

DIPLOMARBEIT

Trigonometrische Interpolation auf  
verallgemeinerten dünnen Gittern mit  
beliebiger Levelstruktur

Angefertigt am  
Institut für Numerische Simulation

Vorgelegt der  
Mathematisch-Naturwissenschaftlichen Fakultät der  
Rheinischen Friedrich-Wilhelms-Universität Bonn

2. September 2014

Von

Kevin Matuschke

geboren am 10. Januar 1986  
in  
Meschede



# Eidesstattliche Erklärung

Hiermit erkläre ich, Kevin Matuschke, an Eides statt, dass ich die Diplomarbeit, Trigonometrische Interpolation auf verallgemeinerten dünnen Gittern mit beliebiger Levelstruktur, selbständig verfasst und keine anderen als die angegebenen Hilfsmittel benutzt sowie Zitate kenntlich gemacht habe.

Bonn, den 2. September 2014



# Danksagung

An dieser Stelle bedanke ich mich bei allen, die zum Gelingen meines Studiums beigetragen haben. Zu allererst sind dies meine Eltern und Familie für die langjährige finanzielle und seelische Unterstützung.

Mein Dank gilt Prof. Dr. Michael Griebel für die Chance meine Abschlussarbeit am Institut für Numerische Simulation zu schreiben und für die Zuteilung dieses interessanten Themas. Weiterhin möchte ich mich bei Prof. Dr. Jochen Garcke für die Übernahme des Zweitgutachtens bedanken.

Mein besonderer Dank gilt Dr. Jan Hamaekers, ohne dessen Hilfe ich diese Arbeit nicht in diesem Umfang hätte Erstellen können. Weiterhin möchte ich mich bei allen Mitarbeitern des Instituts für die gute Arbeitsatmosphäre und zahlreichen Diskussionen bedanken. Besonders erwähne ich in diesem Zusammenhang Jens Oettershagen und Dr. Christian Rieger.

Allen aktuellen und ehemaligen Mitarbeitern, die mich als studentische Hilfskraft im Institut eingearbeitet und unterstützt haben gehört auch meine Dankbarkeit. Hier möchte ich mich besonders bei Dr. Frederik Heber, Ralf Wildenhues, noch einmal bei Dr. Jan Hamaekers, sowie Ralph Thesen und Peter Zaspel bedanken.

Mein Dank gilt auch den Korrekturlesern, die mir bei der Suche nach Rechtschreibfehlern geholfen haben.

Und zu guter Letzt möchte ich mich bei meiner Freundin und allen meinen Freunden bedanken. Ich habe eine sehr harte Zeit durchgemacht um mein Studium zu beenden und hätte dies nie ohne deren Unterstützung geschafft.



# Notation

Symbol	Bedeutung
$\mathbf{x}, \mathbf{g}, \mathbf{p}$ $\mathbf{j}, \mathbf{k}, \mathbf{l}, \mathbf{m}$ $\mathbf{k}' \lesssim \mathbf{k}$ $\mathbf{k} \leq, <, \geq, > \mathbf{l}$	Vektoren mehrdimensionale Indizes mindestens eine Komponente ist kleiner komponentenweise Relation
$\mathbb{T} := [0, 2\pi]/\sim$ $\mathcal{L}^2(\mathbb{T}^D)$ $\mathcal{H}_{\text{mix}}^{t,l}(\mathbb{T}^D)$ $\mathcal{H}_w^\Gamma(\mathbb{T}^D)$ $\ f\ _{\mathcal{H}_{\text{mix}}^{t,l}}$ $\ f\ _{\mathcal{H}_w^\Gamma}$	Intervall, bei welchem die beiden Randpunkte miteinander Identifiziert werden Der Raum der quadratintegrierbaren Funktionen periodische Sobolevräume mit dominierend gemischter Glattheit gewichteter periodischer Sobolevraum Norm für Sobolevräume mit dominierend gemischter Glattheit gewichtete Sobolevnorm
$\mathcal{X}_l, \mathcal{S}_l, \mathcal{T}_l$ $\mathcal{I}, \mathcal{I}_l, \mathcal{I}_L^T$ $\mathcal{J}_v$ $\Omega_u$	Stützstellenmengen Indexmengen für Unterräume eines Gitters Indexmenge zur Indizierung hierarchischer Basisfunktionen in einem Unterraum Indexmenge endlicher Ordnung
$\alpha_{k,j}$ $\hat{f}_k$ $\check{f}_k$ $\phi_{l,j}(\mathbf{x})$ $\Phi_k(x)$ $\Phi_k(x)^*$ $\Psi_k(x)$ $\sigma : \mathbb{N}_0 \rightarrow \mathbb{Z}$	hierarchische Überschüsse bei Hutbasis Fourierkoeffizienten hierarchische Fourierkoeffizienten mehrdimensionale nodale Hut Basisfunktionen Fourier Basisfunktion: $\Phi_k(x) := e^{ikx}$ komplex konjugierte Fourierbasis Funktion hierarchische Fourier Basisfunktion Indexabbildung
$V_1$ $V_l^{SG}$ $V_{\mathcal{I}}$ $W_1$ $\mathbf{I}_{\mathcal{I}}[f]$ $\mathbf{I}_V[f]$ $\Delta_1[f](x)$	Vollgitterraum regulärer Dünngitterraum Gitterraum, der durch die Unterräume der Menge $\mathcal{I}$ aufgespannt wird hierarchischer Unterräume Interpolationsoperator zur Interpolation von $f$ auf dem Raum $V_{\mathcal{I}}$ Interpolationsoperator zur Interpolation von $f$ auf dem Raum $V$ hierarchischer Differenzoperator in den Unterraum $W_1$





# Inhaltsverzeichnis

<b>Danksagung</b>	<b>v</b>
<b>Notation</b>	<b>vii</b>
<b>1 Einleitung</b>	<b>1</b>
<b>2 Grundlagen</b>	<b>9</b>
2.1 Sobolevräume mit dominierend gemischter Glattheit . . . . .	9
2.2 Sobolevräume mit Gewichten endlicher Ordnung . . . . .	13
2.3 Reguläre dünne Gitter . . . . .	14
2.3.1 Interpolation durch stückweise lineare Funktionen . . . . .	15
2.3.2 Konstruktion dünner Gitter . . . . .	17
2.4 Fouriertransformation auf dünnen Gittern . . . . .	19
2.5 Verallgemeinerte dünne Gitter . . . . .	25
2.5.1 Konstruktion der Räume und Interpolationsoperatoren . . . . .	25
2.5.2 Gitterpunkte und Rechenaufwand . . . . .	26
2.5.3 Approximationseigenschaften . . . . .	29
2.5.4 Konvergenzrate in Abhängigkeit der Gitterpunkte und Kosten . . .	31
<b>3 Verallgemeinerte dünne Gitter mit beliebiger Levelstruktur</b>	<b>35</b>
3.1 Motivation . . . . .	35
3.2 Konstruktion des Interpolanten . . . . .	36
3.2.1 Stützstellenwahl . . . . .	37
3.2.2 Eindimensionale Transformationen . . . . .	40
3.2.3 Hierarchisierung und Dehierarchisierung . . . . .	42
3.3 Zahl der Freiheitsgrade . . . . .	44
3.4 Lineare Bestapproximation einer glatten Funktion auf einem PLUS1-Gitter	45
3.5 Rechenaufwand . . . . .	49
<b>4 Methoden</b>	<b>51</b>
4.1 Fouriertransformationen auf verallgemeinerten dünnen Gittern . . . . .	51
4.2 Adaptive Gitterkonstruktion . . . . .	55
4.3 Trigonometrische Interpolation auf verallgemeinerten dünnen Gittern mit beliebiger Levelstruktur . . . . .	58
4.3.1 Hierarchisierung und Dehierarchisierung . . . . .	58

4.3.2	Eindimensionale Transformationen . . . . .	63
4.3.3	Laufzeit und Speicherbedarf Basiskonstruktion . . . . .	66
<b>5</b>	<b>HCFFT Bibliothek</b>	<b>67</b>
5.1	Übersicht über die Funktionen . . . . .	67
5.2	Bibliotheken mit ähnlichen Funktionen . . . . .	68
5.3	Sparse Grid Interpolation Toolbox . . . . .	68
5.3.1	Vergleich von HCFFT und Sparse Grid Interpolation Toolbox . . . . .	69
5.3.2	Interpolationsgenauigkeit . . . . .	71
5.3.3	Geschwindigkeit Gitterkonstruktion . . . . .	75
5.3.4	Geschwindigkeit Funktionsauswertung . . . . .	78
5.4	NHCFFT . . . . .	81
<b>6</b>	<b>Numerische Experimente</b>	<b>83</b>
6.1	Funktionen mit polynomieller Konvergenzrate . . . . .	84
6.1.1	Nicht adaptive Gitter . . . . .	87
6.1.2	Adaptive Gitter . . . . .	93
6.2	Funktionen mit exponentieller Konvergenzrate . . . . .	95
6.2.1	Nicht adaptive Gitter . . . . .	95
6.2.2	Adaptive Gitter . . . . .	101
6.3	Hochdimensionale analytische Funktionen mit Gewichten geringer endlicher Ordnung . . . . .	105
6.4	Konstruktionszeit adaptiver Gitter . . . . .	110
<b>7</b>	<b>Schlussbemerkungen</b>	<b>113</b>
7.1	Zusammenfassung . . . . .	113
7.2	Ausblick . . . . .	115
	<b>Literaturverzeichnis</b>	<b>117</b>

# 1 Einleitung

Numerische Simulationen nehmen einen immer größeren Stellenwert in unserer Gesellschaft ein. Wo früher sehr viele kostspielige Versuche durchgeführt wurden, genügen heute einige wenige, welche durch Computerberechnungen gestützt werden. Bei der Verbesserung der Sicherheit unserer Autos ist es schon aus ökologischen Gesichtspunkten nicht wünschenswert Hunderte von Crashtests durchzuführen um Verbesserungen zu erzielen. Ein weiteres Beispiel ist der Bau von Hochhäusern. Hier können Versuche am ganzen Objekt im Vorfeld nicht durchgeführt werden. Wenn das Gebäude steht und festgestellt wird, dass das Material den Belastungen nicht stand hält, ist es zu spät. Weitere Bereiche in denen Computer Einzug gehalten haben sind die Entwicklung von Medikamenten, Berechnung von Finanzprodukten, Wettervorhersagen und vieles vieles mehr.

In dieser Arbeit geht es um die Verbesserung eines Disretisierungsverfahrens, welches für viele Problemstellungen, die sich bei numerischen Simulationen ergeben, eingesetzt wird. Allen Simulationen ist gemein, dass zuerst das Problem modelliert werden muss. Hierzu werden Prozesse beispielsweise durch Integral- und Differentialgleichungen beschrieben. Diese werden am Computer diskretisiert, das heißt dass von einem kontinuierlichen System in eines mit endlich vielen Datenpunkten übergegangen wird. Bei der Diskretisierung entstehen Fehler, die es zu minimieren gilt. Ein zweiter wichtiger Punkt bei der Diskretisierung ist der Aufwand des Verfahrens, der möglichst gering gehalten werden sollte.

Ein Klasse von Verfahren zur Diskretisierung sind die Spektralverfahren. Diese werden zur Lösung von Differentialgleichungen eingesetzt [CHQZ10]. Hierbei wird die Lösung einer Gleichung durch eine Linearkombination globaler Ansatzfunktionen beschrieben, welche im Falle dieser Arbeit die trigonometrischen Polynome sind. Die so gewählte Lösung stimmt im Allgemeinen nicht auf dem gesamten Gebiet mit der kontinuierlichen Lösung überein. Es verbleibt ein Residuum, welches minimiert wird. Hierzu wird eine Menge von Testfunktionen gewählt und das innere Produkt des Residuums mit jeder der Testfunktionen betrachtet. Die Linearkombination der Ansatzfunktionen soll nun so konstruiert werden, dass die inneren Produkte möglichst gering sind. Wählt man die Diracsche Delta Distribution für eine Menge von Punkten aus dem Gebiet, auf dem die Gleichung gelöst werden soll, so handelt es sich um ein spektrales Kollokationsverfahren. Hier stimmt die diskrete Lösung der Differentialgleichung auf den Punkten, die auch Stützstellen genannt werden, bis auf maschinell bedingte Rundungsfehler, exakt mit der kontinuierlichen Lösung überein.

Das wahrscheinlich einfachste Beispiel für ein spektrales Kollokationsverfahren ist die trigonometrische Interpolation. Diese Art der Interpolation ist besonders für periodische Funktionen geeignet. Diese werden als Linearkombination von Funktionen aus einer Fourierbasis dargestellt. Die Linearkombination soll auf den gewählten Stützstellen mit der Funktion übereinstimmen. Der Vorteil der Darstellung einer Funktion durch eine Kombination von Exponentialfunktionen ist die Möglichkeit diese sehr schnell partiell integrieren und differenzieren zu können. Im Fourierraum entsprechen diese Operationen einer skalaren Multiplikation, welche in linearer Zeit durchführbar ist.

Diese Arbeit beschäftigt sich mit der Konstruktion dünner Gitter zur trigonometrischen Interpolation von periodischen Funktionen, deren Fourierkoeffizienten schneller als polynomiell abfallen. Die verwendeten Gebiete sind kartesische Produkte eines Intervalls. Aus diesem wird eine Auswahl von Stützstellen getroffen und durch Tensorproduktbildung ein Gitter erzeugt, auf dessen Gitterpunkten die Funktion durch ein trigonometrisches Polynom interpoliert wird.

Bei Gitterbasierten Ansätzen zur Diskretisierung tritt das Problem des Fluchs der Dimensionen auf. Dies bedeutet, dass der Aufwand zur Berechnung einer Lösung mit einer festen Genauigkeit exponentiell von der Zahl der Dimensionen abhängt. Ein klassisches Vollgitter welches als Produkt einer eindimensionalen Stützstellenmenge der Länge  $2^L$  entsteht, hat  $2^{LD}$  Gitterpunkte. Schon bei wenig Dimensionen sind diese Gitter nicht mehr handhabbar.

Der Fehler einer Interpolation auf einem Gitter wird in einer Norm gemessen, beispielsweise in der  $\mathcal{L}_2$ -Norm. Der Fehler eines trigonometrischen Interpolanten einer Funktion, welche in eine Fourierreihe entwickelbar ist, hängt von den Koeffizienten der Basisfunktionen ab, die nicht in das Gitter aufgenommen wurden. Verallgemeinerte Sobolevräume mit dominierend gemischter Glattheit klassifizieren Funktionen nach dem Abfallverhalten ihrer Fourierkoeffizienten. Der Betrag der Fourierkoeffizienten fällt für Funktionen aus diesen Räumen polynomiell in Abhängigkeit des Index des Koeffizienten ab. Werden die Gitterpunkte nach dem Kriterium, dass alle Basisfunktionen, deren Koeffizienten größer als eine bestimmte Größenordnung sind, in das Gitter aufgenommen, so erhält man reguläre dünne Gitter. Die Indexmengen der Basisfunktionen dieser Gitter haben die Form eines hyperbolischen Kreuzes. Sie werden deshalb auch hyperbolisches Kreuz genannt.

Dünne Gitter [Zen91, Gri91] benutzen eine eindimensionale Multilevelbasis und verschachtelte Stützstellenmengen. Die Levelstruktur eines solchen Gitters heißt dyadisch, da sich die Zahl der Basisfunktionen von Level zu Level verdoppeln. Aus diesen Levels von Basisfunktionen werden Differenzen gebildet, die wiederum durch einen Tensorproduktansatz zu mehrdimensionalen disjunkten Unterräumen des vollen Gitters ausgebaut werden. Für ein dünnes Gitter werden nur diejenigen Unterräume ausgewählt, deren Index in der 1-Norm kleiner als ein Level  $l$  ist. Dies bildet das Abfallverhalten der Fourierkoeffizienten von Funktionen aus verallgemeinerten Sobolevräumen nach. Die Verteilung

---

der Stützstellen ist bei diesen Gittern äquidistant, weshalb eine Fouriertransformation zur Berechnung der Koeffizienten verwendet werden kann. Die Stützstellenzahl des so konstruierten Gitters hängt nur noch logarithmisch von der Zahl der Dimension ab. Die Interpolationsgenauigkeit ist jedoch, bis auf einen logarithmischen Term, so groß wie die eines vollen Gitters. Dieser Ansatz ermöglicht die Handhabung höher dimensionaler Probleme.

Die Idee der dünnen Gitter wurde über die Jahre weiter entwickelt und wird heute zur Berechnung hochdimensionaler Integrale [GG03], zum Lösen partieller Differentialgleichungen [NTT14] und zur Interpolation von Funktionen eingesetzt [GH14]. Bei manchen Differentialgleichungen werden Gitter benötigt, deren Indexmengen noch kleiner als die eines hyperbolischen Kreuzes sind. Dies führte zur Entwicklung verallgemeinerter dünner Gitter, bei denen eine parametrisierte monotone Indexmengen für die Auswahl der Unterräume verwendet wird. Die so konstruierten Gitter haben Indexmengen, deren Mächtigkeit in bestimmten Fällen unabhängig von der Zahl der Dimensionen ist. So ist es beispielsweise möglich hochdimensionale Funktionen zu interpolieren, deren effektive Dimension gering ist. Dies bedeutet dass die Funktion durch Basisfunktionen dargestellt werden kann, die das Produkt weniger eindimensionaler Funktionen sind. In diesem Rahmen wurde auch ein adaptiver Algorithmus, der die Unterräume nach ihrem Beitrag zur Interpolationsgenauigkeit auswählt.

Treten nun Funktionen auf, deren Fourierkoeffizienten schneller als polynomiell abfallen, ist auch ein verallgemeinertes dünnes Gitter keine gute Wahl. Bei diesen werden immer noch zu viele Basisfunktionen ausgewählt. Die optimale Indexmenge zur Interpolation einer Funktion mit diesem Abfallverhalten ist  $\mathcal{I} = \{\mathbf{k} : \|\mathbf{k}\|_1 < l\}$ . Ein notwendiges Kriterium für ein solches Abfallverhalten ist, dass es sich um eine glatte Funktion handelt. Wenn Funktionen das Tensorprodukt eindimensionaler analytischer Funktionen sind, so fallen die Fourierkoeffizienten sogar exponentiell ab.

Erst dieses Jahr wurden ähnlich konstruierte Gitter bei der Berechnung stochastischer partieller Differentialgleichungen verwendet. In [CCS14] wird ein Gitter mit einer beliebigen monotonen Indexmenge für die Basisfunktionen benutzt um eine Polynominterpolation mit Lagrange Polynomen durchzuführen. Als Stützstellen werden Leja Sequenzen verwendet. Die Parameter der Differentialgleichung genügen in dieser Arbeit gewissen Elliptizitätsannahmen, wodurch eine polynomielle Konvergenz, die unabhängig von der Anzahl der Dimensionen ist, erreicht wird. Weiterhin wird in der Arbeit ein adaptiver Algorithmus präsentiert, der auch eine Greedy Strategie verfolgt und die Basisfunktionen nach ihrem Beitrag zur Interpolationsgenauigkeit auswählt.

In [NTT14] werden auch dünne Gitter mit beliebiger Levelstruktur konstruiert. Es wird eine exponentielle Konvergenzrate für analytische Funktionen erreicht. Diese Arbeit betrachtet die Gitter sehr allgemein und es werden sowohl nicht verschachtelte, als

auch verschachtelte Mengen von Stützstellen behandelt. Die numerischen Experimente werden mit einer elliptischen parametrisierten Differentialgleichung durchgeführt, die als Inklusionsproblem bekannt ist [BNTT11]. Die Lösungen dieser Gleichung sind analytische Funktionen. Zur Berechnung einer Lösung wird zum einen ein Gitter mit verschachtelten Stützstellenmengen betrachtet. Dieses verwendet Clenshaw-Curtis Punkte zusammen mit Chebyshev Polynomen um einen Interpolanten zu konstruieren. Zum anderen wird ein Gitter mit nicht verschachtelten Gauss-Legendre Punkten betrachtet. Bei diesem wird eine Interpolation mit Legendre Polynomen durchgeführt.

In dieser Arbeit wird die Levelstruktur dünner Gitter verallgemeinert. So ist es möglich jede beliebige monotone Indexmenge als Auswahl für die Basisfunktionen zu verwenden. Für die numerischen Experimente werden in dieser Arbeit Gitter mit nur einer Basisfunktion in jedem Unterraum konstruiert. Gitter mit dieser Struktur bekommen den Namen PLUS1-Gitter, da bei den eindimensionalen Interpolationsregeln von Level zu Level eine Basisfunktion hinzukommt. Die Verteilung der Stützstellen ist, da die Mengen verschachtelt sind, zwangsläufig nicht äquidistant. Aus diesem Grund kann bei der Interpolation keine Fouriertransformation verwendet werden

Die auf diese Weise konstruierten Gitter sind optimal bezüglich der Menge der verwendeten Stützstellen, da jede Basisfunktion einzeln betrachtet werden kann. Gitter mit einer solchen Struktur sind auch als Total Degree Gitter bekannt. Diese werden bei der Interpolation von Funktionen durch Polynome verwendet. Die Basisfunktionen sind bei dieser Art Gitter ein Produkt von Monomen und der Name wird von der Summe der Exponenten der Monome abgeleitet.

Ein wesentlicher Erfolg dieser Arbeit ist die Interpolation hochdimensionaler Funktionen mit endlicher Ordnung 3 bis auf Maschinengenauigkeit. Hierzu werden in etwa  $10^5$  Stützstellen benötigt. Dyadische Levelstrukturen führen zu wesentlich größeren Stützstellenanzahlen um eine vergleichbare Genauigkeit zu erreichen. Insgesamt ist die Interpolationsgenauigkeit von Funktionen mit schnellem Abfall der Fourierkoeffizienten bei Verwendung eines PLUS1-Gitters um mehrere Größenordnungen besser als ein dyadisches Gitter. Für Anwendungen in denen periodische Funktionen mit einem solchen Abfallverhalten auftreten ist diese neue Gitterstruktur sehr interessant.

## Aufgabenstellung und Lösungsansatz

In dieser Arbeit wird ein dünnes Gitter für die trigonometrische Interpolation von Funktionen, deren Fourierkoeffizienten mehr als polynomiell abfallen, konstruiert. Unter der Annahme, dass die Fourierkoeffizienten ein Abfallverhalten der Art  $|\hat{f}_{\mathbf{k}}| < \|\mathbf{k}\|_1^{-n}$  für ein beliebig großes  $n \in \mathbb{N}$  haben, wird ein Gitter konstruiert, welches die optimalen Indexmengen für diese Funktionen realisiert. Da die Implementation auf vorhandenem Code basiert, muss dieser zuerst vervollständigt und mit alternativen Bibliotheken verglichen werden.

---

Trigonometrische Interpolation ist die Darstellung einer Funktion  $f$  als Linearkombination von Fourier Basisfunktionen. An die Linearkombination wird die Bedingung gestellt, dass diese auf einer Menge von Gitterpunkten exakt mit der Funktion übereinstimmt. Die Gitterpunkte von dünnen Gittern entstehen durch Tensorproduktbildung eindimensionaler Stützstellenmengen. Die Aufgabe ist es also den folgenden Interpolationsoperator zu implementieren. Hierbei ist  $\mathcal{I}$  eine beliebige monotone Indexmenge und stellt eine Auswahl der Unterräume dar, welche zusammen den Gitterraum zu dem Interpolationsoperator ergeben. Der Interpolationsoperator soll auf einer gegebenen Stützstellenmenge  $\mathcal{S}$  mit der Funktion übereinstimmen. Die genauen Konstruktionsdetails der Stützstellenmenge wird hier vernachlässigt.

$$I_{\mathcal{I}}[f](\mathbf{x}) := \sum_{\mathbf{l} \in \mathcal{I}} \sum_{\mathbf{k} \in \mathcal{J}_1} \hat{f}_{\sigma(\mathbf{k})} \Phi_{\sigma(\mathbf{k})}$$

so dass  $I_{\mathcal{I}}[f](\mathbf{p}) = f(\mathbf{p})$ , für alle  $\mathbf{p} \in \mathcal{S}$  (1.1)

Dies Interpolationsoperator wurde zur Konstruktion verallgemeinerter dünner Gitter verwirklicht. Er entsteht durch Tensorproduktbildung von Differenzen einer Hierarchie von eindimensionalen Interpolationsoperatoren  $\mathbf{I}_l[f](x)$ . Die Struktur der eindimensionalen Operatoren ist jedoch fest vorgegeben. Diese sind levelweise verschachtelt und von Level zu Level wird die Zahl der Basisfunktionen und Stützstellen verdoppelt. Ein Differenzoperator ist wie folgt gegeben

$$\Delta_{\mathbf{l}}[f](\mathbf{x}) = \bigotimes_{i=1}^D (\mathbf{I}_{l_i} - \mathbf{I}_{l_i-1})[f](x) := \sum_{\mathbf{k} \in \mathcal{J}_1} \check{f}_{\mathbf{k}} \Psi_{\mathbf{k}}$$

so dass  $\Delta_{\mathbf{l}}[f](\mathbf{p}) = f(\mathbf{p})$ , für alle  $\mathbf{p} \in \mathcal{S}_1$  (1.2)

Dieser Interpoliert die Funktion  $f$  auf dem Unterraum  $\mathcal{W}_1$  durch Verwendung einer hierarchischen Basis  $\Psi_{\mathbf{k}}$ .

Die optimale Indexmenge der Basisfunktionen zur Interpolation einer Funktion  $f$  mit einem Abfall der Fourierkoeffizienten, welcher schneller als polynomiell ist, sieht so aus  $\mathcal{I}_l = \{\mathbf{k} \in \mathbb{Z}^D : \|\mathbf{k}\|_1 \leq l\}$ . Diese Indexmenge ist monoton wird zur Konstruktion regulärer dünner Gitter, welche ein Spezialfall verallgemeinerter dünner Gitter sind, herangezogen.

In dieser Arbeit wird der Ansatz gewählt die Struktur der Unterräume  $\mathcal{W}_1$  zu verallgemeinern. Dies geschieht, in dem für die Hierarchie der eindimensionalen Interpolationsoperatoren  $\mathbf{I}_l$  eine beliebige Levelstruktur ermöglicht wird. Die allgemeinste Levelstruktur ist die Hinzunahme nur einer weiteren Basisfunktion und Stützstelle pro Level. Diese Struktur wird in dieser Arbeit mit PLUS1 bezeichnet. Eine solche Struktur führt zu Unterräumen, welche nur eine Basisfunktion enthalten. Auf diese Weise entsteht ein Interpolationsoperator, welcher eine beliebige monotone Indexmenge für die Basisfunktionen verwenden kann.

$$I_{\mathcal{I}}[f](\mathbf{x}) := \sum_{\mathbf{k} \in \mathcal{I}} \hat{f}_{\sigma(\mathbf{k})} \Phi_{\sigma(\mathbf{k})} \quad (1.3)$$

Hierzu müssen sowohl die eindimensionalen Transformationen als auch die Konstruktion der hierarchischen Basen überdacht werden. Bisher wurde zur Berechnung der Basisoeffizienten die schnelle Fouriertransformation verwendet. Diese kann nicht mehr benutzt werden, da die Zahl der Stützstellen beliebig ist. Weiterhin sind die Stützstellen nicht mehr äquidistant verteilt und somit kann auch die diskrete Fouriertransformation nicht verwendet werden um die eindimensionalen Interpolationsoperatoren  $\mathbf{I}$  auszuführen.

Die hierarchischen Basen, welche zur Konstruktion der Differenzoperatoren  $\Delta_1$  benutzt wurden, waren bisher die Summe zweier Fourierbasisfunktionen. Dies ist bei einer beliebigen Levelstruktur auch nicht mehr möglich.

Da sowohl die Berechnung der Fourierkoeffizienten bei den eindimensionalen Interpolationsoperatoren, als auch die Konstruktion der hierarchischen Basisfunktionen eine Linearkombination von Fourierbasisfunktionen erfordert, wird in dieser Arbeit auf die Lösung von linearen Gleichungssystemen mit Vandermonde Matrizen, welche aus Exponentialfunktionen bestehen, zurückgegriffen.

## Alternative Ansätze

Es gibt auch andere Lösungsansätze um mehrdimensionale Funktionen zu diskretisieren. Für Gitter, die als Tensorprodukt eindimensionaler Interpolationsregeln aufgebaut werden, gibt es ein Verfahren, das Kombinationstechnik genannt wird. Hierbei werden volle Gitter berechnet und die Koeffizienten eines dünnen Gitters als Differenzen und Summen der vollen Gitter ausgedrückt [GSZ92]. Diese Methode vermeidet die Verwendung hierarchischer Basen. Abbildung 1 veranschaulicht die Arbeitsweise der Kombinationstechnik.

Dieses Verfahren ist für die in dieser Arbeit vorgestellten PLUS1-Gitter ungeeignet, da sehr viele Vollgitter berechnet werden müssen um ein PLUS1-Gitter zu erhalten. Genauer gesagt müssen  $\mathcal{O}(l^{D-1})$  volle Gitter berechnet werden um ein D-dimensionales PLUS1-Gitter mit Level  $l$  zu konstruieren. Außerdem ist das in dieser Arbeit verwendete adaptive Verfahren für die Kombinationstechnik nicht verwendbar.

Gitter die nicht als Tensorprodukt konstruiert werden sind auch möglich. Hier gibt es beispielsweise eindimensionale Lattice Regeln um die Gitterpunkte zu wählen. Dabei wird ein Vektor erstellt und die Gitterpunkte sind ganzzahlige Vielfache dieses Vektors. Das in dieser Arbeit zur Berechnung des  $\mathcal{L}_2$ -Fehlers benutzte Verfahren ist ein solches [CKN06]. Außerdem gibt es ein Lattice Regel Verfahren zur trigonometrischen Interpolation [KKP12]. Dieses Verfahren hat eine bessere Stabilität als ein Dünnigitter bei der trigonometrischen Interpolation. Allerdings ist die Kondition für kleine Gitter, wie sie in dieser Arbeit bei der Interpolation glatter Funktionen auftreten, handhabbar [KK11].



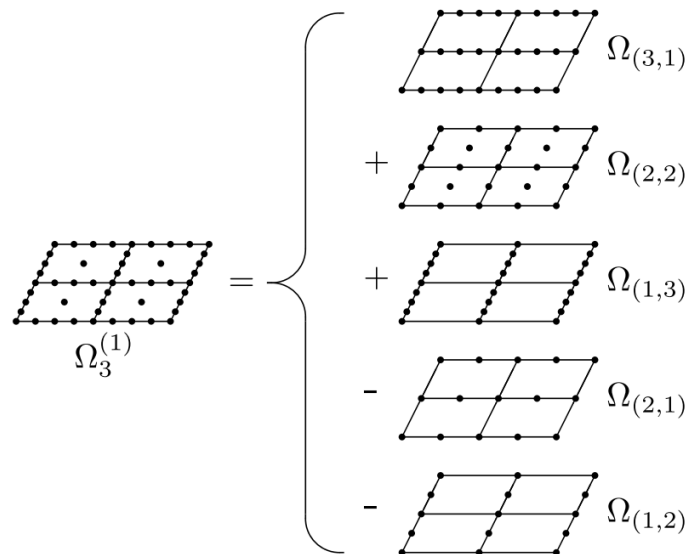


Abbildung 1: Kombinationstechnik zur Konstruktion eines zweidimensionalen dünnen Gitters des Levels 3.<sup>1</sup>

Ein weiterer Vorteil ist die hohe Geschwindigkeit, die sich dadurch ergibt, dass nur eine diskrete Fouriertransformation zur Berechnung der Koeffizienten durchgeführt wird. Ein Nachteil dieses Verfahrens ist, dass es die Struktur einer Funktion nicht ausnutzt. Beispielsweise werden Funktionen mit unterschiedlichem Abfallverhalten entlang der Dimensionen oder Funktionen mit Gewichten geringer endlicher Ordnung nicht gut erfasst.

Zur Interpolation von Funktionen auf Gebieten, die nicht rechteckig sind, gibt es Verfahren, welche Gitterpunkte von Intervallen auf nicht rechteckige Gebiete stetig abbilden [CHQZ10, Kop09]. Solche Gebiete werden in dieser Arbeit nicht betrachtet und sollen deshalb nur am Rande erwähnt werden.

## Eigene Beiträge

Die eigenen Beiträge dieser Arbeit lassen sich wie folgt zusammenfassen

- Untersuchung der Laufzeit und Interpolationsgenauigkeit des vorhandenen Codes, sowie ein Vergleich mit anderen Bibliotheken
- Implementation nützlicher Erweiterungen. Unter anderem wurde ein eindimensionales Lattice Rule Verfahren zur Bestimmung der  $\mathcal{L}_2$ -Fehlers eines Interpolanten

<sup>1</sup>entnommen aus [BG04] Seite 70

eingebaut. Weiterhin ist es jetzt möglich viele Punkte auf einmal auszuwerten, wodurch die Geschwindigkeit pro Auswertung merklich erhöht wurde.

- Es werden theoretische Aussagen über die Anzahl der Gitterpunkte, die Genauigkeit einer linearen Best Approximation und den Rechenaufwand eines PLUS1-Gitters hergeleitet.
- Implementation verallgemeinerter dünner Gitter mit beliebiger Levelstruktur, mit der Möglichkeit hierarchische Basen vorzuberechnen, abzuspeichern und für die Konstruktion der Gitter wieder zu verwenden
- Vergleich der Konvergenzrate, Interpolationsgenauigkeit und Laufzeit der dyadischen und PLUS1-Gitter bei Funktionen mit endlicher Regularität und glatten Funktionen. Es wurden sowohl adaptive als auch nicht adaptive Gitter untersucht.
- Signifikante Verbesserung der Konvergenz bei der Interpolation glatter und hochdimensionaler Funktionen geringer endlicher Ordnung

## Aufbau der Arbeit

Im ersten Kapitel wird die Aufgabenstellung und die Motivation zur Entwicklung verallgemeinerter dünner Gitter mit beliebiger Levelstruktur gegeben. Es folgen die Grundlagen zur Konstruktion dünner Gitter sowie die wichtigsten Aussagen über verallgemeinerte dünne Gitter in Kapitel 2. Das dritte Kapitel beinhaltet Theorie über die Verfahren die zur Konstruktion beliebiger Levelstrukturen notwendig sind sowie eigene Aussagen über PLUS1-Gitter. Im fünften Kapitel werden einige bisher existierende Bibliotheken mit der, in dieser Arbeit verwendeten, verglichen. Kapitel 6 ist das längste dieser Arbeit und enthält zahlreiche numerische Ergebnisse über die Interpolation glatter und nicht glatter Funktionen mit adaptiven und nicht adaptiven Gittern. In Kapitel 7 wird eine Zusammenfassung gegeben die zusammen mit zahlreichen Ideen und Möglichkeiten zur Weiterentwicklung diese Arbeit abschließt.

## 2 Grundlagen

In diesem Kapitel werden die Grundlagen, die zur Entwicklung verallgemeinerter dünner Gitter mit beliebiger Levelstruktur benötigt werden, zusammengetragen. Es wird ein Überblick über die Zahl der Gitterpunkte, den Rechenaufwand und die Konvergenzrate bei der trigonometrischen Interpolation von Funktionen endlicher Regularität auf verallgemeinerten dünnen Gittern gegeben. Die meisten der in diesem Kapitel aufgelisteten Lemmata sind aus [GH14] entnommen. Beweise, die aus langen Rechnungen bestehen werden hier verkürzt, aber mit Referenz zur ausführlichen Rechnung gegeben. Eine vollständig Herleitung verallgemeinerter dünner Gitter inklusive aller Beweise würde den Umfang dieser Arbeit übersteigen.

Zuerst werden Funktionen aus verallgemeinerten periodischen Sobolevräumen mit dominierend gemischter Glattheit definiert. Diese Räume wurden erstmals in [Kna00] konstruiert und stellen eine Verallgemeinerung der Sobolevräume dar. Funktionen aus diesen Räumen lassen sich in eine Fourierreihe entwickeln und werden über den Abfall ihrer Fourierkoeffizienten klassifiziert. Im weiteren Verlauf wird die Interpolation mit stückweise linearen Funktionen auf dünnen Gittern beschrieben. Hier soll eine Idee vermittelt werden, was dünne Gitter sind und worin der Vorteil ihrer Verwendung liegt. Die Konstruktion und Aussagen zur Interpolation mit stückweise linearen Funktionen auf dünnen Gittern sind aus [Gar13] entnommen. Hier findet sich eine kompakte Einführung in dieses Gebiet.

Es folgt eine Beschreibung der Fouriertransformation auf dünnen Gittern welche in [Hal92] entwickelt wurde. Dünne Gitter wurden mit der Zeit weiterentwickelt was zur Entwicklung verallgemeinerter dünner Gitter führt. Hier setzt im nächsten Kapitel die Interpolation auf Gittern mit beliebiger Levelstruktur an.

### 2.1 Sobolevräume mit dominierend gemischter Glattheit

Zu Beginn werden Sobolevräume mit dominierend gemischter Glattheit eingeführt. Diese bieten die Möglichkeit Funktionen über das Abfallverhalten ihrer Fourierkoeffizienten zu beschreiben. Gleichzeitig wird ein Rahmen zur Verfügung gestellt, in welchem Aussagen zur Konvergenzrate von Interpolationen für ganze Klassen von Funktionen getätigt werden können.

