{R}_1\mathbf{t}_2 + \mathbf{t}_1) = (\mathbf{R}_1, \mathbf{t}_1) \circ (\mathbf{R}_2\mathbf{R}_3, \mathbf{R}_2\mathbf{t}_3 + \mathbf{t}_2).$$

G3) For $(\mathbf{R},\mathbf{t}) \in \mathcal{M}$, we have $(\mathbf{I},\mathbf{0}) \circ (\mathbf{R},\mathbf{t}) = (\mathbf{R},\mathbf{t})$. As $(\mathbf{I},\mathbf{0}) \in \mathcal{M}$, it is the neutral element of $\mathcal{M}$.

G4) The inverse element of $\mathcal{M}$ is given by $(\mathbf{R}^\mathsf{T}, -\mathbf{R}^\mathsf{T}\mathbf{t}) \in \mathcal{M}$ as $(\mathbf{R}^\mathsf{T}, -\mathbf{R}^\mathsf{T}\mathbf{t}) \circ (\mathbf{R},\mathbf{t}) = (\mathbf{I},\mathbf{0})$.

■

Definition 5.2.4. *The group* $(\mathcal{M},\circ)$ *is called special Euclidean group* $\mathrm{SE}(3)$.

Remark 5.2.5. *The set* $\mathcal{R} = \{(\mathbf{R},\mathbf{0}) \in \mathrm{SE}(3) \mid \mathbf{R} \in \mathrm{SO}(3)\}$ *is a subgroup of* $\mathrm{SE}(3)$ *isomorphic to the rotation group. Similarly, the set* $\mathcal{T} = \{(\mathbf{I},\mathbf{t}) \in \mathrm{SE}(3) \mid \mathbf{t} \in \mathbb{R}^3\}$ *is a subgroup of* $\mathrm{SE}(3)$.

Lemma 5.2.6. *Any motion* $(\mathbf{R},\mathbf{t}) \in \mathrm{SE}(3)$ *satisfies* $(\mathbf{R},\mathbf{t}) = (\mathbf{R}, z\mathbf{U}\mathbf{e}_z)$ *where* $z \in \mathbb{R}$ *and* $\mathbf{U} \in \mathrm{SO}(3)\backslash\mathcal{Z}$.

Proof. By Corollary 2.2.6 every rotation $\mathbf{S} \in \mathrm{SO}(3)$ can be expressed as $\mathbf{S} = \mathbf{U}\mathbf{Z}$ for $\mathbf{Z} \in \mathcal{Z}$ and $\mathbf{U} \in \mathrm{SO}(3)\backslash\mathcal{Z}$. Using the Euler angle decomposition from Definition 2.2.10 and Lemma 2.2.12, we can write $\mathbf{U} = \mathbf{R}_z(\alpha)\mathbf{R}_y(\beta)$ for $\alpha \in [0,2\pi)$ and $\beta \in [0,\pi]$.
On the other hand, we can express a translation vector $\mathbf{t} \in \mathbb{R}^3$ in spherical coordinates obtaining the coordinate transform

$$\mathbf{t} = \begin{pmatrix} r\cos\varphi\sin\theta \\ r\sin\varphi\sin\theta \\ r\cos\theta \end{pmatrix} = r\,\mathbf{R}_z(\varphi)\mathbf{R}_y(\theta)\mathbf{e}_z$$

where $\varphi \in [0,2\pi)$, $\theta \in [0,\pi]$ and $r > 0$. Identifying $z = r$, $\alpha = \varphi$ and $\beta = \theta$, proves the lemma. ■

A consequence of this lemma is that we can parameterise a motion by five rotation angles $\alpha, \beta, \gamma, \varphi, \theta$ and a one-dimensional translation $z\mathbf{e}_z$.

Definition 5.2.7 (Metric on $\mathrm{SE}(3)$)**.** *A metric on* $\mathrm{SE}(3)$ *can be constructed using the distance on* $\mathrm{SO}(3)$ *from Definition 2.1.7 and the Euclidean norm* $\|\cdot\|$. *Given* $(\mathbf{t}_1,\mathbf{R}_1), (\mathbf{t}_2,\mathbf{R}_2) \in \mathrm{SE}(3)$*, a metric on* $\mathrm{SE}(3)$ *is specified by*

$$|(\mathbf{t}_1,\mathbf{R}_1),(\mathbf{t}_2,\mathbf{R}_2)| = |\mathbf{R}_2\mathbf{R}_1^{-1}| + \|\mathbf{t}_2 - \mathbf{t}_1\|.$$

Next, we consider some aspects of harmonic analysis on the motion group.

Definition 5.2.8. *Integration of a function* $f : \mathrm{SE}(3) \to \mathbb{R}$ *defined on rigid-body motions* $(\mathbf{R},\mathbf{t}) \in \mathrm{SE}(3)$ *reads as*

$$\int_{\mathrm{SE}(3)} f((\mathbf{R},\mathbf{t}))\,\mathrm{d}(\mathbf{R},\mathbf{t}) = \int_{\mathrm{SO}(3)} \int_{\mathbb{R}^3} f((\mathbf{R},\mathbf{t}))\,\mathrm{d}\mathbf{R}\,\mathrm{d}\mathbf{t})$$

where the normalised volume element $\mathrm{d}(\mathbf{R},\mathbf{t})$ *on* $\mathrm{SE}(3)$ *is defined as* $\mathrm{d}(\mathbf{R},\mathbf{t}) = \mathrm{d}\mathbf{R}\,\mathrm{d}\mathbf{t}$ *where* $\mathrm{d}\mathbf{R}$ *is the volume element of* $\mathrm{SO}(3)$ *from Defintion 3.1.1 and* $\mathrm{d}\mathbf{t}$ *is a volume element of the* $\mathbb{R}^3$.

By these definitions, we can express integration of a function $f : \mathrm{SE}(3) \to \mathbb{R}$ in various parameterisations of rotations and translations, e.g., integration of a function $f(\mathbf{R},\mathbf{t})$ with rotations $\mathbf{R} \in \mathrm{SO}(3)$ parameterised in Euler angles and translation vectors $\mathbf{t} \in \mathbb{R}^3$ in Cartesian coordinates reads as

$$\int_{\mathrm{SE}(3)} f((\mathbf{R},\mathbf{t}))\,\mathrm{d}(\mathbf{R},\mathbf{t}) = \frac{1}{8\pi^2}\int_{\mathbb{R}^3}\int_0^{2\pi}\int_0^{\pi}\int_0^{2\pi} f(\mathbf{R}(\alpha,\beta,\gamma),\mathbf{t})\sin\beta\,\mathrm{d}\alpha\,\mathrm{d}\beta\,\mathrm{d}\gamma\,\mathrm{d}\mathbf{t}. \tag{5.12}$$

Regarding the above integration, we define the space $L^2(\mathrm{SE}(3))$ completely analogous to the standard by

$$L^2(\mathrm{SE}(3)) = \left\{ f : \mathrm{SE}(3) \to \mathbb{C} \mid \int_{\mathrm{SE}(3)} |f((\mathbf{R},\mathbf{t}))|^2\,\mathrm{d}(\mathbf{R},\mathbf{t}) < \infty \right\}$$

with an inner product of two functions $f, g \in L^2(SE(3))$ given by

$$\langle f, g\rangle \quad = \quad \int_{SE(3)} f((\mathbf{R},\mathbf{t}))\overline{g((\mathbf{R},\mathbf{t}))}\, d(\mathbf{R},\mathbf{t}). \tag{5.13}$$

The convolution of two such functions $f, g \in L^2(SE(3))$ is written as

$$(f * g)((\mathbf{Q},\mathbf{s})) = \int_{SE(3)} f((\mathbf{R},\mathbf{t}))g((\mathbf{R},\mathbf{t})^{-1} \circ (\mathbf{Q},\mathbf{s}))\, d(\mathbf{R},\mathbf{t}). \tag{5.14}$$

Next, we shall consider Fourier transforms of functions $f(\mathbf{R},\mathbf{t}) \in L^2(SE(3))$. To define Fourier transforms on the motion group, we need to use an orthogonal basis for functions on this group. Like on SO(3), this basis of SE(3) is made up of the matrix elements of the unitary irreducible representations of group that act on subspaces invariant under the application of group elements. In Section 3.2 we characterised an orthogonal basis for the rotation group SO(3) by means of Lemma 3.2.6. We shall use the same reasoning for SE(3).
It is important to note that in case of the motion group these invariant subspaces have infinite many basis elements and hence infinite-dimensional representation matrices. This is because the motion group, in contrast to the rotation group, is not compact. A method to construct representations for certain noncompact groups can be found e.g. in [87]. An extensive description how to obtain the matrix elements of the representations can be found in [17]. Here we simply give their definition and describe very briefly how these representations arise, as the study of harmonic analysis on noncompact groups is not within the scope of this work.
At first, we need to consider a vector space on which the elements of SE(3) act transitively. This will be the $\mathbb{R}^3$, i.e., for all $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^3$, we can find an element $(\mathbf{R},\mathbf{t}) \in SE(3)$ such that $\mathbf{x}_2 = \mathbf{R}\mathbf{x}_1 + \mathbf{t}$. The subgroup of rotations from Remark 5.2.5 acts on $\mathbb{R}^3$ though not transitively. Instead, it divides the space into spheres of radius r,

$$\mathbb{S}^2_r = \{\mathbf{x} \in \mathbb{R}^3 \mid \mathbf{x} = r\mathbf{u},\ r > 0,\ \mathbf{u} \in \mathbb{S}^2\}.$$

We define the following functions.

Definition 5.2.9 (Modified spherical harmonics)**.** *For any $\ell \in \mathbb{N}_0$ and $m, n = -\ell, \dots \ell$ we define functions*

$$h_\ell^{mn}(\mathbf{u}) = h_\ell^{mn}(\varphi,\theta) = e^{-im\varphi} d_\ell^{mn}(\cos\theta) e^{in\varphi} \qquad \mathbf{u} \in \mathbb{S}^2.$$

Note that $h_\ell^{mn}(\varphi,\theta) = D_\ell^{mn}(\varphi,\theta,-\varphi)$ holds true.

Definition 5.2.10 (Unitary representations of SE(3))**.** *The unitary representations of* SE(3) *that act on* $L^2(\mathbb{S}^2_r)$ *are denoted by* $U_s((\mathbf{R},\mathbf{t}),r)$ *for* $s \in \mathbb{Z}$*. By*

$$U_s^{(\ell,m)(\ell',m')}((\mathbf{R},\mathbf{t}),r) = \int_{\mathbb{S}^2} e^{-ir\mathbf{u}^T\mathbf{t}}\, \overline{h_s^{\ell m}(\mathbf{u})} \sum_{n=-\ell}^{\ell} D_{\ell'}^{nm'}(\mathbf{R}) h_s^{\ell' n}(\mathbf{u})\, d\mathbf{u} \tag{5.15}$$

we denote the matrix elements of the unitary representations of SE(3).

The functions $U_s^{(\ell,m)(\ell',m')}$ satisfy the orthogonality relation

$$\begin{aligned} I \quad &= \quad \int_{SE(3)} \overline{U_{s_1}^{(\ell_1,m_1)(\ell_1',m_1')}((\mathbf{R},\mathbf{t}),p_1)} U_s^{(\ell,m)(\ell',m')}((\mathbf{R},\mathbf{t}),p)\, d(\mathbf{R},\mathbf{t}) \\ &= \quad \frac{2\pi^2}{p^2}\delta_{\ell,\ell_1}\delta_{\ell',\ell_1'}\delta_{m,m_1}\delta_{m',m_1'}\delta_{s,s_1}\delta_{p,p_1}, \end{aligned}$$

see [17]. This leads to the following Definition.

Definition 5.2.11 (SE(3) Fourier Transform)**.** *The* SE(3) *Fourier transform of functions* $f \in L^2(\mathrm{SE}(3))$ *is defined as*

$$f((\mathbf{R},\mathbf{t})) = \sum_{s=-\infty}^{\infty} \sum_{\ell=|s|}^{\infty} \sum_{\ell'=|s|}^{\infty} \sum_{m=-\ell}^{\ell} \sum_{m'=-\ell'}^{\ell'} \int_0^{\infty} \hat{f}_s^{(\ell,m)(\ell',m')}(p) U_s^{(\ell,m)(\ell',m')}((\mathbf{R},\mathbf{t}),p)\, p^2\, dp$$

where we denote the SE(3) *Fourier coefficients by* $\hat{f}_s^{(\ell,m)(\ell',m')}(p)$ *with*

$$\hat{f}_s^{\ell,m)(\ell',m'}(p) = \int_{\mathrm{SE}(3)} f((\mathbf{R},\mathbf{t}))\overline{U_s^{(\ell,m)(\ell',m')}((\mathbf{R},\mathbf{t}),p)}\, d\mathbf{R}\, d\mathbf{t}.$$

By this definition and Equation (5.15) we see that the Fourier transform of functions on the motion group can be computed using fast Fourier transforms for $\mathbb{S}^2$, $\mathbf{R}^+$ and the rotation group SO(3). This has been described in [17, Sec. 11.3] in the context of computing convolution of functions $f, g \in L^2(\mathrm{SE}(3))$. Similar to that, in the upcoming Chapter, we will explain an interesting application in which we encounter functions on $L^2(\mathrm{SE}(3))$ and describe how SE(3) Fourier transforms could be employed to compute convolution or correlation of functions on $\mathbb{R}^3$.

6 Protein-Protein Docking

6.1 Overview

The activities of a living cell are manifold, ranging from signal promotion in intra- and extracellular communication to enzyme catalysis and gene expression. Yet they all have a common purpose: transfer of biological information. A central role in these processes is played by proteins, which bring out their respective functions by interacting with each other, i.e., by forming complexes. These complexes may be transient assemblies as in case of enzyme catalysis or stable assemblies like transcriptions factors, ion channel proteins or cell surfaces.
Due to their enormous importance protein-protein interactions have been in focus of molecular biology research for several years now. In vivo proteomics methods like two-hybrid and tandem affinity purification experiments provide extensive information about interaction networks within the living cell. To cope with this vast amount of data on structure, sequences and interactions computational methods are essential to process and combine information. Methods like gene fusion analysis or phylogenetic profiling determine *which* proteins might interact (for a review, see e.g. [45]).
In contrast to that, the approach we are about to discuss here, *protein docking*, aims to predict *how* proteins interact. To understand these interactions, it is essential to determine the three-dimensional structure of the participating proteins. This is a central task of structural biology and is accomplished via X-ray crystallography, NMR spectroscopy or electron microscopy. Based on the analysis on the known structure of proteins determined by the mentioned methods, protein docking procedures calculate the structure of new formed protein complexes. This is particularly useful if we wish to examine transient protein complexes, which are too short-lived for crystallography or spectroscopy, [40]. But also for stable assemblies this computational approach is beneficial.
It has been estimated that the number of natural protein folds is of order of 1000 where each of them has about nine specific interaction partners to constitute one or several of the around 10000 basis types of interaction depending on the mutual binding site and resulting internal conformation [3]. This yields an immense number of possible protein complexes of which many thousands are yet uncovered. Determining all of them experimentally is far from feasible. Thus the prediction of possible protein complex conformations by computational docking methods is highly useful.
An essential tool is the Protein Data Bank (PDB) which stores the structure of around 12000 proteins and protein complexes determined by NMR or X-ray techniques, [11]. Provided with this large collection of structural data single proteins, we formulate the protein-protein-docking problem as the computation of atomic coordinates of a protein complex out of the atomic coordinates of the component molecules. Thus docking procedures identify the binding sites and predict the conformation of the molecules in the complex.
The first automated docking algorithm has been described in 1978 in [92]. Since then many different approaches to tackle the problem have been proposed, see e.g. [27, 40] for current reviews or very recently [75], and the references therein. The common aspect of these approaches is an optimisation problem. The solution space is the set of motions and transformations the molecules can undertake upon complex formation whereas the objective function evaluates the quality of the complex. From this common initial point a vast variety of docking approaches emerged, differing in either the choice

of instances incorporated in the solution space, or the way the molecules are described, the solution space is searched or the quality of the different complexes is evaluated.
With respect to the type of input data, we can distinguish bound and unbound docking problems. For bound docking, we start with a molecule complex and split it into two molecules, which then provide input data for the docking procedure. In contrast to that, the approach where the molecule's atomic coordinates are determined individually is called unbound docking. Naturally, the latter problem is the one we are eventually interested in. But as most molecules change their structure whilst forming a complex, it is also harder to solve as the solution space is much larger. Despite being an artificial setting bound docking is useful for testing and evaluating new docking methods.
According to the objective function of the optimisation problem we can also distinguish two main types of approaches. Methods of the first category are looking for complexes, in which the free energy is minimised. Central to these so-called thermodynamic or direct approaches are different approximations of the enthalpy and entropy of the protein complexes. The other category comprises empirical methods which use chemical or structural complementary of the proteins. These methods also exploit thermodynamics of intermolecular actions but in contrast to the direct methods they do not analyse energy minimisation itself but rather pseudo-energy, i.e., other objective functions related to energy minimisation. That is why these methods are also called indirect methods. Our algorithms, that we will present in Section 6.4 and 6.5 will use such pseudo-energy functions.
An example for such an indirect method is based on so-called *shape complementary*. Although the atoms, of which a protein is composed of, do not have a defined boundary, there exist various models to describe the shape of a protein, or rather its area of influence. In Section 6.2, we will consider some exemplary descriptors of the area of influence, e.g., the van der Waals radius which is half the distance between the nuclei of two atoms that are bound in two different molecules. We will give a new mathematical description of molecules in this context which we present in the paragraph *Molecule Regions* of Section 6.2. Using the van der Waals radius, an atom can be described as a solid ball of this radius, and hence a protein can be described as the union of such balls. This idea is incorporated e.g. in the calotte model of molecules [21]. That way we modelled a geometric shape describing the protein and we also defined its surface. In this setting the most desirable arrangement of the two molecules is the one in which the resulting complex has the smallest surface area under the constraint that the two input molecules do not overlap. The amount of surface burial gives an estimate on the volume of water forced out of the molecules upon complex formation and is as such a measure for the stability of the complex, see e.g. [22]. Other indirect criteria for good docking orientations, originating from the proteins structure are the avoidance of large cavities or the formation of hydrogen bonds.
The objective function and constraints that will be used in this work, will be explained in Section 6.3.
After a suitable representation of the constituent proteins has been found the typical docking procedure consists of two stages. The first stage is the actual optimisation problem, i.e., a search within a solution space. This search consists of a huge amount of calculations, as the objective function in general is highly non-convex possessing several local maxima and minima as we will show in Figure 6.4 in Section 6.3. That being the case we need to consider suitable search strategies. For practical purposes it is therefore beneficial if the computation of the objective function can be done relatively fast or if we restrict the size of the search space. By the search we produced a rather extensive list of about a thousand putative protein complexes sorted according to the objective function. As mentioned before, geometric shape complementary has turned out to be the dominant descriptor in docking processes [46] and hence a good choice for the objective function. Yet other criteria can be used to improve the ranking like models of desolvation, hydrophobicity, charge complementary or the formation of a high number of polar-polar contacts.