Sei  $\mathbb{T} := [0, 2\pi]/\sim$  ein Interval, bei welchem die beiden Randpunkte miteinander identifiziert werden. Dies hat zur Folge, dass alle Funktionen auf  $\mathbb{T}$  periodisch sind.

**Definition 2.1** (verallgemeinerte Sobolevräume mit dominierend gemischter Glattheit). Sei  $f \in \mathcal{L}^2(\mathbb{T}^n)$  und  $-\infty < t, l < \infty$ .

$$\mathcal{H}_{\text{mix}}^{t,l}(\mathbb{T}^D) := \left\{ f(\mathbf{x}) = \sum_{\mathbf{k} \in \mathbb{Z}^D} \hat{f}_{\mathbf{k}} e^{-i\mathbf{k} \cdot \mathbf{x}} : \|f\|_{\mathcal{H}_{\text{mix}}^{t,l}} < \infty \right\} \quad (2.1)$$

$$\|f\|_{\mathcal{H}_{\text{mix}}^{t,l}} := \left( \sum_{\mathbf{k} \in \mathbb{Z}^D} \prod_{d=1}^D (1 + |k_d|)^{2t} \cdot (1 + \|\mathbf{k}\|_{\infty})^{2l} \cdot |\hat{f}_{\mathbf{k}}|^2 \right)^{\frac{1}{2}} \quad (2.2)$$

Sowohl die isotropen Sobolevräume  $\mathcal{H}^s(\mathbb{T}^D) = \mathcal{H}_{\text{mix}}^{0,s}(\mathbb{T}^D)$  als auch Sobolevräume mit dominierend gemischter Glattheit  $\mathcal{H}_{\text{mix}}^t = \mathcal{H}_{\text{mix}}^{t,0}$  sind in dieser Definition enthalten.

Der Vorteil der Definition von Sobolevräumen über das Verhalten der Fourierkoeffizienten einer Funktion ist die Möglichkeit, Konvergenzaussagen zur Interpolation für ganze Klassen von Funktionen zu treffen. Diese leiten sich aus dem Abfall der Fourierkoeffizienten ab. Die Bedingung  $\|f\|_{\mathcal{H}_{\text{mix}}^{t,l}} < \infty$  besagt, dass  $|\hat{f}_{\mathbf{k}}| \lesssim \prod_{d=1}^D (1 + |k_d|)^{-t} \cdot (1 + \|\mathbf{k}\|_{\infty})^{-l}$  für fast alle  $\mathbf{k}$  gilt.

Die Abfallrate der Fourierkoeffizienten einer periodischen Funktion hängt von ihrer Differenzierbarkeit ab. Die nachfolgenden Aussagen zeigen den Zusammenhang zwischen der Differenzierbarkeit einer Funktion und dem Abfallverhalten ihrer Fourierkoeffizienten.

**Lemma 2.1** (spektrale Konvergenz). Sei  $f \in \mathcal{L}^2(\mathbb{T})$  eine periodische Funktion und  $m$ -mal stetig differenzierbar. Dann gilt für  $k \neq 0$

$$|\hat{f}_k| \lesssim |k|^{-m} \quad (2.3)$$

**Beweis.** Da  $f \in \mathcal{L}^2(\mathbb{T})$  ist  $f$  in eine Fourierreihe entwickelbar, welche in der  $\mathcal{L}_2$ -Norm gegen  $f$  konvergiert. Es gilt

$$f(x) = \sum_{k=-\infty}^{\infty} \hat{f}_k e^{ikx} \quad (2.4)$$

$$\hat{f}_k = \frac{1}{2\pi} \int_0^{2\pi} f(x) e^{-ikx} dx \quad (2.5)$$

Das Integral wird mittels partieller Integration ausgerechnet und es ergibt sich

$$\begin{aligned} \hat{f}_k &= \frac{1}{2\pi} \left( \frac{-1}{ik} (f(2\pi)e^{-ik2\pi} - f(0)e^{-ik0}) + \frac{1}{ik} \int_0^{2\pi} f'(x) e^{-ikx} dx \right) \\ &= \frac{1}{2\pi} \left( \frac{-1}{ik} (f(2\pi) - f(0)) + \frac{1}{ik} \int_0^{2\pi} f'(x) e^{-ikx} dx \right) \\ &= \frac{1}{2\pi} \frac{1}{ik} \int_0^{2\pi} f'(x) e^{-ikx} dx \end{aligned}$$

Die Iteration der partiellen Integration führt, da das Integral beschränkt ist, zu dem Ergebnis

$$|\hat{f}_k| = \left| \frac{1}{2\pi i^m} k^{-m} \cdot \int_0^{2\pi} \left( \frac{d}{dx} \right)^m f(x) e^{-ikx} dx \right| \lesssim |k|^{-m} \quad (2.6)$$

□

Für eindimensionale Funktionen gilt somit  $\mathcal{C}^k(\mathbb{T}) \subset \mathcal{H}^k(\mathbb{T})$ . Als Korollar ergibt sich, dass für eine glatte periodische Funktion  $f \in \mathcal{L}^2(\mathbb{T})$ , gilt

$$|\hat{f}_k| \lesssim |k|^{-m}, \text{ m beliebig} \quad (2.7)$$

Bei komplexen periodischen analytischen Funktionen, die auf einem Streifen  $|\operatorname{Im}(z)| < c$  um die reelle Achse definiert sind, ist die Abfallrate der Fourierkoeffizienten sogar exponentiell. Hier gilt

$$|\hat{f}_k| \lesssim e^{-i|k|x} \quad (2.8)$$

was eine Folge des Satzes von Paley und Wiener ist.

Die Aussage des Lemmas wird jetzt auf mehrdimensionale Funktionen übertragen.

**Lemma 2.2** (spektrale Konvergenz für mehrdimensionale Funktionen). *Sei  $f \in \mathcal{L}^2(\mathbb{T}^D)$  und  $m$ -mal stetig partiell differenzierbar, d.h.  $\frac{d^{\|\mathbf{m}\|_1}}{dx_1^{m_1} \dots dx_D^{m_D}} f$  existiert und ist stetig. Dann gilt für*

$$|\hat{f}_{\mathbf{k}}| \lesssim \prod_{d=1}^D |k_d|^{-m_d} \quad (2.9)$$

**Beweis.** Da  $f \in \mathcal{L}_2$ , ist  $f$  in einer Fourierreihe entwickelbar und für die Fourierkoeffizienten gilt

$$f(\mathbf{x}) = \sum_{\mathbf{k} \in \mathbb{Z}^D} \hat{f}_{\mathbf{k}} e^{ik_1 x_1} \dots e^{ik_D x_D} \quad (2.10)$$

$$\hat{f}_{\mathbf{k}} = \frac{1}{(2\pi)^D} \int_{\mathbb{T}^D} f(\mathbf{x}) e^{-ik_1 x_1} \dots e^{-ik_D x_D} d\mathbf{x} \quad (2.11)$$

Die partielle Integration nach der  $D$ -ten Koordinate ergibt

$$\begin{aligned}
 \hat{f}_{\mathbf{k}} &= \frac{1}{(2\pi)^D} \int_{\mathbb{T}} \dots \int_{\mathbb{T}} f(\mathbf{x}) e^{-ik_1 x_1} \cdot e^{-ik_2 x_2} \cdot \dots \cdot e^{-ik_D x_D} dx_D dx_{D-1} \dots dx_1 \\
 &= \frac{1}{(2\pi)^D} \int_{\mathbb{T}} e^{-ik_1 x_1} \int_{\mathbb{T}} e^{-ik_2 x_2} \dots \int_{\mathbb{T}} e^{ik_{D-1} x_{D-1}} \int_{\mathbb{T}} e^{ik_D x_D} f(\mathbf{x}) dx_D dx_{D-1} \dots dx_1 \\
 &= \frac{1}{(2\pi)^D} \int_{\mathbb{T}} e^{-ik_1 x_1} \int_{\mathbb{T}} e^{-ik_2 x_2} \dots \int_{\mathbb{T}} e^{ik_{D-1} x_{D-1}} \\
 &\quad \left( \frac{-1}{ik_D} (f(x_1, \dots, x_{D-1}, 2\pi) - f(x_1, \dots, x_{D-1}, 0)) + \frac{1}{ik_D} \int_{\mathbb{T}} \frac{d}{dx_D} f(\mathbf{x}) e^{-ik_D x_D} dx_D \right) \\
 &\quad dx_{D-1} \dots dx_1 \\
 &= \frac{1}{(2\pi)^D} \frac{1}{ik_D} \int_{\mathbb{T}^D} \frac{d}{dx_D} f(\mathbf{x}) e^{-ik_1 x_1} \cdot \dots \cdot e^{-ik_D x_D} d\mathbf{x}
 \end{aligned}$$

Durch  $m_D$ -fache Iteration der partiellen Integration nach der  $D$ -ten Koordinate folgt analog zum eindimensionalen Fall

$$\begin{aligned}
 \hat{f}_{\mathbf{k}} &= \frac{1}{(2\pi)^D} \frac{1}{ik_D} \int_{\mathbb{T}^D} \frac{d}{dx_D} f(\mathbf{x}) e^{-ik_1 x_1} \cdot \dots \cdot e^{-ik_D x_D} d\mathbf{x} \\
 &= \frac{1}{(2\pi)^D i^{m_D}} k_D^{-m_D} \int_{\mathbb{T}^D} \left( \frac{d}{dx_D} \right)^{m_D} f(\mathbf{x}) e^{-i\mathbf{x} \cdot \mathbf{k}} d\mathbf{x}
 \end{aligned}$$

Da  $f$  stetig ist, sind die Integrale vertauschbar. Jetzt wird dieses Vorgehen auf die  $D-1$ -te Koordinate angewendet

$$\hat{f}_{\mathbf{k}} = \frac{1}{(2\pi)^D} \frac{1}{i^{m_D} \cdot i^{m_{D-1}}} k_D^{-m_D} \cdot k_{D-1}^{-m_{D-1}} \cdot \int_{\mathbb{T}^D} \left( \frac{d}{dx_D} \right)^{m_D} \left( \frac{d}{dx_{D-1}} \right)^{m_{D-1}} f(\mathbf{x}) e^{-i\mathbf{k} \cdot \mathbf{x}} d\mathbf{x}$$

Dieses Vorgehen wird für die verbleibenden Koordinaten durchgeführt und es ergibt sich

$$|\hat{f}_{\mathbf{k}}| = \left| \frac{1}{(2\pi)^D} \prod_{d=1}^D k_d^{-m_d} \cdot \int_{\mathbb{T}^D} \frac{d^{\|\mathbf{m}\|_1}}{dx_1^{m_1} \dots dx_D^{m_D}} f(\mathbf{x}) e^{-i\mathbf{k} \cdot \mathbf{x}} d\mathbf{x} \right| \quad (2.12)$$

Dies führt zum gesuchten Ergebnis. □

Dieses Lemma trägt zum Verständnis des Zusammenhanges der Differenzierbarkeit einer Funktion, dem Abfall der Fourierkoeffizienten dieser und der Zugehörigkeit einer Funktion zu den Sobolevräumen bei. Im mehrdimensionalen Fall gilt, für den Raum der in alle Richtungen  $k$ -fach stetig differenzierbaren Funktionen, die Relation  $\mathcal{C}^k(\mathbb{T}^D) \subset \mathcal{H}_{\text{mix}}^{0,k}(\mathbb{T}^D)$ . Es gilt sogar die noch stärkere Aussage  $\mathcal{C}^k(\mathbb{T}^D) \subset \mathcal{H}_{\text{mix}}^{k,0}(\mathbb{T}^D)$ . Dies ist ersichtlich, wenn die Normen der Räume ausgerechnet werden.

$$\begin{aligned}
 \|f\|_{\mathcal{H}_{\text{mix}}^{t,0}} &= \sum_{\mathbf{k} \in \mathbb{Z}^D} \prod_{d=1}^D (1 + |k_d|)^t \cdot |\hat{f}_{\mathbf{k}}| = \sum_{\mathbf{k} \in \mathbb{Z}^D} \prod_{d=1}^D \sum_{i=0}^t \binom{t}{i} |k_d|^{t-i} \cdot |\hat{f}_{\mathbf{k}}| \\
 \|f\|_{\mathcal{H}_{\text{mix}}^{0,l}} &= \sum_{\mathbf{k} \in \mathbb{Z}^D} (1 + \|\mathbf{k}\|_{\infty})^l \cdot |\hat{f}_{\mathbf{k}}| = \sum_{\mathbf{k} \in \mathbb{Z}^D} \sum_{i=0}^l \binom{l}{i} \|\mathbf{k}\|_{\infty}^{l-i} \cdot |\hat{f}_{\mathbf{k}}|
 \end{aligned} \tag{2.13}$$

Werden die Größenordnungen der beiden Normen verglichen, so ergibt sich

$$\begin{aligned}
 \|f\|_{\mathcal{H}_{\text{mix}}^{t,0}} &\lesssim |k_1 \cdot k_2 \cdot \dots \cdot k_D|^t \cdot |\hat{f}_{\mathbf{k}}| \\
 \|f\|_{\mathcal{H}_{\text{mix}}^{0,l}} &\lesssim \|\mathbf{k}\|_{\infty}^l \cdot |\hat{f}_{\mathbf{k}}|
 \end{aligned} \tag{2.14}$$

Das Produkt der Komponenten des Index ist immer größer gleich der größten Komponente. Damit ist, wenn die Bedingung, dass die gemischte Sobolevnorm einen endlichen Wert für eine Funktion hat, auch die isotrope Sobolevnorm endlich. Somit ist der Raum der dominierend gemischten Glattheit im isotropen Raum gleicher Ordnung enthalten  $\mathcal{H}_{\text{mix}}^{t,0} \subset \mathcal{H}_{\text{mix}}^{0,t}$ .

Wenn eine mehrdimensionale Funktion das Tensorprodukt von eindimensionalen periodischen analytischen Funktionen ist, lässt sich das Abfallverhalten (2.8) übertragen und es ergibt sich

$$\begin{aligned}
 f &= f_1 \cdot \dots \cdot f_D = \sum_{k_1 \in \mathbb{Z}} \hat{f}_1 e^{ik_1 x_1} \cdot \dots \cdot \sum_{k_D \in \mathbb{Z}} \hat{f}_D e^{ik_D x_D} = \sum_{\mathbf{k} \in \mathbb{Z}^D} \hat{f}_{\mathbf{k}} e^{i\mathbf{k} \cdot \mathbf{x}} \\
 \hat{f}_{k_j} &\lesssim e^{-|k_j|}, \text{ für alle } 1 \leq j \leq D \Rightarrow \hat{f}_{\mathbf{k}} \lesssim e^{-\|\mathbf{k}\|_1}
 \end{aligned} \tag{2.15}$$

## 2.2 Sobolevräume mit Gewichten endlicher Ordnung

Es gibt den Fall, dass die Dimensionen einer Funktion getrennt betrachtet werden können. Dies bedeutet, dass hochdimensionale Funktionen durch Basisfunktionen, die ein Produkt weniger eindimensionaler Basisfunktionen sind, interpoliert werden. Diese Eigenschaft einer Funktion wird effektive Dimension genannt. Funktionen dieser Art treten beispielsweise in der Finanzmathematik bei der Berechnung hochdimensionaler Pfadintegrale auf. Eine geringe effektive Dimension ermöglicht die Handhabung hochdimensionaler Funktionen [WW04, SWW04].

In dieser Arbeit werden unter anderem Funktionen mit endlicher Ordnung 1 - 3 und bis zu 500 Dimensionen interpoliert. Hierzu werden sowohl adaptive als auch nicht adaptive Gitter verwendet. Um solch ein Verhalten zu beschreiben, wird die Definition der Sobolevräume um Gewichte endlicher Ordnung erweitert. Gleichzeitig wird die Definition weiter verallgemeinert. Die Faktoren, mit denen die Fourierkoeffizienten in der isotropen und gemischt glatten Norm Multipliziert werden, sind Gewichte. In der nachfolgenden

Definition kann eine allgemeinere Gewichtsfunktion gewählt werden, mit welcher die Fourierkoeffizienten multipliziert werden. Die Definition aus Abschnitt 2.1 ist ein Spezialfall der nachfolgenden Definition.

Es wird der Raum der periodischen Funktionen aus einem gewichteten Sobolevraum endlicher Ordnung eingeführt. Als Anmerkung sei gesagt, dass die hier definierte Indexmenge  $\mathcal{J}_1$  im Verlaufe dieser Arbeit zur Indizierung von bestimmten Basisfunktionen verwendet wird.

**Definition 2.2** (gewichteter Sobolevraum endlicher Ordnung). Sei  $\Gamma := \{\gamma_u\}_{u \subset \{1, \dots, D\}}$  mit  $\gamma_u \geq 0$  eine Menge von Gewichten. Sei  $w : \mathbb{Z}^D \rightarrow \mathbb{R}_+$  eine stetige Gewichtsfunktion. Sei  $\Omega_u$  eine Menge, welche Indizes mit endlicher Ordnung  $q = |u|$  enthält.

$$\Omega_u := \{\mathbf{l} \in \mathbb{N}_0^D : l_d = 0 \text{ für alle } d \in \{1, \dots, D\} \setminus u\} \quad (2.16)$$

Weiterhin sei die Menge  $\mathcal{J}_1 := \mathcal{J}_{l_1} \times \dots \times \mathcal{J}_{l_D}$  ein Tensorprodukt der eindimensionalen Indexmenge

$$\mathcal{J}_v := \begin{cases} \{0\} & , \text{ für } v = 0 \\ \{2^{v-1}, \dots, 2^v - 1\} & , \text{ für } v \geq 1 \end{cases} \quad (2.17)$$

Der gewichtete periodische Sobolevraum endlicher Ordnung ist definiert als

$$\mathcal{H}_w^\Gamma(\mathbb{T}^n) := \left\{ f(\mathbf{x}) = \sum_{\mathbf{k} \in \mathbb{Z}^n} \hat{f}_{\mathbf{k}} e^{i\mathbf{k} \cdot \mathbf{x}} : \|f\|_{\mathcal{H}_w^\Gamma} := \sqrt{\sum_{u \subset \{1, \dots, D\}} \frac{1}{\gamma_u} \sum_{\mathbf{l} \in \Omega_u} \sum_{\mathbf{k} \in \mathcal{J}_1} w(\mathbf{k})^2 |\hat{f}_{\mathbf{k}}|^2} < \infty \right\} \quad (2.18)$$

Die Zerlegung einer Funktion  $f$  auf die folgende Weise heißt ANOVA Zerlegung [ES81] und tritt in der Statistik häufig auf. Hier beschreiben die Gewichte endlicher Ordnung die Korrelationen zwischen den Variablen.

$$f = \sum_{u \subset \{1, \dots, D\}} f_u \quad \text{mit } f_u := \sum_{\mathbf{l} \in \Omega_u} \sum_{\mathbf{k} \in \mathcal{W}_1} \hat{f}_{\mathbf{k}} e^{i\mathbf{k} \cdot \mathbf{x}} \quad (2.19)$$

Für uns interessant ist hier die Menge der Funktionen  $f$  mit Gewichten geringer endlicher Ordnung. Eine Menge von Gewichten  $\Gamma_u = \{\gamma_u\}_{u \subset \{1, \dots, D\}}$  hat endliche Ordnung  $q$ , wenn  $\gamma_u = 0$  für alle  $\gamma_u \in \Gamma$  mit  $|u| > q$  gilt. Für eine Funktion mit  $D$  Dimensionen und endlicher Ordnung  $q$  gibt es  $\binom{D}{q} \lesssim D^q$  mögliche Untermengen  $u \subset \{1, \dots, D\}$  der Ordnung  $q$ . In Kapitel 6 wird eine Funktion konstruiert, bei der über zwei Parameter die Dimension und endliche Ordnung bestimmt werden kann. Bei dieser Funktion werden  $D$  der  $\binom{D}{q}$  möglichen Untermengen realisiert.

## 2.3 Reguläre dünne Gitter

Wir wiederholen erst die Grundlagen über reguläre dünne Gitter, bevor wir eine verallgemeinerte Form selbiger betrachten. Dünne Gitter wurden entwickelt, um dem Fluch der



Dimensionen, der bei vielen Problemen auftritt, zu begegnen. Die Idee geht zurück auf die Arbeit [Smo63], in welcher ein Tensorproduktansatz von verschachtelten eindimensionalen Quadraturregeln wachsender Größe benutzt wurde, um mehrdimensionale Funktionen zu integrieren. Auf die gleiche Weise werden Interpolationsoperatoren für mehrdimensionale Funktionen erzeugt.

Dünne Gitter wurden das erste mal in [Zen91] und [Gri91] vorgestellt und ursprünglich zur Lösung von partiellen Differentialgleichungen verwendet. Mittlerweile werden Sie auch zur Lösung von Integralgleichungen [GOS99], sowie zur Interpolation und Approximation von mehrdimensionalen Funktionen eingesetzt [Kna00, GK00]. Eine umfassende Übersicht über dünne Gitter und ihre Anwendung findet man hier [BG04]. Der Leser, der nur eine kurze Einführung in das Thema wünscht, findet diese hier [Gar13]. Die in dieser Arbeit verwendete Notation lehnt sich unter anderem an die in den beiden zuletzt genannten Artikeln verwendete an.

### 2.3.1 Interpolation durch stückweise lineare Funktionen

Um die Ideen, die hinter der Konstruktion dünner Gittern stecken, zu verstehen, wird zuerst die Interpolation mit stückweisen linearen Funktionen auf dünnen Gittern erklärt. Hier wird auf Aussagen zu Rechenaufwand und Fehlerabschätzungen verzichtet. Diese folgen im Abschnitt 2.5 über verallgemeinerte dünne Gitter, wobei hier die Interpolation durch eine Fourierbasis betrachtet wird. Wir werden den Interpolationsoperator für ein reguläres dünnes Gitter auf dem Einheitswürfel konstruieren und uns ansehen, wie ein solches Gitter aufgebaut ist. Um die Konstruktion lesbarer zu gestalten, wird als Bereich, aus dem die Funktionen interpoliert werden, der Einheitswürfel gewählt.

**Definition 2.3.** Sei  $f : [0, 1]^D \rightarrow \mathbb{R}$  eine reelle Funktion auf dem Einheitswürfel.

Zuerst werden verschachtelten Mengen von äquidistanten Stützstellen auf dem Intervall  $[0, 1]$  konstruiert.

**Definition 2.4.** Seien  $\mathcal{X}_l = \{x_{l,0}, \dots, x_{l,2^l} : x_{l,i} := \frac{i}{2^l}\}$  Stützpunktmengeten und  $\phi_{l,j}(x) := 1 - |2^l x - j|$  für  $x \in [\frac{j-1}{2^l}, \frac{j+1}{2^l}] \cap [0, 1]$  für  $j = 0, \dots, 2^l$  die nodale Hutbasis.

Die Zahl der Stützstellen nimmt von Level zu Level um den Faktor zwei zu. Für die Basisfunktionen eines Levels gelten die Bedingungen  $\phi_{l,j}(x_{l,i}) = \delta_{ij}$ . Aus den eindimensionalen Stützpunktmengeten und Basisfunktionen werden mehrdimensionale Gitter und Basisfunktionen mittels Tensorproduktbildung konstruiert.

**Definition 2.5.** Seien  $\mathbf{l}, \mathbf{j}$  Multiindizes. Eine mehrdimensionale Basisfunktion wird auf die folgende Weise durch Multiplikation gebildet

$$\phi_{\mathbf{l},\mathbf{j}}(\mathbf{x}) := \prod_{i=1}^D \phi_{l_i, j_i}(x_i) \tag{2.20}$$

Für die Stützpunkte gilt genau so  $x_{\mathbf{l},\mathbf{j}} := (x_{l_1,j_1}, \dots, x_{l_D,j_D})$ . Die Basisfunktionen spannen einen Raum auf, der Vollgitterraum heißt

$$V_{\mathbf{l}} := \text{span}\{\phi_{\mathbf{l},\mathbf{j}} | j_i = 0, \dots, 2^{l_i}, i = 1, \dots, D\} \quad (2.21)$$

Die Räume  $V_{\mathbf{l}}$  sind sogenannte Vollgitter, bei denen die Zahl der Stützstellen und damit der Aufwand exponentiell mit der Zahl der Dimensionen steigt. Die Idee der dünnen Gitter ist es die Räume in Unterräume aufzuteilen und nur einige von diesen zur Konstruktion eines Gitters heranzuziehen. Je nachdem welche Unterräume benutzt werden, hängt die Zahl der Stützstellen nur noch logarithmisch oder gar nicht mehr von der Anzahl der Dimensionen ab. Die Logarithmische Abhängigkeit erreicht man mittels regulärer dünner Gitter. Eine Unabhängigkeit von der Dimensionszahl ist bei Energienorm basierten dünnen Gittern der Fall [BG04].

Das Vollgitter  $V_{\mathbf{l}}$  wird nun hierarchisiert und auf die in der folgenden Gleichung stehende Weise, in Unterräume  $W_{\mathbf{l}}$  aufgeteilt, wobei  $e_i$  der  $i$ -te Einheitsvektor ist

$$W_{\mathbf{l}} := V_{\mathbf{l}} \setminus \bigoplus_{i=1}^D V_{\mathbf{l}-e_i} \quad (2.22)$$

Um die Definition zu vervollständigen, wird  $V_{\mathbf{l}} = 0$  gesetzt, wenn mindestens eine der Komponenten des Indexvektors  $\mathbf{l}$  negativ ist. Die Operation ist hierbei in dem Sinne zu verstehen, dass diejenigen Basisfunktionen aus der Basis des Vollgitterraumes  $V_{\mathbf{l}}$  entfernt werden, die in dem Raum  $\bigoplus_{i=1}^D V_{\mathbf{l}-e_i}$  enthalten sind. Der Unterraum  $W_{\mathbf{l}}$  wird dann durch die verbleibenden Basisfunktionen aufgespannt.

Dies wird mit Hilfe der Indexmengen  $\mathcal{J}_{\mathbf{l}}$  verdeutlicht

$$\mathcal{J}_{\mathbf{l}} = \left\{ \mathbf{j} \in \mathbb{N}^D : \begin{cases} 1 \leq j_i \leq 2^{l_i} - 1, j_i \text{ ungerade} & l_i > 0 \\ 0, 1 & l_i = 0 \end{cases}, 1 \leq i \leq D \right\}$$

Der Unterraum  $W_{\mathbf{l}}$  wird durch die Basisfunktionen, welche mit der Indexmenge  $\mathcal{J}_{\mathbf{l}}$  indiziert werden, aufgespannt

$$W_{\mathbf{l}} = \text{span}\{\phi_{\mathbf{l},\mathbf{j}} | \mathbf{j} \in \mathcal{J}_{\mathbf{l}}\}$$

Dies liegt an der Tatsache, dass die in einem Level neu hinzukommenden Stützstellen immer zwischen je zwei alten Stützstellen liegen und ihr Index daher ungerade ist.

Die Unterräume  $W_{\mathbf{l}}$  heißen auch hierarchische Unterräume. Für eine Hutbasis erhält man durch diese Aufteilung in Unterräume automatisch eine hierarchisierte Basis des Raumes  $V_{\mathbf{l}}$  durch  $\{\phi_{\mathbf{k},\mathbf{j}} : \mathbf{j} \in \mathcal{J}_{\mathbf{k}}, \mathbf{k} < \mathbf{l}\}$ . Diese zeichnet sich durch folgende Bedingung aus

$$\phi_{\mathbf{k},\mathbf{j}}(x_i) = 0 \text{ für } x_i \in \mathcal{I}_{\mathbf{k}'} \text{ mit } \mathbf{k}' \lesssim \mathbf{k}$$

Für einen mehrdimensionalen Index  $\mathbf{k}$  ist die Relation  $\mathbf{k}' \lesssim \mathbf{k}$  so zu verstehen

$$\mathbf{k}' \lesssim \mathbf{k} \Leftrightarrow \begin{cases} k'_i \leq k_i & 1 \leq i \leq D \\ k'_j < k_j & \text{mindestens ein } 1 \leq j \leq D \end{cases} \quad (2.23)$$

In der Abbildung 2 auf Seite 18 ist die Zerlegung des Raumes  $V_{3,3}$  in Unterräume zu sehen. Die hierarchischen Unterräume sind so konstruiert, dass die Stützstellen aller Räume, die kleiner als ein beliebiger Raum sind, nicht in der Vereinigung der Träger der Basisfunktionen in dem ausgewählten Raum liegen. Das bedeutet, dass die Zunahme weiterer Unterräume die Koeffizienten der Basisfunktionen in den kleineren Unterräumen nicht verändern. Das Konzept der hierarchischen Unterräume ist sehr wichtig, da es später die Konstruktion eines adaptiven Algorithmus und Gitter mit beliebigen monotonen Indexmengen für Unterräume ermöglicht. Diese Mengen werden in Definition 2.10 beschrieben.

Ein Vollgitterraum von Level  $l$  kann jetzt als direkte Summe von hierarchischen Unterräumen geschrieben werden.

$$V_l = \bigoplus_{\|\mathbf{k}\|_\infty \leq l} W_{\mathbf{k}}$$

Eine Funktion  $f \in V_l$  wird mittels des folgenden Interpolationsoperators interpoliert.

$$\mathbf{I}_{V_l}[f](\mathbf{x}) = \sum_{\|\mathbf{k}\|_\infty \leq l} \sum_{\mathbf{j} \in \mathcal{J}_{\mathbf{k}}} \alpha_{\mathbf{k},\mathbf{j}} \phi_{\mathbf{k},\mathbf{j}}(\mathbf{x})$$

Hierbei sind  $\alpha_{\mathbf{k},\mathbf{j}}$  die hierarchischen Überschüsse oder auch hierarchischen Koeffizienten, die sich als Differenz aus dem Funktionswert von  $f$  und dem Wert der Summe der Funktionswerte der Basisfunktionen  $\phi_{\mathbf{k},\mathbf{j}}$  der Unterräume, die in dem in (2.23) definierten Sinne kleiner als  $\mathbf{k}$  sind, an der Stelle  $x_{\mathbf{k},\mathbf{j}}$ , ergeben.

$$\alpha_{\mathbf{k},\mathbf{j}} = f(x_{\mathbf{j}}) - \sum_{\mathbf{k}' \lesssim \mathbf{k}} \sum_{\mathbf{j}' \in \mathcal{J}_{\mathbf{k}'}} \alpha_{\mathbf{k}',\mathbf{j}'} \phi_{\mathbf{k}',\mathbf{j}'}(x_{\mathbf{j}}) \quad (2.24)$$

Die Approximationseigenschaften eines Vollgitters mit Hutfunktionen sind in dieser Arbeit eher nebensächlich und können unter anderem in [BG04] nachgelesen werden. Dieser Operator unterliegt dem Fluch der Dimensionen. Das heißt dass die benötigte Anzahl an Gitterpunkten für eine festgelegtes Genauigkeit exponentiell mit der Zahl der Dimensionen wächst.

### 2.3.2 Konstruktion dünner Gitter

Die hierarchischen Überschüsse  $\alpha_{\mathbf{k},\mathbf{j}}$  fallen, für Interpolanten von Funktionen aus verallgemeinerten Sobolevräumen mit dominierend geschmischter Glattheit, abhängig vom Index des Unterraumes, in dem die zugehörigen Basisfunktionen enthalten sind, ab. Der Fehler hängt von den Koeffizienten ab, die nicht im Gitter enthalten sind und liegt mindestens in der Größenordnung des größten Koeffizienten, der nicht enthalten ist. Deshalb sollten die Unterräume so ausgewählt werden, dass die wegfallenden Koeffizienten eine gewisse Größe nie überschreiten.

Das folgende Lemma präzisiert den Abfall der Koeffizienten, wird aber ohne Beweis gebracht. Die Herleitung dieses Lemmas findet sich in [Gar13].

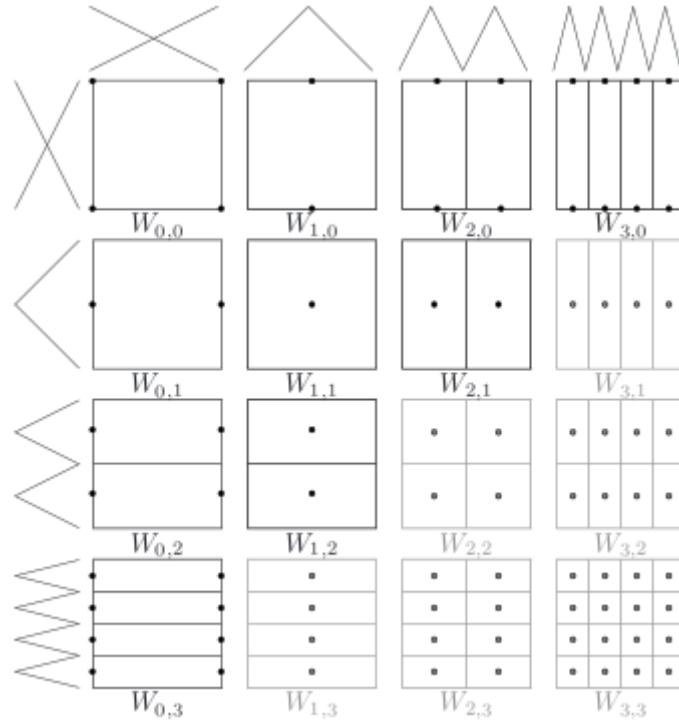


Abbildung 2: Hierarchische Unterraum Zerlegung von  $V_{3,3}$ . Jedes Quadrat repräsentiert einen Unterraum. Jeder Punkt ist ein Stützpunkt. Die Unterräume sind zerlegt, so dass ein inneres Rechteck der Träger der jeweils zu seinem Stützpunkt gehörenden Basisfunktion ist. Die ausgeblendeten Unterräume sind diejenigen, die bei einem dünnen Gitter des Levels 3 nicht enthalten sind. <sup>1</sup>

**Lemma 2.3.** Sei  $f \in \mathcal{H}_{mix}^{0,2}$  eine  $D$  dimensionale Funktion. Sei  $\mathbf{I}_{V_l}[f](\mathbf{x})$  der Interpolant von  $f$  und  $\alpha_{\mathbf{k},j}$  seine hierarchischen Koeffizienten. Dann gilt für die Koeffizienten des Raumes  $W_{\mathbf{k}}$

$$\|\alpha_{\mathbf{k},j}\|_2 \leq C(D) \cdot 2^{-2 \cdot \|\mathbf{k}\|_1} \cdot \|f\|_{\mathcal{H}_{mix}^{0,2}} \quad (2.25)$$

Es zeigt sich also dass die hierarchischen Überschüsse abhängig vom Index des Unterraumes, in welchem die zugehörigen Basisfunktionen liegen, abfallen. Für den Abfall ist die 1-Norm des Indizes von Bedeutung. Alle Unterräume, deren Index in der 1-Norm gleich ist, haben hierarchische Überschüsse gleicher Größenordnung. Die Unterräume sollten also in Form eines Standardsimplex ausgewählt werden. Dies führt zur Definition eines Dünngitterraumes.

$$V_l^{SG} := \bigoplus_{|\mathbf{k}|_1 \leq l} W_{\mathbf{k}} \quad (2.26)$$

<sup>1</sup>entnommen aus [Gar13] Seite 5

Die Gitter, denen diese Räume zu Grunde liegen, heißen reguläre dünne Gitter und erreichen wesentlich bessere Konvergenzraten, als volle Gittern. Das Verhältnis von Approximationsfehler und Zahl der Freiheitsgrade ist hier wesentlich besser, da die Zahl der Freiheitsgrade nur noch logarithmisch in Abhängigkeit der Dimensionen wächst, der Fehler hingegen nur logarithmisch schlechter als bei einem vollen Gitter ist. Auch für diese Aussage wird auf [BG04] verwiesen. Die ersten interessanten Abschätzungen zu Interpolationsfehlern und Zahl der Freiheitsgrade werden in Abschnitt 2.4 über Fouriertransformationen auf dünnen Gittern aufgeführt.

In der Abbildung 3 ist ein reguläres dünnes Gitter für zwei Dimensionen und Level fünf zu sehen. Die linke Grafik zeigt die Indizes der Unterräume. Rechts daneben sind die Gitterpunkte eines solchen Gitters zu sehen.

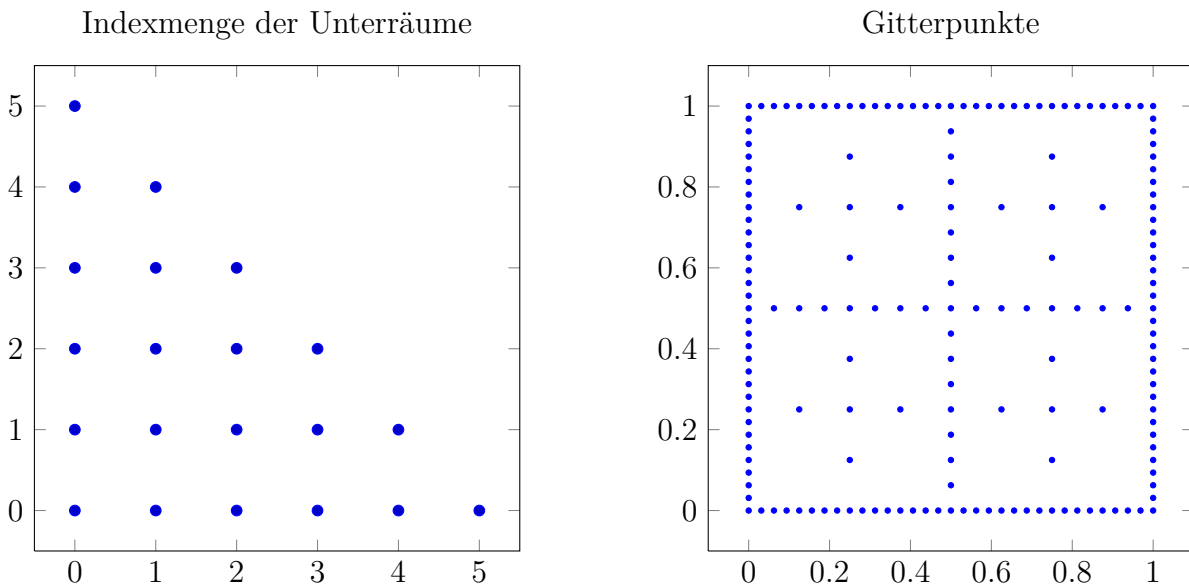


Abbildung 3: Indexmenge der Unterräume und Gitterpunkte eines regulären dünnen Gitters  $V_5^{SG}$ . Es wurde die Stützstellenmenge  $\mathcal{X}_5$  verwendet.

## 2.4 Fouriertransformation auf dünnen Gittern

Fouriertransformationen auf dünnen Gittern wurden in [Hal92] vorgestellt. Der Algorithmus, auf dem die HCFFT Bibliothek basiert und der auch für die trigonometrische Interpolation auf verallgemeinerten dünnen Gittern mit beliebiger Levelstruktur verwendet wird, stammt aus diesem Artikel. Der Grund dünne Gitter für Fouriertransformation zu verwenden, ist der gleiche, weshalb dünne Gitter zur Interpolation von Funktionen mit einer Hutbasis eingesetzt werden. Die Fourierkoeffizienten fallen abhängig vom Index des Unterraumes und der Differenzierbarkeit der zu Interpolierenden Funktion ab. In Abschnitt 2.1 wurde das Abfallverhalten von Fourierkoeffizienten für unterschiedliche

Funktionenklassen hergeleitet.

Jetzt werden periodische Funktionen aus Sobolevräumen mit dominierend gemischter Glattheit betrachtet. Diese werden mit Fourierpolynomen interpoliert und deren Approximationseigenschaften untersucht. Der wichtigste Schritt ist die Konstruktion der hierarchischen trigonometrischen Basis und der Umrechnungsmatrizen. Der Algorithmus für Fouriertransformationen auf einem dünnen Gitter wird in Kapitel 4 beschrieben.

**Definition 2.6.** Sei  $\mathbb{T}^D = [0, 2\pi]^D / \sim$  ein  $D$ -dimensionaler Torus, der durch Identifikation gegenüberliegender Randpunkte entsteht. Sei  $f \in \mathcal{H}_{mix}^{l,l}(\mathbb{T}^D) : \mathbb{T}^D \rightarrow \mathbb{R}$  eine  $D$ -dimensionale  $2\pi$ -periodische Funktion, das heißt dass  $f$  auf je zwei gegenüberliegenden Randpunkten den gleichen Funktionswert annimmt.

Die Stützpunktmengeten sind wieder verschachtelte Mengen äquidistanter Punkte, diesmal aus dem Intervall  $[0, 2\pi]$  anstatt  $[0, 1]$ . Es können auch beliebige Intervalle gewählt werden. Die Koordinaten werden durch eine bijektive Abbildung ineinander überführt. Diese Abbildungen müssen dann mit den Basisfunktionen verkettet werden. So lässt sich das Interpolationsgebiet als ein beliebiges Rechteck wählen, bei dem die gegenüberliegenden Punkte miteinander identifiziert werden. Die Interpolationseigenschaften bleiben unter den Koordinatentransformationen erhalten.

**Definition 2.7** (dyadische Stützstellen und Fourier Basisfunktion). Seien  $\mathcal{S}_l := \{m \frac{2\pi}{2^l} : m = 0, \dots, 2^l - 1\}$  durch  $l$  indizierte verschachtelte Stützstellenmengen und seien  $\Phi_k(x) := e^{ikx}$  die durch  $k$  indizierten Fourier Basisfunktionen.

Da die Entwicklung einer Funktion in eine Fourierreihe durch  $\mathbb{Z}$  indiziert ist, die Notation und implementation aber für Reihen mit Indizes aus  $\mathbb{N}_0$  leichter zu handhaben, wird die Indizierung durch eine bijektive Abbildung  $\sigma : \mathbb{N}_0 \rightarrow \mathbb{Z}$  geändert.

$$\sigma(n) = \begin{cases} -\frac{n}{2} & , \text{ für } n \text{ gerade} \\ \frac{n+1}{2} & , \text{ für } n \text{ ungerade} \end{cases} \quad (2.27)$$

**Definition 2.8** (eindimensionale Fourierbasis). Die Fourierbasis im Level  $l$  ist  $\mathcal{B}_l := \{\Phi_{\sigma(0)}, \dots, \Phi_{\sigma(2^l-1)}\}$  und  $V_l$  sei der von der Basis  $\mathcal{B}_l$  aufgespannte Raum.

Der Interpolationsoperator  $\mathbf{I}_l[f](x)$ , der  $f$  im Raum  $V_l$  interpoliert, wird wie folgt gebildet

$$\mathbf{I}_l[f](x) := \sum_{k=0}^{2^l-1} \hat{f}_{l,k} \Phi_{\sigma(k)}(x) \quad (2.28)$$

$$\hat{f}_{l,k} := 2^{-l} \sum_{x \in \mathcal{S}_l} \Phi_k^* f(x) \quad (2.29)$$

Hierbei ist  $\Phi_k^*$  die komplex konjugierte Fourierbasisfunktion. Der Raum  $V_n$  wird wieder hierarchisiert, das heißt als direkte Summe von Unterräumen  $\mathcal{W}_l$  geschrieben.

$$\begin{aligned}\mathcal{W}_0 &:= \text{span}(\mathcal{B}_0) \\ \mathcal{W}_l &:= \text{span}(\mathcal{B}_l \setminus \mathcal{B}_{l-1}), \text{ für } l \in \mathbb{N} \\ \Rightarrow V_n &= \sum_{l=0}^n \mathcal{W}_l\end{aligned}\quad (2.30)$$

Der Interpolationsoperator wird auch durch Differenzbildung hierarchisiert, so dass jeder Differenzoperator  $\Delta_n[f](x) : \mathcal{H}_{\text{mix}}^{t,l} \rightarrow \mathcal{W}_n$  die Funktion  $f$  auf einem Unterraum  $\mathcal{W}_n$  interpoliert.

$$\begin{aligned}\Delta_0[f](x) &:= \mathbf{I}_0[f](x) \\ \Delta_l[f](x) &:= \mathbf{I}_l[f](x) - \mathbf{I}_{l-1}[f](x) \text{ für } l \in \mathbb{N} \\ \Rightarrow \mathbf{I}_n[f](x) &= \sum_{l=0}^n \Delta_l[f](x)\end{aligned}\quad (2.31)$$

Mehrdimensionale Interpolationsoperatoren entstehen durch ein Tensorprodukt von eindimensionalen Operatoren. Die Indextmengen und Räume werden analog gebildet.

**Definition 2.9** (D-dimensionaler trigonometrischer Interpolationsoperator). *Seien  $\mathbf{k}$  und  $\mathbf{l}$  Multiindizes mit den Komponenten  $(k_i)_{1 \leq i \leq D}$  und  $(l_i)_{1 \leq i \leq D}$ .*

*Die D-dimensionale Stützstellenmenge  $\mathcal{S}_l = \mathcal{S}_{l_1} \times \dots \times \mathcal{S}_{l_D}$  wird durch ein kartesisches Produkt gebildet. Der entsprechende Vollgitterraum mit Index  $\mathbf{l}$  ist  $V_l = V_{l_1} \oplus V_{l_2} \oplus \dots \oplus V_{l_D}$  mit den Basisfunktionen  $\Phi_{\mathbf{k}}(\mathbf{x}) = e^{i\mathbf{k} \cdot \mathbf{x}}$ . Der zugehörige D-dimensionale trigonometrische Interpolationsoperator sieht wie folgt aus*

$$\begin{aligned}\mathbf{I}_l[f](\mathbf{x}) &:= \bigotimes_{i=1}^D \mathbf{I}_{l_i}[f](\mathbf{x}) = \sum_{\mathbf{k} \leq \mathbf{l}} \hat{f}_{l, \sigma(\mathbf{k})} \Phi_{\sigma(\mathbf{k})}(x) \\ \hat{f}_{l, \mathbf{k}} &:= 2^{-\|\mathbf{l}\|_1} \sum_{\mathbf{x} \in \mathcal{S}_l} f(\mathbf{x}) \Phi_{\mathbf{k}}^*\end{aligned}\quad (2.32)$$

Hierbei ist die Summe über einen mehrdimensionalen Index so zu verstehen  $\sum_{\mathbf{k} \leq \mathbf{l}} = \sum_{k_1}^{l_1} \sum_{k_2}^{l_2} \dots \sum_{k_D}^{l_D}$ . Die Relationen  $<$ ,  $>$ ,  $\leq$  und  $\geq$  gelten bei Vektoren und Multiindizes immer komponentenweise, genau so auch die Anwendung von  $\sigma$  auf einen Vektor oder Multiindex.

Da der eindimensionale trigonometrische Interpolationsoperator die Bedingungen  $f(x) = \mathbf{I}_l[f](x)$  für alle  $x \in \mathcal{S}_l$  erfüllt, gilt für das D-dimensionale Tensorprodukt aus diesen ebenfalls die entsprechenden mehrdimensionalen Bedingungen  $f(\mathbf{x}) = \mathbf{I}_l[f](\mathbf{x})$  für alle  $\mathbf{x} \in \mathcal{S}_l$ .

Durch die Differenzoperatoren  $\Delta_l[f](x)$  zerlegt man den Operator  $\mathbf{I}_l[f](\mathbf{x})$  auf dem Vollgitter  $V_l$  in hierarchische Operatoren und den Vollgitterraum entsprechend in hierarchische Unterräume. Dieses Vorgehen ist analog zu dem aus Abschnitt 2.3.2.

$$\begin{aligned}
\mathbf{I}_{\mathbf{n}}[f](\mathbf{x}) &= \bigotimes_{i=1}^D \mathbf{I}_{n_i}[f](\mathbf{x}) \\
&= \bigotimes_{i=1}^D \sum_{l_i=1}^{n_i} \Delta_{l_i}[f](x_i) = \sum_{\mathbf{l}} \bigotimes_{i=1}^D \Delta_{l_i}[f](x_i) \\
&= \sum_{\mathbf{l} \leq \mathbf{n}} (\Delta_{l_1} \times \Delta_{l_2} \times \dots \times \Delta_{l_D}) [f](\mathbf{x}) \\
&= \sum_{\mathbf{l} \leq \mathbf{n}} \Delta_{\mathbf{l}}[f](\mathbf{x}) \tag{2.33}
\end{aligned}$$

Es ist nun die Aufgabe die Unterräume  $\mathcal{W}_{\mathbf{l}}$ , in welche die Differenzoperatoren  $\Delta_{\mathbf{l}}$  interpolieren, zu konstruieren. Das heißt es muss eine Basis für jeden Unterraum angegeben werden. Diese Basis wird hierarchische Basis genannt. Hierzu werden zuerst Basen für die Unterräume  $\mathcal{W}_l$  konstruiert. Die Basen der Unterräume  $\mathcal{W}_{\mathbf{l}}$  entstehen dann durch Tensorproduktbildung.

Im eindimensionalen gilt  $\mathbf{I}_l[f](x) = \mathbf{I}_{l-1}[f](x)$  für alle  $x \in \mathcal{S}_{l-1}$ , woraus  $\Delta_l[f](x) = 0$  für alle  $x \in \mathcal{S}_{l-1}$  folgt. Dies bedeutet, dass jede einzelne Basisfunktion des Unterraumes  $\mathcal{W}_l$  auf den Stützstellen  $\mathcal{S}_{l-1}$  den Wert 0 annimmt. Weiterhin gilt  $\mathbf{I}_l[f](x) = \mathbf{I}_{l-1}[f](x) + \Delta_l[f](x)$ . Der Operator  $\Delta_n$  ist eine Linearkombination von hierarchischen Basisfunktionen, die jede beliebige Kombination von Funktionswerten auf den Stützpunkten von  $\mathcal{S}_l \setminus \mathcal{S}_{l-1}$  ermöglichen. Der Einfachheit halber werden die hierarchischen Basisfunktionen so definiert, dass jede nur auf einem Stützpunkt aus  $\mathcal{S}_l \setminus \mathcal{S}_{l-1}$  den Wert 1 annimmt. So lässt sich der hierarchische Überschuss, also der hierarchische Koeffizient zu der Basisfunktion wieder als Differenz des Funktionswertes und des Wertes des Interpolanten eines Levels kleiner angeben.

Die hierarchischen Basisfunktionen eines Unterraumes  $\mathcal{W}_l$  sind somit durch die folgenden Bedingungen definiert.

$$\Psi_j(x_i) = \sum_{k=0}^{2^l-1} \Phi_k(x_i) := \delta_{ij} \text{ für } j \in \{2^{l-1}, \dots, 2^l - 1\} \text{ und } i \in \{0, \dots, 2^l - 1\} \tag{2.34}$$

In [Hal92] wird hergeleitet, dass die, wie folgt definierten Funktionen die Bedingungen (2.34) erfüllen und eine hierarchische trigonometrische Basis für dyadische Stützstellen bilden. An dieser Stelle erleichtert  $\sigma$  die Notation. Die hierarchischen Basisfunktionen lassen sich im Falle dyadischer Stützstellen als Differenz zweier Exponentialfunktionen schreiben.

$$\Psi_j := \begin{cases} \Phi_0 & , \text{ für } j = 0 \\ \Phi_\sigma(j) - \Phi_\sigma(2^n - 1 - j) & , \text{ für } 2^{n-1} \leq j < 2^n, n > 0 \end{cases} \tag{2.35}$$



Die Differenzoperatoren lassen sich also explizit wie folgt ausschreiben

$$\Delta_n[f](x_i) = f(x_i) - \mathbf{I}_{n-1}[f](x_i) = \sum_{k=2^{n-1}}^{2^n-1} \check{f}_k \Psi_k(x_i) \quad \text{für } i \in \mathcal{S}_n \setminus \mathcal{S}_{n-1} \quad (2.36)$$

Hier ist  $\check{f}_k$  der  $k$ -te hierarchische Fourierkoeffizient und  $\Psi_k(x)$  die  $k$ -te hierarchische trigonometrische Basisfunktion. Die hierarchischen Koeffizienten werden mit einer linearen Abbildung, die Hierarchisierung genannt wird, aus den Fourierkoeffizienten berechnet. Die Basisfunktionen  $\Psi_k$ , die dem Operator  $\Delta_n$  zugeordnet sind, spannen den Raum  $\mathcal{W}_n$  auf. Dieser ist  $2^{n-1}$  dimensional.

Nun lassen sich auch die mehrdimensionalen Differenzoperatoren als Tensorprodukt aausschreiben

$$\Delta_{\mathbf{1}}[f](\mathbf{x}) = \sum_{\mathbf{k} \in \mathcal{J}_{\mathbf{1}}} \check{f}_{\mathbf{k}} \Psi_{\mathbf{k}} = \sum_{k_1=2^{l_1-1}}^{2^{l_1}-1} \cdots \sum_{k_D=2^{l_D-1}}^{2^{l_D}-1} \check{f}_{\mathbf{k}} \Psi_{\mathbf{k}} \quad (2.37)$$

Hier wird die Indexmenge  $\mathcal{J}_{\mathbf{1}}$  aus Definition 2.2 verwendet. Jeder Differenzoperator  $\Delta_{\mathbf{1}}$  spannt einen Unterraum  $\mathcal{W}_{\mathbf{1}} = \text{Im}(\Delta_{\mathbf{1}})$  des Vollgitterraumes auf. Dieser zerfällt somit in disjunkte Teilräume welche als direkte Summe den Vollgitterraum ergeben

$$V_{\mathbf{1}} = \bigoplus_{\|\mathbf{k}\|_{\infty} \leq l} \mathcal{W}_{\mathbf{k}} \quad (2.38)$$

Zur Berechnung der eindimensionalen hierarchischen Überschüsse wird wie folgt vorgegangen. Zuerst werden die Fourierkoeffizienten  $\hat{f}_k$  über eine Fouriertransformation berechnet. Im Anschluss daran werden diese mit einer linearen Abbildung, welche die Fourierbasis auf die hierarchische Basis abbildet, in die hierarchischen Koeffizienten umgerechnet. Diese lineare Abbildung wird, wie schon erwähnt, Hierarchisierung genannt. Da die Dimension der durch die hierarchische und Fourierbasis aufgespannten Räume gleich ist, existiert eine Umkehrabbildung, welche Dehierarchisierung heißt. Die Vorschrift zur Dehierarchisierung ist aus der Definition der hierarchischen Basisfunktionen in der obigen Gleichung (2.35) entnehmbar.

$$\begin{aligned} A_{hierar}^l &: \mathcal{B}_l \rightarrow \text{span}\{\Psi_0, \dots, \Psi_{2^l-1}\} \\ A_{dehier}^l &: \text{span}\{\Psi_0, \dots, \Psi_{2^l-1}\} \rightarrow \mathcal{B}_l \end{aligned}$$

Ein Vektor von hierarchischen Überschüssen ist mittels einer Matrix Vektor Multiplikation in einen Vektor von Fourierkoeffizienten umzurechnen.

$$\begin{pmatrix} \check{f}_0 \\ \vdots \\ \check{f}_{2^l-1} \end{pmatrix} = A_{hierar}^l \cdot \begin{pmatrix} \hat{f}_0 \\ \vdots \\ \hat{f}_{2^l-1} \end{pmatrix} \quad (2.39)$$

Die Linearkombination von hierarchischen Basisfunktionen  $\sum_{k=0}^{2^l-1} \hat{f}_k \Psi_k(x)$ , und die dehierarchisierte Kombination von Fourier Basisfunktionen  $\sum_{k=0}^{2^l-1} \hat{f}_k \Phi_k(x)$  stimmen im gesamten Definitionsbereich der Basisfunktionen überein.

Als Matrix ausgeschrieben sieht die Dehierarchisierungsmatrix für das Level 3 beispielsweise so aus

$$A_{\text{dehier}}^3 = \begin{pmatrix} 1 & -1 & 0 & -1 & 0 & 0 & 0 & -1 \\ 0 & 1 & -1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \quad (2.40)$$

Indem man die Dehierarchisierungsmatrizen invertiert, erhält man die Hierarchisierungsmatrizen. Für das Level 3 sieht diese so aus

$$A_{\text{hierar}}^3 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \quad (2.41)$$

Beide Matrizen sind dünn besetzt und die entsprechenden Matrix-Vektor-Multiplikationen lassen sich so implementieren, dass Sie mit linearem Aufwand eine Umrechnung durchführen. Diese Algorithmen sind in Kapitel 4 zu finden.

Zusammen mit den schnellen Fouriertransformationen, die für einen eindimensionalen Interpolationsoperator die Koeffizienten so berechnen, dass die Interpolationsbedingungen erfüllt sind, sind alle Werkzeuge vorhanden um Fouriertransformationen auf dünnen Gittern durchzuführen.

Um die hierarchischen Koeffizienten und Basisfunktionen eines Unterraumes  $\mathcal{W}_l$  zu bestimmen, benötigt man alle Unterräume  $\mathcal{W}_{l'}$ , deren Index die Bedingung  $l' \leq l$  erfüllt. Als dünne Gitter können somit nur Mengen von Unterräumen  $\{\mathcal{W}_{\mathcal{I}}\}$  ausgewählt werden, deren Indexmenge  $\mathcal{I}$  für jeden Index diese Bedingung erfüllt hat. Solche Indexmengen heißen beispielsweise monoton [CCS14]. Sie werden aber auch in früheren Arbeiten [GG03, GH14] zur Konstruktion von verallgemeinerten dünnen Gittern verwendet und haben dort den Namen zulässige Indexmengen. Beide Definitionen sind äquivalent. In dieser Arbeit wird der Begriff der monotonen Menge verwendet, da die Definition dieser genau der Bedingung an die Indexmenge  $\mathcal{I}$  der Unterräume  $\mathcal{W}_{\mathcal{I}}$  entspricht.

**Definition 2.10** (monotone Menge). Sei  $\mathcal{I} \subset \mathbb{N}_0^D$  eine nicht leere Menge. Diese wird monoton genannt, wenn sie die folgende Bedingung erfüllt

$$\text{Falls } \mathbf{k} \in \mathcal{I} \wedge \mathbf{k}' \leq \mathbf{k} \Rightarrow \mathbf{k}' \in \mathcal{I} \quad (2.42)$$

Die Indexmenge, die zur Konstruktion eines dünnen Gitters in Gleichung (2.26) im vorherigen Abschnitt verwendet wurde, ist eine monotone Menge. Auf einem dünnen Gitter sind Fouriertransformationen mit den in diesem Abschnitt beschriebenen Algorithmen möglich. Diese haben den folgenden Operator

$$\mathbf{I}_l^{SG}[f](\mathbf{x}) := \bigoplus_{\|\mathbf{k}\|_1 \leq l} \Delta_{\mathbf{k}}[f](\mathbf{x}) \quad (2.43)$$

Dieser interpoliert eine Funktion  $f$  auf dem Raum  $V_l^{SG}$  an den zu dem Operator gehörenden Stützstellen mit einer hierarchischen Basis. Diese kann durch sukzessives Anwenden der Dehierarchisierung entlang jeder Richtung in einer Fourierbasis umgerechnet werden. Die Approximationseigenschaften werden im Abschnitt 2.5 über verallgemeinerte dünne Gitter näher behandelt.

## 2.5 Verallgemeinerte dünne Gitter

Im vorherigen Abschnitt wurde ersichtlich, dass Fouriertransformationen nicht nur auf regulären dünnen Gittern, sondern auf dünnen Gittern mit jeder Indexmenge  $\mathcal{I}$ , die monoton ist, durchgeführt werden kann. Hierdurch sind adaptive Verfahren möglich, da in der hierarchischen Darstellung immer Unterräume hinzugefügt werden können, ohne den Interpolanten auf den bisherigen Stützstellen zu verändern. Diese Idee findet vielfältig Verwendung. Beispielsweise bei der Integration von mehrdimensionalen Funktionen [GG98,GG03] oder zur Interpolation mit von Funktionen mit Fourierbasen [GH14].

Jetzt werden verallgemeinerte Dünngitterräume konstruiert und Fouriertransformationen auf diesen durchgeführt. Es werden die Approximationseigenschaften des Interpolanten untersucht. Dabei wird ersichtlich, dass die Definition verallgemeinerter Sobolevräume mit dominierend gemischter Glattheit es ermöglicht die Konvergenzrate der Interpolanten an Hand der Abfallrate der Fourierkoeffizienten der Funktionen aus diesen Räumen abzulesen.

Verallgemeinerte dünne Gitter enthalten die regulären dünnen Gitter und Vollgitter als Spezialfall. Deshalb wurde bisher auf die Untersuchung von Approximationseigenschaften und den Rechenaufwand verzichtet.

### 2.5.1 Konstruktion der Räume und Interpolationsoperatoren

Untersuchungen der Konvergenzeigenschaften von dünnen Gittern zur Lösung von elliptischen partiellen Differentialgleichungen [GK00] ergaben, dass sogar weniger Unterräume als bei einem regulären dünnen Gitter verwendet werden können, ohne die Approximationsrate zu verschlechtern. Die gesuchte Funktion stammte hierbei aus einem Sobolevraum

mit dominierend gemischter Glattheit. Die Indexmengen, die diese Gitter beschreiben sind mit einem Parameter  $T$  versehen und heißen parametrisierte Indexmengen. Die von Ihnen erzeugten Gitter sind die verallgemeinerten dünnen Gitter.

**Definition 2.11** (parametrisierte Indexmenge). *Sei  $T \in [-\infty, 1)$  und  $L \in \mathbb{N}$ . Die parametrische Indexmenge  $\mathcal{I}_L^T$  ist wie folgt definiert*

$$\mathcal{I}_L^T := \{\mathbf{l} : \|\mathbf{l}\|_1 - T\|\mathbf{l}\|_\infty \leq (1 - T)L\}, \quad T < 1 \quad (2.44)$$

Diese Menge ist eine zulässige Indexmenge und führt zu der Definition der Räume der verallgemeinerten dünnen Gitter die als direkte Summe der Unterräume, die in der parametrisierten Indexmenge liegen, geschrieben werden.

$$V_{\mathcal{I}_L^T} := \bigoplus_{\mathbf{l} \in \mathcal{I}_L^T} \mathcal{W}_1 \quad (2.45)$$

**Definition 2.12** (Interpolationsoperator für ein verallgemeinertes dünnes Gitter). *Sei  $\mathcal{I}_L^T$  eine durch  $T$  parametrisierte Indexmenge. Die Gitter, welche von diesen gebildet werden heißen verallgemeinerte dünne Gitter und haben den Interpolationsoperator*

$$\mathbf{I}_{\mathcal{I}_L^T}[f](\mathbf{x}) := \bigoplus_{\mathbf{k} \in \mathcal{I}_L^T} \Delta_{\mathbf{k}}[f](\mathbf{x}) \quad (2.46)$$

In dieser Definition werden die vollen Gitter durch die Indexmenge  $\mathcal{I}_L^{-\infty}$  gebildet, und die regulären dünnen Gitter haben  $\mathcal{I}_L^0$  als Indexmenge.

## 2.5.2 Gitterpunkte und Rechenaufwand

In den Artikeln [GK00, GK09] wurden Abschätzungen zur Zahl der Freiheitsgrade und den Approximationseigenschaften der verallgemeinerten dünnen Gitter hergeleitet, die hier wiederholt werden. Die Zahl der Freiheitsgrade oder auch Gitterpunkte werden als Kosten angesehen, da der Aufwand zur Konstruktion und Auswertung eines Interpolanten von diesen abhängt. Im wissenschaftlichen Rechnen sind vor allem die Konstruktions- und Auswertungszeiten sowie der Speicherbedarf eines Gitters interessant. Im Kapitel 6 ist zu sehen, dass die Auswertungszeit dank effizienter Datenstrukturen der Unterräume linear von der Zahl der Freiheitsgrade abhängt.

Auf einfache Weise kann man die Zahl der Gitterpunkte eines dünnen Gitters, auf Grund der dyadischen Levelstruktur der eindimensionalen Interpolationsoperatoren, wie folgt abschätzen

$$|V_{\mathcal{I}_L^T}| \lesssim \sum_{\mathbf{l} \in \mathcal{I}_L^T} 2^{|\mathbf{l}|_1} \quad (2.47)$$

Von einem Level zum nächsten wächst die Zahl der Gitterpunkte um den Faktor zwei. Dies bedeutet, dass der Unterraum  $\mathcal{W}_l$   $2^l$  Stützstellen hat und der Unterraum  $\mathcal{W}_1$ , welcher als Tensorprodukt gebildet wird, hat  $2^{l_1} \cdot 2^{l_2} \cdot \dots \cdot 2^{l_D} = 2^{|\mathbf{l}|_1}$  Stützstellen.

Diese Abschätzung lässt sich auch weiter ausrechnen und es stellt sich heraus, dass bei Gittern mit  $T > 0$  die Zahl der Freiheitsgrade unabhängig von der Dimension ist. Die folgenden Lemmata sind aus den beiden zu Beginn des Unterabschnitts genannten Arbeiten und aus dem Artikel [GH14] zusammengetragen. Die Beweise werden hier nicht für jedes Lemma gegeben, da diese teilweise aus langen Rechnungen bestehen und den Rahmen dieser Arbeit übersteigen würden. In diesem Kapitel wird ein Überblick über die bisher bekannten Ergebnisse präsentiert.

**Lemma 2.4** (Zahl der Freiheitsgrade). *Sei  $L \in \mathbb{N}_0$  und  $T < 1$ . Die Zahl der Freiheitsgrade eines verallgemeinerten dünnen Gitterraumes  $V_{\mathcal{I}_L^T}$  in Abhängigkeit vom Diskretisierungsparameter  $L$  sind*

$$|V_{\mathcal{I}_L^T}| \lesssim \sum_{\mathbf{k} \in \mathcal{I}_L^T} 2^{|\mathbf{k}|_1} \lesssim \begin{cases} 2^L & \text{für } 0 < T < 1 \\ 2^L L^{D-1} & \text{für } T = 0 \\ 2^L \frac{T-1}{T/D-1} & \text{für } T < 0 \\ 2^{LD} & \text{für } T = -\infty \end{cases} \quad (2.48)$$

Nachfolgend wird der Rechenaufwand abgeschätzt um ein Gitter  $V_{\mathcal{I}_L^T}$  zu konstruieren. Auf Grund der Tensorproduktstruktur des Gitters zerfällt die Konstruktion in die sukzessive Ausführung von eindimensionalen DFTs und Hierarchisierungen sowie Dehierarchisierungen.

Der Aufwand einen Koeffizientenvektor zu hierarchisieren bzw. dehierarchisieren hängt linear von seiner Länge  $n$  ab. Die Gleichung (2.35) der hierarchischen Basis  $\Psi_{\mathbf{k}}$  für eine dyadische Levelstruktur zeigt, dass eine Addition benötigt wird um einen hierarchischen Koeffizienten auszurechnen. Um aus einer Menge von Datenpunkten die Fourierkoeffizienten auszurechnen, hat die schnelle Fouriertransformation eines Vektors der Länge  $2^L$  den Aufwand  $\mathcal{O}(L2^L)$ .

Jetzt wird noch die Zahl der eindimensionalen Koeffizientenvektoren in dem Gitter bestimmt um den Gesamtaufwand zu erhalten. Es werden die Indizes  $\mathcal{M}_d(\mathcal{I}) := \{\mathbf{k} \in \mathcal{I} : \mathbf{k} + \mathbf{e}_d \notin \mathcal{I}\}$  der Unterräume betrachtet, deren Nachfolger entlang einer Dimension nicht mehr im Gitter enthalten ist. Diese Indizes liefern dann die Zahl und Länge der eindimensionalen Koeffizientenvektoren, woraus der Rechenaufwand abgeschätzt werden kann. Die Indexmenge der Gitter wird in eindimensionale Streifen zerlegt und diejenigen der gleichen Länge werden zusammengefasst. Hierdurch entstehen in der Indexmenge der Basisfunktionen Rechtecke gleicher Kantenlänge. In der englischen Literatur wird der Begriff Pencil für ein Rechteck von Indizes von Basisfunktionen benutzt.

Für ein Vollgitter  $V_{\mathcal{I}_L^{-\infty}}$  ist dies sehr einfach, da alle eindimensionalen Koeffizientenvektoren Länge  $2^L$  haben. Insgesamt gibt es näherungsweise  $2^{L(D-1)}$  Koeffizientenvektoren entlang einer Achse. Hieraus folgt für das volle Gitter der Aufwand, der nachfolgend  $\mathcal{T}[V_{\mathcal{I}_L^{-\infty}}]$  genannt wird

$$\mathcal{T}[V_{\mathcal{I}_L^{-\infty}}] \lesssim \sum_{d=1}^D 2^{L(D-1)} \cdot (2^L + \mathcal{O}(L2^L)) = \mathcal{O}(LD \cdot 2^{LD}) \quad (2.49)$$

In den der Abbildung 4 wird dieses Vorgehen für ein zweidimensionales Voll- und Dünngitter gezeigt. In der Abbildung ist die Indexmenge  $\mathcal{M}_d(\mathcal{I})$  rot markiert. Dies ist genau die Indexmenge der Unterräume, mit denen man die Zahl und Länge der Koeffizientenvektoren des Gitters bestimmt. Die Länge eines Koeffizientenvektors, in einer Reihe von Unterräumen, deren letzter entlang einer Richtung durch den Index  $\mathbf{l} \in \mathcal{M}_d$  gegeben ist, ist auf Grund der dyadischen Levelstruktur  $2^{l_d}$ .

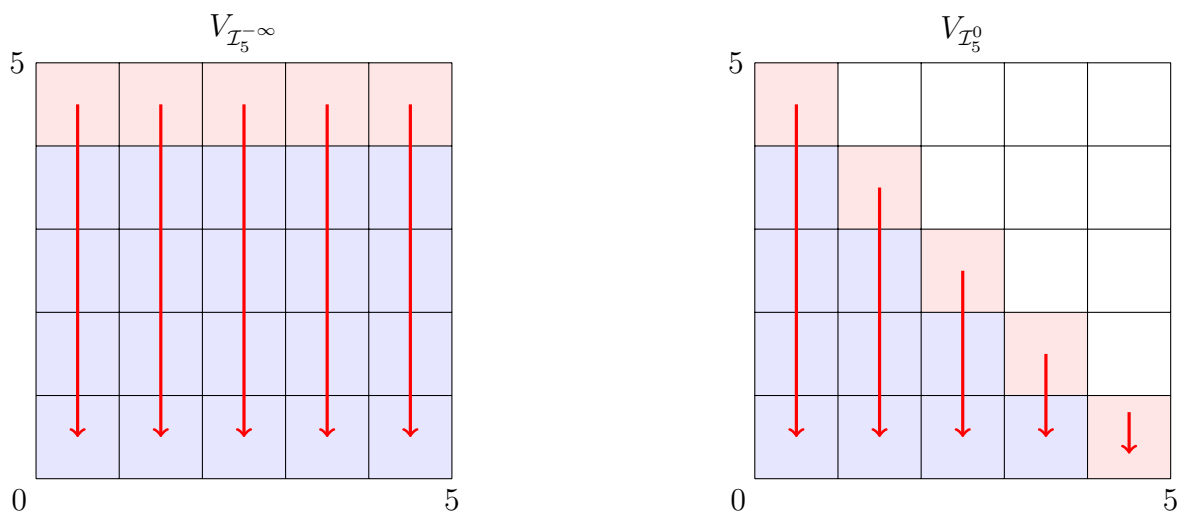


Abbildung 4: Zerlegung von Voll- und Dünngitter in Indexmengen gleicher Länge. Die Indexmenge sind die gefärbten Quadrate, wobei jedes Quadrat einen Unterraum repräsentiert. Die roten Quadrate sind die Unterräume, die keinen Nachbar entlang der zweiten Dimension haben, also durch die Indizes der Menge  $\mathcal{M}_2(\mathcal{I})$  beschrieben werden. Aus diesen werden die Informationen über die Anzahl und Länge der Koeffizientenvektoren entlang einer Achse entnommen. Die als Rechtecke bezeichneten Mengen sind als Pfeile dargestellt.

Mit Hilfe der Indexmenge  $\mathcal{M}_d(\mathcal{I})$  kann der Gesamtaufwand eines Gitters  $\mathbf{I}_{\mathcal{I}}[f](\mathbf{x})$  wie folgt abgeschätzt werden

$$\begin{aligned}
 \mathcal{T}[\mathbf{I}_{\mathcal{I}}[f](\mathbf{x})] &\lesssim \sum_{d=1}^D \sum_{\mathbf{l} \in \mathcal{M}_d(\mathcal{I})} 2^{l_d} l_d 2^{|\mathbf{l}|_1 - l_d} \\
 &= \sum_{d=1}^D \sum_{\mathbf{l} \in \mathcal{M}_d(\mathcal{I})} 2^{|\mathbf{l}|_1} l_d \\
 &\leq \sum_{d=1}^D l_{\max} \sum_{\mathbf{l} \in \mathcal{M}_d(\mathcal{I})} 2^{|\mathbf{l}|_1} \\
 &\leq D l_{\max} \sum_{\mathbf{l} \in \mathcal{I}} 2^{|\mathbf{l}|_1} \tag{2.50}
 \end{aligned}$$

Diese Rechnung ergibt, zusammen mit der Zahl der Freiheitsgrade eines Gitters aus Lemma 2.4, den folgenden Rechenaufwand zur Konstruktion eines Interpolanten auf einem verallgemeinerten dünnen Gitter.

**Lemma 2.5** (Rechenaufwand). *Sei  $L \in \mathbb{N}_0$  und  $T < 1$ . Eine obere Schranke für den Rechenaufwand des Interpolationsoperators  $\mathbf{I}_{\mathcal{I}_L^T}[f](\mathbf{x})$  auf einem verallgemeinerten dünnen Gitter in Abhängigkeit vom Diskretisierungsparameter  $L$  ist gegeben durch*

$$\mathcal{T}[\mathbf{I}_{\mathcal{I}_L^T}[f](\mathbf{x})] \lesssim L \sum_{\mathbf{l} \in \mathcal{I}_L^T} 2^{\|\mathbf{l}\|_1} \lesssim \begin{cases} L2^L & \text{für } 0 < T < 1 \\ L2^L L^{D-1} & \text{für } T = 0 \\ L2^{L \frac{T-1}{T/D-1}} & \text{für } T < 0 \\ L2^{LD} & \text{für } T = -\infty \end{cases} \quad (2.51)$$

### 2.5.3 Approximationseigenschaften

Jetzt werden die Approximationseigenschaften von verallgemeinerten dünnen Gittern betrachtet. Hierbei hilft die Definition der Sobolevräume mit dominierend gemischter Glattheit, da diese über das Abfallverhalten der Fourierkoeffizienten einer Funktion definiert sind. Die bestmögliche lineare Approximation einer Funktion aus einem Sobolevraum auf einem Dünngitterraum wird durch das Abschneiden der Fourierreihe auf diesem Raum abgeschätzt. Da die Fourierkoeffizienten im Allgemeinen nicht exakt bestimmt werden können, kommt bei der Interpolation mittels Fouriertransformationen noch ein Fehler hinzu, der aber Größenordnungsgemäß den Fehler der Bestapproximation nicht übersteigt.