Amongst the complexes in the list the near-native complex is often observed. This will be the one we are interested in. It is not necessarily the global maximum or minimum of the objective function, but might as well be a local one or even just an instance with a high or low value, respectively. So we face the problem how to distinguish this good solution from false positives, i.e., other arrangements of two proteins showing a high energy minimisation. These false positives may occur due to the restriction of the search space, the empirical objective function or chemical constraints that have not been incorporated yet.

That is why most docking procedures have a second stage, the so-called refinement where the list from the first stage is re-ranked by incorporating additional information from previously known protein interfaces, biochemical experiments or in the simplest case visual inspection. This is the so-called data-based and data-driven docking, see e.g. [59] for more information on this stage of docking. Characteristically for this stage are time-consuming heavy-weight calculations. Due to its importance however, there are a variety of free or commercially available programs that facilitate the docking procedure. All of them incorporate more steps to evaluate the quality of a protein complex using various scoring criteria or more experimentally determined data. So far, there is no known method which can list a near native candidate complex within the top ranks in almost all cases tested. From this point of view the field of computational docking is still in its developing phase.

Fourier-Based Rigid-Body Docking In this work, we shall focus on the first stage of docking and present two methods that can be categorised as *Fourier-based rigid-body docking*. This term refers to the search strategy on one hand and to the design of the solution space of the optimisation problem on the other hand. As mentioned before the objective function is highly non-convex, hence we need to think of a way to faciliate the huge amount of computation. Realistically proteins underlie conformational changes when binding to each other, i.e, they not only move with respect to each other but also deform, shear or bend. But as in the setting of bound docking, we will neglect the latter types of transformation. That means we assume that the input coordinates of a single protein remain unchanged with respect to each other, i.e., the proteins are treated as rigid bodies. That way, the only motions the proteins can perform are rigid-body motions as in Definition 5.2.1, combinations of translations and rotations. Rigid-body motions in $\mathbb{R}^3$ have six degrees of freedom. Hence, we drastically reduced the size of our search space such that it becomes six-dimensional. Rigid-body docking is sometimes called ab initio docking, as we incorporate no other experimental structural data than the position of the atomic nuclei of the proteins.

The advantage of rigid-body docking clearly is the reduced size of the search space. The downside is that we have to tolerate some geometric mismatch in comparision to the actual protein complex as in our setting the conformations of the docked molecules can not be changed. However, this neglect of the real setting can be overcome by incorporating protein flexibility in the second stage of docking. In this work, we will not discuss protein flexibility which belongs to the field of molecular dynamics. A comprehensive review on molecular dynamics in protein docking procedures can be found e.g. in [75].

Concerning possible strategies by which we can search the solution space of three-dimensional motion, we can, again, distinguish many different approaches. They fall into three main categories. Methods of the first group use *directed search*, i.e., discrete features of the protein description are extracted. An example is the extraction of concaves and convexes (knobs and holes) of the molecule's surface which are then matched in a geometric hashing, a maximal clique algorithm, [30] or by fast bit-wise arithmetic operations [67]. Another example are triangulated surfaces, [66]. Characteristically these methods are very fast especially if the number of feature points is kept low. However

they are also likely to miss the right complex conformation if inadequate features are chosen. As the objective function highly depends on the chosen feature points and the evaluation of the quality of the found complexes can not be easily adjusted.

The second group comprises *pseudo-random search* strategies like genetic algorithms, simulated annealing or Monte-Carlo methods, e.g. [37].

And finally in the third category, *the Fourier-based search*, we collect methods that use convolutions for searching the solution space and hence employing the fast Fourier transform or one of its variations to efficiently compute the quality of a complex. Most docking procedures actually fall into this category. They not only introduce a certain freedom in designing a scoring function but also provide a desirable precision. The only drawback is that due to the nature of their brute-force search strategy they are usually slower than the directed search methods from the other two categories.

Many, if not most, existing Fourier-based docking algorithms use a regular discrete three-dimensional cartesian grid onto which the molecules are projected. Then a weighting function is used to divide the grid cells into two groups depending on whether or not the underlying atom belongs to the molecules surface. The correlation of the discretised and weighted atoms serves as objective function for the optimisation problem. The correlation between pairs of grid cells is computed via fast Fourier transforms thus implicitly searching over the space of 3D - translation. The remaining rotational degrees of freedom however need to be searched in a brute force global search. Such an approach has been first published by [47] in 1992. Since then, this approach has been adapted and improved many times. An overview on these grid-based docking approaches can be found in [27].

More recently, a set of methods arose that omit the regular equispaced grid, replacing it with a non-equispaced Cartesian one as in [7, 15] or a spherical one as in [55, 76, 80]. The docking algorithm we will consider in Section 6.4 is based on the one from [15]. We will summarise their method and use it for different choices of objective functions and protein models. We will also modify the original algorithm to fit to our descriptions.

The new approach which we present in Section 6.5 starts by following the lines of [76] and [80] using spherical harmonic functions and classical orthogonal polynomials to model molecular shapes. For the first time, we employ the fast $\mathrm{SO}(3)$ Fourier transform algorithms to solve the docking problem. We can thus overcome the limitiations of the algorithms mentioned as they use only spherical harmonics of low orders due to computational complexity. We will also show the connection between the docking procedure and Fourier transforms on the motion group $\mathrm{SE}(3)$ and model proteins using expansions in terms of the unitary representations of the motion group.

In any approach, we construct an objective function that takes three-dimensional motions as input and that is evaluated by global search, not for all possible motions but for sampled motions on a grid on the motion group $\mathrm{SE}(3)$. As a result of a Fourier-based docking procedure, we obtain local and possibly global minima and maxima of the objective function on this grid. After this step however the evaluation of the objective function may be locally refined by a directed search. A suggestion for local refinement concludes our considerations of protein-protein docking in Section 6.6.

6.2 Protein Modelling

The input data for the protein-protein docking are the coordinates of the atomic kernels a molecule is comprised of. In this section, we explain how to obtain a description of the molecule's area of influence out of these coordinates, as well as a description of the molecule's surface.

The molecules, we use in the docking procedure will be called molecule A and molecule B. Whenever we derive some expression or function depending on A, we assume an analogous definition for molecule B, unless otherwise stated.

Areas of Influence of Atoms As the first step in modelling molecules, we describe the area of influence of a single atom. In a typical description, the atom is considered as a ball of certain radius. This radius, the van der Waals radius, is half the distance between two atoms of the same chemical element, at which the repulsive van der Waals force equals the attractive van der Waals force between the two. The van der Waals radius is determined by measuring how close the nuclei of two atoms, bound in two different molecules, can move towards each other. A typical choice of r would be the van der Waals radius of hydrogen atoms which is about $r = 1.2\text{Å}$, see [13].
Without loss of generality, we consider an atom positioned at the point of origin. We examine three different choices of functions $\kappa(\mathbf{x})$ describing the influence of the atom to its neighbourhood. They are all based on empirical descriptions of the energy density of an atom and are rough approximations for the interaction energy of two atoms. We denote the closed ball around $z \in \mathbb{R}^3$ with radius r by $B_r(\mathbf{z}) = \{\mathbf{x} \in \mathbb{R}^3 \mid \|\mathbf{x} - \mathbf{z}\| \leqslant r\}$. The first choice of $\kappa(\mathbf{x}) = \kappa_W(\mathbf{x})$ approximates the energy density by a characteristic function of a closed ball $B_r(\mathbf{0})$ of van der Waals radius r.
Next, $\kappa(\mathbf{x}) = \kappa_G(\mathbf{x})$ is a Gaussian-like kernel approximation of the electron density distribution of a single atom. Finally, $\kappa(\mathbf{x}) = \kappa_L(\mathbf{x})$ is derived from the Lennard-Jones potential. In addition to the term which describes the van der Waals forces, it also contains a term describing repulsion at short ranges from the atomic kernel due to overlapping electron orbitals, e.g. [6, p. 470]. These are in particular:

i) Using the van der Waals ball $B_r(\mathbf{0})$, we can write

$$\kappa_W(\mathbf{x}) = \chi_{B_r(\mathbf{0})}(\mathbf{x}) \tag{6.1}$$

with the characteristic function χ.

ii) The second choice is

$$\kappa_G(\mathbf{x}) = e^{\beta\left(1 - \frac{\|\mathbf{x}\|^2}{r^2}\right)} \tag{6.2}$$

for some parameter β. The parameter β controls the sharpness of the kernel's peak. A typical choice from literature is $\beta = 2.3$ (see [25, 33]).

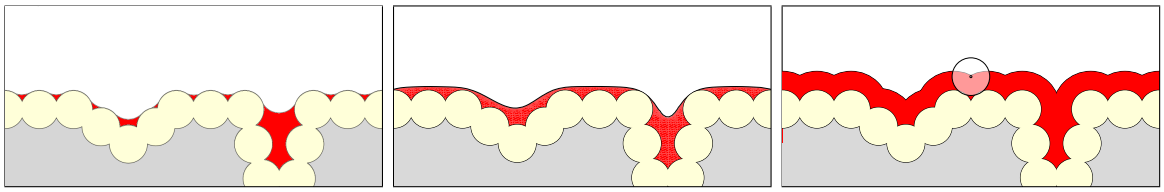
iii) The third choice is given by

$$\kappa_L(\mathbf{x}) = -\left(\frac{r}{\|\mathbf{x}\|}\right)^{12} + \left(\frac{r}{\|\mathbf{x}\|}\right)^{6}, \tag{6.3}$$

as the so-called (12,6)-Lennard-Jones potential.

The next step is to describe molecules composed by single atoms.

Modelling Molecular Surfaces Consider a molecule A that consists of I_A different atoms. We define an index set $\mathcal{A} = \{k \mid k = 1, \ldots, I_A\}$ to number the atoms that are part of A. The centre of the k-th atom will be denoted by $\mathbf{x}_k \in \mathbb{R}^3$. These $\mathbf{x}_k$ for $\mathbf{k} \in \mathcal{A}$ are the input coordinates of the protein-protein docking procedure. Every atom centre $\mathbf{x}_k$ for $k \in \mathcal{A}$ is assigned a ball $B_r(\mathbf{x}_k)$, where



(a) (b) (c)

Figure 6.1: Examples of surface types for a cross section of an example molecule. The interior of the molecule is coloured in grey while the surface layer is coloured yellow. The boundary line between the red region and the white coloured exterior of the molecule describe ∂W_{con} in (a), ∂W_{iso} in (b) and ∂W_{SAS} in (c).

in general r depends on k.
Let us remark that this ball can be described by

$$\begin{aligned} B_r(\mathbf{x}_k) &= \{\mathbf{x} \in \mathbb{R}^3 \mid \kappa_W(\mathbf{x}-\mathbf{x}_k) \geqslant c\} \quad \text{with } c \in (0,1], \\ B_r(\mathbf{x}_k) &= \{\mathbf{x} \in \mathbb{R}^3 \mid \kappa_G(\mathbf{x}-\mathbf{x}_k) \geqslant 1\} \quad \text{or by} \\ B_r(\mathbf{x}_k) &= \{\mathbf{x} \in \mathbb{R}^3 \mid \kappa_L(\mathbf{x}-\mathbf{x}_k) \geqslant 0\}. \end{aligned}$$

Now, the union

$$W_{\mathcal{A}} = \bigcup_{k \in \mathcal{A}} B_r(\mathbf{x}_k)$$

is regarded as the domain filled by the atoms of the molecule, or the area of influence of all atoms of the molecule, [21]. It can be considered as the rigid-body description of the molecule. For the docking procedure we are especially interested in the boundary of the domain $W_{\mathcal{A}}$, as this is the part of the molecules that is exposed to other molecules.
The concept of the surface of a molecule is not mathematically precise. In particular since the molecule described by $W_{\mathcal{A}}$ is a porous cavernous union of balls and the surface $\partial W_{\mathcal{A}}$ of the domain $W_{\mathcal{A}}$ in a mathematical sense has nothing to do with the surface a chemist would have in mind. Therefore, a common approach, found in [58], uses only a verbal description of a surface, called van der Waals surface, which is the molecular surface accessible to water in a practical chemical sense. The atoms accessible to water are called the *surface layer*. Here we present three more mathematical surface concepts.

i) Connolly surface:
The concept of this surface as been introduced in [19], in a constructive manner. Here, we introduce our own description of the Connolly surface in terms of a mathematical definition.
The domain $W_{\mathcal{A}}$ is enlarged by all points which never are reached by a ball of fixed radius r rolling along $W_{\mathcal{A}}$. That is

$$\widetilde{W}_{\text{con}} = \{\mathbf{x} \in \mathbb{R}^3 \mid \mathbf{x} \notin B_r(\mathbf{y}) \text{ for all } \mathbf{y} \text{ with } B_r(\mathbf{y}) \cap W_{\mathcal{A}} = \emptyset\},$$

which is identical to

$$\widetilde{W}_{\text{con}} = \mathbb{R}^3 \setminus \bigcup_{y \in \mathbb{R}^3} \{B_r(\mathbf{y}) \mid B_r(\mathbf{y}) \cap W_{\mathcal{A}} = \emptyset\}.$$

The domain $\widetilde{W}_{\text{con}}$ may have inner cavities, which are completely surrounded by $\widetilde{W}_{\text{con}}$. These cavities are not reachable for reagents, and they have to be considered as part of the interior of the molecule. Hence the Connolly domain is

$$W_{\text{con}} = \bigcap_{V \subseteq \mathbb{R}^3} \{V \mid \widetilde{W}_{\text{con}} \subseteq V, \partial V \text{ simply connected }\}.$$

Now, the Connolly surface is ∂W_{con}, see Figure 6.1(a).

ii) Isovalue surface:
Here, the domain $W_{\mathcal{A}}$ is replaced by sub-level sets of the sum of influence functions, i.e.,

$$\widetilde{W}_{\text{iso}} = \{\mathbf{x} \in \mathbb{R}^3 \mid \sum_{k \in \mathcal{A}} \kappa_G(\mathbf{x} - \mathbf{x}_k) \geqslant c\}.$$

Analogously to the Connolly domain, inner cavities of $\widetilde{W}_{\text{iso}}$ are filled to W_{iso}. The actual isovalue surface is now defined by ∂W_{iso}, see Figure 6.1(b).
More generally, κ_G can be replaced by non-continous functions of influence like κ_W or non-monotonous functions like κ_L, but this is suitable for particular applications only.

iii) Solvent accessible surface:
While the Connolly surface used the contact areas between a molecule and an artificially added ball of radius r rolling along the surface, the solvent accessible surface (SAS) uses the centre of this ball to define the molecule's surface. Consider the set

$$W_{\text{SAS}} = \{\mathbf{x} \in \mathbb{R}^3 \mid \text{ there is an } \mathbf{y} \in W_{\text{con}} : \|\mathbf{x} - \mathbf{y}\| \leqslant r\}.$$

The actual SAS is now given by ∂W_{SAS}. The SAS is depicted in Figure 6.1(c).
Note that, we omitted the definition of $\widetilde{W}_{\text{SAS}}$, that is a surface still containing inner cavities. By using W_{con} instead of $\widetilde{W}_{\text{con}}$ to define the solvent accessible surface, we already got rid of these inner cavities.

Molecule Regions The domain $W_{\mathcal{A}}$ is now divided into two regions with respect to the just defined surfaces: the surface layer or *skin* $\mathcal{S}_{\mathcal{A}}$, described by the outermost layer of atoms of the molecule and the *core* $\mathcal{C}_{\mathcal{A}}$, the interior part of the molecule. Furthermore, we distinguish the surrounding of the molecule between the *exterior skin* $\mathcal{E}_{\mathcal{A}}$, an additional layer around the skin $\mathcal{S}_{\mathcal{A}}$, and the exterior. Based on our descriptions of molecular surface, we give new formal descriptions of these molecular regions.
The indices of the atoms touching the Connolly surface ∂W_{con} are combined in

$$\partial\mathcal{A} = \{k \mid B_r(\mathbf{x}_k) \cap \partial W_{\text{con}} \neq \emptyset\}.$$

Now, the skin is defined by

$$\mathcal{S}_{\mathcal{A}} = \bigcup_{k \in \partial\mathcal{A}} B_r(\mathbf{x}_k).$$

The core is the rest of the molecule

$$\mathcal{C}_{\mathcal{A}} = W_{\mathcal{A}} \backslash \mathcal{S}_{\mathcal{A}}.$$

The exterior is given by

$$\mathcal{E}_{\mathcal{A}} = \bigcup_{\mathbf{y} \in \partial W_{\text{SAS}}} B_r(\mathbf{y}).$$

We can infer some more properties like

$$\mathcal{S}_A \subseteq W_A, \qquad \mathcal{C}_A \subseteq \bigcup_{k \in A \setminus \partial A} B_r(\mathbf{x}_k), \qquad \mathcal{E}_A \cap W_A = \partial W_{\text{con}}.$$

If we consider a molecule in which the atomic areas of influence $B_r(\mathbf{x}_k)$ do not overlap for any $\mathbf{x}_k$ with $k \in \mathcal{A}$ then, due to their definitions, $\mathcal{S}_A$ and $\mathcal{C}_A$ are unions of a finite number of separated balls while $\mathcal{E}_A$ resembles a layer, i.e., a union of infinitely many overlapping balls. We will get back to this issue in Section 6.3.
In the next step, we will define functions that assign different weights to the molecule regions. These functions will be called affinity functions and they are needed in the next section to construct a objective function for the optimisation problem of protein-protein docking. We provide two different approaches to define affinity functions.

i) Symmetric affinity functions:
Consider the characteristic function $\chi_{\mathcal{S}_A}$ of the molecule's skin and the characteristic function $\chi_{\mathcal{E}_A}$ of the exterior skin. The affinity functions $Q^{\chi_{\mathcal{S}_A}} : \mathbb{R}^3 \to \mathbb{R}$ and $Q^{\chi_{\mathcal{E}_A}} : \mathbb{R}^3 \to \mathbb{R}$ are defined as the sums

$$\begin{aligned} Q^{\chi_{\mathcal{S}_A}}(\mathbf{x}) &= \sum_{k \in \mathcal{A}} \chi_{\mathcal{S}_A}(\mathbf{x}_k)\kappa(\mathbf{x} - \mathbf{x}_k) \\ Q^{\chi_{\mathcal{E}_A}}(\mathbf{x}) &= \int_{\mathcal{E}_A} \chi\, \kappa(\mathbf{x} - \mathbf{y})\, d\mathbf{y} \end{aligned} \tag{6.4}$$

where $\mathbf{x}_k$ denote the coordinates of the atomic centres, χ denotes a scaling factor and κ can be set to any of the functions (6.1) or (6.2), (6.3) describing the area of influence of a single atom. We define analogous functions for molecule B. This approach is called symmetric as it will yield the same results after exchanging the two molecules with each other. In contrast to that the outcome of the second approach depends on the order of the molecules.

ii) Asymmetric affinity functions:
Instead of the characteristic functions used in the symmetric approach, we define two functions that assign different values to the respective regions depending on the molecule by

$$\gamma_A(\mathbf{x}) = \begin{cases} \rho i & \text{if } \mathbf{x} \in \mathcal{C}_A, \\ 1 & \text{if } \mathbf{x} \in \mathcal{S}_A, \\ 0 & \text{otherwise} \end{cases} \qquad \text{and} \qquad \gamma_B(\mathbf{x}) = \begin{cases} \rho i & \text{if } \mathbf{x} \in \mathcal{C}_B \cup \mathcal{S}_B, \\ 1 & \text{if } \mathbf{x} \in \mathcal{E}_B, \\ 0 & \text{otherwise} \end{cases}$$

where $\rho \gg 1$. Based on the γ_A and γ_B, we establish the affinity functions $Q^{\gamma_A} : \mathbb{R}^3 \to \mathbb{C}$ and $Q^{\gamma_B} : \mathbb{R}^3 \to \mathbb{C}$ as the sums

$$\begin{aligned} Q^{\gamma_A}(\mathbf{x}) &= \sum_{k \in \mathcal{A}} \gamma_A(\mathbf{x}_k)\kappa(\mathbf{x} - \mathbf{x}_k) \\ Q^{\gamma_B}(\mathbf{x}) &= \sum_{k \in \mathcal{B}} \gamma_B(\mathbf{x}_k)\kappa(\mathbf{x} - \mathbf{x}_k) \end{aligned} \tag{6.5}$$

where $\mathbf{x}_k$ denote the coordinates of the atomic centres and κ is again set to a functions describing the area of influence of a single atom.

6.3 Docking Procedure

We already discussed that the protein-protein docking problem can be thought of as an optimisation problem. With respect to this, we will now consider ways to construct an objective function which will be, in this case, maximised. This function will allow us to determine arrangements of two proteins in which their two areas of influence will have little overlap while the circumference of the resulting area is kept small as well.
To arrange two molecules $\mathcal{A}$ and $\mathcal{B}$, they need to be moved. The assumption molecules are rigid bodies, restricts the motion to rotations and translations in three-dimensional space. Recall from Section 5.2 that any rigid-body motion, i.e., the application of an element $(\mathbf{t},\mathbf{R}) \in \mathrm{SE}(3)$ to a vector $\mathbf{x} \in \mathbb{R}^3$ can be written as a rotation followed by a translation. Hence,

$$f((\mathbf{R},\mathbf{t}) \circ \mathbf{x}) = f(\mathbf{R}^T\mathbf{x} - \mathbf{t})$$

describes a three-dimensional rigid-body motion. We formulate the protein-protein docking problem depending on the classification of the molecule regions as follows.
Let $\mathbf{x}_k$, for $k \in \mathcal{A}$ and $\mathbf{y}_j$, for $j \in \mathcal{B}$ be the coordinates of the atomic centres of two molecules A and B.

i) For the symmetric classification of the molecule regions, we give an objective function

$$\begin{aligned} C((\mathbf{R},\mathbf{t})) &= \int Q^{\chi_{\mathcal{S}_A}}(\mathbf{x}) Q^{\chi_{\mathcal{E}_B}}(\mathbf{R}^T\mathbf{x} - \mathbf{t})\,d\mathbf{x} + \int Q^{\chi_{\mathcal{E}_A}}(\mathbf{x}) Q^{\chi_{\mathcal{S}_B}}(\mathbf{R}^T\mathbf{x} - \mathbf{t})\,d\mathbf{x} \\ &- \rho \int Q^{\chi_{\mathcal{S}_A}}(\mathbf{x}) Q^{\chi_{\mathcal{S}_B}}(\mathbf{R}^T\mathbf{x} - \mathbf{t})\,d\mathbf{x} \end{aligned} \tag{6.6}$$

with $(\mathbf{R},\mathbf{t}) \in \mathrm{SE}(3)$ and $\rho \gg 1$. This function computes the convolution of the fixed molecule A and the molecule B which is moved and rotated. Positions $\mathbf{x} \in \mathbb{R}^3$ where the skin of one molecule overlaps the exterior skin of the other one contribute positive values to the integral. These are favourable arrangements, as the actual molecules are close together without overlapping each other. In contrast to that, positions that are part of the skins of both molecules contribute negative values to the objective function to penalise molecular overlaps.
Hence, *the solution* of the protein docking problem is defined as the pair $(\mathbf{R}_{\max}, \mathbf{t}_{\max})$ with

$$C((\mathbf{R}_{\max}, \mathbf{t}_{\max})) = \max_{(\mathbf{R},\mathbf{t}) \in \mathrm{SE}(3)} C((\mathbf{R},\mathbf{t})). \tag{6.7}$$

ii) For the asymmetric classification of the molecule regions, we give an objective function

$$C((\mathbf{R},\mathbf{t})) = \mathrm{Re}\left(\int Q^{\gamma_A}(\mathbf{x}) Q^{\gamma_B}(\mathbf{R}^T\mathbf{x} - \mathbf{t})\,d\mathbf{x} \right) \tag{6.8}$$

with $(\mathbf{R},\mathbf{t}) \in \mathrm{SE}(3)$. Again, molecule A is kept fix, while the motion is applied to B. For an understanding of the function $C((\mathbf{R},\mathbf{t}))$, we examine the product

$$\gamma_A(\mathbf{x}_k)\gamma_B(\mathbf{x}_k) = \begin{cases} 1 & \text{if } \mathbf{x}_k \in \mathcal{S}_A \cap \mathcal{E}_B \\ \rho i & \text{if } \mathbf{x}_k \in \mathcal{S}_A \cap (\mathcal{C}_B \cup \mathcal{S}_B) \\ \rho i & \text{if } \mathbf{x}_k \in \mathcal{C}_A \cap \mathcal{E}_B \\ -\rho^2 & \text{if } \mathbf{x}_k \in \mathcal{C}_A \cap (\mathcal{C}_B \cup \mathcal{S}_B). \end{cases}$$

The arrangements we are interested in are those where the skin of A overlaps the exterior skin of B while overlaps of the core of A with the molecule B are penalised. The remaining cases of overlaps are of no interest here. The solution of the protein-protein docking problem is again the pair $(\mathbf{R}_{\max}, \mathbf{t}_{\max})$ from (6.7).

To solve the docking problem, we need to compute $C((\mathbf{R},\mathbf{t}))$ for a certain number of different motions. The search space for the solution of the protein-protein docking problem is six-dimensional as a rigid-body motion has six degrees of freedom. For that reason, we denote the total number of motions used in the computation by M^6, i.e., for each of the six parameters of a motion M different values are inserted. Moreover, the number of atoms in the molecules A and B are given by the size of the respective index sets as $|\mathcal{A}|$ and $|\mathcal{B}|$.
In a straight-forward attempt to compute $C((\mathbf{R},\mathbf{t}))$ for all given motions we would have to evaluate the affinity function $Q(\mathbf{x})$ for molecule A which takes $\mathcal{O}(|\mathcal{A}|)$ flops. The same holds true for the evaluation of $Q(\mathbf{R}^T\mathbf{x}-\mathbf{t})$ for one motion $(\mathbf{R},\mathbf{t}) \in SE(3)$. As we examine M^6 different motions, this yields $\mathcal{O}(|\mathcal{A}||\mathcal{B}|M^6)$ flops in total. Finding the maximum value among all $C((\mathbf{R},\mathbf{t}))$ will not contribute to the asymptotic complexity.
In a realistic setting the molecules A and B consist of a few thousand atoms, while we might want to compute 50 different choices per parameter of a rigid-body motion. Considering this, the complexity of a direct computation is too high for a feasible computation and a fast algorithm is needed. On the other hand, a lower bound for number of computation steps is given by the number of input values of the problem which is $|\mathcal{A}| + |\mathcal{B}| + M^6$. Although, we will not reach this lower bound, we shall introduce fast algorithms to solve the docking problem in Sections 6.4 and 6.5 which have a much more favourable complexity than the naive $\mathcal{O}(|\mathcal{A}||\mathcal{B}|M^6)$ flops.

But first, to gain more insight into the problem of protein-protein docking, we examine a simplified scenario of the problem.

Simplified Docking Example Consider two molecules A and B of which we cut out one slice each. Phrased differently, we want to analyse the protein-protein docking procedure for two-dimensional molecules instead of three-dimensional ones. Moreover, their atomic kernels satisfy $\mathbf{x}_k \in \mathbb{Z}^2$ for $k \in \mathcal{A} \cup \mathcal{B}$ and have van der Waals radius $r = \frac{1}{2}$. As possible motions of the proteins we only consider integer-valued translations.
Figure 6.2 presents a visualisation of this model for the two molecules, we are using in this example. With these restrictions the domain W_A is sufficiently described by the grid points $W_{A,\mathbb{Z}^2} = W_A \cap \mathbb{Z}^2$. Now, let

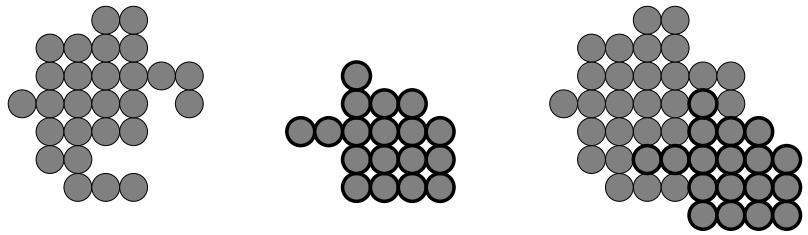
$$\mathcal{N}(\mathbf{x}_k) = \{\mathbf{x}_k, \mathbf{x}_k + (1,0)^T, \mathbf{x}_k - (1,0)^T, \mathbf{x}_k + (0,1)^T, \mathbf{x}_k - (0,1)^T\}$$

describe the five-point stencil of $\mathbf{x}_k$, i.e. it contains all neighbours of $\mathbf{x}_k$ and $\mathbf{x}_k$. Using $\mathcal{N}$ we can easily identify the boundary of the molecule and consequently assign the atoms either to the skin $\mathcal{S}_A$ or the core $\mathcal{C}_A$ which in this case are completely described by

$$\mathcal{C}_{A,\mathbb{Z}^2} = \{\mathbf{x} \in W_{A,\mathbb{Z}^2} \mid \mathcal{N}(\mathbf{x}) \subseteq W_A\}, \qquad \text{and} \qquad \mathcal{S}_{A,\mathbb{Z}^2} = W_{A,\mathbb{Z}^2} \backslash \mathcal{C}_{A,\mathbb{Z}^2}.$$

Recall, that the exterior skin $\mathcal{E}_A$, was defined as a layer of infinite many balls that cover the skin. In this setting here however, we can conveniently describe the exterior skin $\mathcal{E}_A = \mathcal{E}_{A,\mathbb{Z}^2}$ as a finite union of balls, as well, by

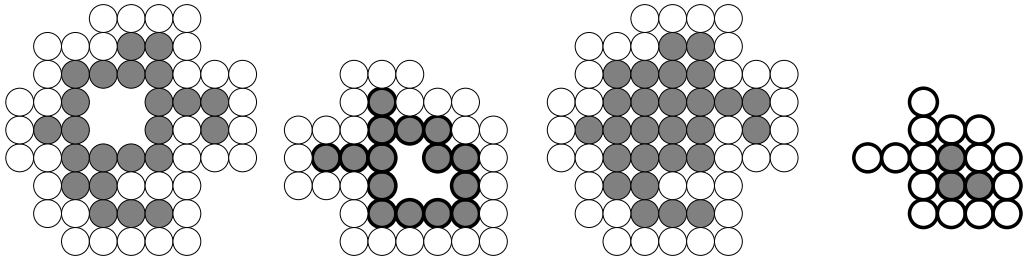
$$\mathcal{E}_{A,\mathbb{Z}^2} = \{\mathbf{x} \in \mathbb{Z}^2 \backslash W_A \mid \mathcal{N}(\mathbf{x}) \cap W_A \neq \emptyset\}.$$



(a) Two example molecules $\mathcal{A}$ and $\mathcal{B}$.

(b) The new formed complex has the smallest possible border avoiding overlap of the two molecules.

Figure 6.2: Simplified Docking Example.



(a) The symmetric approach: A favourable arrangement of the molecules has overlaps between atoms of the exterior skin (white) and atoms of the skin (grey).

(b) The asymmetric approach: A favourable arrangement has many overlaps between white coloured atoms, i.e, atoms with $\gamma_A(\mathbf{x}_k) = 1$ and little or no overlaps of grey coloured atoms with $\gamma_A(\mathbf{x}_k) = \rho i$.

Figure 6.3: Region classification of the molecules.