Wir betrachten nun eine periodische Funktion  $f \in \mathcal{L}_2(\mathbb{T}^D)$  und konstruieren den linearen Approximationsoperator  $\mathcal{Q}_{\mathcal{I}} : \mathcal{L}_2(\mathbb{T}^D) \rightarrow V_{\mathcal{I}}$  mit der Vorschrift

$$\mathcal{Q}_{\mathcal{I}}[f] := \sum_{\mathbf{l} \in \mathcal{I}} \sum_{\mathbf{k} \in \mathcal{I}_1} \hat{f}_{\mathbf{k}} \Phi_{\sigma(\mathbf{k})} \quad (2.52)$$

wobei  $\Phi_{\mathbf{k}} = e^{i\mathbf{k} \cdot \mathbf{x}}$  ist. Dies ist die abgeschnittene Fourierreihe der Funktion  $f$ . Die nachfolgenden Ergebnisse sind dem Artikel [GH14] entnommen und werden hier zusammengefasst wiedergegeben. Hier wird die Bestapproximation für allgemeine gewichtete Sobolevräume und Normen ausgerechnet. Es wird auch eine allgemeine monotone Indexmenge angenommen. Die Bestapproximation auf verallgemeinerten dünnen Gittern in unterschiedlichen Normen sind dann Spezialfälle. Die Idee jedes Beweises ist es die Fourierkoeffizienten durch die Gewichtsfunktion des Raumes abzuschätzen und das Maximum der Gewichte aus der Summe herauszuziehen. Wird nun die Bestapproximation auf einem bestimmten Sobolevraum in einer festen Norm gesucht, müssen die Gewichte der Räume abgeschätzt werden.

**Lemma 2.6** (Bestapproximation für monotone Indexmengen). *Seien  $w, \tilde{w}$  zwei Gewichtsfunktionen und  $\mathcal{H}_w, \mathcal{H}_{\tilde{w}}$  die zugehörigen Sobolevräume. Ohne Beschränkung der Allgemeinheit sei  $\mathcal{H}_w \subset \mathcal{H}_{\tilde{w}} \subset \mathcal{L}_2$  und damit  $w(\mathbf{k}) \lesssim \tilde{w}(\mathbf{k})$ . Sei  $f \in \mathcal{H}_w(\mathbb{T}^D) \subset \mathcal{L}_2(\mathbb{T}^D)$  mit*

der eindeutigen Fourierreihenentwicklung  $f = \sum_{\mathbf{k} \in \mathbb{Z}^D} \hat{f}_{\mathbf{k}} \Phi_{\mathbf{k}}$ . Falls  $\max_{\mathbf{l} \in \mathbb{Z}^D \setminus \mathcal{I}, \mathbf{k} \in \mathcal{J}_I} \frac{\tilde{w}(\sigma(\mathbf{k}))^2}{w(\sigma(\mathbf{k}))^2} < \infty$ , so ist die beste Approximation einer Funktion  $f$  im Raum  $V_{\mathcal{I}}$ , welcher durch eine monotone Indexmenge  $\mathcal{I}$  gegeben ist, die folgende

$$\begin{aligned}
 \inf_{\tilde{f} \in V_{\mathcal{I}}} \|f - \tilde{f}\|_{\mathcal{H}_{\tilde{w}}}^2 &\leq \|f - \mathcal{Q}_{\mathcal{I}} f\|_{\mathcal{H}_{\tilde{w}}}^2 = \left\| \sum_{\mathbf{l} \in \mathbb{Z}^D \setminus \mathcal{I}} \sum_{\mathbf{k} \in \mathcal{J}_I} \hat{f}_{\sigma(\mathbf{k})} \Phi_{\sigma(\mathbf{k})} \right\|_{\mathcal{H}_{\tilde{w}}}^2 \\
 &= \sum_{\mathbf{l} \in \mathbb{Z}^D \setminus \mathcal{I}} \sum_{\mathbf{k} \in \mathcal{J}_I} \tilde{w}(\sigma(\mathbf{k}))^2 |\hat{f}_{\sigma(\mathbf{k})}|^2 \\
 &= \sum_{\mathbf{l} \in \mathbb{Z}^D \setminus \mathcal{I}} \sum_{\mathbf{k} \in \mathcal{J}_I} \frac{\tilde{w}(\sigma(\mathbf{k}))^2}{w(\sigma(\mathbf{k}))^2} |\hat{f}_{\sigma(\mathbf{k})}|^2 w(\sigma(\mathbf{k}))^2 \\
 &\leq \left( \max_{\mathbf{l} \in \mathbb{Z}^D \setminus \mathcal{I}, \mathbf{k} \in \mathcal{J}_I} \frac{\tilde{w}(\sigma(\mathbf{k}))^2}{w(\sigma(\mathbf{k}))^2} \right) \sum_{\mathbf{l} \in \mathbb{Z}^D \setminus \mathcal{I}} \sum_{\mathbf{k} \in \mathcal{J}_I} |\hat{f}_{\sigma(\mathbf{k})}|^2 w(\sigma(\mathbf{k}))^2 \\
 &\leq \left( \max_{\mathbf{l} \in \mathbb{Z}^D \setminus \mathcal{I}, \mathbf{k} \in \mathcal{J}_I} \frac{\tilde{w}(\sigma(\mathbf{k}))^2}{w(\sigma(\mathbf{k}))^2} \right) \sum_{\mathbf{l} \in \mathbb{Z}^D} \sum_{\mathbf{k} \in \mathcal{J}_I} |\hat{f}_{\sigma(\mathbf{k})}|^2 w(\sigma(\mathbf{k}))^2 \\
 &= \left( \max_{\mathbf{l} \in \mathbb{Z}^D \setminus \mathcal{I}, \mathbf{k} \in \mathcal{J}_I} \frac{\tilde{w}(\sigma(\mathbf{k}))^2}{w(\sigma(\mathbf{k}))^2} \right) \|f\|_{\mathcal{H}_w}^2 \tag{2.53}
 \end{aligned}$$

Mit diesem Lemma ist es möglich den Fehler der Bestapproximation mit einem verallgemeinerten dünnen Gitter für Funktionen aus einem Sobolevraum mit dominierend gemischter Glattheit in beliebigen Normen zu berechnen.

**Lemma 2.7** (Bestapproximation für verallgemeinerte dünne Gitter). *Sei  $\mathcal{I}_L^T$  eine parametrisierte Indexmenge für einen verallgemeinerten dünnen Gitterraum  $\mathcal{V}_{\mathcal{I}_L^T}$ . Sei  $f \in \mathcal{H}_{\text{mix}}^{t,r}(\mathbb{T}^D)$ . Für  $L \in \mathbb{N}_0$ ,  $T < 1$ ,  $t' + r' < t + r$  und  $t - t' \geq 0$  gilt*

$$\begin{aligned}
 \inf_{\tilde{f} \in \mathcal{V}_{\mathcal{I}_L^T}} \|f - \tilde{f}\|_{\mathcal{H}_{\text{mix}}^{t',r'}} &\leq \|f - \mathcal{Q}_{\mathcal{I}_L^T} f\|_{\mathcal{H}_{\text{mix}}^{t',r'}} \\
 &\lesssim \begin{cases} 2^{L((r'-r)-(t-t')+(T(t-t')-(r-r'))\frac{n-1}{n-T})} \|f\|_{\mathcal{H}_{\text{mix}}^{t,r}} & \text{für } T \geq \frac{r'-r}{t-t'} \\ 2^{L((r'-r)-(t-t'))} \|f\|_{\mathcal{H}_{\text{mix}}^{t,r}} & \text{für } T \leq \frac{r'-r}{t-t'} \end{cases}
 \end{aligned}$$

Nach Lemma 2.6 muss man die Gewichte der beiden Sobolevräume abschätzen. Der Lesbarkeit wegen werden die Gewichte mit  $\lambda_{\text{iso}}(\mathbf{k}) := 1 + \|\mathbf{k}\|_{\infty}$  für den isotropen Anteil des Sobolevraumes und  $\lambda_{\text{mix}}(\mathbf{k}) := \prod_{d=1}^D (1 + |k_d|)$  für den gemischten Anteil abgekürzt.

$$\max_{\mathbf{l} \in \mathbb{Z}^D \setminus \mathcal{I}, \mathbf{k} \in \mathcal{J}_I} \frac{\lambda_{\text{mix}}(\sigma(\mathbf{k}))^{t'} \lambda_{\text{iso}}(\sigma(\mathbf{k}))^{r'}}{\lambda_{\text{mix}}(\sigma(\mathbf{k}))^t \lambda_{\text{iso}}(\sigma(\mathbf{k}))^r} \tag{2.54}$$

Diese Rechnung findet man in [GK09].

Da die Fourierkoeffizienten  $\hat{f}_{\mathbf{k}} = \int_{\mathbb{T}^D} f(\mathbf{x}) \Phi_{\mathbf{k}}^*(\mathbf{x}) d\mathbf{x}$  der Funktion  $f$  im allgemeinen nicht bekannt sind und das Lösen eines Integrals für jeden Koeffizienten viel zu aufwendig wäre,



setzt man diskrete Fouriertransformationen ein, um die Koeffizienten bei der trigonometrischen Interpolation zu bestimmen. Hierbei entsteht ein weiterer Fehler, der Aliasing Fehler heißt. Dieser tritt auf, wenn die Indexmenge der Fourierkoeffizienten nicht endlich ist, oder wenn zu viele Fourierkoeffizienten abgeschnitten werden. Genauer gesagt ist die Aussage, dass die Zahl der Stützstellen entlang einer Achse mindestens doppelt so hoch wie der höchste Index eines Fourierkoeffizienten entlang dieser Achse sein muss, um einen Aliasing Fehler zu vermeiden. Dies ist als Nyquist-Shannon-Abtasttheorem [Sha48] bekannt. Da die Funktionen hier im Allgemeinen eine unendliche Fourier Reihenentwicklung besitzen, tritt ein Fehler auf von welchem erst einmal nicht sicher ist, ob dieser die Interpolationsgenauigkeit zu sehr stört.

Um den Interpolationsfehler für verallgemeinerte dünne Gitter zu bestimmen ist einiges an Vorarbeit nötig, welche hier nicht vollständig ausgeführt werden kann. Daher fassen wir hier die Ergebnisse zusammen. Die Herleitung des Interpolationsfehlers für ein Vollgitter und reguläres Dünngitter findet man in [GH14]. Analoge Lemmata sind in [Kup97, Kup99] zu finden, welche auf den Arbeiten [Pas80, Tem93] basieren. Die Herleitung des Interpolationsfehlers von verallgemeinerten dünnen Gittern wurde erstmals in [GH14] durchgeführt.

**Lemma 2.8** (Interpolationsfehler volles Gitter). *Für  $L \in \mathbb{N}_0$ ,  $f \in \mathcal{H}^s$ ,  $s > \frac{D}{2}$  und  $0 \leq r < s$  gilt*

$$\|f - \mathbf{I}_{\mathcal{I}_L^\infty} f\|_{\mathcal{H}^r} \lesssim 2^{-(s-r)L} \|f\|_{\mathcal{H}^s} \quad (2.55)$$

**Lemma 2.9** (Interpolationsfehler dünnes Gitter). *Für  $L \in \mathbb{N}_0$ ,  $f \in \mathcal{H}_{mix}^t$ ,  $t > \frac{1}{2}$  und  $0 \leq r < t$  gilt*

$$\|f - \mathbf{I}_{\mathcal{I}_L^0} f\|_{\mathcal{H}^r} \lesssim 2^{-(t-r)L} L^{D-1} \|f\|_{\mathcal{H}_{mix}^t} \quad (2.56)$$

Man sieht hier, dass für ein dünnes Gitter der Term  $L^{D-1}$  zum Fehler hinzukommt. Da der Fehler aber mit  $2^{-L}$  abnimmt, ist dies hinnehmbar. Für ein verallgemeinertes dünnes Gitter ist der Interpolationsfehler

**Lemma 2.10** (Interpolationsfehler verallgemeinertes dünnes Gitter). *Sei  $L \in \mathbb{N}_0$ ,  $T < 1$ ,  $r < t$ ,  $t > \frac{1}{2}$  und  $f \in \mathcal{H}_{mix}^t$  mit einer punktweise konvergenten Fourierreihe. Dann gilt*

$$\|f - \mathbf{I}_{\mathcal{I}_L^T} f\|_{\mathcal{H}^r} \lesssim \begin{cases} 2^{-((t-r)+(Tt-r)\frac{D-1}{D-T})L} L^{D-1} \|f\|_{\mathcal{H}_{mix}^t} & \text{für } T \geq \frac{r}{t} \\ 2^{-(t-r)L} \|f\|_{\mathcal{H}_{mix}^t} & \text{für } T < \frac{r}{t} \end{cases} \quad (2.57)$$

## 2.5.4 Konvergenzrate in Abhängigkeit der Gitterpunkte und Kosten

Abschließend wird die Konvergenzrate in Abhängigkeit der Zahl der Freiheitsgrade und des Rechenaufwandes hergeleitet. Diese Ergebnisse sind auch aus der Arbeit [GH14] entnommen. Es müssen die Ergebnisse aus den beiden vorherigen Unterabschnitten miteinander kombiniert werden, um die gewünschten Aussagen zu erhalten.

**Lemma 2.11** (Konvergenzrate in Abhängigkeit der Gitterpunkte). *Sei  $L \in \mathbb{N}_0$ ,  $0 < r < t$ ,  $t > \frac{1}{2}$ ,  $0 < T < \frac{r}{t}$ , und  $f \in \mathcal{H}_{mix}^t$  mit einer punktweise konvergenten Fourierreihe. Dann gilt*

$$\|f - \mathbf{I}_{\mathcal{I}_L^T} f\|_{\mathcal{H}^r} \lesssim M^{-(t-r)} \|f\|_{\mathcal{H}_{mix}^t} \quad (2.58)$$

mit den Kosten  $M := |V_{\mathcal{I}_L^T}|$

**Beweis.** Dies folgt aus den Lemmata 2.4 und 2.10, in dem die Zahl der Freiheitsgrade in die Interpolationsgenauigkeit eingesetzt wird. Die Aussage aus Lemma 2.4 wird umgestellt zu

$$M \leq c_1(D) \cdot 2^L \Rightarrow 2^{-L} \leq c_1(D) \cdot M^{-1} \quad (2.59)$$

Dies wird für  $0 < T < \frac{r}{t}$  in Lemma 2.10 eingesetzt und man erhält

$$\begin{aligned} \|f - \mathbf{I}_{\mathcal{I}_L^T} f\|_{\mathcal{H}^r} &\lesssim c_2(D) \cdot 2^{-(t-r)L} \|f\|_{\mathcal{H}_{mix}^t} \\ &\leq c_2(D) (c_1(D) \cdot M^{-1})^{t-r} \|f\|_{\mathcal{H}_{mix}^r} \\ &= C(D) \cdot M^{-(t-r)} \|f\|_{\mathcal{H}_{mix}^r} \end{aligned} \quad (2.60)$$

mit der Konstanten  $C(D) = c_1(D)^{(t-r)} c_2(D)$ .  $\square$

Die Konvergenzrate in Abhängigkeit des Rechenaufwandes wird analog abgeschätzt. Hierzu wird Lemma 2.5 in Lemma 2.10 eingesetzt.

**Lemma 2.12** (Konvergenzrate in Abhängigkeit des Rechenaufwandes). *Für  $0 < r < t$ ,  $t > \frac{1}{2}$ ,  $0 < T < \frac{r}{t}$ , und  $f \in \mathcal{H}_{mix}^r$  mit einer punktweise konvergenten Fourierreihe gilt*

$$\|f - \mathbf{I}_{\mathcal{I}_L^T} f\|_{\mathcal{H}^r} \lesssim R^{-(t-r)} \log(R)^{t-r} \|f\|_{\mathcal{H}_{mix}^t} \quad (2.61)$$

mit dem Aufwand  $R := \mathcal{T}[\mathcal{I}_{\mathcal{I}_L^T}]$ .

**Beweis.** Durch Umstellen der Aussage aus Lemma 2.5 erhält man analog zum Beweis von 2.11

$$\begin{aligned} \|f - \mathbf{I}_{\mathcal{I}_L^T} f\|_{\mathcal{H}^r} &\lesssim 2^{-(t-r)L} \|f\|_{\mathcal{H}_{mix}^t} \\ &\leq (L \cdot R^{-1})^{t-r} \|f\|_{\mathcal{H}_{mix}^t} \\ &= L^{t-r} R^{-(t-r)} \|f\|_{\mathcal{H}_{mix}^t} \end{aligned} \quad (2.62)$$

Hieraus folgt die Aussage, da der Aufwand mindestens so groß wie die Zahl der Stützstellen ist, also  $2^L \lesssim R$ . Diese Abschätzung ist nach unten scharf, da das Achsenkreuz, also die Indizes, deren Einträge bis auf einen 0 sind, in jedem Fall in dem Gitter enthalten ist. Dies ergibt mindestens  $2^L$  Gitterpunkte.  $\square$

Abschließend wird noch eine Aussage bezüglich der Konvergenzrate in Abhängigkeit der Kosten für reguläre dünne Gitter präsentiert. Diese erlaubt es die Konvergenzrate eines nicht adaptiven dünnen Gitters für eine Funktion aus einem gemischt glatten Sobolevraum zu bestimmen. In Kapitel 6 wird diese theoretische Konvergenzaussage numerisch bestätigt.

**Lemma 2.13** (Konvergenzrate in Abhängigkeit der Gitterpunkte für ein reguläres dünnes Gitter). Für  $L \in \mathbb{N}_0$ ,  $t > \frac{1}{2}$  und  $f \in \mathcal{H}_{\text{mix}}^t$  mit einer punktweise konvergenten Fourierreihe gilt die Abschätzung

$$\|f - \mathbf{I}_{\mathcal{I}_L^0}\|_{\mathcal{L}_2} \lesssim M^{-t} L^{(t+1)(D-1)} \|f\|_{\mathcal{H}_{\text{mix}}^t} \lesssim M^{-t} \log(M)^{(t+1)(D-1)} \|f\|_{\mathcal{H}_{\text{mix}}^t} \quad (2.63)$$

Dies ist wieder eine Konsequenz der Lemmata 2.4 und 2.10.

**Beweis.** Der Fehler wird in der  $\mathcal{L}_2$ -Norm abgeschätzt. Dies bedeutet  $r = 0$  und da es sich um ein reguläres dünnes Gitter handelt ist auch  $T = 0$ . Nach Lemma 2.4 gilt

$$M \lesssim 2^L L^{D-1} \Rightarrow 2^{-L} \lesssim M^{-1} L^{D-1} \quad (2.64)$$

Dies wird in Lemma 2.10 eingesetzt

$$\begin{aligned} \|f - \mathbf{I}_{\mathcal{I}_L^0} f\|_{\mathcal{L}_2} &\lesssim 2^{-tL} L^{D-1} \|f\|_{\mathcal{H}_{\text{mix}}^t} \\ &= (M^{-1} L^{D-1})^t L^{D-1} \|f\|_{\mathcal{H}_{\text{mix}}^t} \\ &= M^{-t} L^{t(D-1)} L^{D-1} \|f\|_{\mathcal{H}_{\text{mix}}^t} \\ &= M^{-t} L^{(t+1)(D-1)} \|f\|_{\mathcal{H}_{\text{mix}}^t} \end{aligned} \quad (2.65)$$

Die Aussage folgt hieraus Analog zum Beweis des Lemmas 2.12, da auch hier die Zahl der Gitterpunkte in jedem Fall größer gleich  $2^L$  sind.  $\square$

Im Kapitel 6 wird eine Funktion aus dem Raum  $\mathcal{H}^{p+\frac{1}{2}-\epsilon}$  interpoliert. Der Fehler wird in der  $\mathcal{L}_2$ -Norm berechnet und die Konvergenzrate in Abhängigkeit der Zahl der Freiheitsgrade bestimmt. Für die untersuchte Funktion gilt  $\|f\|_{\mathcal{H}_{\text{mix}}^t} = 1$ . Aus dem letzten Lemma folgt mit  $t = p + \frac{1}{2} - \epsilon$  Konvergenzrate

$$\begin{aligned} \|f - \mathbf{I}_{\mathcal{I}_L^0}\|_{\mathcal{L}_2} &\lesssim M^{-t} \log(M)^{(t+1)(D-1)} \|f\|_{\mathcal{H}_{\text{mix}}^t} \\ &= M^{p+\frac{1}{2}-\epsilon} \log(M)^{(p+\frac{3}{2}-\epsilon)(D-1)} \end{aligned} \quad (2.66)$$

Für den eindimensionalen Fall ist bei der Interpolation einer Funktion endlicher Differenzierbarkeit mit einem regulären dünnen Gitter eine polynomielle Konvergenzrate von der Ordnung in Höhe ihrer Differenzierbarkeit zu erwarten. Diese nimmt mit steigender Dimension etwas ab. Für die Funktion aus dem Raum  $\mathcal{H}^{p+\frac{1}{2}-\epsilon}$  wird mit einem dünnen Gitter eine Konvergenzrate von  $\mathcal{O}(p + \frac{1}{2})$  erwartet. Bei den numerischen Untersuchungen stellt sich diese Rate auch ein.



# 3 Verallgemeinerte dünne Gitter mit beliebiger Levelstruktur

## 3.1 Motivation

Verallgemeinerte dünne Gitter mit beliebiger Levelstruktur erweitern die möglichen Indextmengen der Basisfunktionen, die zur Konstruktion der Gitter herangezogen werden können. Bisher wurden Gitter untersucht, welche eine dyadische Levelstruktur haben und deren Stützstellen äquidistant vorliegen. Für Funktionen, deren Fourierkoeffizienten polynomiell abfallen, ist diese Struktur optimal [Hal92].

Haben die Koeffizienten jedoch ein schnelleres Abfallverhalten, wie es beispielsweise bei Funktionen, die als Tensorprodukt analytischer Funktionen geschrieben werden können, der Fall ist, so werden Gitter mit allgemeineren Levelstrukturen benötigt.

Bei einer Funktion, die in eine Fourierreihe entwickelbar ist, schneidet man die Reihenentwicklung so ab, dass alle Koeffizienten bis zu einer frei wählbaren Größenordnung enthalten sind. Der Fehler der Interpolation ist von der Größenordnung immer so groß, wie der vom Betrage höchste Fourierkoeffizient einer Basisfunktion, die nicht in der abgeschnittenen Reihenentwicklung enthalten sind. Dieses Prinzip gilt für jede orthonormale Basis von Funktionen.

Wenn nun eine in eine Fourierreihe entwickelbare Funktion mit dem folgenden Abfallverhalten der Koeffizienten vorliegt

$$f(\mathbf{x}) = \sum_{\mathbf{k}} \hat{f}_{\mathbf{k}} e^{i\mathbf{k}\cdot\mathbf{x}}, \text{ mit } |\hat{f}_{\mathbf{k}}| \lesssim e^{-\|\mathbf{k}\|_1} \quad (3.1)$$

so sollte die Fourierreihe so abgeschnitten werden, dass alle Indizes  $\mathbf{k}$  der Basisfunktionen, die der Bedingung  $\|\mathbf{k}\|_1 < l$  genügen, enthalten sind. Bei einem verallgemeinerten dünnen Gitter mit dyadischer Levelstruktur ist dies nicht möglich. Bei diesen können zwar beliebige monotone Indextmengen  $\mathcal{I}$  für die Unterräume  $\mathcal{W}_{\mathcal{I}}$ , die im Gitterraum  $V_{\mathcal{I}}$  enthalten sind, verwendet werden, die Strukturen der Unterräume sind jedoch fest vorgeben.

$$V_{\mathcal{I}} = \bigoplus_{\mathbf{l} \in \mathcal{I}} \mathcal{W}_{\mathbf{l}} \quad (3.2)$$

In dieser Arbeit wird die Verallgemeinerung der Unterraumstrukturen untersucht, implementiert und numerische Experimente durchgeführt. Funktionen mit einem exponentiellen Abfallverhalten der Fourierkoeffizienten treten unter anderem bei der Uncertainty Quantification auf [NTT14]. In dem zitierten Artikel werden partielle Differentialgleichungen mit

Gittern gelöst, die auch eine beliebige Levelstruktur haben. Die Lösungen der Differentialgleichungen sind analytische Funktionen und es wird dementsprechend eine exponentielle Konvergenz erreicht.

## 3.2 Konstruktion des Interpolanten

Das Ziel ist es einen Interpolanten zu konstruieren, der Funktionen mit exponentiellem Abfall der Fourierkoeffizienten möglichst effizient interpoliert. Hierzu wird eine Indexmenge der folgenden Form benötigt

$$\mathcal{I}_L = \{\mathbf{k} \in \mathbb{Z}^D : \|\mathbf{k}\|_1 \leq L\} \quad (3.3)$$

Für Funktionen, deren Koeffizienten schneller als polynomiell Abfallen, ist eine solche Indexmenge auch optimal. Zuerst werden wieder durch ein Level indizierte verschachtelte Menge von Stützstellen konstruiert. In dieser Arbeit wurde eine Stützstellenmenge gewählt, die die Punkte der dyadischen Stützstellenmengen verwendet, jedoch von Level zu Level nur einen Stützpunkt hinzunimmt. Es wurden keinerlei Untersuchungen gemacht, ob und wann diese Auswahl an Stützstellen optimal ist. Hier muss in Zukunft noch Forschungsarbeit geleistet werden.

Es wird der gleiche Interpolationsoperator  $I_{\mathcal{I}_L}$ , der schon für die Konstruktion von verallgemeinerten dünnen Gittern benutzt wurde, verwendet. Allerdings werden die möglichen Strukturen der Unterräume verallgemeinert. Es ist jetzt möglich eine beliebige Anzahl an Stützstellen von Level zu Level hinzuzunehmen. Die verbleibenden Einschränkungen sind durch die Tensorproduktstruktur gegeben und die Verschachtelung der Stützstellenmengen gegeben.

In dieser Arbeit wird das allgemeinste Gitter untersucht, bei dem von Level zu Level nur eine Stützstelle hinzugenommen wird. Diese Struktur bekommt den Namen PLUS1-Levelstruktur und ein Gitter mit dieser Struktur heißt verallgemeinertes dünnes Gitter mit beliebiger monotoner Indexmenge oder kurz PLUS1-Gitter. Der Grund für die Benennung liegt darin, dass die Unterräume wie bei den verallgemeinerten dünnen Gittern ausgewählt und verwaltet werden. Als Menge der Basisfunktionen ist mit der PLUS1-Struktur der Stützstellenmengen aber jede beliebige monotone Menge möglich. Eine Konsequenz aus dieser Levelstruktur ist, dass jeder Unterraum nur eine Basisfunktion enthält. Die Indexmenge der Unterräume und Basisfunktionen sind bei einem PLUS1-Gitter also äquivalent. Der einzige Unterschied besteht hier noch darin, dass die Indexmenge der Basisfunktionen durch das Anwenden der  $\sigma$ -Funktion aus Gleichung (2.27) aus der Indexmenge der Unterräume hervorgeht.

### 3.2.1 Stützstellenwahl

Für diese Arbeit wurde eine Stützstellenmenge konstruiert, die der dyadischen ähnlich sieht. Zur Erinnerung, die dyadischen Stützstellenmengen sind

$$\mathcal{S}_l := \left\{ m \frac{2\pi}{2^l} : m = 0, \dots, 2^l - 1 \right\} \quad (3.4)$$

Diese Stützstellen werden wie folgt durchnummeriert, bis mindestens ein maximaler Punkt  $p_{\max}$  erreicht wurde. Dieser Punkt wird im Algorithmus durch die Anzahl an Stützstellen im höchsten Level vorgegeben.

---

**Algorithmus 1:** Durchnummerierung der dyadischen Stützstellen
 

---

```

 $p_0 := 0$ 
 $p_1 := \pi$ 
 $l := 1$ 
while  $l \leq \lceil \log(p_{\max}) \rceil$  do
  for  $i = 2^{l-1}$  to  $2^l - 1$  do
     $p_{2^l+i} := p_i - \frac{2\pi}{2^l}$ 
     $p_{2^l+2^{l-1}+i} := p_i + \frac{2\pi}{2^l}$ 
  end
   $l = l + 1$ 
end

```

---

In Abbildung 5 sieht man die Durchnummerierung für  $p_{\max} = 16$ . Diese Durchnummerierung hat zur Folge, dass ein Stützpunkt mit geraden Index und der nachfolgende Stützpunkt den Abstand  $\pi$  haben. Nun wird die Stützstellenmenge zur Interpolation mit einem PLUS1-Gitter so aufgebaut, dass jedes Level einen Punkt mehr als das vorherige hat.

$$\mathcal{T}_l := \{p_0, \dots, p_{l-1}\} \quad (3.5)$$

Das resultierende Gitter ist in Abbildung 6 zu sehen. Hier wird ein zweidimensionales Gitter mit Level 16 gezeigt. Die linke Abbildung ist die Indexmenge der Basisfunktionen, welche rautenförmig ist, und das rechte Bild zeigt das Gitter mit seinen Stützstellen.

Um die Interpolationsgenauigkeit dieser Stützstellenwahl zu betrachten, wurden Funktionen unterschiedlicher Regularität eindimensional mit PLUS1- und dyadischer Levelstruktur interpoliert. Dies ist in Abbildung 7 zu sehen. Erwartungsgemäß wird bei gleicher Zahl an Stützstellen die gleiche Genauigkeit erreicht. Der Verlauf der Interpolationsgenauigkeit ist bei eindimensionaler Interpolation treppenförmig. Bei Funktionen mit kleiner Regularität werden deutlich bessere Ergebnisse erst bei der Verdopplung der Stützstellenzahl erreicht. Mit steigender Differenzierbarkeit nimmt dies Verhalten ab und bei einer Funktion mit exponentiellem Abfall der Fourierkoeffizienten erhält man auch bei der Hinzunahme weniger Stützstellen deutlich bessere Ergebnisse.

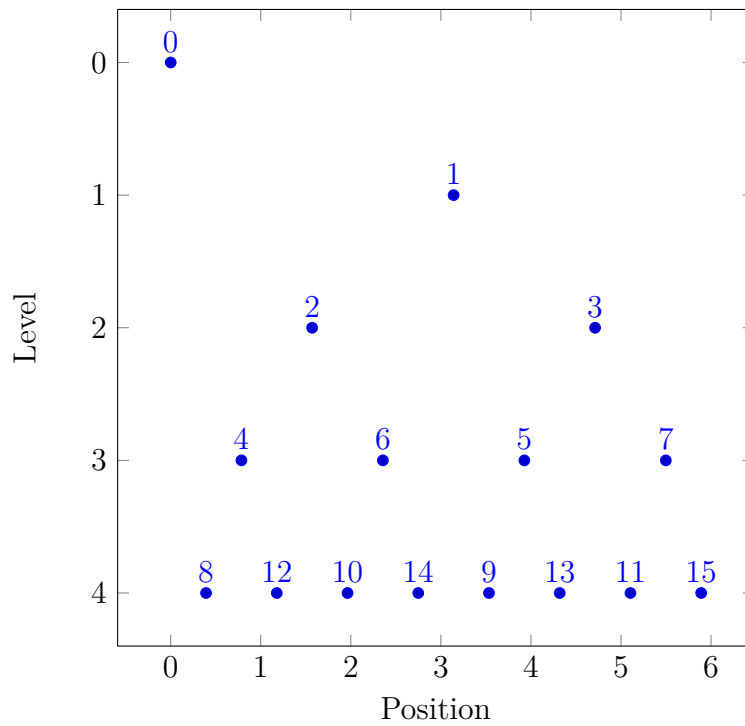


Abbildung 5: Durchnummerierung der ersten 16 Stützstellen zur Verwendung für Gitter mit beliebigen Levelstrukturen

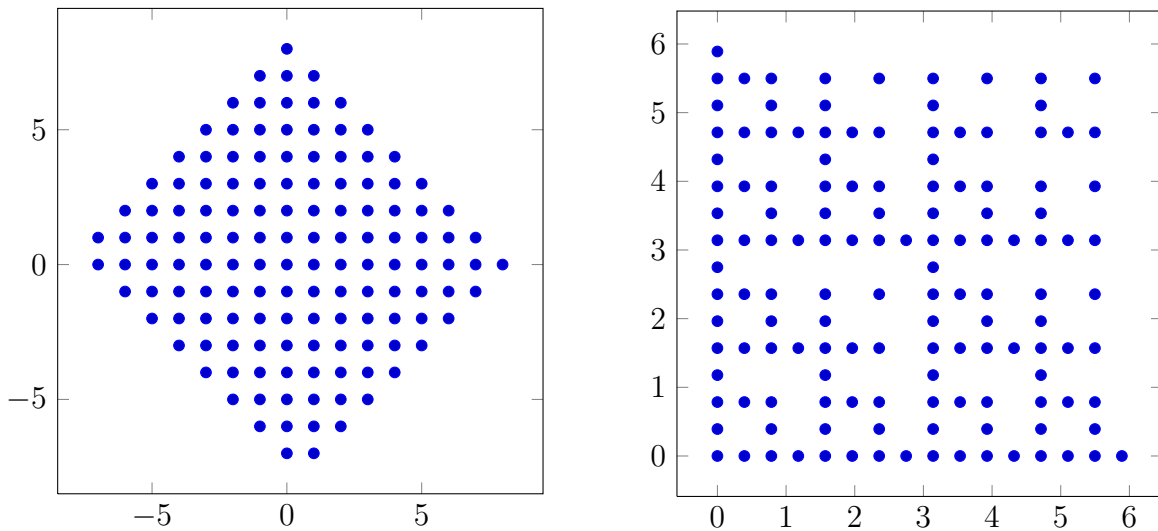


Abbildung 6: PLUS1-Gitter mit Level 16. links: Indexmenge der Basisfunktionen, rechts: Gitterpunkte



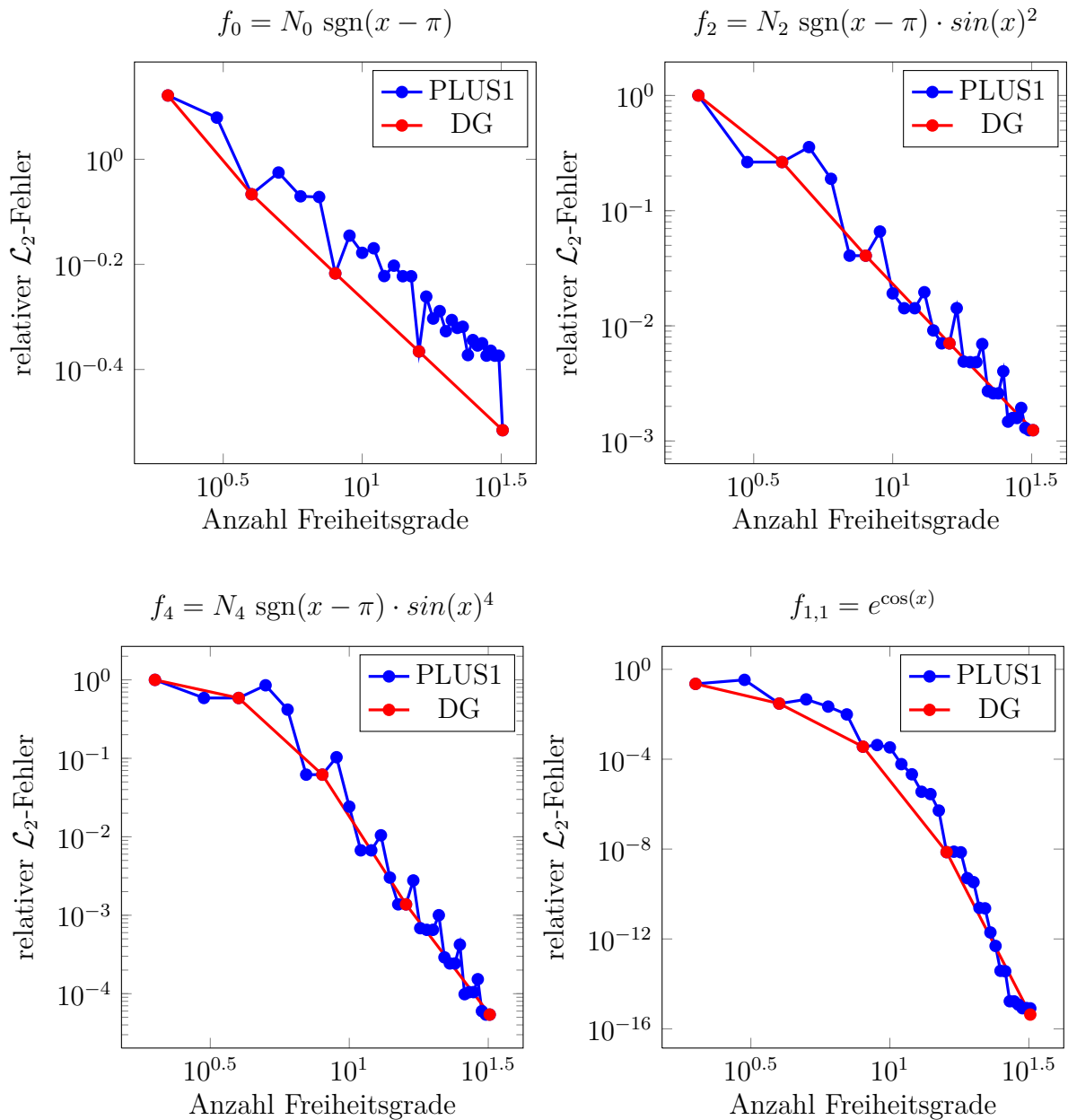


Abbildung 7: Eindimensionale Interpolation von Funktionen unterschiedlicher Differenzierbarkeit. Die Funktionen  $f_p$  sind Funktionen mit endlicher Differenzierbarkeit und sind aus dem Raum  $\mathcal{H}^{p+\frac{1}{2}+\epsilon}$ . Die Funktion  $f_{1,1}$  ist analytisch. PLUS1 ist das Gitter mit einer Stützstelle mehr pro Level. DG ist ein reguläres dünnes Gitter mit dyadischer Levelstruktur. Zu sehen ist der relative  $\mathcal{L}_2$ -Fehler in Abhängigkeit der Anzahl der Freiheitsgrade des Gitters. Je glatter die Funktion ist, desto schneller wächst die Genauigkeit des Gitters und die Ausprägung der Treppenstufen nimmt ab.