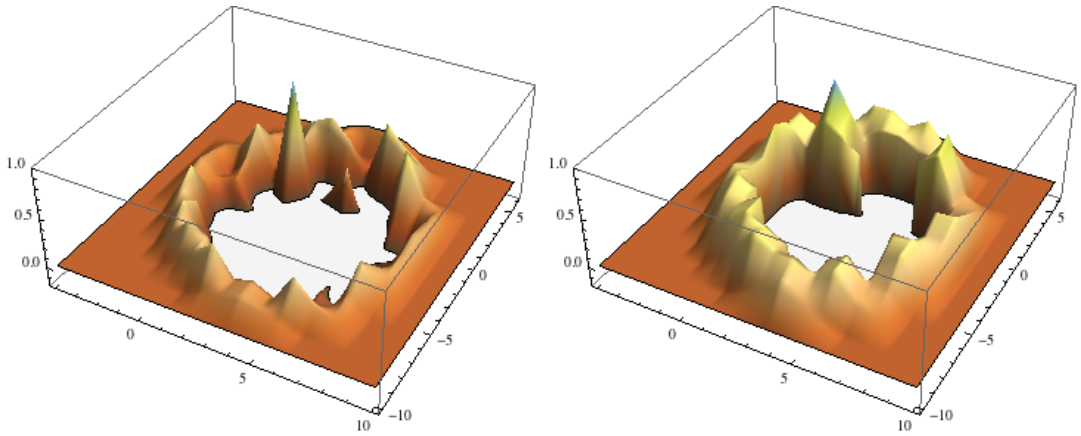
We can assign the molecule regions of B analogously.
The objective functions $C((\mathbf{R},\mathbf{t}))$, we defined in (6.6) and (6.8) to solve the docking problem (6.7) will also be slightly modified to exploit the chosen simplifications. On one hand the integrals are exchanged for sums over $\mathbb{Z}^2$ and on the other hand we shall omit rotations. This yields

$$\begin{aligned} C(\mathbf{t}) &= \sum_{\mathbf{x}\in\mathbb{Z}^2} Q^{\chi_{\mathcal{S}_A}}(\mathbf{x}) Q^{\chi_{\mathcal{E}_B}}(\mathbf{x}-\mathbf{t}) + \sum_{\mathbf{x}\in\mathbb{Z}^2} Q^{\chi_{\mathcal{E}_A}}(\mathbf{x}) Q^{\chi_{\mathcal{S}_B}}(\mathbf{x}-\mathbf{t}) \\ &- \rho \sum_{\mathbf{x}\in\mathbb{Z}^2} Q^{\chi_{\mathcal{S}_A}}(\mathbf{x}) Q^{\chi_{\mathcal{S}_B}}(\mathbf{x}-\mathbf{t}), \qquad \text{for } \mathbf{t}\in\mathbb{Z}^2 \end{aligned} \tag{6.9}$$

for the symmetric approach, and

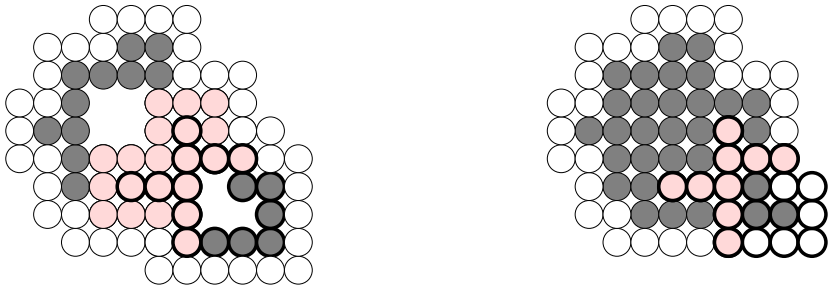
$$C(\mathbf{t}) = \mathrm{Re}\left(\sum_{\mathbf{x}\in\mathbb{Z}^2} Q^{\gamma_A}(\mathbf{x}) Q^{\gamma_B}(\mathbf{x}-\mathbf{t})\right), \qquad \text{for } \mathbf{t}\in\mathbb{Z}^2 \tag{6.10}$$

for the asymmetric approach. Note that the choice of function κ describing the atomic area of influence, has no influence on the solution of the docking problem, in this particular setting. In Figure 6.3 we show the classification of the molecule regions, for both the symmetric and the asymmetric approach.



(a) The objective function in the symmetric approach.

(b) The objective function in the asymmetric approach.



(c) The resultant complex with symmetric region classification.

(d) The resultant asymmetric region classification.

Figure 6.4: The plots show the values of the normalised correlation function $\frac{C(\mathbf{t})}{C(\mathbf{t}_{max})}$ for the two molecules from Figure 6.2 and the resultant protein complex after applying the translation $\mathbf{t}_{max}$ which is the solution of the docking problem. In (a) and (b), we computed 441 translations $\mathbf{t} \in \mathbb{Z}^2$ and interpolated the remaining data. Both region classification approaches find the same translation to produce the best arrangement, shown in (c) and (d). The red coloured grid cells are contributing positively to the function $C(\mathbf{t})$. In this example, there are no undesired overlaps.

The objective functions $C(\mathbf{t})$ from (6.10) and (6.9) have been computed for $\mathbf{t} \in \mathbb{Z}^2$ with $\mathbf{t} \in [-10, 10]^2$ and are displayed in Figure 6.4 (a), (b). These plots demonstrate the highly non-convex behaviour of the objective functions. The function values have been plotted for $C(\mathbf{t}) \geqslant 0$ only, which leads to the crater in the middle of the plot that occurs when the two proteins have large overlaps. All minima of the objective function are contained in this area. This negative crater containing highly indesired translations is directly bounded by a ring-like structure on which numerous local maxima of the objective function are located. These are translations leading to favourable arrangements. Among the translations positioned on the ring, lies the one we are eventually interested in. Moving away from the ring the objective function becomes zero. At these positions the molecules will not overlap at all.
These plots illustrate why it is sensible to perform a global search for the maxima of the objective function, even in this very simple example. If we employed a directed search instead, we would hardly find the correct value among the tremendous amount of local maxima encountered.

One might interject that, we could use search at least the ring only to find the maxima. But recall, that we are in a simplified setting here. In the realistic protein-protein docking, when four more degrees of freedom are introduced, the objective function would exhibit even more local maxima rendering the directed search even more difficult. Cast in this light, it seems sensible to employ a global search over the whole space of possible molecule motions.
The translation $\mathbf{t}$ that maximises the function $C(\mathbf{t})$ leading to the arrangement of molecules depicted in Figure 6.4 (c),(d). This figure also illustrates the different overlaps.
This very simple example also gives us an understanding of some more of the problems, we will encounter during the docking procedure other then the non-convex objective function. Among these problems are occurring errors and unfavourable complexity, the latter surely a consequence of the global search. Errors or inaccuracies arise in different situation here, for example from considering the atoms only positions $\mathbf{x}_k \in \mathbb{Z}^2$. In realistic molecules however, atoms are not positioned equispaced. We will consider this in the next section. Note that already the input data from the Protein Data Base may be defective due to errors of measurement but we will neglect this type of error here.
Although, so far we have not exploited the convolution-like nature of the objective function. The direct approach used in this example is computationally expensive despite the simplifications made. For the molecule's motion we analysed two degrees of freedom, that leads to evaluating M^2 times the objective function $C(\mathbf{t})$ and therefore to a total complexity of $\mathcal{O}(|\mathcal{A}||\mathcal{B}|M^2)$ flops. Already for the small example in Figure 6.2 computing the correlations in Figure 6.4 is expensive. For the actual docking problem we have to add one more dimension to the proteins and include one more direction of translation, as well as three-dimensional rotations. Computing this with the direct evaluation of the objective function is far from feasible.

6.4 Fast Translational Matching

In this section, we introduce an algorithm to efficiently solve the protein-protein docking problem from (6.7) by means of fast Fourier transforms. The presented algorithm, called *fast translational matching* has been introduced by [15], and newly extended in [7]. There the asymmetric affinity functions (6.5) were used to describe the molecule regions. The resulting objective function

$$C((\mathbf{R},\mathbf{t})) = \mathrm{Re}\left(\int Q^{\gamma_A}(\mathbf{x})Q^{\gamma_B}(\mathbf{R}^T\mathbf{x}-\mathbf{t})\,d\mathbf{x}\right) \tag{6.11}$$

is computed for a set of translations $\mathbf{t} \in \mathbb{R}^3$ and a set of rotations $\mathbf{R} \in \mathrm{SO}(3)$ by using the NFFT algorithm and its adjoint [71]. We shall briefly summarise this method and apply it to our own protein descriptions. Moreover, we will adapt the algorithm to the symmetric region classification for the first time and propose improvements of the algorithm from [15].
The key idea, we are using to efficiently compute the integral (6.11), is the following elementary property of a correlation.

Lemma 6.4.1. *Let*

$$f(\mathbf{x}) = \sum_{\mathbf{k}\in\mathbb{Z}^3} \hat{f}_{\mathbf{k}} e^{2\pi i \mathbf{x}^T\mathbf{k}}, \qquad g(\mathbf{x}) = \sum_{\mathbf{k}\in\mathbb{Z}^3} \hat{g}_{\mathbf{k}} e^{2\pi i \mathbf{x}^T\mathbf{k}}$$

be given. Then their correlation $C(\mathbf{t}) = \int_{\mathbb{R}^3} f(\mathbf{x})g(\mathbf{t}+\mathbf{x})\,d\mathbf{x}$ *for* $\mathbf{t} \in \mathbb{R}^3$ *has the Fourier expansion*

$$C(\mathbf{t}) = \sum_{\mathbf{k}\in\mathbb{Z}^3} \hat{f}_{\mathbf{k}}\hat{g}_{\mathbf{k}} e^{2\pi i \mathbf{t}^T\mathbf{k}}.$$

Recall that a motion $(\mathbf{R},\mathbf{t}) \in \mathrm{SE}(3)$ can be separated into a rotation followed by a translation, i.e., we have $(\mathbf{R},\mathbf{t}) = (\mathbf{I},\mathbf{t}) \circ (\mathbf{R},\mathbf{0})$, cf. Definition 5.2.1. That, on the other hand, means that for fixed $\mathbf{R} \in \mathrm{SO}(3)$ the function $C((\mathbf{R},\mathbf{t}))$ in (6.11) is a correlation of the two affinity functions $Q^{\gamma_A}, Q^{\gamma_B} : \mathbb{R}^3 \to \mathbb{C}$ and we can compute it by means of fast Fourier transforms applying Lemma 6.4.1.
Unfortunately, we need to recalculate the correlation for all desired choices of rotations $\mathbf{R} \in \mathrm{SO}(3)$. As only the different translations will be computed by fast algorithms, this docking procedure is called fast translational matching.
To describe a motion, we are using the Euler angles for the rotation $\mathbf{R} \in \mathrm{SO}(3)$ and Cartesian coordinates for the translation $\mathbf{t} \in \mathbb{R}^3$. For each of the six parameters we choose M different values. The motions that result from all possible combinations of these choices make up the set $\mathcal{M}$. More specifically

$$\mathcal{M} = \{(\mathbf{R}(\alpha_{i_1}, \beta_{i_2}, \gamma_{i_3}), (x_{i_4}, y_{i_5}, z_{i_6})^T) \in \mathrm{SE}(3) \mid i_1, \dots, i_6 = 1, \dots, M\}. \tag{6.12}$$

Hence, $\mathcal{M}$ contains M^6 motions for which we compute the correlation $C((\mathbf{R},\mathbf{t}))$. These motions are put together from M^3 rotations and M^3 translations.
Consider two molecules A and B. In contrast to the example in Section 6.3, their atom centres are not positioned on an equispaced grid. Furthermore, by choosing the functions κ_G from (6.2) to describe the atomic area of influence, the affinity functions of the molecules will be of unbounded support. This needs to be considered for the expansion into a Fourier sum that is needed to employ Lemma 6.4.1. In the following, we shall describe the necessary modifications of the affinity functions in the fast translational matching approach. Exemplarily, we are using the Gaussian-like kernel function (6.2) to describe the atomic area of influence.
First, we relocate and scale the molecules such that their domains W_A and W_B fit into the unit cube $\left[-\frac{1}{2}, \frac{1}{2}\right]^3$. To do this, we compute the diameters

$$p_A = \max_{j,k \in \mathcal{A}} \|\mathbf{x}_j - \mathbf{x}_k\| \qquad \text{and} \qquad p_B = \max_{j,k \in \mathcal{B}} \|\mathbf{y}_j - \mathbf{y}_k\|$$

of the molecules and the molecule centres

$$c_A = \frac{1}{|\mathcal{A}|} \sum_{k \in \mathcal{A}} \mathbf{x}_k \qquad \text{and} \qquad c_B = \frac{1}{|\mathcal{B}|} \sum_{k \in \mathcal{B}} \mathbf{y}_k.$$

We also need to consider a certain margin w that depends on parameters chosen in the function κ_G. By adding w to the molecules' diameter, we will ensure that the areas of influence of the outermost atoms are still included in the unit cube. Consequently, we calculate the modified atomic centres by

$$\mathbf{z}_k = \frac{\mathbf{x}_k - \mathbf{c}_A}{2p}, \quad k \in \mathcal{A} \qquad \text{and} \qquad \tilde{\mathbf{z}}_k = \frac{\mathbf{y}_k - \mathbf{c}_B}{2p}, \quad k \in \mathcal{B}$$

with $p = \max\{p_A, p_B\} + w$. A suggestion to compute the margin w is given by

$$w = \sqrt{r^2 \left(1 - \frac{\ln 10^{-16}}{\beta}\right)}$$

where r is the van der Waals radius of the atom and the parameter β controls the sharpness of the peak of κ_G. Using the above formula, we cut off the area of influence of the outermost atom for at most $\kappa_G \leqslant 10^{-16}$.

Having relocated the modified atomic centres, we need to adjust the area of influence of the atoms as well, namely by scaling the van der Waals radii by the factor $\frac{1}{2p}$. Now, we can approximate the influence function of an atom by the Fourier sum

$$\kappa_G(\mathbf{x}) \approx \tilde{\kappa}_G(\mathbf{x}) = \sum_{\ell \in \mathcal{I}_n} h_\ell e^{2\pi i \mathbf{x}^T \ell} \tag{6.13}$$

for a set of indices $\mathcal{I}_n = \{\ell \in \mathbb{Z}^3 \mid \ell \in [-\frac{n}{2}, \frac{n}{2})^3 \text{ for } n \in \mathbb{N}\}$ and with Fourier coefficients

$$h_\ell = \int_{[-\frac{1}{2},\frac{1}{2}]^3} \kappa_G(\mathbf{x}) e^{-2\pi i \mathbf{x}^T \ell} d\mathbf{x}. \tag{6.14}$$

A symmetry property we will be using later on is given by $h_\ell = h_{-\ell}$. This follows easily from $\kappa_G(\mathbf{x}) = \kappa_G(-\mathbf{x})$ as we have

$$h_\ell = \int_{[-\frac{1}{2},\frac{1}{2}]^3} \kappa_G(\mathbf{x}) e^{-2\pi i \mathbf{x}^T \ell} d\mathbf{x} = \int_{[-\frac{1}{2},\frac{1}{2}]^3} \kappa_G(-\mathbf{x}) e^{-2\pi i \mathbf{x}^T \ell} d\mathbf{x} = h_{-\ell}. \tag{6.15}$$

Inserting (6.13) into the affinity function Q^{γ_A}, we get

$$\begin{aligned} Q^{\gamma_A}(\mathbf{x}) \approx \tilde{Q}^{\gamma_A}(\mathbf{x}) &= \sum_{k \in \mathcal{A}} \gamma_A(\mathbf{z}_k) \tilde{\kappa}_G(\mathbf{x} - \mathbf{z}_k) \\ &= \sum_{k \in \mathcal{A}} \gamma_A(\mathbf{z}_k) \sum_{\ell \in \mathcal{I}_n} h_\ell e^{2\pi i (\mathbf{x} - \mathbf{z}_k)^T \ell} \\ &= \sum_{\ell \in \mathcal{I}_n} h_\ell e^{2\pi i \mathbf{x}^T \ell} \sum_{k \in \mathcal{A}} \gamma_A(\mathbf{z}_k) e^{-2\pi i \mathbf{z}_k^T \ell}. \end{aligned} \tag{6.16}$$

Note that in this approximated affinity function we separated the function into a molecule dependent sum and a molecule independent one. We shall denote the molecule dependent terms by

$$\alpha_\ell = \sum_{k \in \mathcal{A}} \gamma_A(\mathbf{z}_k) e^{-2\pi i \mathbf{z}_k^T \ell}. \tag{6.17}$$

The sums α_ℓ are in fact three-dimensional standard Fourier sums and as the atomic centres are not necessarily equispaced, we will employ the NFFT algorithm [71] for its computation, or to be more precise its adjoint. The computation will take $\mathcal{O}(|\mathcal{A}| + n^3 \log n)$ flops. Next, we consider the effect of a motion $(\mathbf{R}, \mathbf{t}) \in \mathrm{SE}(3)$ on the affinity function, i.e., we examine $Q^{\gamma_B}((\mathbf{R}, \mathbf{t}) \circ \mathbf{x})$ and its Fourier approximation. Recall from Lemma 2.1.5 that a rotation $\mathbf{R} \in \mathrm{SO}(3)$ preserves lengths of vectors, hence

$$\kappa_G(\mathbf{R}^T \mathbf{x} - \mathbf{z}_k) = e^{\beta(1 - \frac{\|\mathbf{R}^T \mathbf{x} - \mathbf{z}_k\|^2}{r^2})} = e^{\beta(1 - \frac{\|\mathbf{R}(\mathbf{R}^T \mathbf{x} - \mathbf{z}_k)\|^2}{r^2})} = \kappa_G(\mathbf{x} - \mathbf{R}\mathbf{z}_k)$$

is satisfied. This can be used to rewrite

$$\begin{aligned} \tilde{\kappa}_G((\mathbf{R}, \mathbf{t}) \circ \mathbf{x}) &= \tilde{\kappa}_G(\mathbf{R}^T \mathbf{x} - \mathbf{t} - \mathbf{z}_k) = \tilde{\kappa}_G(\mathbf{x} - \mathbf{t} - \mathbf{R}\mathbf{z}_k) \\ &= \sum_{\ell \in \mathcal{I}_n} h_\ell e^{2\pi i (\mathbf{x} - \mathbf{t} - \mathbf{R}\mathbf{z}_k)^T \ell} = \sum_{\ell \in \mathcal{I}_n} h_\ell e^{2\pi i (\mathbf{x} - \mathbf{t})^T \ell} e^{-2\pi i \mathbf{R}\mathbf{z}_k^T \ell}. \end{aligned}$$

Inserting this into the affinity function yields

$$\begin{aligned} \tilde{Q}^{\gamma_B}((\mathbf{R}, \mathbf{t}) \circ \mathbf{x}) &= \sum_{k \in \mathcal{B}} \gamma_B(\mathbf{z}_k) \tilde{\kappa}_G((\mathbf{R}, \mathbf{t}) \circ \mathbf{x}) \\ &= \sum_{\ell \in \mathcal{I}_n} h_\ell e^{2\pi i (\mathbf{x} - \mathbf{t})^T \ell} \sum_{k \in \mathcal{B}} \gamma_B(\mathbf{z}_k) e^{-2\pi i \mathbf{R}\mathbf{z}_k^T \ell}. \end{aligned} \tag{6.18}$$

Again, we separated the affinity function into a molecule dependent sum and a molecule independent one. Corresponding to (6.17), we denote the molecule dependent sum

$$\beta_{\ell}^{\mathbf{R}} = \sum_{k \in \mathcal{B}} \gamma_B(\mathbf{x}_k) e^{-2\pi i \mathbf{R}\mathbf{z}_k^T \ell}. \tag{6.19}$$

Note, that the sums $\beta_{\ell}^{\mathbf{R}}$ depend on the rotation $\mathbf{R} \in \mathrm{SO}(3)$. Again, we can use the adjoint NFFT algorithm to compute the coefficients. Due to the dependency on the rotation $\mathbf{R}$, we might need to evaluate $\beta_{\ell}^{\mathbf{R}}$ for all M^3-many rotations and the computational cost is raised to $\mathcal{O}((|\mathcal{B}|+n^3 \log n) \cdot M^3)$ flops.
Finally, the approximated affinity functions (6.16) and (6.18) are put in the objective function (6.11) which yields

$$\begin{aligned}
C((\mathbf{R},\mathbf{t})) &= \mathrm{Re}\left(\int_{[-\frac{1}{2},\frac{1}{2}]^3} Q^{\gamma_A}(\mathbf{x}) Q^{\gamma_B}(\mathbf{R}^T\mathbf{x} - \mathbf{t})\, d\mathbf{x} \right) \\
&= \mathrm{Re}\left(\int_{[-\frac{1}{2},\frac{1}{2}]^3} \sum_{\ell \in \mathcal{I}_n} h_\ell e^{2\pi i \mathbf{x}^T \ell} \alpha_\ell \sum_{\ell' \in \mathcal{I}_n} h_{\ell'} e^{2\pi i (\mathbf{x}-\mathbf{t})^T \ell'} \beta_{\ell'}^{\mathbf{R}}\, d\mathbf{x} \right) \\
&= \mathrm{Re}\left(\sum_{\ell \in \mathcal{I}_n} \sum_{\ell' \in \mathcal{I}_n} h_\ell h_{\ell'} \alpha_\ell \beta_{\ell'}^{\mathbf{R}} \int_{[-\frac{1}{2},\frac{1}{2}]^3} e^{2\pi i \mathbf{x}^T \ell} e^{2\pi i (\mathbf{x}-\mathbf{t})^T \ell'}\, d\mathbf{x} \right) \\
&= \mathrm{Re}\left(\sum_{\ell \in \mathcal{I}_n} \sum_{\ell' \in \mathcal{I}_n} h_\ell h_{\ell'} \alpha_\ell \beta_{\ell'}^{\mathbf{R}} e^{-2\pi i \mathbf{t}^T \ell'} \int_{[-\frac{1}{2},\frac{1}{2}]^3} e^{2\pi i \mathbf{x}^T \ell} e^{2\pi i \mathbf{x}^T \ell'}\, d\mathbf{x} \right).
\end{aligned}$$

As the integral satisfies $\int_{[-\frac{1}{2},\frac{1}{2}]^3} e^{2\pi i \mathbf{x}^T \ell} e^{2\pi i \mathbf{x}^T \ell'}\, d\mathbf{x} = \delta_{\ell,-\ell'}$ and by exploiting the symmetry from (6.15), we simplify the term to

$$C((\mathbf{R},\mathbf{t})) = \mathrm{Re}\left(\sum_{\ell \in \mathcal{I}_n} h_\ell^2 \alpha_\ell \beta_{-\ell}^{\mathbf{R}} e^{2\pi i \mathbf{t}^T \ell} \right).$$

For given, or rather precomputed, coefficients h_ℓ, α_ℓ and $\beta_{\ell}^{\mathbf{R}}$ this sum represents a three-dimensional Fourier sum that can be evaluated by either the FFT from [32] or the NFFT from [49]. We consider M^3-many translations here, and therefore the total cost of this step is $\mathcal{O}((M^3 + n^3 \log n) \cdot M^3)$ flops.

Recall, that finding $(\mathbf{R}_{\max}, \mathbf{t}_{\max}) \in \mathrm{SE}(3)$ with

$$C((\mathbf{R}_{\max}, \mathbf{t}_{\max})) \geqslant C((\mathbf{R},\mathbf{t}))$$

for all $(\mathbf{R},\mathbf{t}) \in \mathrm{SE}(3)$ would solve the docking problem. However, we compute the correlation only for a set of motions $\mathcal{M} \subset \mathrm{SE}(3)$. By choosing sufficiently many different motions, we still might encounter the solution or at least a motion that almost leads to the solution. In the numerical tests from Section 6.6 we will comment on this aspect. But, to conclude this section, we summarise the algorithm for the docking procedure using fast translational matching from Algorithm 1.

The advantages of the fast translational matching in comparison to the straightforward approach mentioned in the previous section is the separation of molecule dependent parts and molecule independent part of the function and the computation of the correlation integral with respect to the translations by means of fast Fourier transforms. The computational complexity is improved from previous $\mathcal{O}(|\mathcal{A}| \cdot |\mathcal{B}| \cdot M^6)$ necessary steps to $\mathcal{O}((|\mathcal{A}| + |\mathcal{B}| + M^3) \cdot M^3)$.

6.5 Fast Rotational Matching

In the fast translational matching algorithm from the previous section we were able to improve the computational complexity of the docking procedure by accelerating the computation of the objective function (6.11) for the three translational degrees of freedom encountered in a motion. Key to the fast translational matching was the realisation that for fixed rotations $\mathbf{R}$ the objective function is a correlation integral and can be computed by fast Fourier transforms. Thus we obtained a fast algorithm to search for the maximum of the objective function with respect to the different translations. On the other hand, we still have to repeat the computation of the correlation for the different rotations.
The idea of the new algorithm, we are presenting here is to use correlation again, but instead of correlating functions defined on the unit cube as previously, we will use functions defined on $\mathbb{R}^3$ but

Algorithm 1: Fast Translational Matching

Input:
$\mathcal{I}_n$: a set of n^3 indices ;
$\mathcal{A}, \mathcal{B}$: two index sets of atomic coordinates;
$\mathcal{M} \subset \mathrm{SE}(3)$: a set of motions as in (6.12);

foreach $\mathbf{x}_k$ *with* $k \in \mathcal{A} \cup \mathcal{B}$ **do**
  Calculate the coordinates $\mathbf{z}_k$ of the relocated atoms
end
foreach $\ell \in \mathcal{I}_n$ **do**
  Calculate the coefficients h_ℓ from (6.14);
  Calculate the coefficients α_ℓ from (6.17) by means of an adjoint NFFT;
  foreach $\mathbf{R}$ *with* $(\mathbf{R}, \mathbf{t}) \in \mathcal{M}$ **do**
    Calculate $\beta_\ell^{\mathbf{R}}$ from (6.19) by means of an adjoint NFFT;
  end
end
foreach $(\mathbf{R}, \mathbf{t}) \in \mathcal{M}$ **do**
  Calculate $C((\mathbf{R}, \mathbf{t})) = \sum_{\ell \in I_n} h_\ell^2 \alpha_\ell \beta_{-\ell}^{R} e^{2\pi i \mathbf{t}^T \ell}$ by means of an NFFT or FFT;
end
foreach $(\mathbf{R}, \mathbf{t}) \in \mathcal{M}$ **do**
  Find the maximum value of $C((\mathbf{R}, \mathbf{t}))$;
end

Output: the solution of the docking problem (6.7)
Complexity: $\mathcal{O}((|\mathcal{A}| + |B| + M^3)M^3)$ flops

split into $\mathbb{R}^+ \times \mathbb{S}^2$. In Section 3.4 we already discussed how to compute correlations of functions in $L^2(\mathbb{S}^2)$ (see (3.31)) by means of Fourier transforms on the rotation group. This way, we will be able to accelerate the computation of the objective function for the rotational degrees of freedom instead of the translational degrees of freedom. Although not immediately appearent, the idea of exploiting the rotational invariance (3.29) of the spherical harmonics that serve as basis functions in the Fourier expansion of a functions in $L^2(\mathbb{S}^2)$ instead of the translation-invariant Fourier expansion from Section 6.4 has some advantages. By means of Lemma 5.2.6 a translation $\mathbf{t} \in \mathbb{R}^3$ can be uniquely expressed as $\mathbf{t} = r\,\mathbf{R}_z(\varphi)\mathbf{R}_y(\theta)\mathbf{e}_z$ for $\varphi \in [0, 2\pi), \theta \in [0, \pi]$ and $r \in \mathbb{R}^+$. Phrased differently a three-dimensional translation can be expressed as a translation along the z-axis followed by two rotations, one about the y-axis and one about the z-axis. Hence, it has two rotational degrees of freedom and one translational. Combining, this in a motion $(\mathbf{R}, \mathbf{t}) \in \mathrm{SE}(3)$, we have five rotation angles that describe a motion and one absolute value of a translation along one axis. If we are able to speed up the computation for the rotations by correlating functions on the sphere, we get an improved complexity for five of the six degrees of freedom instead of the previous three.
This approach has been suggested in [54] in a context similar to the protein-protein docking, called *proteinmatching*. The term matching refers to evaluating the similarity of two molecules, rather than their complementary in which we are interested in. Although the tasks seem similar, protein matching only looks for maximal overlaps and does not include an additional constraint as the protein docking which seeks minimal overlaps while also minimising the resulting boundary of the molecule complex' domain. In the following, we will see that this approach for matching can be used for protein docking as well, after adapting it to our protein descriptions. The algorithm, we are about to present, combines for the first time the molecule descriptions in terms of the affinity function (6.5) using non-equispaced atom centres and the accelerated computation of the rotations.

The starting point of this approach is a coordinate transform of vectors $\mathbf{x} \in \mathbb{R}^3$ from Cartesian to spherical coordinates, i.e., we rewrite any $\mathbf{x} \in \mathbb{R}^3$ as $\mathbf{x} = r\mathbf{u}$ with $r = |\mathbf{x}|$ and $\mathbf{u} \in \mathbb{S}^2$. The inner product of two square-integrable functions $f, g : \mathbb{R}^3 \to \mathbb{C}$ parameterised in spherical coordinates is given by

$$\langle f, g\rangle_{L^2(\mathbb{S}^2,\mathbb{R}^+)} = \int_{\mathbb{R}^+}\int_{\mathbb{S}^2} f(r\mathbf{u})\overline{g(r\mathbf{u})}r^2 d\mathbf{u}dr. \tag{6.20}$$

While we already introduced an orthogonal basis of the $L^2(\mathbb{S}^2)$, namely the spherical harmonics Y_ℓ^m, cf. (3.11) and (3.12), we need to consider a set of basis functions for the positive real axis $\mathbb{R}^+$. For $k \in \mathbb{N}$ and $\alpha \in \mathbb{R}$ the Laguerre polynomials, are given as

$$L_k^\alpha(x) = \frac{e^x x^{-\alpha}}{k!}\frac{d^k}{dx^k}\left(e^{-x}x^{k+\alpha}\right),$$

cf. [81]. The Laguerre polynomials $L_k^\alpha(x)$ constitute an orthogonal basis of $L^2([0,\infty))$ with respect to the weighting function $e^{-x}x^\alpha$. We have

$$\int_0^\infty L_k^\alpha(x)L_\ell^\alpha(x)e^{-x}x^\alpha\,dx = \frac{\Gamma(k+\alpha+1)}{k!}\delta_{k,\ell}. \tag{6.21}$$

For an orthogonal basis suitable to our protein docking procedure, we introduce a weighted version of the Laguerre polynomials, denoted by $R_k^\ell(r)$. These functions are used to describe the radial part of the orbitals of hydrogenic atoms and are also known as radial wavefunctions, see [6, pp. 368ff] for general informations. In [74] these functions have been employed in the protein-protein docking procedure.

Definition 6.5.1. *For* $r \in \mathbb{R}_0^+,\ \ell, k \in \mathbb{N}_0,\ k > \ell$*, we define the functions* $R_k^\ell : \mathbb{R}^+ \to \mathbb{R}$ *by*

$$R_k^\ell(r) = \beta_{k\ell} e^{-\frac{r^2}{2}} r^\ell L_{k-\ell-1}^{\ell+\frac{1}{2}}\left(r^2\right)$$

with

$$\beta_{k\ell} \quad = \quad \sqrt{\frac{2(k-\ell-1)!}{\Gamma(k+\frac{1}{2})}}.$$

Lemma 6.5.2. *For* $r \in \mathbb{R}_0^+,\ \ell, k \in \mathbb{N}_0,\ k > \ell$*, the functions* $R_k^\ell(r)$ *satisfy*

$$\int_0^\infty R_k^\ell(r) R_n^l(r) r^2 dr = \delta_{k,n}.$$

Proof. We have

$$\int_0^\infty R_k^\ell(r) R_n^\ell(r) r^2 dr \quad = \quad \beta_{k\ell}\beta_{n\ell} \int_0^\infty e^{-r^2} r^{2\ell} L_{k-\ell-1}^{\ell+\frac{1}{2}}\left(r^2\right) L_{n-\ell-1}^{\ell+\frac{1}{2}}\left(r^2\right) r^2 dr.$$

By substituting first $r^2 = x$, and then $\alpha = \ell + \frac{1}{2}, k' = k - \ell - 1$ and $n' = n - \ell - 1$, we get

$$\begin{aligned}\int_0^\infty R_k^\ell(r) R_n^\ell(r) r^2 dr &= \beta_{k\ell}\beta_{n\ell} \int_0^\infty e^{-x} x^\ell L_{k-\ell-1}^{\ell+\frac{1}{2}}(x) L_{n-\ell-1}^{\ell+\frac{1}{2}}(x) \frac{x}{2\sqrt{x}} dx \\ &= \frac{1}{2}\beta_{k\ell}\beta_{n\ell} \int_0^\infty e^{-x} x^\alpha L_{k'}^\alpha(x) L_{n'}^\alpha(x) dx.\end{aligned}$$

By the orthogonality relation (6.21) and back substitution of k' and n' this simplifies to

$$\begin{aligned}\int_0^\infty R_k^\ell(r) R_n^\ell(r) r^2 dr &= \frac{1}{2}\beta_{k\ell}\beta_{n\ell} \frac{\Gamma(k'+\alpha+1)}{k'!}\delta_{k'n'} = \frac{1}{2}\beta_{k\ell}\beta_{n\ell} \frac{\Gamma(k+\frac{1}{2})}{(k-\ell-1)!}\delta_{kn} \\ &= \beta_{k\ell}^2 \frac{\Gamma(k+\frac{1}{2})}{2(k-\ell-1)!}\delta_{kn} = \delta_{kn}\end{aligned}$$

after inserting $\beta_{k\ell}$ from Definition 6.5.1. ∎

Based on the previous lemma and the orthogonality relation from (3.11), we see that the functions $R_k^\ell(r) Y_\ell^m(\mathbf{u})$ for $k, \ell \in \mathbb{N},\ k > l \geqslant |m|$ are orthonormal with respect to the inner product from (6.20). This follows immediately by

$$\begin{aligned}\langle R_k^\ell(r) Y_\ell^m(\mathbf{u}), R_{k'}^{\ell'}(r) Y_{\ell'}^{m'}(\mathbf{u})\rangle &= \int_0^\infty R_k^\ell(r) R_{k'}^{\ell'}(r) r^2 dr \int_{\mathbb{S}^2} Y_\ell^m(\mathbf{u}) \overline{Y_{\ell'}^{m'}(\mathbf{u})} d\mathbf{u} \\ &= \delta_{k,k'}\delta_{\ell,\ell'}\delta_{m,m'}.\end{aligned}$$

Moreover, these products of functions constitute an orthogonal basis of the space of square-integrable functions on $\mathbb{R}^3$. Therefore, we find a unique series expansion of functions $f : \mathbb{R}^3 \to \mathbb{R}$ as in the following definition.