### 3.2.2 Eindimensionale Transformationen

Aufgrund der Verschachtelung der Stützstellen liegen diese bei Gittern mit beliebiger Levelstruktur nicht äquidistant vor. Die Zahl der Stützstellen ist auch beliebig. Die bisher verwendeten Algorithmen wie die schnelle und diskrete Fouriertransformation können in diesem Fall nicht verwendet werden um die Fourierkoeffizienten zu berechnen. Es gibt Pakete um eine diskrete Fouriertransformation mit beliebig vielen nicht äquidistanten Stützstellen durchzuführen wie zum Beispiel die NFFT Bibliothek [KKP08]. Auf diese wurde jedoch verzichtet, da sie nicht die Möglichkeit bieten, Berechnungen im Vorfeld durchzuführen und diese abzuspeichern um Zeit einzusparen.

Im Rahmen der Diplomarbeit wurde auf die Vorberechnung von linearen Gleichungssystemen zur Lösung der eindimensionalen Interpolationsgleichungen zurückgegriffen. Der Grund hierfür liegt in der Flexibilität. Die PLUS1-Gitter arbeiten so auch mit den anderen, in der HCFFT Bibliothek implementierten, Basisfunktionen zusammen. Weiterhin kann nahezu jede beliebige Menge von Basisfunktionen und Stützstellen auf diese Weise eingebaut werden. Es stellt sich allerdings die Frage, welche Mengen von Stützstellen und Basisfunktionen zu lösaren Gleichungssystemen führen. Dies wird für die Fourierbasis in diesem Abschnitt beantwortet.

**Definition 3.1** (eindimensionale trigonometrische Interpolation). *Sei  $f : [0, 2\pi] \rightarrow \mathbb{R}$  eine periodische Funktion. Sei  $\mathcal{X}_l = \{x_0, \dots, x_l\}$  mit  $x_i \in [0, 2\pi)$  für  $0 \leq i \leq l$  eine Menge von Stützstellen und  $\mathcal{B}_l = \{\Phi_{\sigma(0)}(x), \dots, \Phi_{\sigma(l)}(x)\}$  eine Menge von Basisfunktionen. Die eindimensionale trigonometrische Interpolation ist die Berechnung eines Interpolanten  $I_l[f](x) := \sum_{k=0}^l c_k \Phi_{\sigma(k)}(x)$ , mit den Basisfunktionen  $\Phi_k(x) = e^{ikx}$  und komplexen Koeffizienten.*

$\sigma : \mathbb{N}_0 \rightarrow \mathbb{Z}$  ist die Indexabbildung aus Gleichung (2.27). Der Interpolant soll den folgenden Bedingungen genügen

$$I_l[f](x_i) = f(x_i) \quad \forall x_i \in \mathcal{X}_l \quad (3.6)$$

Dies führt zu einem komplexen linearen Gleichungssystem mit einer quadratischen Matrix, die sowohl bei der Interpolation, als auch bei der Hierarchisierung und Dehierarchisierung auftaucht. Diese wird ab jetzt mit  $A_{\text{exp}}$  bezeichnet.

$$\begin{pmatrix} \Phi_{\sigma(0)}(x_0) & \Phi_{\sigma(1)}(x_0) & \dots & \Phi_{\sigma(l)}(x_0) \\ \Phi_{\sigma(0)}(x_1) & & & \Phi_{\sigma(l)}(x_1) \\ \vdots & & & \vdots \\ \Phi_{\sigma(0)}(x_l) & \dots & & \Phi_{\sigma(l)}(x_l) \end{pmatrix} \cdot \begin{pmatrix} c_0 \\ \vdots \\ c_l \end{pmatrix} = \begin{pmatrix} f(x_0) \\ \vdots \\ f(x_l) \end{pmatrix} \quad (3.7)$$

Es können aber auch beliebige Intervalle für die Interpolation gewählt werden. Zwei Intervalle sind durch eine bijektive stetige Abbildung  $\tau$  ineinander überführbar und an der Lösbarkeit der in der Interpolation auftretenden Gleichungssysteme ändert sich nichts,

falls die Basisfunktionen durch eine Verkettung mit der Abbildung  $\tau^{-1}$  an das neue Intervall angepasst werden.  $\mathcal{B}_l = \{\Phi_1(\tau^{-1}(x)), \dots, \Phi_l(\tau^{-1}(x))\}$  sind dann die angepassten Basisfunktionen.

Es ist anzumerken, dass sich die Matrix des Gleichungssystems nur ändert, wenn sich die Zahl oder Position der Stützstellen ändern. Für eine gegebene Levelstruktur können die Matrizen im Vorfeld invertiert und abgespeichert werden. Hierdurch ist die eindimensionale Interpolation auch mit quadratischem Aufwand durchführbar.

Es stellt sich heraus, dass das Gleichungssystem 3.7 für paarweise disjunkte Stützstellen immer lösbar ist. Diese Eigenschaft wird Unisolvenz genannt und der Beweis ist ähnlich zu dem Beweis der Unisolvenz der Funktionen  $\{1, e^{ix}, e^{i2x}, \dots, e^{ilx}\}$  [Dav75].

**Lemma 3.1** (Unisolvenz der trigonometrischen Interpolation). *Das lineare Gleichungssystem 3.7 ist für jede beliebige Menge paarweise verschiedener Stützstellen  $\{x_0, \dots, x_l\}$  mit  $x_i \in [0, 2\pi)$  lösbar.*

**Beweis.** *Es ist zu zeigen, dass die Determinante der Matrix*

$$\begin{pmatrix} 1 & e^{ix_0} & e^{-ix_0} & e^{2ix_0} & e^{-2ix_0} & \dots & e^{\sigma(l)ix_0} \\ 1 & e^{ix_1} & e^{-ix_1} & e^{2ix_1} & e^{-2ix_1} & \dots & e^{\sigma(l)ix_1} \\ \vdots & & & & & & \vdots \\ 1 & e^{ix_l} & e^{-ix_l} & e^{2ix_l} & e^{-2ix_l} & \dots & e^{\sigma(l)ix_l} \end{pmatrix} \quad (3.8)$$

nicht verschwindet, wobei  $\sigma(l) = -\frac{l}{2}$  oder  $\sigma(l) = \frac{l+1}{2}$ . Durch Vertauschen der Spalten, welches den Betrag der Determinante nicht verändert, erreicht man dass die Spalten nach dem Index der Exponentialfunktionen sortiert sind.

$$\begin{pmatrix} \dots & e^{-2ix_0} & e^{-ix_0} & 1 & e^{ix_0} & e^{2ix_0} & \dots \\ \dots & e^{-2ix_0} & e^{-ix_0} & 1 & e^{ix_0} & e^{2ix_0} & \dots \\ \vdots & & & & & & \vdots \\ \dots & e^{-2ix_l} & e^{-ix_l} & 1 & e^{ix_l} & e^{2ix_l} & \dots \end{pmatrix} \quad (3.9)$$

wobei die erste Spalte entweder  $e^{-\frac{1}{2}x_j}$  oder  $e^{-\frac{l-1}{2}x_j}$  ist. Jetzt Multipliziert man die  $j$ -te Zeile mit dem komplex konjugierten des ersten Eintrages in der  $j$ -ten Zeile. Da die Exponentialfunktion keine Nullstelle hat, bleibt die Determinante bei diesem Vorgehen ungleich 0. Das Ergebnis ist eine Vandermonde Matrix und sieht so aus

$$\begin{pmatrix} 1 & e^{ix_0} & e^{2ix_0} & \dots & e^{lix_0} \\ 1 & e^{ix_1} & e^{2ix_1} & \dots & e^{lix_1} \\ \vdots & & & & \vdots \\ 1 & e^{ix_l} & e^{2ix_l} & \dots & e^{lix_l} \end{pmatrix} \quad (3.10)$$

Die Determinante dieser Matrix ist  $\prod_{0 \leq k < j \leq l} (e^{ix_k} - e^{ix_j})$  und da die Stützstellen paarweise disjunkt und aus  $[0, 2\pi)$  sind, ist diese ungleich 0. Diese Determinante unterscheidet sich

durch den Faktor  $e^{-i\sigma(j)(x_0+x_1+\dots+x_l)}$ , welcher ungleich 0 ist, von der Determinante der Matrix  $A_{\text{exp}}$ , welche im linearen Gleichungssystem (3.7) auftritt. Hier ist  $j = l$  oder  $j = l - 1$ , je nachdem ob  $l$  gerade oder ungerade ist.  $\square$

Die Kondition einer Vandermonde Matrix wächst mit zunehmender Größe und die Genauigkeit der Interpolation sinkt. Die hier auftretenden Matrizen sind aber klein. Da die Funktionen, die mit den PLUS1-Gittern interpoliert werden, einen sehr raschen Abfall der Fourierkoeffizienten haben, wird die maschinell bedingte Genauigkeit oft mit Basen mit maximal 32 Stützstellen erreicht. Die zugehörigen Gleichungssysteme sind entsprechend auch von der Größe  $32 \times 32$ .

Weiterhin beeinflusst die Wahl der Stützstellen die Determinante und damit auch die Eigenwerte der Matrix. Dies hat wiederum einen Einfluss auf die Kondition der Matrix und die Genauigkeit der Interpolation. Doch diese Überlegungen müssen in zukünftigen Arbeiten betrachtet werden.

### 3.2.3 Hierarchisierung und Dehierarchisierung

Die Verwendung einer beliebigen Levelstruktur ändert an der Aufgabenstellung, eine hierarchische Basis auszurechnen, nichts. Die Konstruktion dieser ist analog zu den Matrizen (2.40) und (2.41) bei der Fouriertransformation auf dünnen Gittern. Im Unterschied zur dyadischen Levelstruktur haben die hier auftretenden Matrizen im Allgemeinen eine dichte Dreiecksgestalt. Die Hierarchisierung und Dehierarchisierung kann bei diesen Matrizen nicht in linearer Zeit implementiert werden. Auch hier werden wieder lineare Gleichungssysteme konstruiert und gelöst. Diese sehen genau so aus, wie das Gleichungssystem 3.7 und sind damit für paarweise disjunkte Stützstellen auch immer lösbar. Die Interpolation auf verallgemeinerten dünnen Gittern mit beliebiger Levelstruktur ist somit für paarweise disjunkte Stützstellen immer durchführbar.

Sei  $L \in \mathbb{N}$  ein maximales Level,  $D \in \mathbb{N}$  die Dimension der Funktion  $f$  und  $\mathbf{g} \in \mathbb{N}^L$  eine Levelstruktur, wobei  $\mathbf{g}_l > \mathbf{g}_{l-1}$  für  $0 \leq l \leq L$  gelte. Seien  $\mathcal{T}_l = \{x_0, \dots, x_{g_l-1}\}$  verschachtelte Mengen von Stützstellen, das heißt  $\mathcal{T}_{l-1} \subset \mathcal{T}_l$ . Seien  $\Phi_j(x)$  für  $j \in \mathbb{Z}$  Basisfunktionen.

Bei der Interpolation mit dünnen Gittern ist es die Aufgabe den Interpolationsoperator für eine monotone Indexmenge  $\mathcal{I} \in \mathbb{N}^D$  für eine Funktion  $f$  zu berechnen.

$$\mathbf{I}_{\mathcal{I}}[f](\mathbf{x}) = \bigoplus_{\mathbf{l} \in \mathcal{I}} \Delta_{\mathbf{l}}[f](x) \quad (3.11)$$

Hierzu werden wieder eindimensionale Interpolationsoperatoren für die Level  $l$  definiert

$$\mathbf{I}_l[f](x) := \sum_{k=0}^{\mathbf{g}_l} \hat{f}_k \Phi_{\sigma(k)} \quad (3.12)$$

welche den folgenden Bedingungen genügen.

$$\mathbf{I}_l[f](x_j) = f(x_j) , \text{ für } 1 \leq j \leq \mathbf{g}_l \quad (3.13)$$

Die Operatoren werden wieder hierarchisiert

$$\begin{aligned}\Delta_l[f](x) &:= \mathbf{I}_l[f](x) - \mathbf{I}_{l-1}[f](x), \text{ für } 1 \leq l \leq L \\ \Delta_0[f](x) &:= \mathbf{I}_0[f](x)\end{aligned}\quad (3.14)$$

Dabei ist jeder Differenzoperator wieder eine Linearkombination von hierarchischen Basisfunktionen, die den folgenden Bedingungen genügen

$$\Delta_l[f](x) = \sum_{j=g_{l-1}}^{g_l-1} \check{f}_j \Psi_j(x) \quad (3.15)$$

$$\Psi_j(x_i) = \sum_{k=0}^{g_l-1} \Phi_k(x_i) := \delta_{ij} \text{ für } j \in \{g_{l-1}, \dots, g_l - 1\} \text{ und } i \in \{0, \dots, g_l - 1\} \quad (3.16)$$

Hierbei wird  $g_{-1} := 0$  gesetzt.

Für jede hierarchische Basisfunktion muss ein lineares Gleichungssystem gelöst werden. Die Größe des Gleichungssystems ist von der Größe des Levels, zu dem die hierarchische Basisfunktion gehört, abhängig. Die Matrix der Gleichungssysteme ist dieselbe, die auch bei der eindimensionalen trigonometrischen Interpolation in Gleichung (3.7) auftritt. Von diesen wurde gezeigt, dass Sie für paarweise disjunkte Stützstellen immer lösbar sind. Und auch hier ändern sich die Gleichungssysteme nur, wenn sich die Levelstruktur oder Position der Stützstellen ändert. Eine hierarchische Basis muss also nur einmal für eine vorgegebene Gitterstruktur berechnet zu werden und kann dann immer wieder verwendet werden. Auf weitere Details zur Konstruktion der Hierarchisierungs- und Dehierarchisierungsmatrix wird in Kapitel 4 eingegangen.

Die Differenzoperatoren  $\Delta_{\mathbf{1}} = \Delta_{l_1} \times \dots \times \Delta_{l_D}$  bilden mittels einer hierarchischen Basis  $\Psi_{\mathbf{k}} = \Psi_{k_1} \cdot \dots \cdot \Psi_{k_D}$ , die Funktion  $f$  in die disjunkten Unterräume  $\mathcal{W}_{\mathcal{I}}$  des Gitters  $V_{\mathcal{I}}$  ab. Sie werden als Tensorprodukt eindimensionaler Operatoren  $\Delta_l = (\mathbf{I}_l - \mathbf{I}_{l-1})[f]$  gebildet. Alle Details zur Konstruktion wurden in Abschnitt 2.4 hergeleitet. Die Notation wurde hier an die beliebige Levelstruktur angepasst. Die hierarchische Basisfunktionen müssen, um die eindimensionalen Operatoren  $\Delta_l$  nachzubilden, auf den Stützstellen aller Räume den Wert 0 annehmen und den Raum  $\text{span}\{B_l \setminus B_{l-1}\}$  aufspannen. Dies führt zu den linearen Gleichungssystemen

Bei den in dieser Arbeit verwendeten PLUS1-Gittern sieht der Vektor  $\mathbf{g}$  so aus

$$g_L = (1, 2, 3, \dots, L + 1) \quad (3.17)$$

Ein verallgemeinertes dünnes Gitter mit beliebiger Levelstruktur ist durch ein maximales Level  $L$ , den Vektor  $\mathbf{g} \in \mathbb{N}^L$ , einem Vektor von Stützstellen  $\mathbf{p} \in \mathbb{R}^L$  und schließlich einer monotonen Indexmenge  $\mathcal{I} \in \mathbb{N}^D$  für die Unterräume vollständig beschrieben.

### 3.3 Zahl der Freiheitsgrade

Ab jetzt wird immer das Gitter mit PLUS1-Levelstruktur verwendet und alle Aussagen beziehen sich auf dieses. Es ist die allgemeinste Form eines Gitters mit Tensorproduktstruktur und verschachtelten Stützstellenmengen und erlaubt beliebige monotone Mengen als Indexmenge für die Basisfunktionen. Von Level zu Level wird lediglich eine Stützstelle hinzugenommen. Es werden die der dyadischen Levelstruktur ähnlichen Stützstellen, die zu Beginn dieses Kapitels beschrieben wurden, verwendet. Bei Verwendung dieser Gitterstruktur ist es dem in dieser Arbeit beschriebenen adaptiven Algorithmus in Kapitel 4 möglich jede Basisfunktion einzeln auszuwählen. Ein adaptiv konstruiertes Gitter mit dieser Struktur ist also optimal bezüglich der Zahl der Freiheitsgrade.

In diesem Abschnitt wird die Zahl der Freiheitsgrade eines PLUS1-Gitters mit Level  $l$  berechnet. Die Indexmenge der Unterräume eines solchen Gitters ist ein Standard Simplex  $\Delta_D$  der Dimension  $D$  mit Kantenlänge  $l$ , also die Indexmenge  $\mathcal{I}_l := \{\mathbf{k} \in \mathbb{N}_0^D : \|\mathbf{k}\|_1 \leq l\}$ .

**Lemma 3.2** (Anzahl Gitterpunkte im PLUS1-Gitter). *Sei  $\mathcal{I}_l = \{\mathbf{k} \in \mathbb{N}_0^D : \|\mathbf{k}\|_1 \leq l\}$  eine Indexmenge für ein  $D$ -Dimensionales PLUS1-Gitter. Es gilt*

$$|\mathcal{I}_l| \lesssim l^D \quad (3.18)$$

**Beweis.** *Die Anzahl der Gitterpunkte entspricht der Anzahl der Möglichkeiten die Zahlen  $0 \leq k \leq l$  als Summe  $D$  nicht negativer ganzer Zahlen zu schreiben. In der Kombinatorik ist dies als schwache Zerlegung bekannt. Die Formel hierfür lautet*

$$\sum_{\|\mathbf{k}\|_1 \leq l} 1 = \sum_{k=0}^l \binom{D+k-1}{D-1} \quad (3.19)$$

*Jeder Binomialkoeffizient kann abgeschätzt werden durch*

$$\binom{D+k-1}{D-1} = \frac{(D+k-1)!}{(D-1)!k!} = \frac{(k+D-1) \cdot \dots \cdot (k+1)}{(D-1)!} \lesssim k^{D-1} \quad (3.20)$$

*Hieraus folgt*

$$\sum_{k=0}^l \binom{D+k-1}{D-1} \lesssim l^{D-1} + (l-1)^{D-1} + \dots + 2^{D-1} + 1 \lesssim l \cdot l^{D-1} = l^D \quad (3.21)$$

□

Die Zahl der Gitterpunkte wächst exponentiell mit dem Level  $l$ . Hier zeigt sich wieder der Fluch der Dimensionen. Allerdings ist die Zahl der Gitterpunkte ohne eine Abschätzung der Genauigkeit des Interpolanten weniger interessant. Diese Gitter wurden für Funktionen konstruiert, deren Konvergenzrate größer als polynomiell und im Falle analytischer Funktionen sogar exponentiell ist. Mit einem solchen Gitter werden nur wenige Gitterpunkte für eine gute Interpolation benötigt und wenn der Zuwachs an Genauigkeit in Abhängigkeit vom Level schneller ist als der Zuwachs der Stützstellen, so ist auch ein Gitter, dessen Freiheitsgrade exponentiell von der Dimension abhängen, hinnehmbar.

### 3.4 Lineare Bestapproximation einer glatten Funktion auf einem PLUS1-Gitter

Es werden Funktionen mit exponentiellem Abfall der Fourierkoeffizienten, wie in Gleichung (3.1), approximiert. Zuerst wird die bestmögliche Approximation an eine solche Funktion auf einem Gitterraum  $V_{\mathcal{I}_l}$  zur Indexmenge  $\mathcal{I}_l = \{\mathbf{k} : \|\mathbf{k}\|_1 \leq l\}$  betrachtet. Hierzu wird die abgeschnittene Fourierreihe der Funktion abgeschätzt. Die Reihenentwicklung von  $f$  in eine Fourierreihe ist

$$f = \sum_{\mathbf{k} \in \mathbb{N}_0^D} \hat{f}_{\sigma(\mathbf{k})} \Phi_{\sigma(\mathbf{k})} \quad (3.22)$$

Der lineare Approximationsoperator  $\mathcal{Q}_{\mathcal{I}_l}$  ist definiert durch

$$\mathcal{Q}_{\mathcal{I}_l} f := \sum_{\mathbf{k} \in \mathcal{I}_l} \hat{f}_{\sigma(\mathbf{k})} \Phi_{\sigma(\mathbf{k})} \quad (3.23)$$

Die Abschätzung des Fehlers wird in der  $\mathcal{L}_2$ -Norm für analytische Funktionen durchgeführt. Bei diesen ist das Abfallverhalten der Fourierkoeffizienten  $|\hat{f}_{\mathbf{k}}| \lesssim e^{-\|\mathbf{k}\|_1}$ .

$$\begin{aligned} \inf_{\tilde{f} \in V_{\mathcal{I}_l}} \|f - \tilde{f}\|_{\mathcal{L}_2}^2 &\leq \|f - \mathcal{Q}_{\mathcal{I}_l} f\|_{\mathcal{L}_2}^2 \\ &= \left\| \sum_{\mathbf{k} \in \mathbb{N}_0^D \setminus \mathcal{I}_l} \hat{f}_{\sigma(\mathbf{k})} \Phi_{\sigma(\mathbf{k})} \right\|_{\mathcal{L}_2}^2 \\ &= \sum_{\mathbf{k} \in \mathbb{N}_0^D \setminus \mathcal{I}_l} |\hat{f}_{\sigma(\mathbf{k})}|^2 \\ &= \sum_{\|\mathbf{k}\|_1 > l} |\hat{f}_{\sigma(\mathbf{k})}|^2 \\ &= \sum_{j > l} \sum_{\|\mathbf{k}\|_1 = j} |\hat{f}_{\sigma(\mathbf{k})}|^2 \\ &\lesssim \sum_{j > l} \sum_{\|\mathbf{k}\|_1 = j} e^{-\|\mathbf{k}\|_1} \end{aligned} \quad (3.24)$$

Hier wird die Parseval Identität, und in der letzten Zeile das Abfallverhalten der Fourierkoeffizienten benutzt. Die Funktion  $\sigma$ , halbiert ungefähr den Wert des Indizes in der 1-Norm, was durch das Quadrat der Beträge der Koeffizienten wieder aufgehoben wird. Jetzt gilt es die Summe  $\sum_{\|\mathbf{k}\|_1 = j} 1$  abzuschätzen. Diese ist gleich der Differenz der Mächtigkeit der Indexmengen  $\mathcal{I}_j$  und  $\mathcal{I}_{j-1}$ . Es gilt also

$$\sum_{\|\mathbf{k}\|_1 = j} 1 = \sum_{\|\mathbf{k}\| \leq j} 1 - \sum_{\|\mathbf{k}\| \leq j-1} 1 \lesssim j^D - (j-1)^D \lesssim j^{D-1} \quad (3.25)$$

Dies wird in die Abschätzung eingesetzt und ergibt

$$\begin{aligned}
 \inf_{\tilde{f} \in V_{\mathcal{I}_l}} \|f - \tilde{f}\|_{\mathcal{L}_2}^2 &\lesssim \sum_{j>l} \sum_{\|\mathbf{k}\|_1=j} e^{-\|\mathbf{k}\|_1} \\
 &= \sum_{j>l} e^{-j} \sum_{\|\mathbf{k}\|_1=j} 1 \\
 &\lesssim \sum_{j>l} e^{-j} j^{D-1} \\
 &\leq \int_l^\infty e^{-j} j^{D-1} dj
 \end{aligned} \tag{3.26}$$

Hier wurde das Integralkriterium für Reihen angewendet. Das Integral wird durch D-1 fache partielle Integration ausgerechnet.

$$\begin{aligned}
 \int_l^\infty j^{D-1} e^{-j} dj &= [j^{D-1}(-e^{-j})]_l^\infty + (D-1) \int_l^\infty j^{D-2} e^{-j} dj \\
 &= l^{D-1} e^{-l} + (D-1) \int_l^\infty j^{D-2} e^{-j} dj \\
 &= (l^{D-1} + (D-1)l^{D-2})e^{-l} + (D-1)(D-2) \int_l^\infty j^{D-3} e^{-j} dj \\
 &\quad \vdots \\
 &= [l^{D-1} + (D-1)l^{D-2} + \dots + (D-1)(D-2) \cdot \dots \cdot 2 l] e^{-l} + \\
 &\quad + (D-1)(D-2) \cdot \dots \cdot 2 \int_l^\infty e^{-j} dj \\
 &= [l^{D-1} + (D-1)l^{D-2} + \dots + (D-1)(D-2) \cdot \dots \cdot 2 l + \\
 &\quad + (D-1)(D-2) \cdot \dots \cdot 2] e^{-l}
 \end{aligned} \tag{3.27}$$

Hieraus ergibt sich das folgende Lemma

**Lemma 3.3** (Bestapproximation auf PLUS1-Gitterraum einer analytischen Funktion).  
 Sei  $f = \sum_{\mathbf{k} \in \mathbb{N}_0^D} \hat{f}_{\sigma(\mathbf{k})} e^{i\sigma(\mathbf{k}) \cdot \mathbf{x}}$  mit  $\hat{f}_{\mathbf{k}} \lesssim e^{-\|\mathbf{k}\|_1}$ ,  $l \in \mathbb{N}$  und  $D \in \mathbb{N}$ . Für ein PLUS1-Gitterraum  $V_{\mathcal{I}_l}$  mit der Indexmenge  $\mathcal{I}_l = \{\mathbf{k} : \|\mathbf{k}\|_1 \leq l\}$  gilt

$$\inf_{\tilde{f} \in V_{\mathcal{I}_l}} \|f - \tilde{f}\|_{\mathcal{L}_2}^2 \leq \|f - \mathcal{Q}_{\mathcal{I}_l} f\|_{\mathcal{L}_2}^2 = \mathcal{O}(e^{-l} l^{D-1}) \tag{3.28}$$



Bei einer Funktion, deren Fourierkoeffizienten nicht exponentiell, sondern nur schneller als algebraisch abfallen, also  $\hat{f}_{\mathbf{k}} < \|\mathbf{k}\|_1^{-n}$  für beliebiges  $n \in \mathbb{N}$ , gelangt man zu der folgenden Abschätzung.

$$\begin{aligned}
 \inf_{\tilde{f} \in V_{\mathcal{I}_l}} \|f - \tilde{f}\|_{\mathcal{L}_2}^2 &\leq \sum_{j>l} \sum_{\|\mathbf{k}\|_1=j} |\hat{f}_{\sigma(\mathbf{k})}|^2 \\
 &< \sum_{j>l} \sum_{\|\mathbf{k}\|_1=j} \|\mathbf{k}\|_1^{-n} \\
 &= \sum_{j>l} j^{-n} \sum_{\|\mathbf{k}\|_1=j} 1 \\
 &\lesssim \sum_{j>l} j^{-n} j^{D-1} \\
 &\lesssim \sum_{j>l} j^{-n} \\
 &\leq \int_l^\infty j^{-n} dj
 \end{aligned} \tag{3.29}$$

Dieses Integral kann man sehr leicht ausrechnen und erhält

$$\int_l^\infty j^{-n} dj = \left[ \frac{1}{1-n} j^{1-n} \right]_l^\infty \lesssim l^{-n} \tag{3.30}$$

Dies ergibt die nachfolgende Aussage, wobei diese nicht so scharf ist, wie das Lemma bei exponentiellem Abfall der Fourierkoeffizienten. Ein solches Abfallverhalten wird bei Funktionen beobachtet, welche ein Tensorprodukt eindimensionaler glatter Funktionen sind.

**Lemma 3.4** (Bestapproximation bei schneller als polynomiell abfallender Funktion auf PLUS1-Gitterraum). Sei  $f = \sum_{\mathbf{k} \in \mathbb{N}_0^D} \hat{f}_{\sigma(\mathbf{k})} e^{i\sigma(\mathbf{k}) \cdot \mathbf{x}}$  mit  $\hat{f}_{\mathbf{k}} < \|\mathbf{k}\|_1^{-n}$  wobei  $n \in \mathbb{N}$  beliebig ist. Sei  $l \in \mathbb{N}$  und  $D \in \mathbb{N}$ .

Für ein PLUS1-Gitterraum  $V_{\mathcal{I}_l}$  mit der Indexmenge  $\mathcal{I}_l = \{\mathbf{l} : \|\mathbf{l}\|_1 \leq l\}$  gilt

$$\inf_{\tilde{f} \in V_{\mathcal{I}_l}} \|f - \tilde{f}\|_{\mathcal{L}_2}^2 \leq \|f - Q_{\mathcal{I}_l} f\|_{\mathcal{L}_2}^2 < l^{-n} \tag{3.31}$$

Eine einfache Übertragung der Beweise zur Interpolationsgenauigkeit verallgemeinerter dünner Gitter auf PLUS1-Gitter ist nicht möglich. Ein zentraler Bestandteil der Beweise ist die Abschätzung des Aliasing Fehlers, der bei der diskreten Fouriertransformation auftritt. Bei PLUS1-Gittern wird keine diskrete Fouriertransformation durchgeführt. Es werden lineare Gleichungssysteme gelöst um die Fourierkoeffizienten zu erhalten und hier fehlt ein Zusammenhang zwischen dem Integral, durch welches die exakten Fourierkoeffizienten definiert sind, und dem Ergebnis, welches bei dem Lösen der Gleichungssysteme entsteht. Die numerischen Ergebnisse legen nahe, dass der Interpolationsfehler im Bereich

der Bestapproximation und des Aliasingfehlers von DFTs der entsprechenden Längen liegen sollte. Aber dies ist nur eine Vermutung und auch hier wird noch weitere Arbeit nötig sein.

Eine kleine Überlegung zum Abschluss. Die Konvergenzrate eines vollen Gitters mit dyadischer Levelstruktur für eine Funktion  $f \in \mathcal{H}^s$  erhält man durch Kombination der Zahl der Freiheitsgrade aus Lemma 2.4 und des Interpolationsfehlers aus Lemma 2.8. Es ergibt sich eine Konvergenzrate von

$$\|f - I_{\mathcal{I}_L^\infty} f\|_{\mathcal{H}^r} \lesssim M^{\frac{r-s}{D}} \|f\|_{\mathcal{H}^s} \quad (3.32)$$

für ein volles Gitter mit dyadischer Levelstruktur. Der Fehler wird in den numerischen Experimenten in der  $\mathcal{L}_2$ -Norm bestimmt und die  $\mathcal{L}_2$ -Norm der zu interpolierenden Funktion ist 1. Dies führt zu

$$\|f - I_{\mathcal{I}_L^\infty} f\|_{\mathcal{L}^2} \lesssim M^{\frac{-s}{D}} \quad (3.33)$$

Für eine Funktion  $f \in \mathcal{H}^{p+\frac{1}{2}+\epsilon}$  wird eine Konvergenzrate von  $\frac{p+\frac{1}{2}}{D}$  erwartet. Diese Konvergenzrate wird auch bei der Interpolation von Funktionen mit polynomielltem Abfall der Fourierkoeffizienten durch PLUS1-Gitter in Kapitel 6 beobachtet.

In [Hal92] wird gezeigt, dass die Indexmenge der Fourierkoeffizienten, welche vom Betrage größer oder gleich einer beliebigen Größenordnung sind, ein hyperbolisches Kreuz bilden. Werden die Koeffizienten in Form eines Quadrates oder Dreiecks ausgeschnitten, so wie es bei den vollen und PLUS1-Gittern gemacht wird, liegen die größten Koeffizienten, die nicht in das Gitter aufgenommen wurden, auf den Koordinatenachsen und haben dort den Index  $(0, \dots, 0, l+1, 0, \dots, 0)$ , wobei  $l$  das Level der Gitter ist. Sowohl bei einem vollen, als auch bei einem PLUS1-Gitter mit Level  $l$  ist dieser Index nicht enthalten. Dies bedeutet, dass beide Gitter von der Größenordnung her den gleichen Interpolationsfehler erreichen sollten. Da ein PLUS1-Gitter nur um den Faktor  $D!$ , welcher eine Konstante ist, weniger Punkte als ein volles Gitter hat, ist die Konvergenzrate bei beiden Gittern annähernd gleich.

Diese Überlegung wird in der nachfolgenden Abbildung 8 verdeutlicht. Die grüne Linie stellt die Fourierkoeffizienten bis zu einer Größenordnung  $c$  dar. Alle Indizes, die vollständig unter dieser Linie sind, haben Fourierkoeffizienten, welche größer als  $c$  sind. Die rote Indexmenge stellt die Unterräume eines PLUS1-Gitters und die rote und blaue Indexmenge die eines vollen Gitters dar. Beide Gitter haben Level 9. Hier ist zu sehen, dass die größten Fourierkoeffizienten, die nicht mehr im Gitter enthalten sind, diejenigen auf den Achsen mit Index  $(10, 0)$  und  $(0, 10)$  sind.

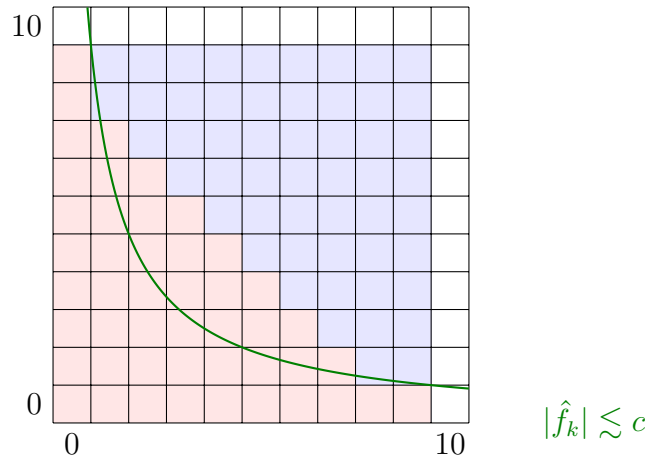


Abbildung 8: Es sind die Indexmengen der Unterräume, sowie das Abfallverhalten der Fourierkoeffizienten zu sehen. Die rote Indexmenge ist die eines PLUS1-Gitters und die rote zusammen mit der blauen stellt die Indexmenge eines vollen Gitters dar. Beide haben Level 9. Die Kurve entspricht der Menge der Fourierkoeffizienten bis zu einer gewissen Größenordnung.