Definition 6.5.3. *Any square-integrable function* $f : \mathbb{R}^3 \to \mathbb{R}$ *can be expanded into*

$$f(\mathbf{x}) = f(r\mathbf{u}) \quad = \quad \sum_{k=1}^{\infty} \sum_{\ell=0}^{k-1} \sum_{m=-\ell}^{l} \hat{f}_{k\ell m} R_k^{\ell}(r) Y_{\ell}^{m}(\mathbf{u})$$

with coefficients

$$\hat{f}_{k\ell m} \quad = \quad \int_0^{\infty} \int_{\mathbb{S}^2} f(r\mathbf{u}) \overline{R_{\ell}^{k}(r) Y_{\ell}^{m}(\mathbf{u})} r^2 d\mathbf{u} dr.$$

Now, we will apply Definition 6.5.3 to approximate the affinity functions Q^{γ_A} and Q^{γ_B} by the finite sum

$$Q^{\gamma_A}(r\mathbf{u}) \approx Q_N^{\gamma_A}(r\mathbf{u}) = \sum_{k=1}^{N} \sum_{\ell=0}^{k-1} \sum_{m=-\ell}^{\ell} h_{k\ell m}^{A} R_k^{\ell}(r) Y_{\ell}^{m}(\mathbf{u}) \tag{6.22}$$

for $N \in \mathbb{N}$ and coefficients

$$\begin{aligned} h_{k\ell m}^{A} \quad &= \quad \int_0^{\infty} \int_{\mathbb{S}^2} Q_N^{\gamma_A}(r, \mathbf{u}) R_k^{\ell}(r) \overline{Y_{\ell}^{m}(\mathbf{u})} r^2 d\mathbf{u} dr. \\ &= \quad \int_0^{\infty} \int_{\mathbb{S}^2} \sum_{j \in \mathcal{A}} \gamma_A(r_j \mathbf{u}_j) \kappa(r\mathbf{u} - r_j \mathbf{u}_j) R_k^{\ell}(r) \overline{Y_{\ell}^{m}(\mathbf{u})} r^2 d\mathbf{u} dr \end{aligned} \tag{6.23}$$

where $\mathbf{x}_j = r_j \mathbf{u}_j$ denote the centres of the molecule's atoms. The coefficients $h_{k\ell m}$ consist of molecule dependent and molecule independent terms. As in the fast translational matching, we seek to separate them. But due to the coordinate transform to spherical coordinates, this is not conveniently done. Expressing $\kappa(r\mathbf{u} - r_j \mathbf{u}_j) = \kappa_G(r\mathbf{u} - r_j \mathbf{u}_j)$ in spherical coordinates with $\mathbf{u} = (\varphi, \theta)$ leads to

$$\kappa_G(r\mathbf{u} - r_j \mathbf{u}_j) \quad = \quad e^{\frac{\beta}{\nu^2}(\nu^2 - r^2 + r_j^2 - 2r r_j \left(\cos(\varphi - \varphi_j) \sin\theta \sin\theta_j + \cos\theta \cos\theta_j\right)},$$

where ν denotes the van der Waals radius of the atom. Owing to the term

$$K = -2r r_j \left(\cos(\varphi - \varphi_j) \sin\theta \sin\theta_j + \cos\theta \cos\theta_j\right)$$

which is a product of molecule dependent and independent parts the function κ_G cannot be separated in the same manner as in the fast translational matching. Still, it is possible to find a way how this can be done. We find a detailed derivation of the separation along with the description of a method how to compute the coefficients (6.23) in [9, pp. 45-52]. The approach presented there is based on a power series expansion of the critical term e^K as

$$\begin{aligned} e^K = \sum_{p=0}^{\infty} \frac{K^p}{p!} \quad &= \quad \sum_{p=0}^{\infty} \sum_{q=0}^{p} \sum_{s=0}^{q} \sum_{u=0}^{q} \sum_{\nu=0}^{p-q} \binom{p}{q} \binom{q}{s} \binom{q}{u} \binom{p-q}{\nu} (-1)^{p+q-u} \frac{(2r)^p}{p!} \\ &\times \quad e^{i\theta(p-2(u+\nu))} e^{i\varphi(q-2s)} r_j^p \left(\frac{i}{2} \sin\theta_j\right)^q (\cos\theta_j)^{p-q} e^{-i\varphi_j(q-2s)}. \end{aligned} \tag{6.24}$$

The seperation of molecule dependent and independent parts of the coefficients $h_{k\ell m}^{A}$ follows by inserting (6.24) into (6.23). We get

$$\begin{aligned} h_{k\ell m}^{A} \quad &= \quad \sum_{p=0}^{\infty} \sum_{q=0}^{p} \sum_{s=0}^{q} \sum_{t=0}^{q} \sum_{\nu=0}^{p-q} \binom{p}{q} \binom{q}{s} \binom{q}{t} \binom{p-q}{\nu} (-1)^{p+q-t} \frac{(2\beta)^p}{\nu^p p!} \alpha_p^{qs} \\ &\times \quad \int_0^{\infty} \int_0^{2\pi} \int_0^{\pi} e^{\beta(1 - \frac{r^2}{\nu^2})} e^{i\theta(p-2(t+\nu))} e^{i\varphi(q-2s)} R_k^{\ell}(r) \overline{Y_{\ell}^{m}((\varphi, \theta))} r^{2+p} \sin\theta \, d\theta d\varphi dr \end{aligned}$$

with

$$\alpha_p^{qs} = \sum_{j\in\mathcal{A}} \gamma_A(r_j(\varphi_j,\theta_j))e^{\frac{\beta r_j^2}{\nu^2}} r_j^p \left(\frac{i}{2}\sin\theta_j\right)^q (\cos\theta_j)^{p-q} e^{-i\varphi_j(q-2s)}$$

denoting the molecule dependent terms.
The approach from [9] continues by cutting the series expansion at a certain degree and computing only terms for $p \leqslant P$, expanding the whole integral into a power series and then integrating every term separately. The work also discusses complexity and accuracy of these computations depending on the cut-off P.

Having considered a possible method to compute the coefficients $h_{k\ell m}$ of the expansion (6.22), we return to the bigger scope of the fast rotational matching.
As in the previous section, we seek to evaluate the objective function (6.11) to find the best arrangement of two molecules. This time however, we will not keep molecule A fixed while only moving molecule B.

Lemma 6.5.4. *The motion* $(\mathbf{R}_{max}, \mathbf{t}_{max}) \in \mathrm{SE}(3)$ *in*

$$(\mathbf{R}_{max}, \mathbf{t}_{max}) = (\mathbf{R}_{max}, z_{max}\mathbf{U}_{max}\mathbf{e}_z) = \max_{(\mathbf{R},z\mathbf{U}\mathbf{e}_z)\in\mathrm{SE}(3)} \mathrm{Re}\left(\int Q^{\gamma_A}(\mathbf{U}^T\mathbf{x} - z\mathbf{e}_z)Q^{\gamma_B}(\mathbf{R}^T\mathbf{x})\,d\mathbf{x}\right)$$

for $\mathbf{R} \in \mathrm{SO}(3)$, $\mathbf{U} \in \mathrm{SO}(3)\backslash\mathcal{Z}$ *and* $z \in \mathbb{R}$ *is a solution of the docking problem* (6.7) *when using the asymmetric approach, i.e, the objective function* (6.8)*.*

Proof. Suppose $(\mathbf{S}, \mathbf{t})$ to be the solution of the docking problem (6.7) the resulting molecule complex is described by the product of affinity functions

$$Q^{\gamma_{A\cup B}}(\mathbf{x}) = Q^{\gamma_A}(\mathbf{x})Q^{\gamma_B}(\mathbf{S}^T\mathbf{x} + \mathbf{t}).$$

By means of Lemma 5.2.6, we can replace $\mathbf{t} = z\mathbf{U}\mathbf{e}_z$ to get

$$Q^{\gamma_{A\cup B}}(\mathbf{x}) = Q^{\gamma_A}(\mathbf{x})Q^{\gamma_B}(\mathbf{S}^T\mathbf{x} + z\mathbf{U}\mathbf{e}_z)$$

Applying the rotation $\mathbf{U}^T$ to the whole complex yields

$$Q^{\gamma_{A\cup B}}(\mathbf{U}^T\mathbf{x}) = Q^{\gamma_A}(\mathbf{U}^T\mathbf{x})Q^{\gamma_B}(\mathbf{U}^T\mathbf{S}^T\mathbf{x} + z\mathbf{e}_z).$$

We continue by applying the translation $-z\mathbf{e}_z$ to the whole complex. This yields

$$Q^{\gamma_{A\cup B}}(\mathbf{U}^T\mathbf{x} - z\mathbf{e}_z) = Q^{\gamma_A}(\mathbf{U}^T\mathbf{x} - z\mathbf{e}_z)Q^{\gamma_B}(\mathbf{R}^T\mathbf{x})$$

where we set $\mathbf{R} = \mathbf{S}\mathbf{U}$. As we maintained the position of the molecules with respect to one another, this proves the lemma. ∎

Note that the objective function (6.8) can easily be replaced with the objetive function (6.6) of the symmetric approach.
We shall now examine how an affinity function expanded as in Definition 6.5.3 behaves under the application of a rotation. Recall the representation property (3.29) of spherical harmonics

$$Y_\ell^n(\mathbf{R}^T\mathbf{u}) = \sum_{m=-\ell}^{\ell} Y_\ell^m(\mathbf{u})D_\ell^{mn}(\mathbf{R}), \qquad \text{for } |m| \leqslant \ell, \mathbf{u} \in \mathbb{S}^2, \mathbf{R} \in \mathrm{SO}(3).$$

showing their rotational invariance. Due to this, we have

$$Q^{\gamma_B}(\mathbf{R}^T\mathbf{x}) = Q^{\gamma_B}(r\mathbf{R}^T\mathbf{u}) \quad = \quad \sum_{k=1}^{N}\sum_{\ell=0}^{k-1}\sum_{m,n=-\ell}^{\ell} h_{k\ell n} D_\ell^{mn}(\mathbf{R}) R_k^\ell(r) Y_\ell^m(\mathbf{u}).$$

Note that the rotation does not affect the radial parts of the function as a rotation preserves distance, see Lemma 2.1.5.
Next, we need to consider the translation along the z-axis. In spherical coordinates a translation of the vector $\mathbf{x}$ about $z\mathbf{e}_z$ is given by

$$\mathbf{x} - z\mathbf{e}_z = r_z\mathbf{u}_z$$

with $r_z = \sqrt{r^2 + 2rz\cos\theta + z^2}$ and $\mathbf{u}_z = (\arccos\left(\frac{r}{r_z}\sin\theta\right), \varphi)$. We point out that the longitude φ does not change during a translation along the z-axis. Now we have all ingredients to compute the modified objective function from Lemma 6.5.4. We get

$$\begin{aligned} C((\mathbf{R}, \mathbf{U}z\mathbf{e}_z)) \quad &= \quad \int Q^{\gamma_A}(\mathbf{U}^T\mathbf{x} - z\mathbf{e}_z) Q^{\gamma_B}(\mathbf{R}^T\mathbf{x})\, d\mathbf{x} \\ &= \quad \int_{\mathbb{R}^+}\int_{\mathbb{S}^2} \sum_{k=1}^{N}\sum_{\ell=0}^{k-1}\sum_{m,n=-\ell}^{\ell} h_{k\ell n} D_\ell^{mn}(\mathbf{U}) R_k^\ell(r_z) Y_\ell^m(\mathbf{u}_z) \\ &\times \quad \sum_{k'=1}^{N}\sum_{\ell'=0}^{k'-1}\sum_{m',n'=-\ell'}^{\ell'} h_{k'\ell'n'} D_{\ell'}^{m'n'}(\mathbf{R}) R_{k'}^{\ell'}(r) Y_{\ell'}^{m'}(\mathbf{u}) r^2\, d\mathbf{u}\, dr. \end{aligned}$$

By extracting the rotation dependent terms from the integral, this can be rearranged to

$$C((\mathbf{R}, \mathbf{U}z\mathbf{e}_z)) \quad = \quad \sum_{\ell,\ell'=0}^{N-1}\sum_{m,n=-\ell}^{\ell}\sum_{m',n'=-\ell'}^{\ell'} J_{\ell\ell'}^{mm'nn'}(z)\, D_\ell^{mn}(\mathbf{U}) D_{\ell'}^{m'n'}(\mathbf{R})$$

with

$$\begin{aligned} J_{\ell\ell'}^{mm'nn'}(z) \quad &= \quad \int_{\mathbb{R}^+}\int_0^{2\pi}\int_0^{\pi} \sum_{k=\ell+1}^{N}\sum_{k'=\ell'+1}^{N} h_{k\ell n} h_{k'\ell'n'} R_k^\ell(r_z) R_{k'}^{\ell'}(r) \\ &\times \quad Y_\ell^m((\varphi, \theta_z)) Y_{\ell'}^{m'}((\varphi, \theta)) r^2 \sin\theta\, d\theta\, d\varphi\, dr. \end{aligned}$$

Consider the following equation

$$\int_0^{2\pi}\int_0^{\pi} P_\ell^{m'}(\cos\theta) P_\ell^m(\cos\theta_z) e^{im\varphi} e^{im'\varphi} \sin\theta\, d\theta\, d\varphi = \int_0^{\pi} P_\ell^m(\cos\theta) P_\ell^m(\cos\theta_z) \sin\theta\, d\theta.$$

Inserting the above equation into $J_{\ell\ell'}^{mm'nn'}(z)$ for the spherical harmonics Y_ℓ^m, we conclude that the integral $J_{\ell\ell'}^{mm'nn'}(z)$ evaluates to zero for all $m' = -m$. Note that this property originates from the orthogonality relation (3.11).
The correlation $C((\mathbf{R}, \mathbf{U}z\mathbf{e}_z))$ can hence be computed by

$$C((\mathbf{R}, \mathbf{U}z\mathbf{e}_z)) = \sum_{\ell,\ell'=0}^{N}\sum_{n=-\ell}^{\ell}\sum_{m',n'=-\ell'}^{\ell'} J_{\ell\ell'}^{-m'm'nn'}(z) D_\ell^{-m'n}(\mathbf{U}) D_{\ell'}^{m'n'}(\mathbf{R}). \tag{6.25}$$

To compute the integrals $J_{\ell\ell'}^{-m'm'nn'}(z)$, we refer to two different approaches, in [9] an approach is presented which expands them into a power series, followed by component-wise integration. The other approach, found in [74] computes the integral by means of a spherical Bessel transform.

Computing all necessary $J_{\ell\ell'}^{-m'm'nn'}(z)$ takes $\mathcal{O}((|\mathcal{A}|+|\mathcal{B}|)M)$ operations where M refers to the number of one-dimensional translation that we perform, and $|\mathcal{A}|, |\mathcal{B}|$ are the number of atoms in the two molecules $\mathcal{A}$ and $\mathcal{B}$.
We shall now simply assume that we are given the precomputed values of $J_{\ell\ell'}^{-m'm'nn'}$, as we are more interested in the application of the SO(3) Fourier transform here.
Writing the Wigner-D functions in terms of their Euler angles by (3.23), we get

$$C((\mathbf{R},\mathbf{U}z\mathbf{e}_z)) = \sum_{\ell,\ell'=0}^{N}\sum_{n=-\ell}^{\ell}\sum_{m',n'=-\ell'}^{\ell'} J_{\ell\ell'}^{-m'm'nn'}(z)e^{-in\varphi}e^{-im'\alpha}e^{-in'\gamma}d_{\ell'}^{m'n'}(\cos\beta)d_{\ell}^{-m'n}(\cos\theta)$$

for $\mathbf{R}=\mathbf{R}(\alpha,\beta,\gamma)\in SO(3)$ and $\mathbf{U}=\mathbf{U}(0,\theta,\varphi)\in SO(3)\backslash\mathcal{Z}$. By rearranging the sums in (6.25), we see that we can employ two consecutive SO(3) Fourier transforms here. First we compute

$$\tilde{J}_{\ell'}^{m'n'} = \sum_{\ell=0}^{N}\sum_{n=-\ell}^{\ell} J_{\ell\ell'}^{-m'm'nn'}e^{-in\varphi}d_{\ell}^{-m'n}(\cos\theta).$$

This is actually not a complete SO(3) Fourier transform as we are missing one Euler angle. Hence this sum can be computed by a Wigner-d transform followed by a two-dimensional standard Fourier transform, in contrast to the three-dimensional one in the NFSOFT algorithm. Another nice feature of this sum is, that it resembles a sum over the modified spherical harmonics $h_\ell^{mn}(\mathbf{u})$ introduced in Definition 5.2.9.
The next step is to compute the sum

$$C((\mathbf{R},\mathbf{U}z\mathbf{e}_z)) = \sum_{\ell'=0}^{N}\sum_{m'=-\ell}^{\ell}\sum_{n'=-\ell'}^{\ell'} \tilde{J}_{\ell'}^{m'n'}e^{-im'\alpha}e^{-in'\gamma}d_{\ell'}^{m'n'}(\cos\beta)$$

which can be conveniently computed by the NFSOFT algorithm. As we need to compute the objective function $C((\mathbf{R},\mathbf{U}z\mathbf{e}_z))$ for five rotational degrees of freedom, i.e., $\mathcal{O}(M^5)$ different Euler angles, the computational complexity yields $\mathcal{O}((|\mathcal{A}|+|\mathcal{B}|+M^5)M)$. We summarise the whole procedure in Algorithm 2.

A docking example To conclude the section, we present an example result of the docking procedure. We used the so-called cAMP-dependent protein kinase (2CPK) and its catalytic subunit to test the docking procedures. We performed a bound docking procedure. That is, we extracted the subunit from the experimentally determined protein complex and computed the overlaps to reproduce the molecular arrangement.
The 2CPK molecule we used consists of 2666 atoms while the sub-unit is made up by 158 atoms. By adding an artificial exterior skin to the sub-unit its size grew to 1818 atoms.
For each degree of freedom we computed 24 different values, leading to 24^6 different analysed motions. Both, the fast translational matching and fast rotational matching list the experimentally determined arrangement of the complex among the twenty highest values of the objective function $C((\mathbf{R},\mathbf{t}))$.