### 3.5 Rechenaufwand

Der Rechenaufwand zur Konstruktion eines PLUS1-Gitters ist auf ähnliche Weise bestimmbar wie der eines dünnen Gitters. Für die Indexmenge der Unterräume eines PLUS1-Gitters  $\mathcal{I}_L = \{\mathbf{k} \in \mathbb{N}_0^D : \|\mathbf{k}\|_1 \leq L\}$  werden wieder die Menge Unterräume gebildet, die keinen Nachfolger haben.

$$\mathcal{M}_d(\mathcal{I}_L) = \{\mathbf{k} \in \mathcal{I}_L : \mathbf{k} + \mathbf{e}_d \notin \mathcal{I}_L\} \quad (3.34)$$

Hier gilt allerdings zu beachten, dass der Rechenaufwand der Transformation, sowie der Hierarchisierung und Dehierarchisierung eines Vektors der Länge  $l$  von der Ordnung  $\mathcal{O}(l^2)$  ist. Ein weiterer Unterschied ist, dass ein Unterraum nur eine Basisfunktion enthält und damit ein Unterraum der Indexmenge  $\mathcal{M}_d(\mathcal{I}_L)$  nur einen Koeffizientenvektor der Länge  $l_d$  indiziert. Dies führt zu einem Rechenaufwand, der ähnlich hergeleitet wird, wie der Aufwand aus Gleichung (2.50) auf Seite 28.

$$\mathcal{T}[\mathcal{I}_{\mathcal{I}_L}] \lesssim \sum_{d=1}^D \sum_{\mathbf{l} \in \mathcal{M}_d(\mathcal{I}_L)} l_d^2 \lesssim L^2 |\mathcal{M}_d(\mathcal{I}_L)| \quad (3.35)$$

Betrachtet man nun die Indexmenge  $\mathcal{I}_L$  eines D-dimensionalen PLUS1-Gitters mit Level  $L$ , so ist die Mächtigkeit der Menge  $\mathcal{M}_d(\mathcal{I}_L)$  genau so groß wie die Zahl der Gitterpunkte eines  $D-1$  dimensionalen PLUS1-Gitters gleichen Levels. Dies liegt daran, dass man statt den Unterräumen ohne Nachfolger auch die Unterräume ohne Vorfahren als Indizierung der eindimensionalen Koeffizientenvektoren entlang einer Richtung in Betracht ziehen

kann. Die Menge der Unterräume ohne Vorfahren wird wie folgt definiert

$$\mathcal{M}'_d(\mathcal{I}_L) := \{\mathbf{k} \in \mathcal{I}_L : \mathbf{k} - \mathbf{e}_d \notin \mathcal{I}_L\} \quad (3.36)$$

Diese Idee wird aus der Abbildung 9 ersichtlich. Es sind die Menge der Unterräume eines PLUS1-Gitters mit Level 4 zu sehen. Die rote Indexmenge ist  $\mathcal{M}_d$  und die grüne ist  $\mathcal{M}'_d$ .

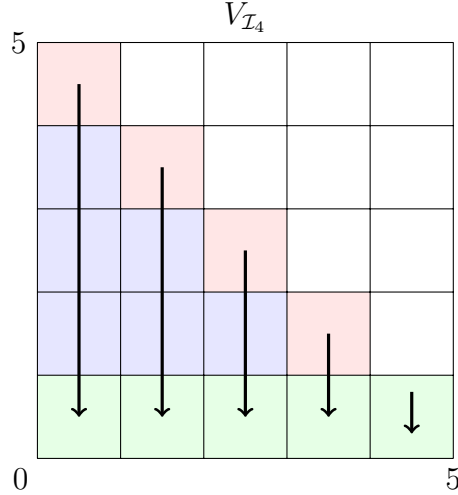


Abbildung 9: Zerlegung eines PLUS1-Gitters in Indexmengen gleicher Länge. Die Indexmenge der Unterräume sind die gefärbten Quadrate. Die roten Quadrate sind die Indizes der Menge  $\mathcal{M}_2(\mathcal{I})$ . Die grüne Indexmenge ist  $\mathcal{M}'_2(\mathcal{I})$  und die Pfeile zeigen die Abbildung der einen Indexmenge auf die andere. Beide haben die gleiche Mächtigkeit und mit Hilfe der grünen Indexmenge kann der Rechenaufwand bestimmt werden.

Für einen Index  $\mathbf{l}$  aus  $\mathcal{M}'_d(\mathcal{I}_L)$  ist auf Grund der Monotonie der Menge  $\mathcal{I}_L$ ,  $l_d = 0$ . In dieser Menge sind also alle Kombinationen von  $D - 1$  nicht negativen ganzen Zahlen enthalten, deren Summe kleiner gleich  $L$  ist. Dies sind genau so viele wie die Zahl der Freiheitsgrade eines  $D - 1$  dimensionalen PLUS1-Gitters des Levels  $L$ , also

$$|\mathcal{M}_d(\mathcal{I}_L)| \lesssim L^{D-1} \quad (3.37)$$

Diese Überlegungen ergeben die folgende Abschätzung für den Rechenaufwand.

**Lemma 3.5** (Rechenaufwand eines PLUS1-Gitters). *Sei  $L \in \mathbb{N}_0$ . Eine obere Schranke für den Rechenaufwand eines Interpolationsoperators  $\mathbf{I}_{\mathcal{I}_L}[f](\mathbf{x})$  auf einem PLUS1-Gitter in Abhängigkeit des Diskretisierungsparameters  $L$  ist gegeben durch*

$$\mathcal{T}[\mathcal{I}_{\mathcal{I}_L}] \lesssim L^2 |\mathcal{M}_d(\mathcal{I}_L)| \lesssim L^{D+1} \quad (3.38)$$

## 4 Methoden

In diesem Kapitel werden die einzelnen Algorithmen vorgestellt, die zur trigonometrischen Interpolation auf verallgemeinerten dünnen Gittern mit beliebiger Levelstruktur benötigt werden. Zu Beginn werden die Algorithmen für dyadische Levelstrukturen gezeigt. Zu diesen zählt die Berechnung der Basiskoeffizienten und der adaptive Algorithmus, welcher die Unterräume der Reihe nach dem Gitter hinzufügt. Als Grundlage für die Implementierung wurde die Hyperbolic Cross Fast Fourier Transform Library verwendet, welche am Institut für Numerische Simulation an der Universität Bonn entwickelt wurde.

Die wesentlichen Neuerungen sind die Hierarchisierung bzw. Dehierarchisierung der Koeffizienten bei beliebiger Levelstruktur, sowie die eindimensionalen Interpolationsregeln für beliebig viele Stützstellen. Diese werden auch Transformationsregeln genannt. Nicht erwähnte kleinere Änderungen betrafen vor allem die Teile der Bibliothek, in denen die Verdopplung der Zahl der Stützstellen von einem Level zum nächst höheren fest im Code implementiert war.

Der Algorithmus zur Berechnung der Fouriertransformation auf dünnen Gittern wurde in [Hal92] entwickelt und um eine effiziente Verwaltung der Unterräume, sowie einen adaptiven Greedy Algorithmus erweitert [GG03]. Außerdem können auch verallgemeinerte dünne Gitter [GH14] für die Fouriertransformationen benutzt werden.

Zu jedem Algorithmus wird eine Laufzeitanalyse und für die hierarchische Basis mit allen Umrechnungsmatrizen zusätzlich eine Untersuchung des Speicherbedarfs durchgeführt.

### 4.1 Fouriertransformationen auf verallgemeinerten dünnen Gittern

In Abschnitt 2.4 sind alle Verfahren hergeleitet, die für eine Fouriertransformation auf verallgemeinerten dünnen Gittern, benötigt werden. Der Interpolationsoperator  $\mathbf{I}_{\mathcal{I}_L^T}[f](\mathbf{x})$  aus Gleichung (2.46) auf Seite 26 muss implementiert werden.

Der erste Schritt bei der Gitterkonstruktion ist das Erstellen der eindimensionalen Interpolationsoperatoren. Sei  $D \in \mathbb{N}$ ,  $L \in \mathbb{N}$  und  $\mathbf{g} \in \mathbb{N}^D$  ein Vektor, welcher die Zahl der Stützpunkte in jedem Level speichert. Weiterhin seien  $\Phi_k(x) := e^{ikx}$  die Fourierbasisfunktionen und  $\sigma : \mathbb{N}_0 \rightarrow \mathbb{Z}$  die aus Gleichung (2.27) bekannte Indexabbildung. Bei einer dyadischen Stützstellenwahl sind die Komponenten des Vektors wie folgt gegeben:  $\mathbf{g}_i = 2^i$  für  $0 \leq i \leq L$ . Die entsprechenden Stützstellenmengen sind

$$\mathcal{S}_l := \left\{ m \frac{2\pi}{2^l} : 0 \leq m < \mathbf{g}_l \right\} \quad (4.1)$$

Die eindimensionalen Interpolationsoperatoren bei dyadischer Levelstruktur sind

$$\begin{aligned} \mathbf{I}_l[f](x) &= \sum_{k=0}^{\mathbf{g}_l-1} \hat{f}_k \Phi_{\sigma(k)} \\ \mathbf{I}_l[f](p) &= f(p), \text{ für alle } p \in \mathcal{S}_l \end{aligned} \quad (4.2)$$

Diese können bei dyadischer Levelstruktur mit der schnellen Fouriertransformation berechnet werden, da die Stützstellenanzahl immer eine Zweierpotenz und die Verteilung dieser äquidistant ist.

Der zweite Schritt ist die Konstruktion des Raumes  $V_{\mathcal{I}_L^T}$ . Es werden alle Unterräume  $\mathcal{W}_l : \mathbf{l} \in \mathcal{I}_L^T$  konstruiert. Die Indexmenge  $\mathcal{I}_L^T$  wurde in Gleichung (2.44) auf Seite 26 definiert. Jeder Unterraum speichert seine vorherigen und nachfolgenden Nachbarräume entlang jeder Dimension. Außerdem speichert jeder Unterraum die Position des ersten Basiskoeffizienten im Koeffizientenvektor. Die Größe des Unterraumes ist durch seinen Index und den Einträgen des Vektors  $\mathbf{g}$  gegeben.

Als nächstes werden die Differenzoperatoren konstruiert. Dies bedeutet, dass die hierarchische Basis konstruiert wird sowie die Hierarchisierungs- und Dehierarchisierungsmatrizen aufgestellt werden. Die Differenzoperatoren sind so definiert

$$\Delta_l[f](x) = \sum_{k=\mathbf{g}_{l-1}}^{\mathbf{g}_l-1} \check{f}_k \Psi_k \quad (4.3)$$

Die hierarchischen Basen sind dabei für das Level  $l$  wie folgt definiert

$$\Psi_j(x_i) = \sum_{k=0}^{\mathbf{g}_l-1} c_k \Phi_{\sigma(k)}(x_i) = \delta_{ij} \quad (4.4)$$

Für eine dyadische Levelstruktur wurden die hierarchischen Basisfunktionen in 2.35 definiert und die Umrechnungsmatrizen waren auch in Abschnitt 2.4 beschrieben. In den beiden nachfolgenden Algorithmen ist die Hierarchisierung und Dehierarchisierung beschrieben. Diese haben einen linearen Aufwand in Abhängigkeit der Länge eines Koeffizientenvektors  $(\hat{f}_k)_{0 \leq k < \mathbf{g}_l}$ .

Mit der hierarchischen Basis können nun die Unterräume konstruiert werden. Zuerst werden noch die Indexmengen der hierarchischen Basisfunktionen eines Unterraumes definiert

$$\begin{aligned} \mathcal{J}_l &:= \begin{cases} \{0, \dots, \mathbf{g}_0 - 1\} & , \text{ für } l = 0 \\ \{\mathbf{g}_{l-1}, \dots, \mathbf{g}_l - 1\} & , \text{ für } l \geq 1 \end{cases} \\ \mathcal{J}_1 &= \mathcal{J}_{l_1} \times \dots \times \mathcal{J}_{l_D} \end{aligned} \quad (4.5)$$

Mit dieser Indexmenge lassen sich die Unterräume explizit durch die hierarchischen Basisfunktionen angeben.

$$\mathcal{W}_l = \text{span}\{\Psi_j : j \in \mathcal{J}_l\} \quad (4.6)$$

---

**Algorithmus 2:** Hierarchisierung eines Koeffizientenvektors

---

**Input :**  $\mathbf{u}$  mit Fourierkoeffizienten,  $n \in \mathbb{N}$  die Länge von  $\mathbf{u}$   
**Output :**  $\mathbf{u}$  mit hierarchischen Koeffizienten  
**for**  $j=1$  **to**  $n$  **do**  
    **for**  $k = 2^{j-1}$  **to**  $2^j - 1$  **do**  
         $\mathbf{u}_{2^j-1-k} = \mathbf{u}_{2^j-1-k} - \mathbf{u}_k$   
    **end**  
**end**  
// Jetzt liegen die hierarchischen Koeffizienten vor  
**return**  $\mathbf{u}$

---



---

**Algorithmus 3:** Dehierarchisierung eines Koeffizientenvektors

---

**Input :**  $\mathbf{u}$  mit hierarchischen Koeffizienten,  $n \in \mathbb{N}$  die Länge von  $\mathbf{u}$   
**Output :**  $\mathbf{u}$  mit Fourierkoeffizienten  
**for**  $j=1$  **to**  $n$  **do**  
    **for**  $k = 2^j - 1$  **downto**  $2^{j-1}$  **do**  
         $\mathbf{u}_{2^j-1-k} = \mathbf{u}_{2^j-1-k} + \mathbf{u}_k$   
    **end**  
**end**  
// Jetzt liegen die Fourierkoeffizienten vor  
**return**  $\mathbf{u}$

---

So lässt sich schließlich der Raum  $V_{\mathcal{I}_L^T}$  wie folgt aufspannen

$$V_{\mathcal{I}_L^T} = \sum_{\mathbf{l} \in \mathcal{I}_L^T} \mathcal{W}_{\mathbf{l}} = \bigoplus_{\mathbf{l} \in \mathcal{I}_L^T} \text{span}\{\Psi_{\mathbf{j}} : \mathbf{j} \in \mathcal{J}_{\mathbf{l}}\} \quad (4.7)$$

Insgesamt ergibt sich der Interpolationsoperator auf einem verallgemeinerten dünnen Gitter in dieser Notation so

$$\begin{aligned} \mathbf{I}_{\mathcal{I}_L^T}[f](\mathbf{x}) &= \sum_{\mathbf{l} \in \mathcal{I}_L^T} \sum_{\mathbf{k} \in \mathcal{J}_{\mathbf{l}}} \check{f}_{\mathbf{k}} \Psi_{\mathbf{k}}(\mathbf{x}) \\ \mathbf{I}_{\mathcal{I}_L^T}[f](\mathbf{p}) &= f(\mathbf{p}), \text{ für alle } \mathbf{p} \in \bigcup_{\mathbf{l} \in \mathcal{I}_L^T} \mathcal{S}_{\mathbf{l}} \end{aligned} \quad (4.8)$$

Sobald der Gitterraum mit den Unterräumen konstruiert ist, können die hierarchischen Koeffizienten berechnet werden. Diese werden durch sukzessive Berechnung der Fourierkoeffizienten mit anschließender Hierarchisierung entlang jeder Dimension bestimmt. Hierzu wird zuerst ein Datenfeld  $u$  welches die Funktionswerte der zu interpolierenden Funktion an den Gitterpunkten enthält. In diesem werden Rechtecke gebildet, welche Vektoren enthalten, die entlang einer Dimension gleich lang sind. Diese Rechtecke können durch die Unterräume indiziert werden, welche entlang der Richtung keinen Nachfolger haben.

Diese sind der jeweils letzte Raum eines Rechteckes. Um eine Operation auf ein Rechteck anzuwenden, müssen dann alle Vorgänger des Unterraumes hinzugenommen werden.

$$\mathcal{M}_d(\mathcal{I}_L^T) := \{\mathbf{l} \in \mathcal{I}_L^T : \mathbf{l} + \mathbf{e}_d \notin \mathcal{I}_L^T\} \quad (4.9)$$

Auf Grund der Tatsache, dass jeder Unterraum seinen Vorgänger entlang jeder Richtung kennt, sind die Rechtecke durch diese Menge vollständig konstruierbar. Ein Rechteck  $\mathcal{R}_d(\mathbf{l})$  für ein Index  $\mathbf{l} \in \mathcal{M}_d(\mathcal{I}_L^T)$  ist dann die Vereinigung der folgenden Unterräume des Gitters

$$\mathcal{R}_d(\mathbf{l}) := \bigcup_{k=0}^{\mathbf{l}_d} \mathcal{W}_{\mathbf{l} - k\mathbf{e}_d} \quad (4.10)$$

---

**Algorithmus 4:** Berechnung der Basiskoeffizienten
 

---

**Input :**  $u_j$  mit Funktionswerten

**Output :**  $u_j$  mit Fourierkoeffizienten

// Tensorproduktstruktur wird ausgenutzt. Eindimensionale Arbeitsschritte werden entlang jeder Dimension ausgeführt.

**for**  $d = 1$  to  $D$  **do**

    // Gehe durch alle Rechtecke entlang der Dimension  $d$

**forall the**  $l \in \mathcal{M}_d(\mathcal{I}_L^T)$  **do**

        // Wende Operationen auf jeden Koeffizientenvektor in dem Rechteck an

**forall the**  $j \in \mathcal{J}_{l_1} \times \dots \times \mathcal{J}_{l_{d-1}} \times \mathcal{J}_0 \times \mathcal{J}_{l_{d+1}} \times \dots \times \mathcal{J}_{l_D}$  **do**

            1D FFT( $u_{j_1, \dots, j_{d-1}, 0, j_{d+1}, \dots, j_D}, \dots, u_{j_1, \dots, j_{d-1}, 2^{l_d-1}, j_{d+1}, \dots, j_D}$ )

            Hierarchisierung( $u_{j_1, \dots, j_{d-1}, 0, j_{d+1}, \dots, j_D}, \dots, u_{j_1, \dots, j_{d-1}, 2^{l_d-1}, j_{d+1}, \dots, j_D}$ )

**end**

**end**

**end**

// Dehierarchisiere jeden Koeffizientenvektor

**for**  $d = 1$  to  $D$  **do**

**forall the**  $l \in \mathcal{M}_d(\mathcal{I}_L^T)$  **do**

**forall the**  $j \in \mathcal{J}_{l_1} \times \dots \times \mathcal{J}_{l_{d-1}} \times \mathcal{J}_0 \times \mathcal{J}_{l_{d+1}} \times \dots \times \mathcal{J}_{l_D}$  **do**

            Dehierarchisierung( $u_{j_1, \dots, j_{d-1}, 0, j_{d+1}, \dots, j_D}, \dots, u_{j_1, \dots, j_{d-1}, 2^{l_d-1}, j_{d+1}, \dots, j_D}$ )

**end**

**end**

**end**

// Jetzt liegen die Fourierkoeffizienten vor

**return**  $\mathbf{u}$

---



## 4.2 Adaptive Gitterkonstruktion

Dünne Gitter mit vorgegebener Indexmengen sind geeignet, wenn das Abfallverhalten der Fourierkoeffizienten bekannt ist. Beispielsweise sind reguläre dünne Gitter optimal für Funktionen mit polynomielltem Abfall der Fourierkoeffizienten. Wenn jedoch im Vorfeld über die zu interpolierende Funktion nicht genug Informationen vorhanden sind, ist es besser einen Algorithmus zu verwenden, der sich die Unterräume so auswählt, dass diejenigen mit den größten Koeffizienten im Gitter enthalten sind. Diese leisten den größten Beitrag zur Genauigkeit des Interpolanten. Ein Algorithmus der sukzessive Unterräume mit den jeweils größten Koeffizienten hinzunimmt ist ein Greedy Algorithmus. In diesem Abschnitt wird der in der HCFFT benutzte Algorithmus vorgestellt. Dieser basiert auf der Arbeit [GG03] in welcher er zur Integration hochdimensionaler Funktionen benutzt wurde. Der Algorithmus wurde angepasst um auch Sobolevräume mit Gewichten endlicher Ordnung adäquat behandeln zu können. Hierzu wurde die Möglichkeit eingebaut eine maximale endliche Ordnung für die Gewichte der im Gitter verwendeten Basisfunktionen vorzugeben.

Der Algorithmus verwaltet zwei Indexmengen. Einmal die Indexmenge  $\mathcal{I}$  der Unterräume, die im Gitter enthalten sind, und eine Menge der aktiven Indizes  $\mathcal{A}$ , von denen jeder einzelne, zusammen mit der Menge  $\mathcal{I}$  wieder eine monotone Indexmenge ergibt. Die Räume mit Indizes aus  $\mathcal{A}$  werden untersucht und deren Gewicht bestimmt. Falls ein Gewicht größer als eine Schwelle  $\epsilon$  ist, wird der Unterraum mit dem größten Gewicht dem Gitter hinzugefügt und sein Index aus  $\mathcal{A}$  entfernt. Anschließend werden aus den Nachbarn des Raumes diejenigen Indizes bestimmt, die wieder, zusammen mit den Indizes aus  $\mathcal{I}$ , eine monotone Menge ergeben. Diese werden dann der aktiven Menge  $\mathcal{A}$  hinzugefügt und die Gewichte neu berechnet.

Vorab wird daran erinnert, dass die Ordnung eines Multiindizes die Zahl der Komponenten ungleich Null ist. Für das Gewicht  $g(\mathcal{W}_1)$  eines Unterraumes stehen sechs Kriterien zu Verfügung. Das einfachste ist die Summe der Beträge aller ihm enthaltenen Fourierkoeffizienten. Es können auch die hierarchischen Koeffizienten zur Berechnung des Gewichtes in Betracht gezogen werden. Weiterhin kann auch der Laplace Operator auf die Unterräume angewandt werden und von diesem die Fourier- oder hierarchischen Koeffizienten als Gewichte verwendet werden. In der nachfolgenden Tabelle 4.1 sind die möglichen Gewichte aufgelistet.

Nachfolgend ist der Algorithmus zur adaptiven Interpolation auf einem verallgemeinerten dünnen Gitter aufgeschrieben. Die Schwäche dieses Algorithmuses liegt darin, dass jedesmal wenn ein Unterraum zum Gitter hinzugefügt wurde, die Gewichte neu berechnet werden. Dies führt später bei den Gittern mit beliebiger Levelstruktur dazu, dass die Konstruktionszeiten für größere Gitter stark ansteigen. Jeder Unterraum eines PLUS1-Gitters hat nur eine Basisfunktion und dementsprechend viele Unterräume werden zur Verwaltung des Gitters konstruiert.

$$\begin{aligned} g_1(\mathcal{W}_1) &:= \sum_{\mathbf{k} \in \mathcal{J}_1} \Delta \check{f}_{\mathbf{k}} + \check{f}_{\mathbf{k}} & g_4(\mathcal{W}_1) &:= \sum_{\mathbf{k} \in \mathcal{J}_1} \Delta \hat{f}_{\mathbf{k}} + \hat{f}_{\mathbf{k}} \\ g_2(\mathcal{W}_1) &:= \sum_{\mathbf{k} \in \mathcal{J}_1} \check{f}_{\mathbf{k}} & g_5(\mathcal{W}_1) &:= \sum_{\mathbf{k} \in \mathcal{J}_1} \hat{f}_{\mathbf{k}} \\ g_3(\mathcal{W}_1) &:= \sum_{\mathbf{k} \in \mathcal{J}_1} \Delta \check{f}_{\mathbf{k}} & g_6(\mathcal{W}_1) &:= \sum_{\mathbf{k} \in \mathcal{J}_1} \Delta \hat{f}_{\mathbf{k}} \end{aligned}$$

Tabelle 4.1: Mögliche Gewichte der Unterräume  $\mathcal{W}_1$  zur Verwendung im adaptiven Algorithmus

Bei Gittern mit dyadischer Levelstruktur ist dies kein Problem, da die Zahl der Unterräume logarithmisch mit der Zahl der Freiheitsgrade wächst. Hierfür wurde der Algorithmus entwickelt. Dies muss zukünftig verbessert werden. Erste Ideen sind im Ausblick dieser Arbeit aufgeführt.

Hohe Dimensionen sind für beide Levelstrukturen ein Problem, da die Zahl der Unterräume bis zu einem bestimmten Level exponentiell mit der Zahl der Dimensionen steigt. Bei vielen Dimensionen steigt auch dementsprechend die Zahl der Unterräume bei dyadischer Levelstruktur stark an.

---

**Algorithmus 5:** Adaptive Gitterkonstruktion

---

**Input :**  $f \in \mathcal{H}_{\text{mix}}^{t,r}$ ,  $L_{\text{init}} \in \mathbb{N}_0$ ,  $L_{\text{max}} \in \mathbb{N}$ ,  $O_{\text{max}} \in \mathbb{N}_0$ ,  $\epsilon \in \mathbb{R}_+$ **Output :**  $\mathcal{I} \in \mathbb{N}_0^D$ // Erstelle Gitter mit Level  $L_{\text{init}}$  und maximalem Gewicht endlicher  
Ordnung  $O_{\text{max}}$  $\mathcal{I} := \{\mathbf{l} \in \mathbb{N}_0^D : \|\mathbf{l}\|_1 \leq L_{\text{init}} \wedge \text{Ordnung}(\mathbf{l}) \leq O_{\text{max}}\}$  $\mathcal{A} := \{\mathbf{l} \in \mathbb{N}_0^D : \mathbf{l} \notin \mathcal{I} \wedge O(\mathbf{l}) \leq O_{\text{max}} \wedge \forall 1 \leq d \leq D : \mathbf{l} - \mathbf{e}_d \in \mathcal{I}, \text{ falls } \mathbf{l}_d > 0\}$ Berechne Gewichte( $\mathcal{A}$ )// Füge so lange Unterräume hinzu, bis in der aktiven Menge keiner  
mehr mit einem passenden Gewicht existiert.**while**  $\exists \mathbf{l} \in \mathcal{A}$  mit  $g(\mathcal{W}_{\mathbf{l}}) > \epsilon$  **do**     $\mathcal{I} = \mathcal{I} \cup \mathbf{l}$      $\mathcal{A} = \mathcal{A} \setminus \mathbf{l}$     // Teste Nachbarn des hinzugefügten Blocks auf Level, Ordnung und  
    Monotonie der potentiellen neuen Menge.    **for**  $d = 1$  to  $D$  **do**        **if**  $\mathbf{l}_d + 1 \leq L_{\text{max}}$  **then**            **if**  $O(\mathbf{l} + \mathbf{e}_d) \leq O_{\text{max}}$  **then**                **if**  $(\mathbf{l} + \mathbf{e}_d) - \mathbf{e}_q \in \mathcal{I} \forall q = 1, \dots, D$  **then**                     $\mathcal{A} = \mathcal{A} \cup \mathbf{l}$                 **end**            **end**        **end**    **end**    Berechne Gewichte( $\mathcal{A}$ )**end**return  $\mathcal{I}$ 

---

## 4.3 Trigonometrische Interpolation auf verallgemeinerten dünnen Gittern mit beliebiger Levelstruktur

Um trigonometrische Interpolationen auf verallgemeinerten dünnen Gittern mit beliebiger Levelstruktur durchführen zu können werden neue Algorithmen für die Interpolationsregeln, Hierarchisierung und Dehierarchisierung benötigt. Die Gitterkonstruktion, sowie das Verfahren zum Berechnen der Koeffizienten aus Algorithmus 4 und der adaptive Algorithmus 5 können ohne Veränderungen weiter verwendet werden. Es müssen lediglich die FFT gegen allgemeinere Interpolationsregeln und die Hierarchisierung und Dehierarchisierung in dem Algorithmus ausgetauscht werden. Außerdem muss der Vektor  $\mathbf{g}$ , welcher die Anzahl der Stützstellen in jedem Level speichert, verändert werden. Dieser ist, mit den in dieser Arbeit konstruierten Gittern, nun frei wählbar. Die Datenstrukturen zur Verwaltung des Gitters wurden nicht verändert. Dies hat den Nachteil, dass die Zahl der Unterräume bei PLUS1-Gittern sehr stark gestiegen ist.

### 4.3.1 Hierarchisierung und Dehierarchisierung

Die Hierarchisierung- und Dehierarchisierung entsprechen jeweils einem Basiswechsel von einer beliebigen Basis in die zugehörige hierarchische Basis bzw. von der hierarchischen Basis zurück in die Ausgangsbasis. Dies wird mittels einer Matrix-Vektor Multiplikation realisiert. Die hierarchischen Basisfunktionen sind durch das lineare Gleichungssystem (4.4) gegeben, welches im Vorfeld für jede Basisfunktion gelöst werden muss.

Es werden Koeffizienten erhalten, welche die hierarchische Basisfunktion als Linearkombination der Ausgangsbasis angeben. Diese Umrechnungskoeffizienten werden anschließend in eine Matrix geschrieben. Dies ist die Dehierarchisierungsmatrix. Die Hierarchisierungsmatrix wird durch Invertierung konstruiert. Dies ist immer möglich, da beide Basen die gleiche Mächtigkeit haben. So entspricht die Umrechnung von einer Basis in eine andere einer Matrix-Vektor Multiplikation.

Als Verfahren zum Lösen der linearen Gleichungssysteme kann beispielsweise die QR Zerlegung verwendet werden. Zur Invertierung von Matrizen gibt es unter anderem die LU-Zerlegung. Für diese Arbeit wurden die LU- bzw. QR Routinen der GNU Scientific Library [Gou09] verwendet. Nachfolgend wird der Algorithmus zum Erhalt der Matrizen beschrieben sowie die Laufzeit untersucht.

#### Algorithmus

Der Algorithmus zur Erzeugung der Dehierarchisierungs- und Hierarchisierungsmatrizen wird hier im Detail beschrieben. Dieser ist auf Seite 61 zu sehen. Sei  $L \in \mathbb{N}$  ein maximales Level und  $\mathbf{g}$  ein Vektor, welcher die Levelstruktur enthält. Dies bedeutet, dass in ihm die Zahl der Stützstellen in jedem Level gespeichert sind. Sei weiterhin  $\{p_0, \dots, p_{\mathbf{g}_L-1}\}$  die aus Abschnitt 1 beschriebenen durchnummerierten dyadischen Stützstellen.

Die Dehierarchisierung entspricht einem Basiswechsel von der hierarchischen Basis

$$\{\Psi_0(x), \dots, \Psi_{\mathbf{g}_L-1}(x)\} \quad (4.11)$$

in die Ausgangsbasis. Diese ist im Falle der trigonometrischen Interpolation die Fourierbasis

$$\{\Phi_{\sigma(0)}(x), \dots, \Phi_{\sigma(\mathbf{g}_L-1)}(x)\} \quad (4.12)$$

Für jedes Level werden nun die hierarchischen Basisfunktionen berechnet. Diese sollen auf allen Stützstellen, in jedem Level, welches kleiner dem aktuellen Level ist, die Eigenschaft haben, den Wert 0 anzunehmen. In dem aktuellen Level hingegen sollen Sie an der Stützstelle, welche den gleichen Index wie die hierarchische Basisfunktion hat, den Wert 1 annehmen und an den restlichen Stützstellen wieder den Wert 0. Es wird also eine Lagrange Basis für die Stützstellen jedes Levels konstruiert.

Die Konstruktion wird mittels einer Linearkombination der Basisfunktionen  $\Phi_{\sigma(k)}$  durchgeführt.

$$\Psi_j(x) = \sum_{k=0}^{\mathbf{g}_l-1} c_{kj} \Phi_{\sigma(k)}(x), \text{ so dass } \Psi_j(p_i) = \delta_{ij} \quad (4.13)$$

Somit ergibt sich für jedes Level  $l$  eine Matrix  $A_{\text{exp}}$  mit mehreren rechten Seiten  $\mathbf{e}_j, j \in \{\mathbf{g}_{l-1}, \dots, \mathbf{g}_l - 1\}$ . Die Matrix  $A_{\text{exp}}$  sieht folgendermaßen aus und entspricht der Matrix  $A_{\text{exp}}$  aus Gleichung (3.7) für welche die Unisolvenz für paarweise disjunkt Stützstellen bewiesen wurde.

$$A_{\text{exp}} = \begin{pmatrix} \Phi_{\sigma(0)}(p_0) & \Phi_{\sigma(1)}(p_0) & \dots & \Phi_{\sigma(\mathbf{g}_L-1)}(p_0) \\ \Phi_{\sigma(0)}(p_1) & & & \Phi_{\sigma(\mathbf{g}_L-1)}(p_1) \\ \vdots & & & \vdots \\ \Phi_{\sigma(0)}(p_{\mathbf{g}_L-1}) & \dots & \dots & \Phi_{\sigma(\mathbf{g}_L-1)}(p_{\mathbf{g}_L-1}) \end{pmatrix} \quad (4.14)$$

Nun wird für jede hierarchische Basisfunktion  $\Psi_j$  das Gleichungssystem gelöst, welches durch  $A_{\text{exp}}$  und  $\mathbf{e}_j$  gegeben ist. Dies ergibt den Koeffizientenvektor  $\mathbf{c}_j$ , der die Koeffizienten der  $j$ -ten hierarchischen Basis enthält. Der Koeffizientenvektor ist gleichzeitig die  $j$ -te Spalte der Dehierarchisierungsmatrix  $A_{\text{dehier}}$ , da dieser der Umrechnung von einer hierarchischen Basisfunktion in die Ausgangsbasis entspricht.

$\mathbf{I}_L[f]_{\text{hierar}}$  ist eine Interpolation einer Funktion  $f$  mit hierarchischen Basen, dann ist der  $\mathbf{I}_L[f]$  der Interpolant bezüglich der Ausgangsbasis und beide nehmen die gleichen Funktionswerte an. Diese sind auf folgende Weise ineinander umrechenbar

$$\begin{aligned}
 \mathbf{I}_L[f](x) &= \sum_{j=0}^{\mathbf{g}_L-1} \check{f}_j \Psi_j(x) \\
 &\stackrel{(4.13)}{=} \sum_{j=0}^{\mathbf{g}_L-1} \check{f}_j \sum_{k=0}^{\mathbf{g}_L-1} c_{kj} \Phi_{\sigma(k)}(x) \\
 &= \sum_{k=0}^{\mathbf{g}_L-1} \hat{f}_{\sigma(k)} \Phi_{\sigma(k)}(x) = \mathbf{I}_L[f](x) \\
 \text{mit } \hat{f}_{\sigma(k)} &= \sum_{j=0}^{\mathbf{g}_L-1} \check{f}_j \cdot c_{kj} \tag{4.15}
 \end{aligned}$$

Hier ist der Zusammenhang zwischen Dehierarchisierung und der Multiplikation der Dehierarchisierungsmatrix mit dem hierarchischem Koeffizientenvektor ersichtlich. Die Dehierarchisierungsmatrix sieht also wie folgt aus

$$A_{dehier} = \begin{pmatrix} c_{0,0} & c_{0,1} & \cdots & c_{0,\mathbf{g}_L-1} \\ c_{1,0} & \ddots & & c_{1,\mathbf{g}_L-1} \\ \vdots & & & \vdots \\ c_{\mathbf{g}_L-1,0} & \cdots & & c_{\mathbf{g}_L-1,\mathbf{g}_L-1} \end{pmatrix} \tag{4.16}$$

Mit dieser Matrix können auch alle Level kleiner als  $L$  dehierarchisiert werden. Wird ein Koeffizientenvektor  $(\check{f}_0, \dots, \check{f}_{\mathbf{g}_l-1})$  aus einem Level  $l < L$  dehierarchisiert, so wird nur die entsprechende obere linke  $\mathbf{g}_l - 1 \times \mathbf{g}_l - 1$  Matrix zur Multiplikation verwendet. Nachfolgend ist der Algorithmus zur Konstruktion der beiden Matrizen gezeigt. Diese haben, wie im Algorithmus zu sehen ist, eine obere rechte Dreiecksstruktur, welche durch die Levelstruktur  $\mathbf{g}$  gegeben ist.

---

**Algorithmus 6:** Berechnung De- und Hierarchisierungsmatrix

---

```

Input :  $L \in \mathbb{N}, \mathbf{g} \in \mathbb{N}^D, \mathbf{p} \in \mathbb{T}^D, \Phi_j(x), \sigma(j)$ 
Output :  $A_{dehier}, A_{hierar} \in \mathbb{R}^{\mathbf{g}_L-1 \times \mathbf{g}_L-1}$ 
// Für jedes Level ein LGS mit mehreren rechten Seiten aufstellen
for  $l = 0$  to  $L$  do
  for  $i, j = 0$  to  $\mathbf{g}_l - 1$  do
    |  $A_{\text{exp}}[i, j] = \Phi_{\sigma(j)}(p_i)$ 
  end
  // Zerlege die Matrix, so dass die Lösungen nachher schnell
  // berechnet werden können.
  LUdecomp( $A_{\text{exp}}$ )
  for  $i = \mathbf{g}_{l-1}$  to  $\mathbf{g}_l - 1$  do
    //  $\mathbf{e}_i$  ist Einheitsvektor der Länge  $\mathbf{g}_l - 1$ , c enthält die Lösung
    LUSolve( $A_{\text{exp}}, \mathbf{c}, \mathbf{e}_i$ )
    // Kopiere den Lösungsvektor in die i-te Spalte der
    // Dehierarchisierungsmatrix
    for  $j = 0$  to  $\mathbf{g}_l - 1$  do
      |  $A_{dehier}[i, j] = \mathbf{c}_j$ 
    end
  end
end
// Invertiere Dehierarchisierungsmatrix und erhalte
// Hierarchisierungsmatrix
 $A = A_{dehier}$ 
LUdecomp( $A$ )
LUinvert( $A, A_{hierar}$ )
return  $A_{dehier}, A_{hierar}$ 

```

---

## Laufzeit

Da die Hierarchisierung und Dehierarchisierung einer Matrix-Vektor Multiplikation entspricht, hängt die Laufzeit quadratisch von der Länge des Vektors ab. Im Unterschied zum regulären dünnen Gitter ist die Matrix bei beliebiger Levelstruktur im Allgemeinen voll besetzt. Die Laufzeit der Hierarchisierung und Dehierarchisierung bei regulären dünnen Gittern hängt linear von der Länge der Vektoren ab [Hal92]. Da beliebige dünne Gitter für glatte Funktionen eingesetzt werden, bei denen die Fourierkoeffizienten sehr schnell abfallen, ist die Zahl der Stützstellen und Basisfunktionen klein und der Unterschied der Laufzeiten fällt nicht zu sehr ins Gewicht.

Die Laufzeit zur Berechnung der Inversen einer Matrix ist  $\mathcal{O}(n^3)$  mit der LU Zerlegung. Diese wird nur einmal berechnet, während zur Bestimmung der Dehierarchisierungsmatrix viele Gleichungssysteme gelöst werden. Nachfolgend wird der Rechenaufwand zur Konstruktion einer hierarchischen Basis bei einer Levelstruktur der Art  $\mathbf{g}_l = k(l + 1)$  bestimmt. Es zeigt sich, dass dieser Aufwand nur von der Zahl der Level und der maximalen Anzahl der Stützstellen abhängt.

**Lemma 4.1** (Aufwand zur Berechnung der Hierarchisierung und Dehierarchisierung). *Sei  $L \in \mathbb{N}$  das maximale Level der hierarchischen Basis und  $\mathbf{g}_l = k(l + 1)$  für  $0 \leq l \leq L$  die Levelstruktur. Der Rechenaufwand zur Berechnung der Hierarchisierungs- und Dehierarchisierungsmatrix ist*

$$\mathcal{O}(L \cdot \mathbf{g}_L^3) \quad (4.17)$$

**Beweis.** *In der nachfolgenden Tabelle 4.2 ist der Rechenaufwand, welcher in einem Level anfällt, aufgeführt.*

Operation	Aufwand	Anzahl pro Level	Aufwand pro Level
Füllen der Matrix $A_{\text{exp}}$	$\mathcal{O}((kl)^2)$	1	$\mathcal{O}((kl)^2)$
LUdecomp( $A_{\text{exp}}$ )	$\mathcal{O}((kl)^3)$	1	$\mathcal{O}((kl)^3)$
LUSolve( $A_{\text{exp}}, \mathbf{c}, \mathbf{e}_i$ )	$\mathcal{O}((kl)^2)$	$k$	$\mathcal{O}(k^3 l^2)$
Kopieren in $A_{\text{dehier}}$	$\mathcal{O}(kl)$	$k$	$\mathcal{O}(k^2 l)$

Tabelle 4.2: Rechenaufwand der einzelnen Operationen in jedem Level zur Konstruktion von  $A_{\text{dehier}}$  und  $A_{\text{hierar}}$

*Die Speicheroperationen sind hier nur der Vollständigkeit aufgeführt und können für das Laufzeitverhalten vernachlässigt werden. Insgesamt ist es erforderlich  $L$  Level zu berechnen, wobei  $l$  jedesmal um 1 erhöht wird. Dies ist eine arithmetische Folge über  $l$ . Mit Hilfe der bekannten Formeln zur Berechnung von arithmetischen Folgen ergibt sich der folgende Rechenaufwand in Tabelle 4.3*



Operation	Aufwand
Füllen der Matrix A	$\mathcal{O}(k^2 L^3)$
LUDecomp(A)	$\mathcal{O}(k^3 L^4)$
LUSolve(A,x,e <sub>i</sub> )	$\mathcal{O}(k^3 L^3)$
Kopieren in A <sub>dehier</sub>	$\mathcal{O}(k^2 L^2)$

Tabelle 4.3: Gesamter Rechenaufwand der einzelnen Operationen zur Konstruktion von A<sub>dehier</sub> und A<sub>hierar</sub>

Die dominierenden Terme sind der Rechenaufwand der LU Zerlegung und der des Vorwärts- bzw. Rückwärtseinsetzen zum Erhalt der Lösung. Dies ergibt einen Gesamtaufwand von  $\mathcal{O}(k^3 L^4 + k^3 L^3) = \mathcal{O}(L \cdot \mathbf{g}_L^3)$  für die Berechnung der Hierarchisierungs und Dehierarchisierungsmatrizen.  $\square$

Die berechneten Matrizen hängen von der Levelstruktur und Stützstellenauswahl ab und werden im Vorfeld berechnet und abgespeichert. Solange sich die Strukturen nicht ändern, können die Matrizen wiederverwendet werden. Der Speicheraufwand ist mit  $\mathcal{O}(\mathbf{g}_L^2)$  gegenüber dem Speicherbedarf der vorberechneten Transformationsmatrizen vernachlässigbar. Weiterhin werden bei Funktionen mit einem mehr als polynomiellen Abfall der Fourierkoeffizienten nur wenige Stützstellen benötigt um Maschinengenauigkeit zu erreichen. Die meisten der in dieser Arbeit durchgeführten Berechnungen in Kapitel 6 benötigten weniger als 32 Stützstellen entlang einer Richtung, um Maschinengenauigkeit zu erreichen.

### 4.3.2 Eindimensionale Transformationen

Der zweite wichtige Algorithmus für die Interpolation auf einem beliebigen dünnen Gitter ist die Durchführung beliebig langer eindimensionaler Transformationen. Bei einem regulärem dünnen Gitter mit dyadischer Levelstruktur kann auf die schnellen Fouriertransformationen, mit einer Laufzeit von  $\mathcal{O}(n \cdot \log(n))$ , zur Berechnung der Fourierkoeffizienten zurückgegriffen werden. Bei beliebiger Levelstruktur treten Transformationen beliebiger Länge und mit beliebig gewählten Stützstellen auf. Zur Berechnung der Koeffizienten werden lineare Gleichungssysteme aufgestellt und gelöst.

Da ein Tensorproduktansatz für die Konstruktion des Gitters verwendet wird, müssen viele eindimensionale Transformationen durchgeführt werden. Das Gleichungssystem, welches für ein Level  $l$  zu lösen ist, sieht wie folgt aus

$$I_l[f](\mathbf{p}_j) = \sum_{k=0}^{\mathbf{g}_l-1} \hat{f}_{\sigma(k)} \Phi_{\sigma(k)}(\mathbf{p}_j) = f(\mathbf{p}_j) \quad (4.18)$$

Die Matrix dieses Gleichungssystemes ist wieder die bekannte A<sub>exp</sub> aus Gleichung (4.14) mit entsprechender Größe, die vom Level der Transformation abhängt. Da sich die Transfor-

mationsregeln nicht ändern, so lange die Gitterstruktur gleich bleibt, können die auftretenden Matrizen im Vorfeld invertiert und abgespeichert werden. Auf diese Weise entspricht eine eindimensionale Transformation einer Matrix-Vektor-Multiplikation. Der Aufwand ist von der Ordnung  $\mathcal{O}(n^2)$ .

Der Nachteil ist ein hoher Speicherbedarf, da für jedes Level zwei Matrizen der Größe  $\mathbf{g}_l \times \mathbf{g}_l$  abgespeichert werden müssen. Doch nur so ist es überhaupt möglich größere Gitter in angemessener Zeit zu konstruieren. In der Abbildung 10 werden die Zeiten für die Fouriertransformationen von zweidimensionalen PLUS1-Gittern mit maximal 128 Levels durch Matrix-Vektor-Multiplikationen und dem Lösen von Gleichungssystemen miteinander verglichen. Der Unterschied sieht nicht besonders groß aus, jedoch werden bei der Konstruktion größer adaptiver Gitter sehr häufig alle Basiskoeffizienten neu berechnet. Und die Anzahl der eindimensionalen Transformationen hängt exponentiell von der Dimension des Problems ab. Dies wurde in Kapitel 3 bei der Abschätzung des Rechenaufwandes zur Berechnung der Fourierkoeffizienten gezeigt.

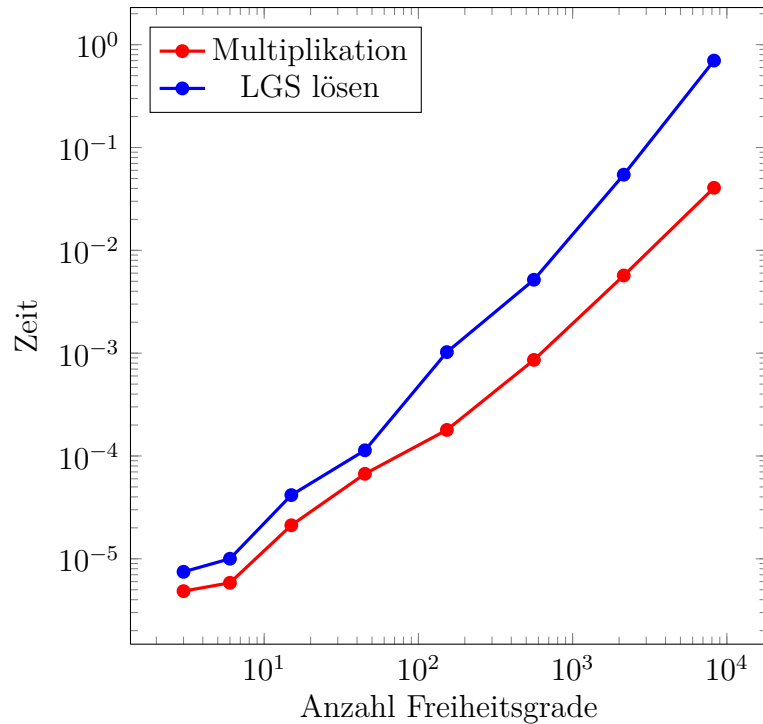


Abbildung 10: Vergleich der Rechenzeit von Matrix-Vektor Multiplikation und Lösen des linearen Gleichungssystems zur Berechnung der eindimensionalen Interpolationsoperatoren.

Es wird wieder eine Levelstruktur der Art  $\mathbf{g}_l = k(l+1)$  angenommen. Der Speicheraufwand für  $L$  Level ist bei dieser Struktur

$$2k^2 \sum_{l=0}^L (l+1)^2 = \mathcal{O}(k^2 L \cdot (L+1)^2) = \mathcal{O}(L \cdot \mathbf{g}_L^2) \quad (4.19)$$

Die inversen Transformationsmatrizen können direkt aufgestellt werden. Wenn ein Koeffizientenvektor  $\hat{f}$  gegeben ist, erhält man die Funktionswerte an den Stützstellen  $\mathbf{p}$  durch Einsetzen in die Interpolationsvorschrift (4.18). Dies entspricht der Multiplikation der Matrix  $A_{\text{exp}}$  aus Gleichung (4.14) mit dem Vektor  $\hat{f}$ .

Die Transformationsmatrizen, um aus den Stützstellen die Koeffizienten für die Basisfunktionen zu erhalten, werden dann durch Invertierung erhalten. In der HCFFT wird die LU Invertierung aus der GNU Scientific Library verwendet. Nachfolgend ist der Algorithmus zur Konstruktion der Transformationsmatrizen dargestellt.

---

**Algorithmus 7:** Berechnung Transformationsmatrizen

---

```

Input :  $L \in \mathbb{N}$ ,  $\mathbf{g} \in \mathbb{N}^L$ ,  $\mathbf{p} \in \mathbb{T}^L$ ,  $\Phi_j(x)$ ,  $\sigma(j)$ 
Output :  $\text{TM}[L]$ ,  $\text{ITM}[L] \subset \{\mathbb{R}^{\mathbf{g}_0 \times \mathbf{g}_0}, \dots, \mathbb{R}^{\mathbf{g}_L \times \mathbf{g}_L}\}$ 
// Für jedes Level ITM durch einsetzen der Funktionswerte der
  Basisfunktionen an Stützstellen aufstellen
for  $l = 0$  to  $L$  do
  | for  $i, j = 0$  to  $\mathbf{g}_l$  do
  | |  $\text{ITM}[i,j] = \Phi_{\sigma(j)}(\mathbf{p}_i)$ 
  | end
end
// Errechne TM durch Invertieren von ITM
for  $l = 0$  to  $L$  do
  |  $A = \text{ITM}[l]$ 
  |  $\text{LUdecomp}(A)$ 
  |  $\text{LUinvert}(A, \text{TM}[l])$ 
end
return  $\text{TM}[L]$ ,  $\text{ITM}[L]$ 

```

---

Der Rechenaufwand zur Berechnung der Transformationsmatrizen mit  $k(l+1)$  Punkten pro Level  $l$  wird analog zu dem Aufwand von Algorithmus 6 bestimmt.

**Lemma 4.2.** *Sei  $L \in \mathbb{N}$  das maximale Level der hierarchischen Basis und  $\mathbf{g}_l = k(l+1)$  für  $0 \leq l \leq L$  die Levelstruktur. Der Rechenaufwand zur Berechnung der Transformationsmatrizen ist*

$$\mathcal{O}(L \cdot \mathbf{g}_L^3) \tag{4.20}$$

**Beweis.** *In der nachfolgenden Tabelle 4.4 ist der Aufwand der einzelnen Rechenschritte in jedem Level angeben. Diese werden nur einmal pro Level aufgerufen.*

*Die dominierenden Terme sind die der LU Zerlegung und Invertierung. Der Gesamtaufwand errechnet sich wie folgt*

$$\sum_{l=0}^L \mathcal{O}(k^3 l^3) = \mathcal{O}(k^3 (L+1)L^3) = \mathcal{O}(L \cdot \mathbf{g}_L^3) \tag{4.21}$$

Operation	Aufwand pro Level
Füllen der Matrix ITM	$\mathcal{O}((kl)^2)$
LUDecomp(A)	$\mathcal{O}((kl)^3)$
LUinvert(A, TM)	$\mathcal{O}((kl)^3)$

Tabelle 4.4: Aufwand zur Konstruktion der Transformationsmatrizen, welche zur Berechnung der Basiskoeffizienten verwendet werden.

Wie im ersten Algorithmus ist der Gesamtaufwand  $\mathcal{O}(L \cdot \mathbf{g}_L^3)$ . Auch hier werden die Berechnungen für jede Gitterstruktur nur einmal durchgeführt und die erhaltenen Matrizen werden gespeichert. Eine Basis kann auch für jedes Gitter verwendet werden, welches die gleiche Levelstruktur verwendet, aber ein kleineres maximales Level hat.

### 4.3.3 Laufzeit und Speicherbedarf Basiskonstruktion

In diesem Abschnitt wird eine Übersicht über den einer PLUS1-Levelstruktur auftretenden Speicherbedarf und die Zeit, die benötigt wird, um eine Basis auszurechnen, gegeben. In Tabelle 4.5 ist der Speicherbedarf und die Zeiten zur Berechnung einer Basis und zum Laden einer vorberechneten Basis aufgeführt. Der Speicherbedarf steigt sehr stark an. Eine Verdoppelung der Stützstellen hat den achtfachen Speicherbedarf zur Folge.

Es ist zu bedenken, dass PLUS1-Gitter für die Verwendung bei Funktionen mit mehr als polynomiell abfallenden Fourierkoeffizienten gedacht sind. Um Maschinengenauigkeit zu erreichen, genügen häufig schon Basen der Größe 32. Die größte in dieser Arbeit verwendete Basis hatte 128 Basisfunktionen und wurde zur Konstruktion eines adaptiven Gitters für eine Funktion mit polynomieller Konvergenzrate verwendet. Die Daten wurden auf einem Xeon X7460 Prozessor mit 2.66 Ghz Taktrate bestimmt.

Stützstellen	Berechnung	Importieren	Speicherbedarf
2	$8.58 \cdot 10^{-3}$	$5.65 \cdot 10^{-4}$	332
4	$1.61 \cdot 10^{-2}$	$5.86 \cdot 10^{-4}$	1,556
8	$9.92 \cdot 10^{-3}$	$7.74 \cdot 10^{-4}$	8,740
16	$1.91 \cdot 10^{-2}$	$8.14 \cdot 10^{-4}$	56,388
32	$6.2 \cdot 10^{-2}$	$1.66 \cdot 10^{-3}$	$3.99 \cdot 10^5$
64	0.33	$6.73 \cdot 10^{-3}$	$2.99 \cdot 10^6$
128	2.78	$3.81 \cdot 10^{-2}$	$2.32 \cdot 10^7$
256	31.71	0.24	$1.82 \cdot 10^8$
512	363.33	1.74	$1.44 \cdot 10^9$
1,024	6,438.11	15.98	$1.15 \cdot 10^{10}$

Tabelle 4.5: Speicherbedarf und Konstruktionszeit der Basen bei PLUS1-Gittern. Die Zeit wird in Sekunden und der Speicherbedarf in Byte angegeben. Es wurde der Datentyp double verwendet.

# 5 HCFFT Bibliothek

Für die Implementation der verallgemeinerten dünnen Gitter mit beliebigen Levelstrukturen wurde die Hyperbolic Cross Fast Fourier Transform Bibliothek, welche am Institut für Numerische Simulation der Universität Bonn entwickelt wurde, erweitert. Die Bibliothek ist in C geschrieben und stellt nicht adaptive und adaptive verallgemeinerte dünne Gitter zur Verfügung. Zuerst wurden die vorhandenen Funktionalitäten vervollständigt und auf Fehler überprüft. Zu diesem Zweck wird ein Vergleich der Interpolationsgenauigkeit und Laufzeit der HCFFT mit anderen Bibliotheken, welche ähnliche Möglichkeiten bieten, gezeigt.

In diesem Kapitel werden die wichtigsten Funktionen, die die HCFFT Bibliothek bietet, vorgestellt. Anschließend folgt ein Vergleich mit zwei frei erhältlichen MATLAB Bibliotheken, die in Teilen gleiche Funktionen bieten. Sowohl die Konstruktionsgeschwindigkeit und die Zeit für eine Funktionsauswertung als auch die Interpolationsgenauigkeit, soweit hierfür Routinen existieren, sind Bestandteil des Vergleichs. Es stellt sich heraus, dass die HCFFT Bibliothek in den meisten Fällen wesentlich schneller ist. Allerdings zeigt sich auch, dass im Bereich der adaptiven Gitterkonstruktion, noch Verbesserungen möglich sind.

## 5.1 Übersicht über die Funktionen

Die HCFFT Bibliothek stellt alle benötigten Algorithmen zur Verfügung um verallgemeinerte dünne Gitter mit unterschiedlichen Basisfunktionen zu konstruieren. Es besteht die Möglichkeit eine Fourier-, Sinus- oder Cosinusbasis sowie Chebyshevpolynome zur Interpolation auszuwählen. Als Gitter stehen verallgemeinerte dünne Gitter zur Verfügung, welche volle und reguläre dünne Gitter als Spezialfälle enthalten. Den verallgemeinerten dünnen Gittern liegt die parametrisierte Indexmenge  $\mathcal{I}_L^T$  zu Grunde und es kann der Parameter  $T$  vorgeben oder ein adaptives Konstruktionsverfahren verwendet werden.

Bei der adaptiven Verfeinerung werden so lange Unterräume hinzugenommen, bis keine erlaubten Unterräume mehr existieren, deren Summe der in ihnen enthaltenen Koeffizienten größer als eine wählbare Schranke  $\epsilon$  ist.

Nachdem ein Gitter konstruiert und die hierarchischen Überschüsse bestimmt wurden, besteht die Möglichkeit verschiedene Operationen auf den Interpolanten anzuwenden. Eine Interpolation mittels Chebyshevpolynomen ist in eine mit Legendrepolynomen umrechenbar. Außerdem können partielle Ableitungen und der Laplace Operator auf den

Interpolanten angewendet werden. Falls die zu Interpolierende Funktion gewisse Differenzierbarkeits Eigenschaften erfüllt, konvergieren die Ableitungen und der Laplace Operator des Interpolanten gegen die entsprechenden Ableitungen der Funktion. Weiterhin stehen Algorithmen zur Verfügung um unterschiedliche Normen, z.B. die  $\mathcal{L}_2$ -Norm oder den  $\mathcal{L}_2$ -Fehler der Interpolation zu bestimmen.

## 5.2 Bibliotheken mit ähnlichen Funktionen

Bibliotheken mit ähnlicher Funktionalität sind beispielsweise die Sparse Grid Interpolation Toolbox [KW05, Kli07] und die Nonequispaced Hyperbolic Cross Fast Fourier Transform [DKP10, KK11]. Beides sind MATLAB Programme und verwenden reguläre dünne Gitter zur Interpolation von Funktionen.

Der Vergleich wurde auf einer Workstation mit 4 Xeon X7460 CPUs mit einer Taktrate von 2.66 Ghz und 256 GB RAM durchgeführt. Da die HCFFT Bibliothek noch nicht parallelisiert ist, wurde der Vergleich auf einen Prozessorkern beschränkt.

Für den Vergleich wurde eine glatte Funktion mit zwei bis fünf Dimensionen interpoliert. Interessant ist vor allem ein Vergleich der Interpolationsgenauigkeit und Geschwindigkeit beider Bibliotheken. Auch der Aufwand der verwendeten Algorithmen wird betrachtet, da sich hier zeigt, ob Verbesserungsmöglichkeiten bestehen.

Vorab sei gesagt, dass es keine freie Bibliothek gibt, die so vielseitig ist, wie die HCFFT. Die NHCFFT bietet zwar alle nötigen Routinen an um eine Fouriertransformation auf einem hyperbolischem Kreuz auszuführen, kann aber keine adaptiven Gitter konstruieren. Sie wurde vor allem für Fouriertransformationen mit nicht äquidistanten Stützstellen entwickelt. Dies könnte interessant sein, da bei Gittern mit beliebigen monotonen Indexmengen solche Fouriertransformationen auftreten.

Die Sparse Grid Interpolation Toolbox wiederum ist in der Lage adaptive Gitter zu konstruieren, stellt aber nur Chebyshev Polynome bereit um hohe Interpolationsgenauigkeiten zu erreichen.

## 5.3 Sparse Grid Interpolation Toolbox

Die Sparse Grid Interpolation Toolbox, kurz SpInterp, verwendet unter anderem stückweise lineare Funktionen und Chebyshev Polynome zur Interpolation von Funktionen. Es stehen drei verschiedene Gitter für stückweise lineare und zwei Gitter für polynomielle Basisfunktionen zur Auswahl. In diesem Vergleich werden die nicht adaptiven und adaptiven Chebyshev-Gauss-Lobatto Gitter mit Chebyshev Polynomen als Basisfunktionen verwendet. Diese sind auch in der HCFFT Bibliothek implementiert.

Bei den Gittern der beiden Bibliotheken gibt es einen kleinen Unterschied. Die Sparse Grid Interpolation Toolbox benutzt im 0. Level den Mittelpunkt des gewählten Intervalls als Stützstelle, während die HCFFT Bibliothek die beiden Randpunkte nimmt. Im ersten Level verhält sich dies genau anders herum und die nachfolgenden Level stimmen in beiden

Bibliotheken überein. Dies führt dazu, dass bei der HCFFT der Rand stärker besetzt ist, und in der Sparse Grid Interpolation Toolbox die Mitte des Gebietes.

Nachfolgend sind als Beispiel CGL-Gitter mit Level 3 und Dimension 2 abgebildet. Das rechte ist von der HCFFT Bibliothek und das linke von der Sparse Grid Interpolation Toolbox.

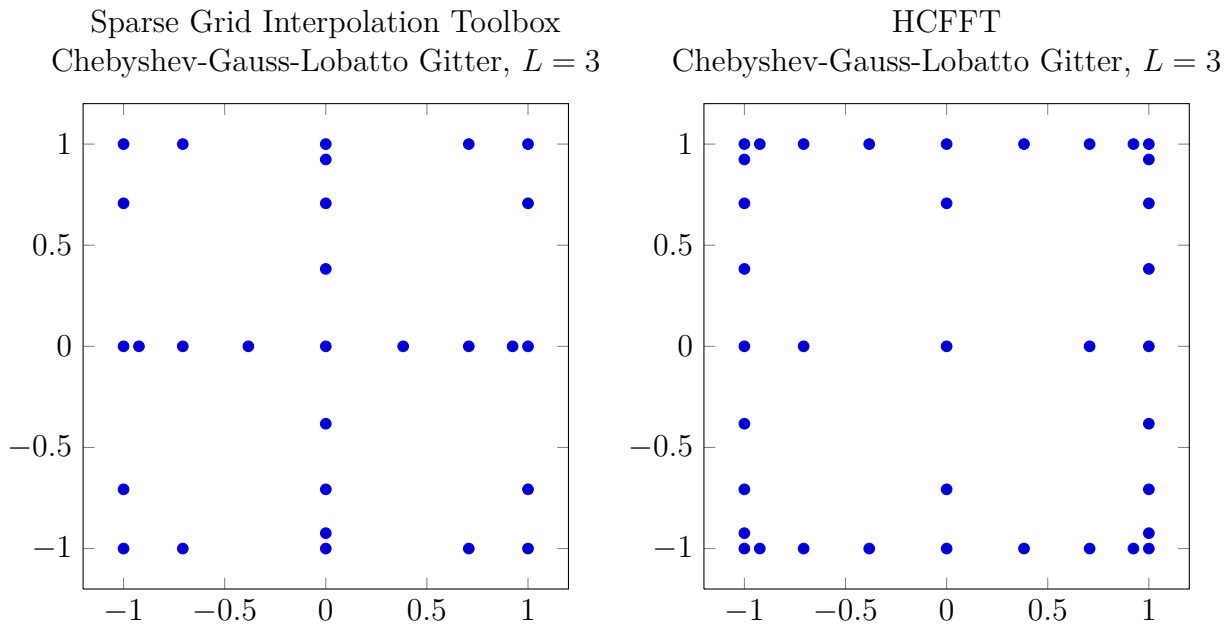


Abbildung 11: Zweidimensionale Chebyshev-Gauss Lobatto Gitter des Level 3 der Sparse Grid Interpolation Toolbox und HCFFT. Die Toolbox gewichtet das innere des Gebietes stärker. Bei der HCFFT ist der Rand stärker besetzt.

Mit der Toolbox können Gitter adaptiv und nicht adaptiv konstruiert werden. Für nicht adaptive Gitter ist das maximale Level, welches ein Unterraum haben kann, zehn. Dies entspricht 1025 Punkten entlang einer Achse. Die Unterräume werden so ausgewählt, dass die Bedingung  $\|\mathbf{1}\|_1 \leq L$  für jeden Unterraum  $\mathcal{W}_l$  des Gitters erfüllt ist.

Bei der adaptiven Gitterkonstruktion kann ein maximales Level, eine maximale Anzahl an Stützstellen und eine Mindestgrenze für den geschätzten relativen und absoluten Fehler angegeben werden. Das Gitter wird so aufgebaut, dass entweder der geschätzte Fehler oder die Stützstellenzahl erreicht wurde, falls dies mit dem gegebenen maximalem Level möglich ist.

### 5.3.1 Vergleich von HCFFT und Sparse Grid Interpolation Toolbox

Für den Vergleich wurde die folgende analytische Funktion auf dem Gebiet  $[-1, 1]^D$  interpoliert. Diese Funktion ist auf dem Gebiet nicht periodisch und hat die endliche Ordnung  $D$ . Weiterhin sind alle Dimensionen gleich Gewichtet um möglichst große adaptive Gitter

zu erhalten.

$$f(\mathbf{x}) = e^{\cos\|\mathbf{x}\|_1} \quad (5.1)$$

Es wurde einmal mit einem nicht adaptiven dünnen Gitter der Level 1 - 10 und dann mit einem adaptiven Gitter mit maximalem Level 10 und immer kleiner werdender Fehlertoleranz interpoliert. Im Falle der HCFFT Bibliothek heißt dies, dass die Grenze für die Gewichte der Unterräume, bei denen diese dem Gitter hinzugefügt werden, immer kleiner gesetzt wurde. Bei der Sparse Grid Interpolation Toolbox wurde die geschätzte relative Genauigkeit für die Interpolation immer weiter verringert.

Von dem Interpolanten wurde der relative  $\mathcal{L}_2$ -Fehler mit einem 1D Lattice Rule Verfahren [CKN06] bestimmt. Dies ist ein Quasi Monte Carlo Verfahren, bei dem die Punkte so konstruiert werden, dass spektrale Konvergenz bei der Berechnung des Integrals einer analytischen Funktion erreicht wird. Außerdem wurde die benötigte Zeit zum Erstellen des Gitters und der Berechnung der Koeffizienten sowie die durchschnittliche Zeit für eine Funktionsauswertung bestimmt.

Die Sparse Grid Interpolation Toolbox hat die Möglichkeit eine ganze Menge von Punkten mit nur einem Lauf durch die Datenstrukturen des Gitters auszuwerten. Der Vorteil dieser Methode liegt auf der Hand. Die Datenstrukturen des Gitters werden auf diese Weise nur einmal durchlaufen was zu schnelleren Auswertungszeiten führt. Diese Möglichkeit wurde im Rahmen der Diplomarbeit auch in der HCFFT Bibliothek implementiert, was unter anderem die Berechnung des  $\mathcal{L}_2$ -Fehlers beschleunigt hat, da hier alle Auswertungspunkte im Vorfeld durch den Lattice Vektor gegeben sind.

Es tritt bei der Verwendung von Chebyshev Polynomen und kleinen Gittern mit geringer Dimension eine Besonderheit auf. Es ist zu erwarten, dass die Auswertung von mehreren Punkten auf einmal schneller als eine Schleife von Einzelauswertungen ist. Bei der Fourierinterpolation mit komplexen Exponentialfunktionen als Basis ist dies immer der Fall. Bei der Verwendung von Chebyshev Polynomen als Basisfunktionen hingegen ist die Einzelauswertung in einer Schleife teilweise schneller. Da dies nur bei Gittern mit wenig Dimensionen geschieht, wird dieses Phänomen hier nicht weiter untersucht.



### 5.3.2 Interpolationsgenauigkeit

Erwartungsgemäß ist der relative  $\mathcal{L}_2$ -Fehler der nicht adaptiven Gitter bei beiden Bibliotheken vergleichbar. Bei steigender Dimension ist eine bessere Genauigkeit der HCFFT Bibliothek zu beobachten. Dies liegt an der unterschiedlichen Wahl der Stützstellen. Ein Gitter, welches den Rand stärker Gewichtet, ist für diese Funktion offenbar die bessere Wahl.

Bei bis zu vier Dimensionen stellt sich bei beiden Programmen die gleiche Konvergenzrate ein. Bei fünf Dimensionen hingegen ist die Konvergenzrate in den letzten fünf Leveln bei der HCFFT besser als bei der Sparse Grid Interpolation Toolbox. Hier reicht die Gittergröße der Toolbox noch nicht aus um alle wichtigen Strukturen der Funktion nachzubilden.

Nachfolgend werden adaptiv konstruierte Gitter für die gleiche Funktion mit zwei bis fünf Dimensionen untersucht. Der relative  $\mathcal{L}_2$ -Fehler verhält sich hier bei beiden Bibliotheken gleich. Bei bis zu vier Dimensionen wird die Genauigkeit maschinell bedingt begrenzt. Beide Pakete erreichen die Genauigkeitsgrenze mit vergleichbar vielen Stützstellen. Bei fünf Dimensionen reicht die maximale Gittergröße die man bei der Sparse Grid Interpolation Toolbox einstellen kann, nicht mehr aus, um Maschinengenauigkeit zu erreichen.

Die unterschiedlich gewählten Stützstellen in den ersten beiden Level fallen nicht mehr auf. Das adaptive Verfahren wählt bei beiden Paketen die Unterräume so aus, dass gleiche Interpolationsgenauigkeiten erreicht werden.

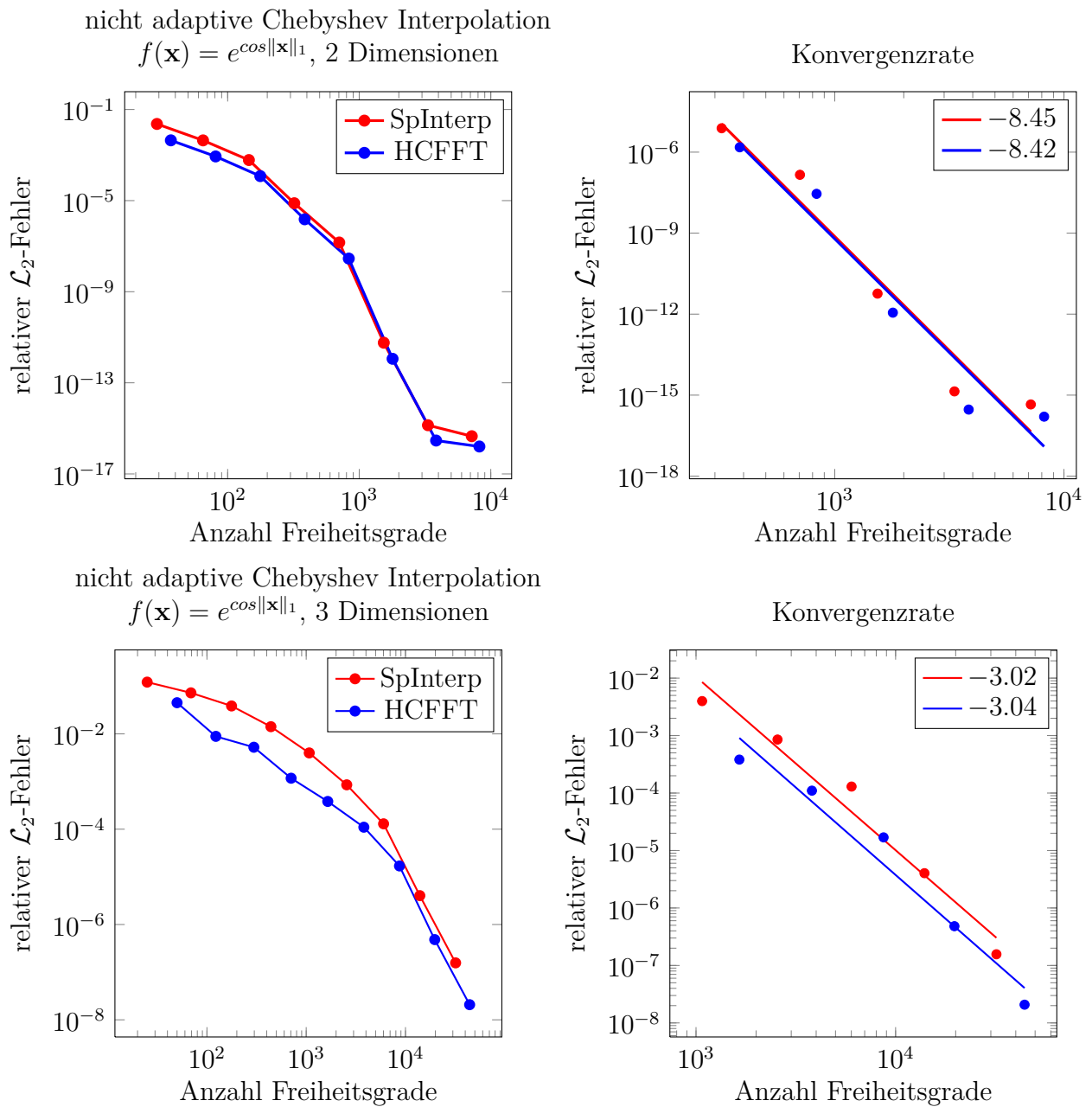


Abbildung 12: Vergleich des Konvergenzverhaltens der HCFFT Bibliothek mit der Sparse Grid Interpolation Toolbox bei nicht adaptiven Gittern mit zwei und drei Dimensionen. Es wurde die Funktion  $f(\mathbf{x}) = e^{\cos\|\mathbf{x}\|_1}$  auf einem Chebyshev-Gauss-Lobatto Gitter mit maximalem Level zehn interpoliert. Die Konvergenzrate wurde für die letzten fünf Datenpunkte durch eine lineare Regressionsgerade bestimmt.