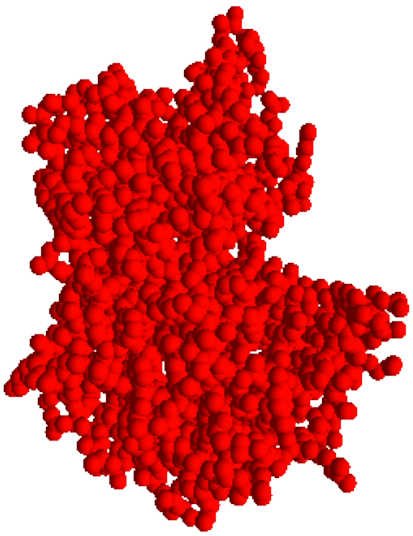
Algorithm 2: Fast Rotational Matching

Input:
P, N : the cut-offs;
$\mathcal{A}, \mathcal{B}$: two index sets of atomic coordinates;
$\mathcal{M} \subset \mathrm{SE}(3)$: a set of motions as in (6.12);

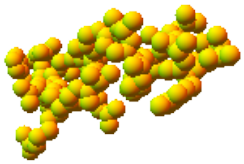
foreach $\mathbf{x}_k$ *with* $k \in \mathcal{A} \cup \mathcal{B}$ **do**
  Calculate the centres c_A and c_B of the molecules and compute the relocated atoms $\mathbf{z}_k = x_k - c_{A/B}$
end
foreach (p, q, s) *with* $s \leqslant q \leqslant p \leqslant P$ **do**
  Calculate the coefficients α_p^{qs} and β_p^{qs};
end
foreach (k, ℓ, m) *with* $|m| \leqslant \ell \leqslant k \leqslant N$ **do**
  Calculate the coefficients $h^A_{k\ell m}$ and $h^B_{k\ell m}$;
end
foreach $z \in R^+$ *in* $(\mathbf{R}, z\mathbf{U}\mathbf{e}_z) \in \mathcal{M}$, (ℓ, ℓ', m', n, n') *with* $|m'|, |n'| \leqslant \ell' \leqslant N$ *and* $|n| \leqslant \ell \leqslant N$ **do**
  Calculate the integrals $J_{\ell\ell'}^{-m'm'nn'}(z)$;
end
foreach $\mathbf{U}$ *in* $(\mathbf{R}, z\mathbf{U}\mathbf{e}_z) \in \mathcal{M}$ **do**
  Calculate $\tilde{J}_{\ell'}^{m'n'}$ by means of an reduced NFSOFT;
end
foreach $\mathbf{R}$ *in* $(\mathbf{R}, z\mathbf{U}\mathbf{e}_z) \in \mathcal{M}$ **do**
  Calculate $C((\mathbf{R}, z\mathbf{U}\mathbf{e}_z))$ by means of an NFSOFT;
end
foreach $(\mathbf{R}, \mathbf{t}) \in \mathcal{M}$ **do**
  Find the maximum value of $C((\mathbf{R}, z\mathbf{U}\mathbf{e}_z))$;
end

Output: the solution of the docking problem
Complexity: $\mathcal{O}((|\mathcal{A}| + |B| + M^5)M)$ flops

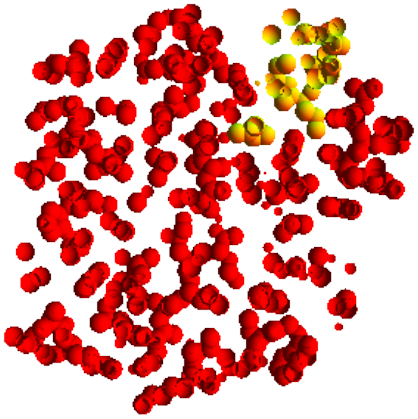
Figure 6.5 shows the two proteins and the docked complex. The cross-section of the molecule in Figure 6.5(c) demonstrates nicely how the catalytic sub-unit is positioned in an open pocket of the 2CPK protein.
A problem encountered upon producing the list of putative arrangements is that for a finer grid of motions. the list exhibits certain clusters of high ranked motions which lead to almost identical arrangements. For a ranking or re-ranking it might be helpful to identify these as belonging to the same molecular arrangement. As we mentioned in Section 6.1, to distinguish the right complex from the false positives is a widely discussed issue in protein-protein docking and beyond the scope of this work. However it would be interesting to re-score the results from the fast rotational and translational matching using other functions to construct the objective function than the area of influence of the proteins presented here.



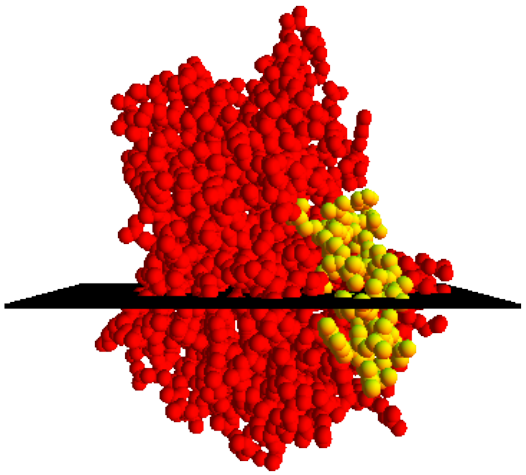
(a) The cAMP-dependent protein kinase (2CPK) without the catalytic subunit.



(b) The catalytic subunit has been previously extracted from the protein.



(c) A cross-section of the resultant protein complex.



(d) The resultant complex. We also marked a plane at which we sliced the protein in (c).

Figure 6.5: An example for a docking procedure.

6.6 Refinement

For the docking procedures we defined a finite set of rigid-body motion for which we calculated the objective function of the docking problem. Given the fact that we are actually looking for the maximum of this function, we might not have completed the task after performing the fast matching procedures as the maximising motion might not be in the given sampling set. But surely one can think of different strategies to overcome this flaw. Having found a motion $(\mathbf{R}, \mathbf{t}) \in \mathrm{SE}(3)$ which yields a high value of the objective function, we could examine more motions $(\mathbf{S}, \mathbf{u})$ such that $\|\mathbf{t} - \mathbf{u}\|$, and

$|\mathbf{R}\mathbf{S}^{\mathsf{T}}|$ (cf. Definition 2.1.7) are sufficiently small. One idea would be to simply compute the objective function for these additional motions by one of the fast matching procedures from Sections 6.4 and 6.5. Another idea is a refinement step based on Taylor expansions which will provide convenient results providing sufficiently small motions.
Given a function $\mathrm{Q} : \mathbb{R}^3 \to \mathbb{C}$ as in (6.4) or (6.5), we can approximate $\mathrm{Q}((\mathbf{R},\mathbf{t}) \circ \mathbf{x})$, $(\mathbf{R},\mathbf{t}) \in \mathrm{SE}(3)$ for small $\mathbf{R} \in \mathrm{SO}(3)$ and small $\mathbf{t} \in \mathbb{R}^3$ by

$$\begin{aligned}\mathrm{Q}(\mathbf{R}^{\mathsf{T}}\mathbf{x}-\mathbf{t}) &= \mathrm{Q}(\mathbf{x}+(\mathbf{R}^{\mathsf{T}}-\mathbf{I})\mathbf{x}-\mathbf{t}) \\ &\approx \mathrm{Q}(\mathbf{x}) + \nabla\mathrm{Q}(\mathbf{x})\left((\mathbf{R}^{\mathsf{T}}-\mathbf{I})\mathbf{x}-\mathbf{t}\right) + \frac{1}{2}\left((\mathbf{R}^{\mathsf{T}}-\mathbf{I})\mathbf{x}-\mathbf{t}\right)^{\mathsf{T}}\nabla\nabla\mathrm{Q}(\mathbf{x})\left((\mathbf{R}^{\mathsf{T}}-\mathbf{I})\mathbf{x}-\mathbf{t}\right).\end{aligned}$$

Consequently, the objective function for two molecules $\mathcal{A}$ and $\mathcal{B}$ becomes

$$\begin{aligned}\mathrm{C}((\mathbf{t},\mathbf{R})) &= \int \mathrm{Q}^{\mathcal{A}}(\mathbf{x})\mathrm{Q}^{\mathcal{B}}((\mathbf{t},\mathbf{R})\circ\mathbf{x})\,\mathrm{d}\mathbf{x} \\ &\approx \int \mathrm{Q}^{\mathcal{A}}(\mathbf{x})\mathrm{Q}^{\mathcal{B}}(\mathbf{x}) + \mathrm{Q}^{\mathcal{A}}(\mathbf{x})\nabla\mathrm{Q}^{\mathcal{B}}(\mathbf{x})\left((\mathbf{R}^{\mathsf{T}}-\mathbf{I})\mathbf{x}-\mathbf{t}\right) \\ &\quad + \frac{\mathrm{Q}^{\mathcal{A}}(\mathbf{x})}{2}\left((\mathbf{R}^{\mathsf{T}}-\mathbf{I})\mathbf{x}-\mathbf{t}\right)^{\mathsf{T}}\nabla\nabla\mathrm{Q}^{\mathcal{B}}(\mathbf{x})\left((\mathbf{R}^{\mathsf{T}}-\mathbf{I})\mathbf{x}-\mathbf{t}\right)\,\mathrm{d}\mathbf{x}.\end{aligned}$$

Setting $\nabla_{\mathbf{t}}\mathrm{C} = \mathbf{0}$, we have

$$\mathbf{0} = \nabla_{\mathbf{t}}\mathrm{C} \approx -\int \mathrm{Q}^{\mathcal{A}}(\mathbf{x})\nabla\mathrm{Q}^{\mathcal{B}}(\mathbf{x}) + \mathrm{Q}^{\mathcal{A}}(\mathbf{x})\nabla\nabla\mathrm{Q}^{\mathcal{B}}(\mathbf{x})\left((\mathbf{R}^{\mathsf{T}}-\mathbf{I})\mathbf{x}-\mathbf{t}\right)\,\mathrm{d}\mathbf{x},$$

and obtain by

$$\int \mathrm{Q}^{\mathcal{A}}(\mathbf{x})\nabla\mathrm{Q}^{\mathcal{B}}(\mathbf{x}) + \mathrm{Q}^{\mathcal{A}}(\mathbf{x})\nabla\nabla\mathrm{Q}^{\mathcal{B}}(\mathbf{x})\left((\mathbf{R}^{\mathsf{T}}-\mathbf{I})\mathbf{x}\right)\,\mathrm{d}\mathbf{x} = \left(\int \mathrm{Q}^{\mathcal{A}}(\mathbf{x})\nabla\nabla\mathrm{Q}^{\mathcal{B}}(\mathbf{x})\,\mathrm{d}\mathbf{x}\right)\mathbf{t}$$

a linear system of equations for $\mathbf{t}$ depending on $\mathbf{R}$. For $\mathrm{C}((\mathbf{R},\mathbf{t})) = \mathrm{C}((\mathbf{R},\mathbf{t}(\mathbf{R})))$ the maximum of the objective function can be found by solving the 3×3 system of equations $\frac{\partial}{\partial\mathbf{R}}\mathrm{C}((\mathbf{R},\mathbf{t}(\mathbf{R}))) = \mathbf{0}$ for small Euler angles.
Either way, we are now left with a list of putative protein-protein complexes, ranked according to their values of the objective function. As mentioned in Section 6.1 also the correct complex might be in the list, it may well not be the one with the highest score. To refine the ranking in this sense, we would need to incorporate more biological information. Although, geometric surface complementary is the dominant descriptor in docking processes and hence a good choice for the scoring function it is not the only one. Other criteria can be used to design affinity functions like models of desolvation, hydrophobicity, hydrogen bonds or electrostatics. Our fast matching procedures will conveniently work on any of them. Using such a new affinity function one could hence simply reuse the fast matching procedures and re-rank the current list of complexes accordingly.
Note, that truly identifying the near native complex in the ranked list is beyond the scope of the Fourier based matching procedures. One main fact being that they consider molecules to be rigid-bodies. Incorporation of additional information from previously known protein interfaces, biochemical experiments or even visual inspection will be helpful for this, as well. This is the so-called data-based and data-driven docking, see e.g. [47] for more information on this stage of docking. Characteristically for this stage are time-consuming heavy-weight calculations.

7 Conclusion

This thesis presented a broad investigation of Fourier transforms on the rotation group. We introduced the approximation of arbitrary functions $f \in L^2(SO(3))$ by finite sums

$$f(\mathbf{R}_q) \approx \sum_{\ell=0}^{L} \sum_{m=-\ell}^{\ell} \sum_{n=-\ell}^{\ell} \widehat{f}_\ell^{mn} D_\ell^{mn}(\mathbf{R}_q)$$

for Q different arbitrary rotations $\mathbf{R}_q \in SO(3)$ with $q = 1, \ldots, Q$ as the nonequispaced discrete $SO(3)$ Fourier transform, NDSOFT. The naive computation of the above sum by the NDSOFT has a high computational complexity of $\mathcal{O}(L^3 Q)$ operations in contrast to only $\mathcal{O}(L^3 + Q)$ input values. With the NFSOFT, we developed strategies for a faster computation of the $SO(3)$ Fourier transform. The general concept of exploiting the Euler angle representation of elements of $SO(3)$ and the possible separation of the Wigner-D functions $D_\ell^{mn}(\mathbf{R})$ according to these angles in

$$f(\mathbf{R}(\alpha_q, \beta_q, \gamma_q)) \quad = \quad \sum_{m=-L}^{L} e^{-im\alpha_q} \sum_{n=-L}^{L} e^{-in\gamma_q} \sum_{\ell=L_0}^{L} \hat{f}_\ell^{mn} d_\ell^{mn}(\cos\beta_q).$$

	Complete Name	Complexity	Reference
FFT	Fast Fourier transform	$\mathcal{O}(L^d \log L)$	[32]
NFFT	Nonequispaced fast Fourier transform	$\mathcal{O}(Q + L^d \log L)$	[49]
SOFT	(Equispaced) $SO(3)$ Fourier transform	$\mathcal{O}(L^4)$, $Q = 8L^3$	[53]
NFSOFT	Nonequispaced fast $SO(3)$ Fourier transform	$\mathcal{O}(Q + L^3 \log L)$	Theorem 4.2.2
NDSOFT	Nonequispaced discrete $SO(3)$ Fourier transform	$\mathcal{O}(QL^3)$	Definition 3.2.11
NFSUFT	Nonequispaced fast $SU(2)$ Fourier transform	$\mathcal{O}(Q + L^3 \log L)$	Theorem 5.1.9
NDSUFT	Nonequispaced discrete $SU(2)$ Fourier transform	$\mathcal{O}(QL^3)$	Definition 5.1.3
FWT-C	Fast Wigner transform based on cascade summation	$\mathcal{O}(L \log^2 L)$	Section 4.1.1
FWT-S	Fast Wigner transform based on semiseparable matrices	$\mathcal{O}(L \log L)$	Section 4.1.2
DWT	Discrete Wigner transform	$\mathcal{O}(L^2)$	Definition 4.1.1

Table 7.1: A list of the transforms mentioned in this work with references, implementation and their asymptotic complexities depending on the bandwidth L and the number of input nodes Q.

was explained. We suggested how to handle the innermost sum of the term, by proposing two algorithms for transforming the sum over Wigner-d functions into Chebyshev polynomials independent of the sampled rotations $\mathbf{R}(\alpha_q, \beta_q, \gamma_q)$. The first algorithm, newly presented here, the one based on the cascade summation had the complexity $\mathcal{O}(L \log^2 L)$. The second one introduced in this work, based on semiseparable matrices has an even more favourable complexity of $\mathcal{O}(L \log L)$. And indeed the numerical results showed that this second algorithm is an improvement over the previously one with the cascade summation even though both of them outperform the previously existing ones.
Moreover it does not seem reasonable trying to improve the asymptotical complexity of this transform further then the achieved $\mathcal{O}(L \log L)$. The remaining step necessary to compute an NFSOFT, the rotation dependent application of the three-dimensional NFFT has complexity of $\mathcal{O}(L^3 \log L + Q)$ operations. Hence, improving the Wigner-d transform in terms of asymptotical complexity would not have an effect on the overall complexity of the algorithm. Still it would be interesting to consider further developments of the NFSOFT, like a derivation of an inverse algorithm. Comparing our NFSOFT to the naive evaluation of a discrete $\mathrm{SO}(3)$ Fourier sum, our numerical tests verified the improved complexity.
The examination of the NFSOFT is the central aspect of this work. To conclude the thesis, we look back on its title and summarise the contributions made with respect to this topic.

i) **The Nonequispaced Fast** $\mathrm{SO}(3)$ **Fourier transform:** The main part of this thesis was devoted to the development of an efficient algorithm to evaluate the above sum. In Theorem 4.2.2 we stated such an algorithm, called the nonequispaced fast $\mathrm{SO}(3)$ Fourier transform, in short, NFSOFT. The NFSOFT is able to compute the above sum with $\mathcal{O}(L^3 \log L + Q)$ operations instead of $\mathcal{O}(L^3 Q)$ needed in a naive attempt. The implementation of the NFSOFT using the cascade summation for the transform of Wigner-d functions has been incorporated in the public available NFFT library [49].

ii) **Generalisations:** Motivated by the good results of the NFSOFT, it seemed natural to examine whether the concepts used for $\mathrm{SO}(3)$ Fourier transforms are applicable to other, similar groups. This was done in Chapter 5. There we saw that indeed, an almost immediate generalisation are the nonequispaced Fourier transforms on the complex rotation group $\mathrm{SU}(2)$, NDSUFT. We provided the theoretical background and a first implementation of the necessary routines in Mathematica for a fast algorithm, called the NFSUFT and stated it in Theorem 5.1.9. It would be an interesting future work to use this transform, the NFSUFT, in suitable applications.
Not that immediate, arose the possibility for computing Fourier transforms on the motion group, which we briefly discussed, pointing out the difficulties in defining Fourier transforms and especially their inverse. Nevertheless, this is a promising generalisation which would be useful also for the application, we discussed in Section 6.

iii) **Applications:** A direct application was presented with the fast summation of radial functions on $\mathrm{SO}(3)$. By splitting sums of rotation dependent functions as in Equation (3.35), we were able to compute them by means of the NFSOFT and its adjoint. The theoretical error estimates in Lemma 3.5.3, as well as the numerical tests demonstrated the advantageous runtime while maintaining good accuracy.
As laid out in Section 6, the protein-protein docking is a much regarded problem from biochemistry. Here, we formalised the search process as a first stage of docking focusing on a sound mathematical description of proteins and the formulation of the underlying non-convex optimisation problem. The objective functions we introduced in (6.6) and (6.8) exhibit various extrema justifying the application of a global search scheme. The evaluation of the objective functions

at discrete grid points of $\mathrm{SE}(3)$ have been carried out by the fast translational matching from Algorithm 1 and by the new fast rotational matching from Algorithm 2. The application of our nonequispaced $\mathrm{SO}(3)$ Fourier transform was established in Lemma 6.5.4. This way we are now able to solve the Docking problem in $\mathcal{O}((|\mathcal{A}| + |\mathcal{B}| + \mathrm{M}^5)\mathrm{M})$ arithmetic operations instead of $\mathcal{O}(|\mathcal{A}||\mathcal{B}|\mathrm{M}^6)$.

Bibliography

[1] M. Abramowitz and I. A. Stegun. *Handbook of Mathematical Functions*. National Bureau of Standards, 1972.

[2] B. L. Adams, S. I. Wright, and K. Kunze. Orientation imaging: The emergence of a new microscopy. *Journal Metallurgical and Materials Transactions A*, 24:819 – 831, 1993.

[3] P. Aloy and R. Russell. Ten thousand interactions for the molecular biologist. *Nature Biotechnology*, 22:1317 – 1321, 2004.

[4] B. K. Alpert and V. Rokhlin. A fast algorithm for the evaluation of Legendre expansions. *SIAM J. Sci. Stat. Comput.*, 12:158 – 179, 1991.

[5] R. Askey. *Orthogonal Polynomials and Special functions*. SIAM, 1975.

[6] P. Atkins and J. de Paula. *Physical Chemistry for the Life Sciences*. Oxford University Press, 2006.

[7] C. Bajaj, R. Chowdhury, and V. Siddahanavalli. F2dock: Fast fourier protein-protein docking. *IEEE/ACM Trans. Comput. Biol. Bioinf*, 2009.

[8] G. Baszenski and M. Tasche. Fast polynomial multiplication and convolution related to the discrete cosine transform. *Linear Algebra Appl.*, 252:1 – 25, 1997.

[9] T. Becker. Protein–protein-docking mit fast rotational matching. Master thesis, Institute of Mathematics, University of Lübeck, 2008.

[10] H. Berens and Y. Xu. On Bernstein-Durrmeyer polynomials with Jacobi weights. In C. K. Chui, editor, *Approximation, Interpolation and Functional Analysis*, pages 25 – 46. Academic Press Inc., Boston, 1991.

[11] H. Berman, K. Henrick, and H. Nakamura. Announcing the worldwide protein data bank. *Nature Structural Biology*, 10:980, 2003.

[12] G. Beylkin. On the fast Fourier transform of functions with singularities. *Appl. Comput. Harmon. Anal.*, 2:363 – 381, 1995.

[13] A. Bondi. Van der Waals volumes and radii. *J. Phys. Chem.*, 68:441 – 451, 1964.

[14] H. J. Bunge. *Mathematische Methoden der Texturanalyse*. Akademie Verlag, Berlin, 1969.

[15] J. E. Castrillon-Candas, V. Siddavanahalli, and C. Bajaj. Nonequispaced Fourier transforms for protein-protein docking. *ICES Report 05-44, Univ. Texas*, 199:122 – 140, 2005.

[16] S. Chandrasekaran and M. Gu. A divide-and-conquer algorithm for the eigendecomposition of symmetric block-diagonal plus semiseparable matrices. *Numer. Math.*, 96:723 – 731, 2004.

[17] G. S. Chirikjian and A. Kyatkin. *Engineering Applications of Noncommutative Harmonic Analysis: with Emphasis on Rotation and Motion Groups*. CRC Press, Boca Raton, 2001.

[18] C. W. Clenshaw. A note on the summation of Chebyshev series. *Math. Comput.*, 9:118 – 120, 1955.

[19] M. Connolly. Analytical molecular surface calculation. *J. Appl. Cryst.*, 16:548 – 558, 1983.

[20] J. W. Cooley and J. W. Tukey. An algorithm for machine calculation of complex Fourier series. *Math. Comput.*, 19:297 – 301, 1965.

[21] R. B. Corey and L. Pauling. Molecular models of amino acids, peptides, and proteins. *Review of Scientific Instruments*, 24:621 – 627, 1953.

[22] T. Creighton. *Protein Structures and Molecular Properties*. Freeman, New York, 1997.

[23] J. R. Driscoll and D. Healy. Computing Fourier transforms and convolutions on the 2–sphere. *Adv. Appl. Math.*, 15:202 – 250, 1994.

[24] J. R. Driscoll, D. Healy, and D. Rockmore. Fast discrete polynomial transforms with applications to data analysis for distance transitive graphs. *SIAM J. Comput.*, 26:1066 – 1099, 1996.

[25] B. Duncan and A. Olson. Shape analysis of molecular surfaces. *Biopolymers*, 33:231 – 238, 1993.

[26] A. Dutt and V. Rokhlin. Fast Fourier transforms for nonequispaced data. *SIAM J. Sci. Stat. Comput.*, 14:1368 – 1393, 1993.

[27] M. Eisenstein and E. Katchalski-Katzir. On proteins, grids, correlations, and docking. *C R Biol*, 327:409 – 420, 2004.

[28] G. E. Fasshauer. *Meshfree approximation methods with MATLAB*. World Scientific Publishing, Inc., Singapore, 2007.

[29] F. Filbir and D. Schmid. Stability results for approximation by positive definite functions on $\mathrm{SO}(3)$. *J. Approx. Theory*, 153:170 – 183, 2008.

[30] D. Fischer, S. Li, H. Wolfson, and R. Nussinov. A geometry-based suite of molecular docking processes. *J. Mol. Biol.*, 248:459 – 477, 1995.

[31] W. Freeden, T. Gervens, and M. Schreiner. *Constructive Approximation on the Sphere*. Oxford University Press, Oxford, 1998.

[32] M. Frigo and S. G. Johnson. FFTW, C subroutine library. URL `http://www.fftw.org`.

[33] R. Gabdoulline and R. Wade. Analytically defined surfaces to analyse molecular interaction properties. *J. Mol. Graph.*, 14:341 – 353, 1996.

[34] I. Gradstein and I. Ryshik. *Tables of Series, Products, and Integrals, Volume 2*. Verlag Harri Deutsch, Frankfurt am Main, 1981.

[35] M. Gräf and S. Kunis. Stability results for scattered data interpolation on the rotation group. *Electron. Trans. Numer. Anal.*, 31:30 – 39, 2008.

[36] M. Gräf and D. Potts. Sampling sets and quadrature formulae on the rotation group. *Numer. Funct. Anal. Optim.*, 30:665 – 688, 2009.

[37] J. J. Gray, S. Moughon, C. Wang, O. Schueler-Furman, B. Kuhlman, C. A. Rohl, and D. Baker. Protein–protein docking with simultaneous optimization of rigid-body displacement and side-chain conformations. *J. Mol. Biol.*, 331:281 – 299, 2003.

[38] L. Greengard and V. Rokhlin. A fast algorithm for particle simulations. *J. Comput. Phys.*, 73:325 – 348, 1987.

[39] T. Gutzmer. Interpolation by positive definite functions on locally compact groups with application to $\mathrm{SO}(3)$. *Results Math.*, 29:69 – 77, 1996.

[40] I. Halperin, B. Ma, H. Wolfson, and R. Nussinov. Principles of docking: An overview of search algorithms and a guide to scoring functions. *PROTEINS: Struct. Funct. Genet.*, 47:409 – 443, 2002.

[41] S. Helgason. *Groups and Geometric Analysis*. Academic Press Inc., New York, 1984.

[42] R. Hielscher. The Radon Transform on the Rotation Group – Inversion and Application to Texture Analysis. Dissertation, Department of Geology, Technical University Bergakademie Freiberg, 2007.

[43] R. Hielscher, J. Prestin, and A. Vollrath. Fast summation of functions on SO(3). *Math. Geosci.*, to appear.

[44] J.-S. Huang. *Lecture on Representation Theory*. World Scientific Publishing, Inc., Singapore, 1999.

[45] J. Janin and B. Seraphin. Genome-wide studies of protein-protein interaction. *Curr. Op. Struct. Biol.*, 13:383 – 388, 2003.

[46] Y. K. Kang, G. Némethy, and H. Scheraga. Free energies of hydration of solute molecules. 1. improvement of the hydration shell model by exact computations of overlapping volumes. *J. Phys. Chem.*, 91:4105 – 4109, 1987.

[47] E. Katchalski-Katzir, I. Shariv, M. Eisenstein, A. Friesem, C. Aflalo, and I. Vakser. Molecular surface recognition: determination of geometric fit between proteins and their ligands by correlation techniques. *Proc. Nat. Acad. Sci. USA*, 89:2195 – 2199, 1992.

[48] J. Keiner. Computing with Expansions in Gegenbauer Polynomials. *SIAM J. Sci. Comput.*, 31:2151 – 2171, 2009.

[49] J. Keiner, S. Kunis, and D. Potts. NFFT 3.0, C subroutine library. URL `http://www.tu-chemnitz.de/~potts/nfft`.

[50] J. Keiner, S. Kunis, and D. Potts. Fast summation of radial functions on the sphere. *Computing*, 78:1 – 15, 2006.

[51] J. Keiner and D. Potts. Fast evaluation of quadrature formulae on the sphere. *Math. Comput.*, 77:397 – 419, 2008.

[52] J. Keiner and A. Vollrath. Fast Fourier transform on the rotation group. in preparation.

[53] P. J. Kostelec and D. N. Rockmore. FFTs on the rotation group. *J. Fourier Anal. Appl.*, 14:145 – 179, 2008.

[54] J. A. Kovacs, P. Chacón, Y. Cong, E. Metwally, and W. Wriggers. Fast rotational matching of rigid bodies by fast Fourier transform acceleration of five degrees of freedom. *Acta Crystallogr. Sect. D*, 59:1371 – 1376, 2003.

[55] J. A. Kovacs and W. Wriggers. Fast rotational matching. *Acta Crystallogr. Sect. D*, 58:1282 – 1286, 2002.

[56] S. Kunis. Nonequispaced FFT - Generalisation and Inversion. Doctoral thesis, Institute of Mathematics, University of Lübeck, 2006.

[57] S. Kunis and D. Potts. Fast spherical Fourier algorithms. *J. Comput. Appl. Math.*, 161:75 – 98, 2003.

[58] B. Lee and F. Richards. The interpretation of protein structures: estimation of static accessibility. *J. Mol. Biol.*, 55:379 – 400, 1971.

[59] L. Li, R. Chen, and Z. Weng. RDOCK: refinement of rigid-body protein docking predictions. *PROTEINS: Struct. Funct. Genet.*, 53:693 – 707, 2003.

[60] A. Makadia and K. Daniilidis. Direct 3D Rotation Estimation via a generalized Shift Theorem. *Computer Vision Pattern Recognition*, 2003.

[61] P. Martinsson and V. Rokhlin. An accelerated kernel-independent fast multipole method in one dimension. *SIAM J. Sci. Comput.*, 29:1160 – 1178, 2007.

[62] D. K. Maslen and D. N. Rockmore. Generalized FFTs - A Survey of Some Recent Results. In L. Finkelstein and W. Kantor, editors, *DIMACS Workshop in Groups and Computation*, volume 28, pages 183 – 238, 1995.

[63] S. Matthies, G. Vinel, and K. Helmig. *Standard Distributions in Texture Analysis*. Akademie-Verlag Berlin, 1987.

[64] W. Miller. *Lie Theory and Special Functions*. Academic Press Inc., New York, 1968.

[65] M. J. Mohlenkamp. A fast transform for spherical harmonics. *J. Fourier Anal. Appl.*, 5:159 – 184, 1999.

[66] R. Norel, D. Petrey, H. Wolfson, and R. Nussinov. Examination of shape complementary in docking of unbound proteins. *Proteins*, 39:178 – 194, 1999.

[67] P. Palma, L. Krippahl, J. Wampler, and J. Moura. BiGGER: a new (soft) docking algorithm for predicting protein interactions. *PROTEINS: Struct. Funct. Genet.*, 39:372 – 384, 2000.

[68] D. Potts, J. Prestin, and A. Vollrath. A fast algorithm for nonequispaced Fourier transforms on the rotation group. *Numer. Algorithms*, 52:355 – 384, 2009.

[69] D. Potts and G. Steidl. Fast summation at nonequispaced knots by NFFTs. *SIAM J. Sci. Comput.*, 24:2013 – 2037, 2003.

[70] D. Potts, G. Steidl, and M. Tasche. Fast algorithms for discrete polynomial transforms. *Math. Comput.*, 67:1577 – 1590, 1998.

[71] D. Potts, G. Steidl, and M. Tasche. Fast Fourier transforms for nonequispaced data: A tutorial. In J. J. Benedetto and P. J. S. G. Ferreira, editors, *Modern Sampling Theory: Mathematics and Applications*, chapter 12, pages 247 – 270. Birkhäuser, Boston, 2001.

[72] D. Potts, G. Steidl, and M. Tasche. Numerical stability of fast trigonometric transforms - a worst case study. *J. Concrete Appl. Math.*, 1:1 – 36, 2003.

[73] T. Risbo. Fourier transform summation of Legendre series and D-Functions. *J. Geod.*, 70:383 – 396, 1996.

[74] D. W. Ritchie. High order analytic translation matrix elements for real six-dimensional polar Fourier correlations. *J. Appl. Cryst.*, 38:808 – 818, 2005.

[75] D. W. Ritchie. Recent progress and future directions in protein-protein docking. *Curr. Prot. Pep. Sci.*, 9:1 – 15, 2008.

[76] D. W. Ritchie and G. J. L. Kemp. Protein docking using spherical polar Fourier correlations. *PROTEINS: Struct. Funct. Genet.*, 39:178 – 194, 2000.

[77] V. Rokhlin and M. Tygert. Fast algorithms for spherical harmonic Expansions. *SIAM J. Sci. Comput.*, 27:1903 – 1928, 2006.

[78] M. Ruzhansky and V. Turunen. *Pseudo-Differential Operators and Symmetries*. Birkhäuser, Basel, 2009.

[79] D. Schmid. Marcinkiewicz-Zygmund inequalities and polynomial approximation from scattered data on $\mathrm{SO}(3)$. *Numer. Funct. Anal. Optim.*, 29:855 – 882, 2008.

[80] K. Sumikoshi, T. Terada, S. Nakamura, and K. Shimizu. A fast protein-protein docking algorithm using series expansions in terms of spherical basis functions. *Genome Informatics*, 16:161 – 193, 2005.

[81] G. Szegő. *Orthogonal Polynomials*. Amer. Math. Soc., Providence, 4th edition, 1975.

[82] T. Tilma and E. Sudarshan. Generalized Euler angle parameterization for SU(n). *J. Phys. A: Math. Gen.*, 35:10467 – 10501, 2002.

[83] R. Vandebril, M. V. Barel, G. Golub, and N. Mastronardi. A bibliography on semiseparable matrices. *Calcolo*, 42:249 – 270, 2005.

[84] D. Varshalovich, A. Moskalev, and V. Khersonski. *Quantum Theory of Angular Momentum*. World Scientific Publishing, Singapore, 1988.

[85] K. G. v.d. Boogaart, R. Hielscher, J. Prestin, and H. Schaeben. Kernel-based methods for inversion of the radon transform on $\mathrm{SO}(3)$ and their applications to texture analysis. *J. Comput. Appl. Math.*, 199:122 – 140, 2007.

[86] N. Vilenkin. *Special Functions and the Theory of Group Representations*. Amer. Math. Soc., Providence, 1968.

[87] N. J. Vilenkin and A. U. Klimyk. *Representation of Lie Groups and Special Functions*. Kluwer Academic Publishers, Dordrecht, 1991.

[88] N. O. Virchenko and I. Fedotova. *Generalized associated Legendre functions and their applications*. World Scientific Publishing, Inc., Singapore, 2001.

[89] A. Vollrath. Fast Fourier transforms on the rotation group and applications. Diploma thesis, Institute of Mathematics, University of Lübeck, 2006.

[90] H. Weyl. *The Theory of Groups and Quantum Mechanics*. Dover Publications, New York, 1931.

[91] E. P. Wigner. *Group Theory and its application to the Quantum Mechanics of Atomic Spectra*. Academic Press Inc., New York, 1959.

[92] S. Wodak and J. Janin. Computer analysis of protein-protein interaction. *J. Mol. Biol.*, 124:323 – 342, 1978.

[93] A. Yershova and S. M. LaValle. Deterministic sampling methods for spheres and SO(3). In *Proceedings. IEEE International Conference on Robotics and Automation.*, pages 3974 – 3980, 2004.