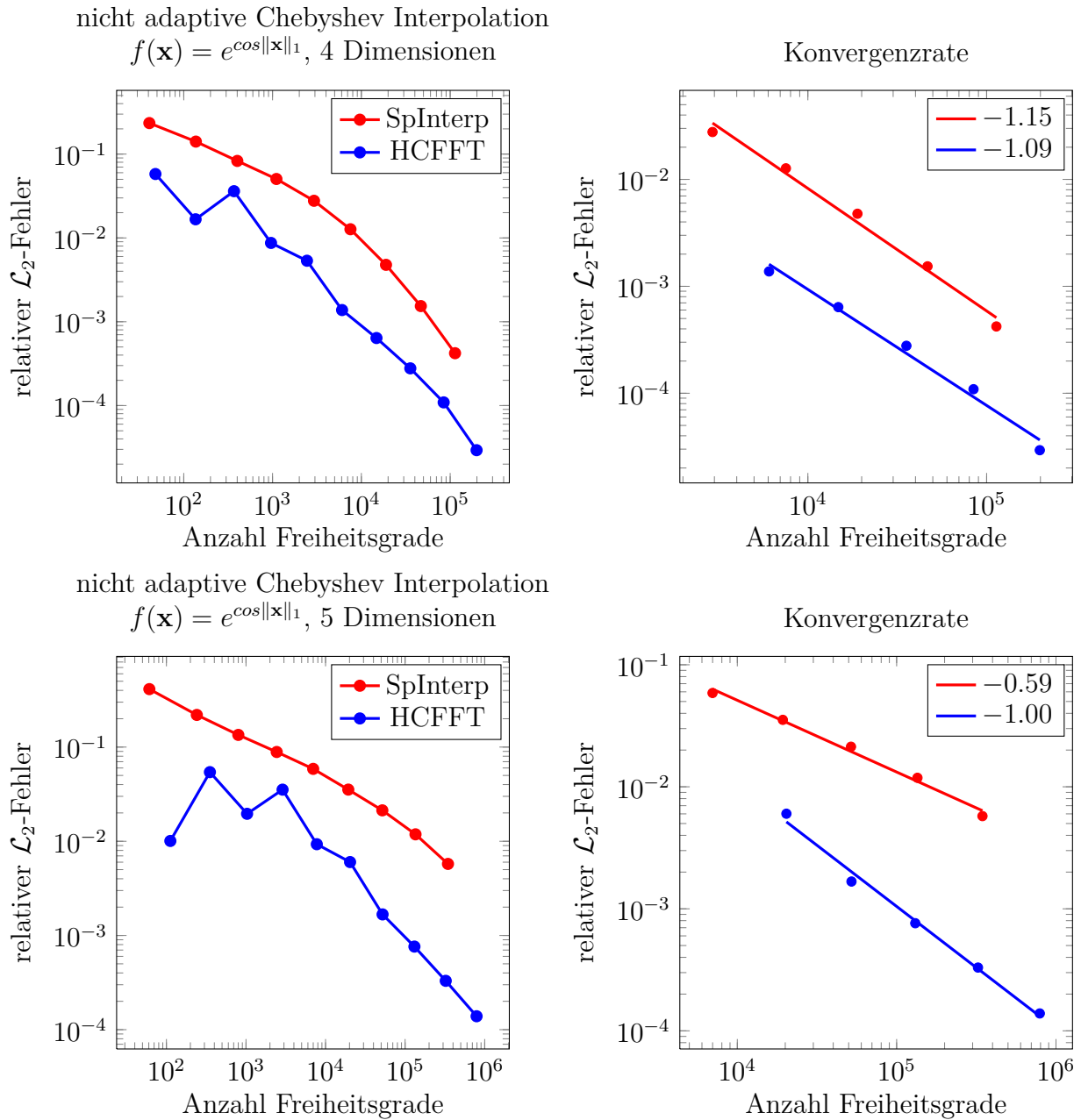


Abbildung 13: Vergleich des Konvergenzverhaltens der HCFFT Bibliothek mit der Sparse Grid Interpolation Toolbox (SpInterp) bei nicht adaptiven Gittern mit vier und fünf Dimensionen. Es wurde die Funktion  $f(\mathbf{x}) = e^{\cos\|\mathbf{x}\|_1}$  auf einem Chebyshev-Gauss-Lobatto Gitter mit maximalem Level zehn interpoliert. Die Konvergenzrate wurde für die letzten fünf Datenpunkte durch eine lineare Regressionsgerade bestimmt.

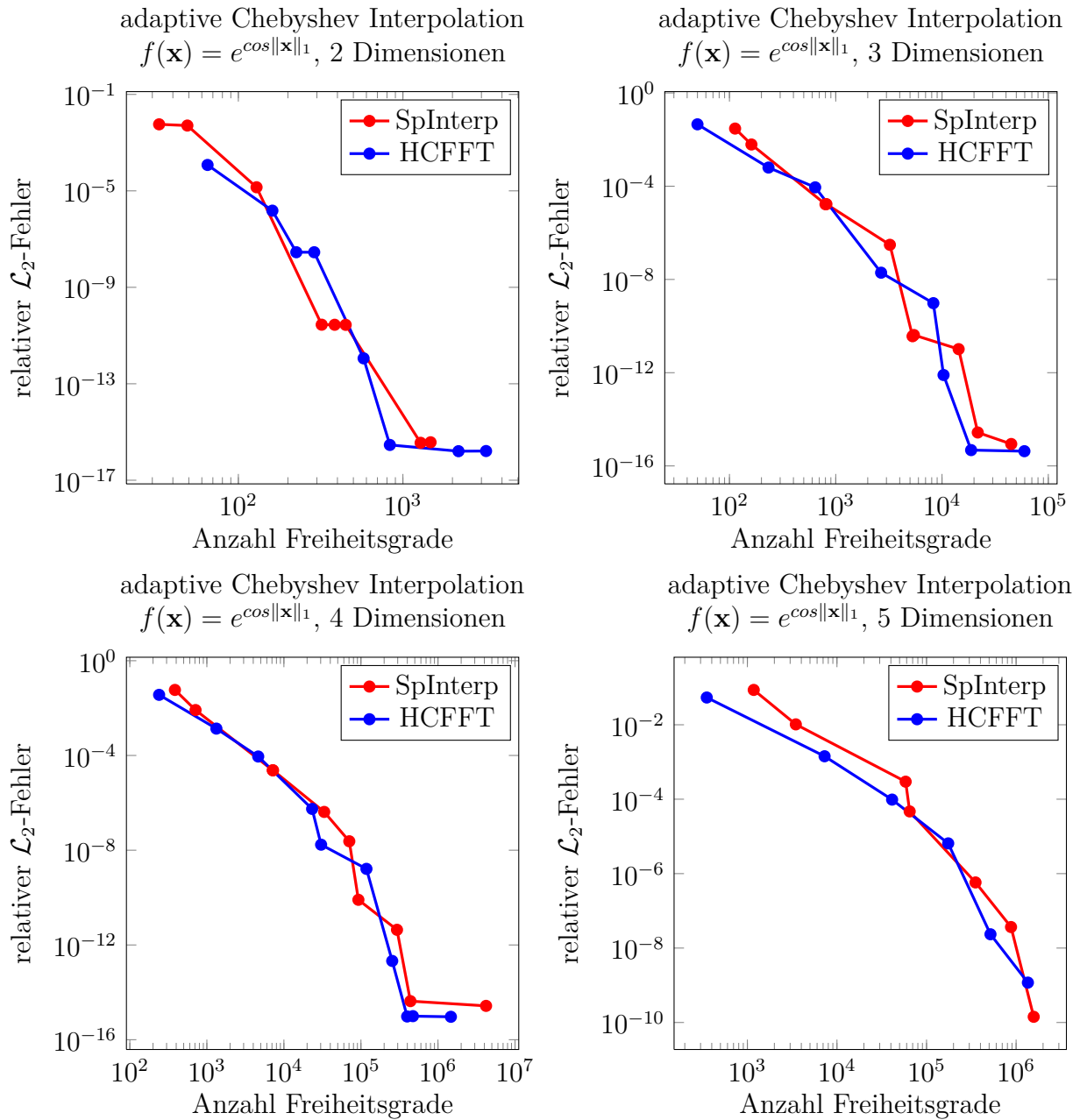


Abbildung 14: Vergleich der Interpolationsgenauigkeit der HCFFT Bibliothek mit der Sparse Grid Interpolation Toolbox bei adaptiven Gittern mit zwei bis fünf Dimensionen. Es wurde die Funktion  $f(\mathbf{x}) = e^{\cos\|\mathbf{x}\|_1}$  auf einem Chebyshev-Gauss-Lobatto Gitter interpoliert.

### 5.3.3 Geschwindigkeit Gitterkonstruktion

Zuerst wurde die Geschwindigkeit zur Konstruktion von nicht adaptiven Gittern mit zwei bis fünf Dimensionen betrachtet. Es wurden Gitter bis zu einem maximalem Level von Zehn konstruiert. In die gemessene Zeit fließt sowohl die Konstruktion der Datenstrukturen zur Verwaltung des Gitters, als auch die Zeit, die benötigt wird um die Basiskoeffizienten zu bestimmen, ein. Hier ist zu sehen, dass die Algorithmen bei beiden Bibliotheken einen vergleichbaren Rechenaufwand haben. Auf Grund der unterschiedlichen Implementationen ist die HCFFT jedoch um fast zwei Größenordnungen schneller. In den nachfolgenden Grafiken ist die Konstruktionszeit gegen die Zahl der Gitterpunkte aufgetragen. In der Tabelle

Anzahl der Dimensionen	SpInterp	HCFFT
2	0.49	0.59
3	0.70	0.83
4	0.81	0.92
5	0.89	0.95

Tabelle 5.1: Die Tabelle zeigt den Rechenaufwand in Abhängigkeit der Zahl der Freiheitsgrade zur Konstruktion nicht adaptiver Gitter mit zwei bis fünf Dimensionen. Es wurde eine lineare Regression verwendet und der Wert gibt die Steigung der Regressionsgeraden an.

Bei den adaptiven Gittern fällt bei steigender Dimension auf, dass die verwendeten Algorithmen einen unterschiedlichen Aufwand haben. Die HCFFT Bibliothek ist oft schneller als die Sparse Grid Interpolation Toolbox, doch der Geschwindigkeitsvorteil schwindet sowohl mit steigender Dimension als auch wachsender Zahl der Freiheitsgrade. Dies deutet darauf hin, dass der bei der HCFFT Bibliothek verwendete Greedy Algorithmus verbessert werden kann. Dies zeigt sich auch bei weiteren Untersuchungen.

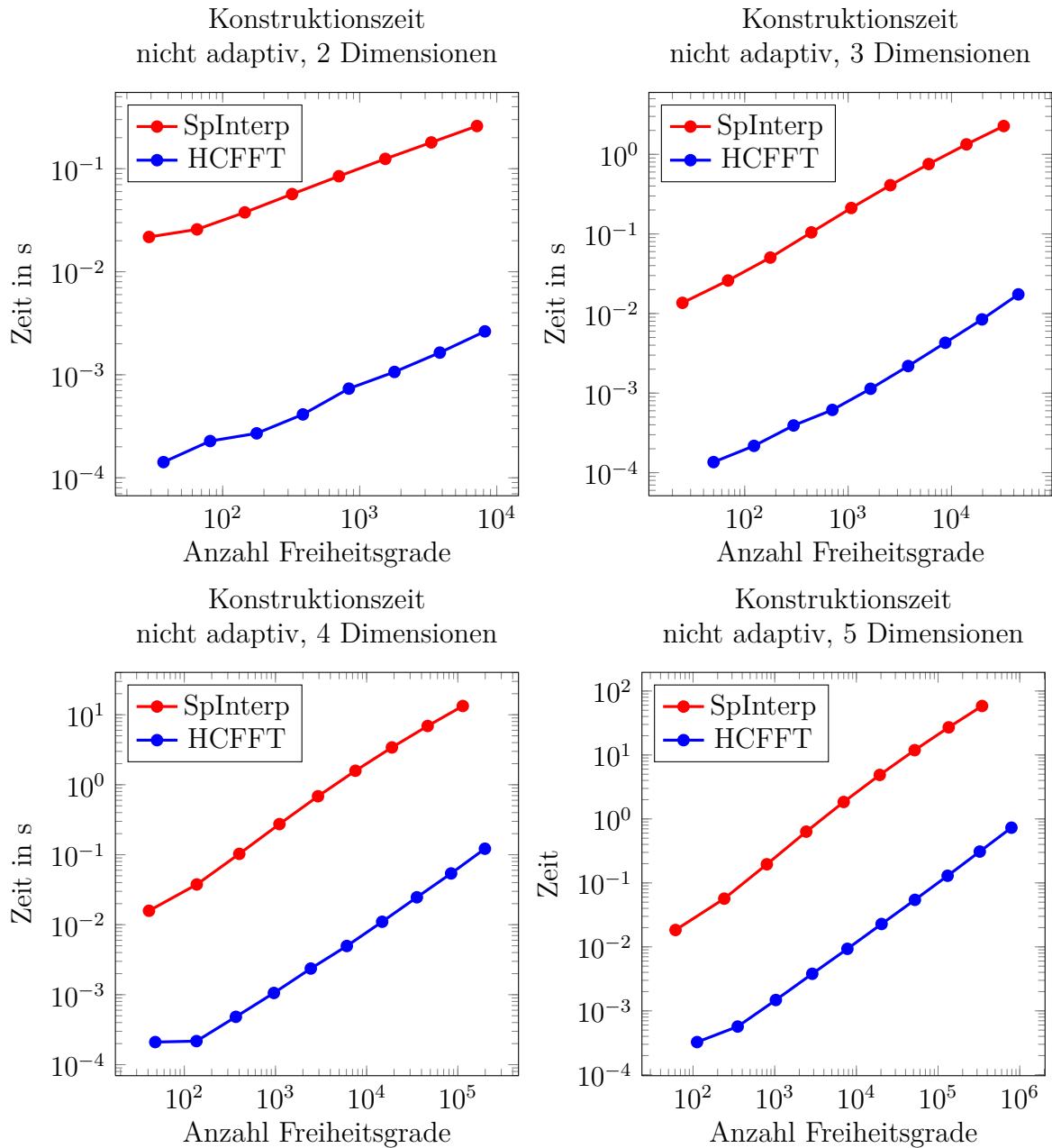


Abbildung 15: Vergleich der Zeiten der HCFFT und Sparse Grid Interpolation Toolbox zur Konstruktion nicht adaptiver Gitter mit zwei bis fünf Dimensionen. Es wurde die Zeit zur Konstruktion des Gitterraumes und der Berechnung der Basiskoeffizienten gemessen.

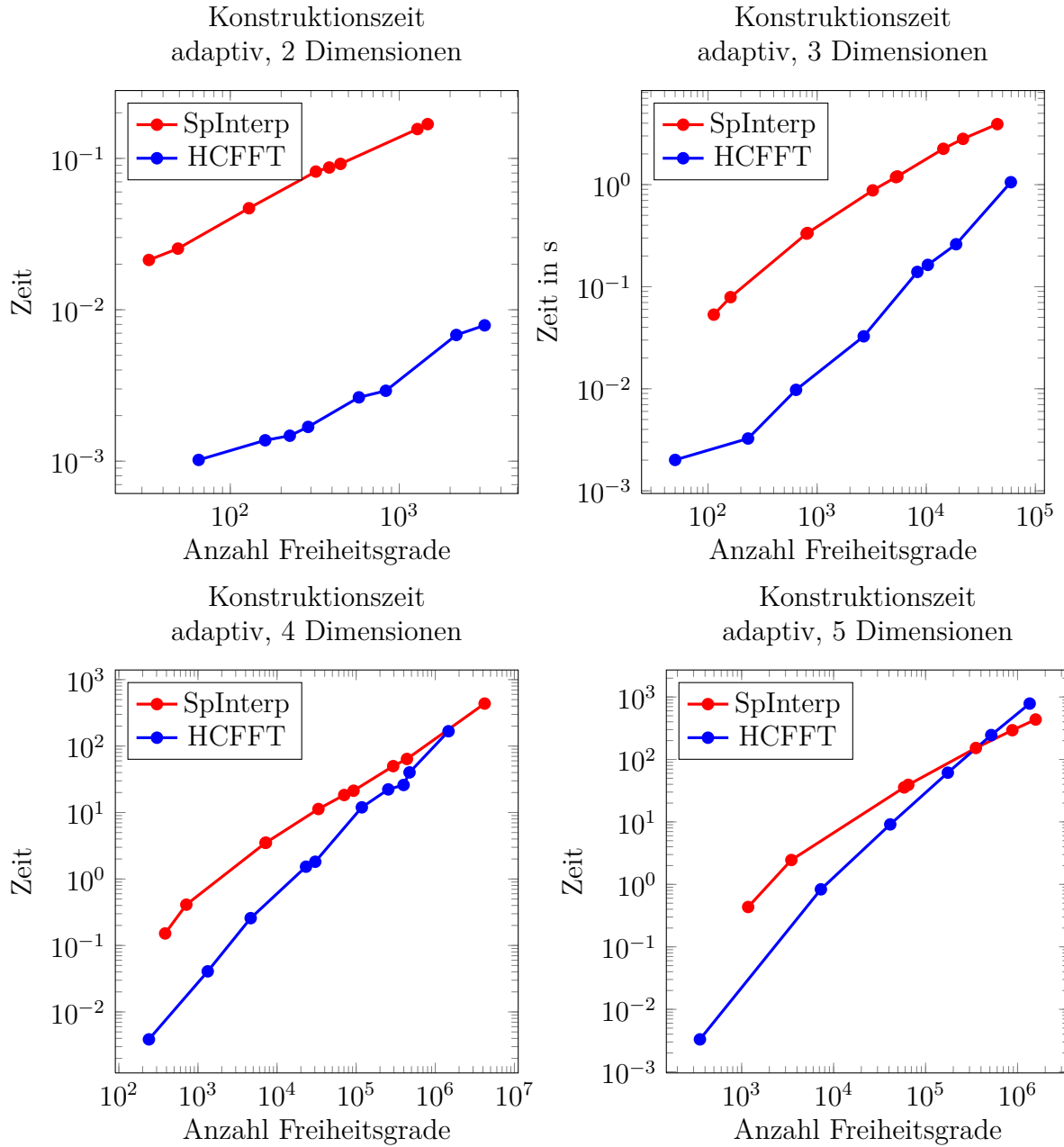


Abbildung 16: Vergleich der Zeiten der HCFFT und Sparse Grid Interpolation Toolbox zur Konstruktion nicht adaptiver Gitter mit zwei bis fünf Dimensionen. Es wurde die Zeit zur Konstruktion des Gitterraumes und der Berechnung der Basiskoeffizienten gemessen.

### 5.3.4 Geschwindigkeit Funktionsauswertung

Je nach Anwendung sind die durchschnittlichen Zeiten für eine oder viele Funktionsauswertungen wesentlich interessanter als die benötigte Zeit zur Konstruktion des Gitters. Der Interpolant kann an beliebig vielen Stellen gleichzeitig ausgewertet werden.

Falls im Vorfeld die Auswertungspunkte bekannt sind wird so Zeit eingespart. In den Grafiken wird dies als Array bezeichnet. In diesem Fall werden die Unterräume nur einmal durchlaufen und in jedem die Basisfunktionen an vielen Stellen ausgewertet. Der Vergleich der Zeit für eine Funktionsauswertung wird hier einmal mit der Auswertung von 100 Datenpunkten als Array und als Schleife über 100 einzelne Auswertungen durchgeführt. Beides sind Fälle, die in der Praxis vorkommen und von Interesse sind. Es wird jeweils die durchschnittliche Zeit für eine Auswertung gemessen und in den Grafiken aufgetragen.

Es ist hier ein deutlicher Unterschied im Laufzeitverhalten beider Bibliotheken bei steigender Stützstellenzahl zu sehen. Bei der Verwendung von vielen einzelnen Auswertungen ist die HCFFT in jedem Fall schneller und wird erst bei maximaler Gittergröße, welche die Sparse Grid Interpolation Toolbox konstruieren kann, von dieser eingeholt.

Im Falle von 100 Auswertungen auf einmal ist die Toolbox jedoch schon ab moderaten Gittergrößen, mit vierstelliger Stützstellenzahl je nach Anzahl an Dimensionen, schneller als die HCFFT. Der Auswertungszeiten der HCFFT haben einen steileren Verlauf, als die der Toolbox. Dies bedeutet, dass der Aufwand der Algorithmen in der HCFFT größer ist. Hier zeigt sich ein Vorteil der Datenstrukturen, die in der Toolbox verwendet werden und es deuten sich Verbesserungsmöglichkeiten bei der HCFFT an.

Werden die Graphen für adaptive und nicht adaptive Gitter betrachtet, so fällt auf, dass kein Unterschied bei der Zeit zur Funktionsauswertung besteht. Die Art auf welche die Gitter erzeugt wurden ist also unerheblich für die Auswertung. Diese hängt vor allem von der Zahl der Gitterpunkte ab.



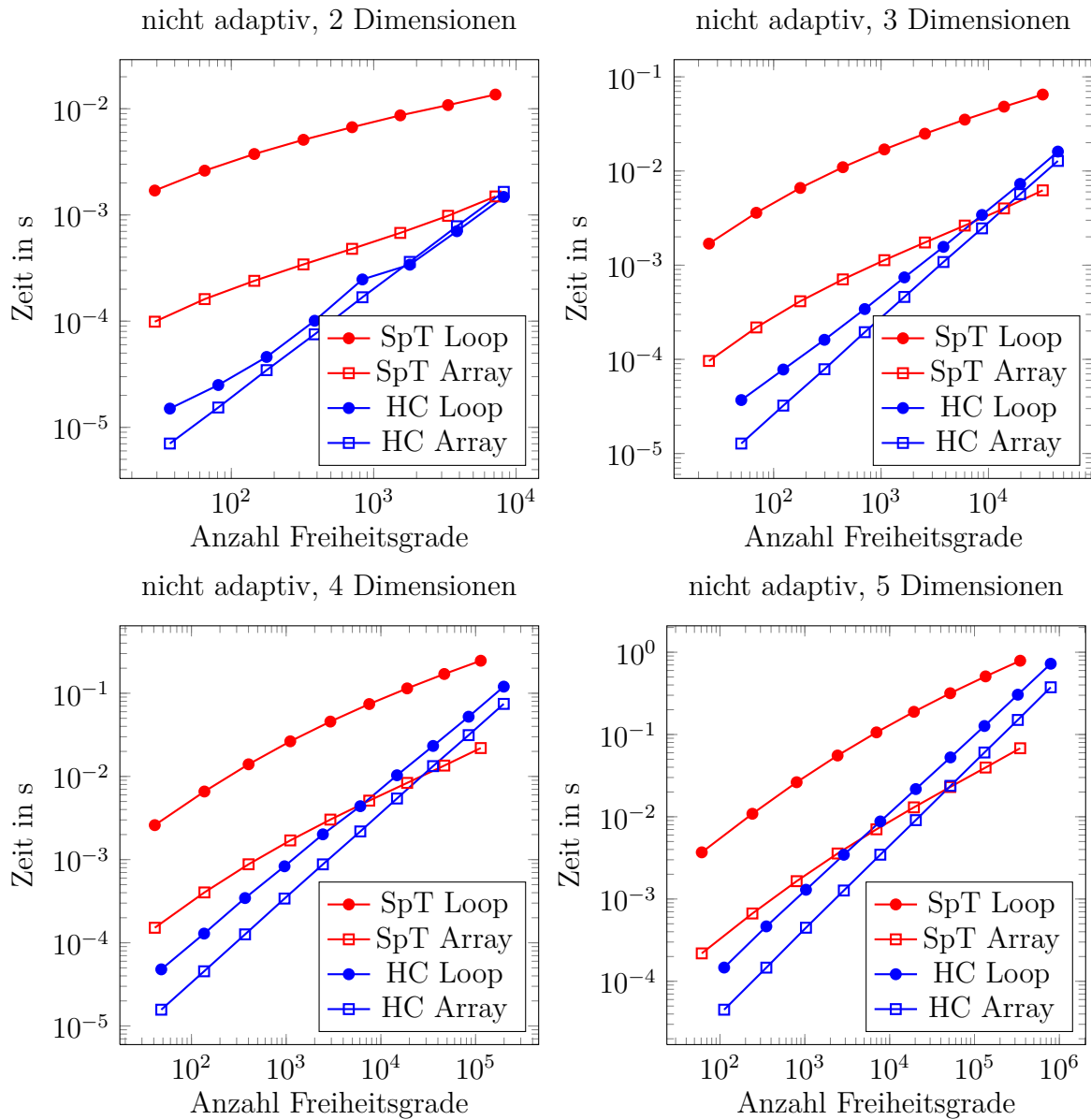


Abbildung 17: Es ist die durchschnittliche Zeit für eine Funktionsauswertung bei nicht adaptiven Gittern zu sehen. Hierbei werden zwei Verfahren verwendet. Einmal werden 100 einzelne Auswertungen gemessen. Dies wird in den Diagrammen als Loop bezeichnet. Das zweite Verfahren sind 100 Auswertungen in einer Schleife, welches in den Diagrammen als Array bezeichnet wird. HC steht für die HCFIT und SpT für die Sparse Grid Interpolation Toolbox.

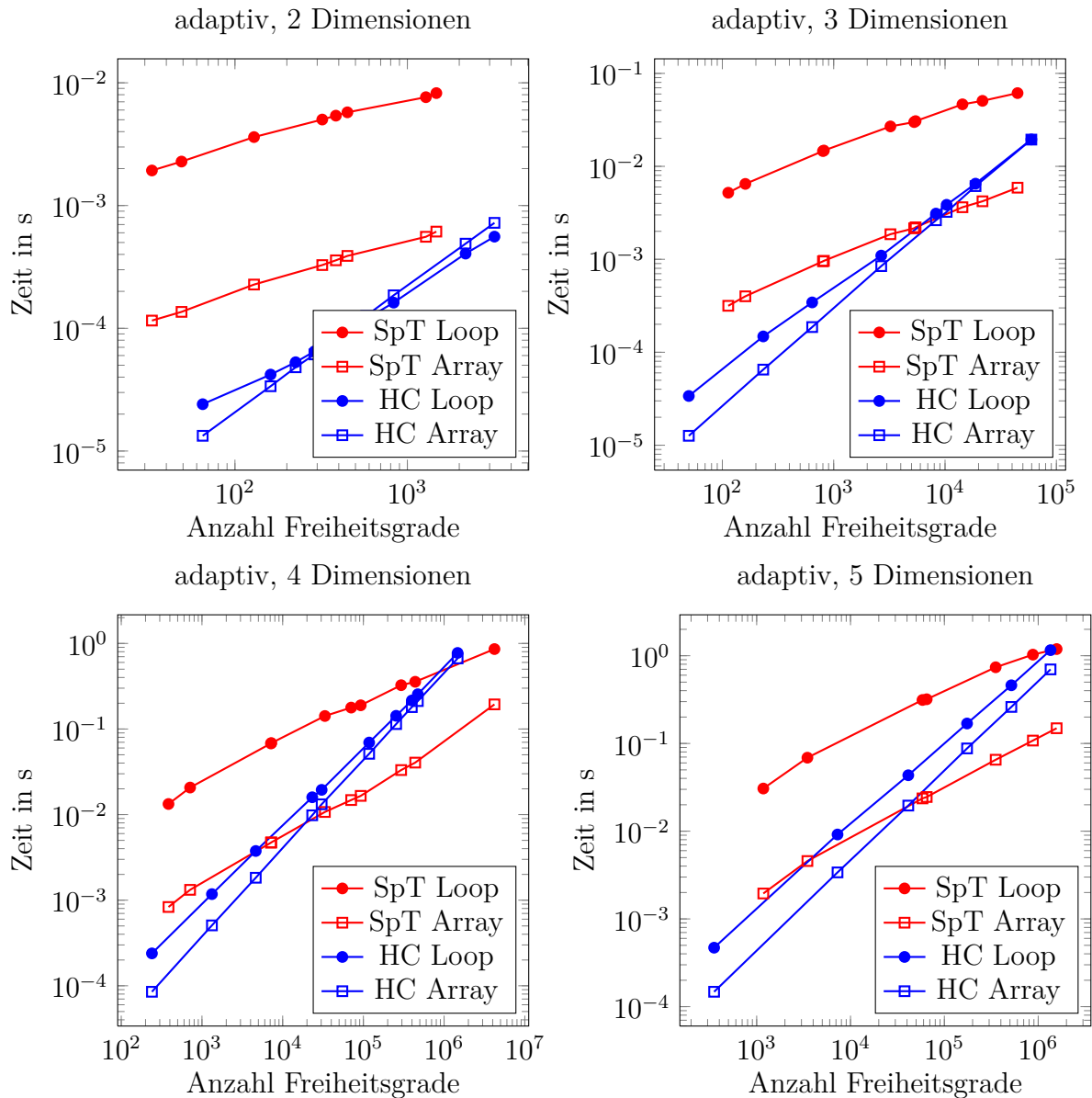


Abbildung 18: Es ist die durchschnittliche Zeit für eine Funktionsauswertung bei adaptiven Gittern zu sehen. Hierbei werden zwei Verfahren verwendet. Einmal werden 100 einzelne Auswertungen gemessen. Dies wird in den Diagrammen als Loop bezeichnet. Das zweite Verfahren sind 100 Auswertungen in einer Schleife, welches in den Diagrammen als Array bezeichnet wird. HC steht für die HCFFT und SpT für die Sparse Grid Interpolation Toolbox.

## 5.4 NHCFFT

Die Non Equispaced Hyperbolic Cross Fast Fourier Transform Bibliothek konstruiert Gitter mit nicht äquidistanten Stützstellen und benutzt Fourierpolynome um Funktionen auf diesen zu interpolieren. Die Stützstellen können auch äquidistant gewählt werden. Dies führt zu genau den gleichen Gittern, welche auch die HCFFT Bibliothek zur Interpolation mit Fourierpolynomen verwendet.

Da eine Routine zur Funktionsauswertung bei der NHCFFT fehlt, wurde nur die Zeit für die Konstruktion eines Gitters und Berechnung der Basiskoeffizienten verglichen. Die Interpolationsgenauigkeit sollte, da es sich um die gleichen Stützstellen und Basisfunktionen handeln, vergleichbar sein. Es wurden nicht adaptive Gitter mit maximalem Level zehn, für zwei bis fünf Dimensionen konstruiert.

Es ist zu sehen, dass die Algorithmen zur Gitterkonstruktion einen ähnlichen Aufwand haben. Allerdings ist die HCFFT Bibliothek um mehrere Größenordnungen schneller, als die NHCFFT.

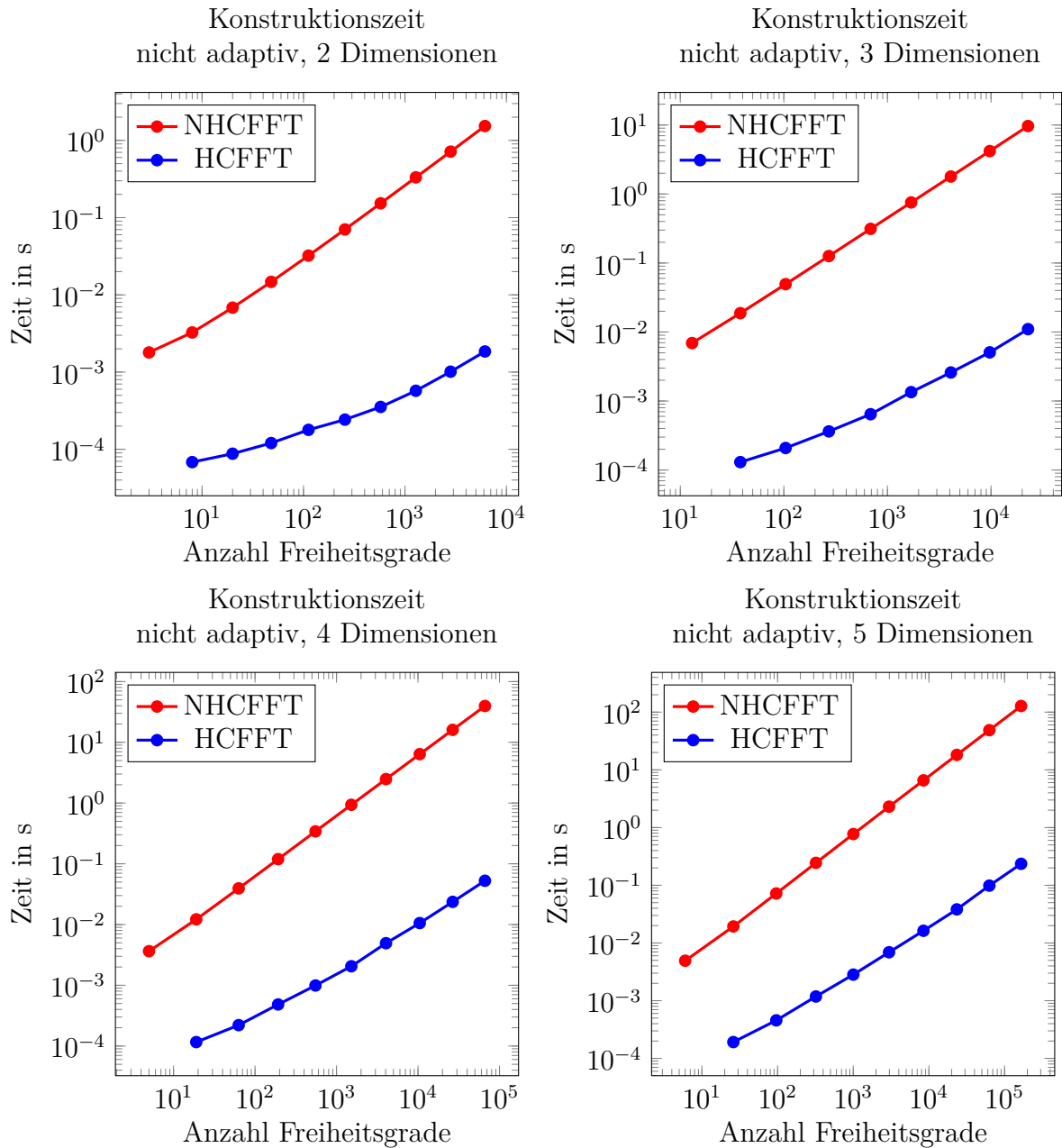


Abbildung 19: Vergleich der Konstruktionszeit nicht adaptiver regulärer dünner Gitter eines maximalen Levels Zehn zur Fouriertransformation. Es wurde die Zeit für die Erstellung des Gitters und Berechnung der Fourierkoeffizienten für ein Gitter mit zwei bis fünf Dimensionen gemessen.

## 6 Numerische Experimente

In diesem Kapitel werden die numerischen Ergebnisse präsentiert. Zur Untersuchung werden zwei verschiedene periodische Testfunktionen benutzt. Eine Funktion unterliegt einem polynomiellen Abfall der Fourierkoeffizienten und wird vor allem benutzt um zu zeigen, dass dyadische Levelstrukturen in diesem Fall die bessere Wahl sind. Die andere Funktion ist ein Tensorprodukt aus analytischen Funktionen und bietet zusätzlich die Möglichkeit die endliche Ordnung über einen Parameter zu steuern. Diese zeigt die Überlegenheit der PLUS1-Gitter bei einem Abfall der Fourierkoeffizienten, welcher schneller als polynomiell ist. Eine PLUS1-Struktur bedeutet, dass jedes Level nur eine neue Stützstelle enthält und folglich jeder Unterraum nur eine hierarchische Basisfunktion. Die Ausgangsbasis ist eine Fourierbasis

$$\mathcal{B}_l := \{e^{i\sigma(0)x}, e^{i\sigma(1)x}, \dots, e^{i\sigma(l)x}\} \quad (6.1)$$

Es werden die durchnummerierten dyadischen Stützstellen  $\mathbf{p}$  aus Abschnitt 3.2.1 zur Konstruktion der PLUS1-Gitter verwendet.

Bei den Experimenten werden nicht adaptive und adaptive Gitter für beide Funktionen verwendet und miteinander verglichen. In den ersten Untersuchungen wird die Funktion mit polynomiellen Abfall betrachtet. Es zeigt sich, dass die Wahl einer dyadischen Stützstellenmenge besser geeignet ist. Im Falle nicht adaptiver Gitter liefern diese die bessere Konvergenzrate und im Falle adaptiver Gitter die bessere Konstruktionszeit bei gleicher Interpolationsgenauigkeit.

Adaptiv konstruierte Gitter erreichen mit beiden Levelstrukturen gleiche Konvergenzraten. Hier zeigen sich allerdings auch die ersten Probleme des adaptiven Verfahrens. Die möglichen Verfeinerungskriterien sind bei für dyadische Levelstrukturen der verwendeten Testfunktion zu ungenau. Bei schnellem polynomiellen Abfall werden auf einmal viele Unterräume hinzugenommen, so dass hier keine verwertbaren Daten entstanden sind. Bei den PLUS1-Gittern zeigt sich hier auch der rasante Anstieg der Konstruktionszeiten bei dem adaptiven Verfahren, was an der großen Anzahl der Unterräume liegt.

Im zweiten Teil wird eine Funktion mit exponentiellem Abfall der Fourierkoeffizienten untersucht. Die Überlegenheit der PLUS1-Gitter wird sehr deutlich sichtbar. Sowohl bezüglich der Zahl der Stützstellen, als auch bezüglich der Konstruktionszeit erreichen diese eine bessere Genauigkeit bei nicht adaptiv konstruierten Gittern. Es werden zuerst Funktionen mit voller Ordnung betrachtet und erste Daten bezüglich der Zeit für eine Funktionsauswertung gegeben. Bei dieser zeigt sich der Vorteil von mehreren Auswertungen auf einmal und auch, dass die Auswertungszeiten, auch bei mehrfachen Durchlauf

der Unterräume, fast nur von der Zahl der Gitterpunkte abhängt. Hier werden auch an zwei Beispielen der Zeitbedarf der einzelnen Schritte, das heißt Konstruktionszeit der Unterräume und Berechnung der Basiskoeffizienten gezeigt.

Als nächstes werden adaptive Gitter untersucht. Hier ist die PLUS1-Struktur wieder überlegen, da der adaptive Algorithmus die Indexmengen genauer ausschneiden kann. Hier werden auch das erste Mal Funktionen mit endlicher Ordnung verwendet. Ein direkter Vergleich adaptiver und nicht adaptiver Gitter zeigt bei der PLUS1-Struktur, dass adaptive Gitter bei einer Funktion mit voller Ordnung keinen Vorteil bieten.

Der vorletzte Absatz untersucht Funktionen mit geringer endlicher Ordnung aber sehr hoher Dimension. Hier zeigt sich bei den adaptiv konstruierten Gittern der Vorteil der beliebigen Levelstruktur am deutlichsten. Mit diesen ist es bei hochdimensionalen Funktionen überhaupt erst möglich Gitter zu konstruieren, die die Maschinengenauigkeit erreichen. Die dyadische Levelstruktur übernimmt zu viele Basisfunktionen, die keinen Beitrag zur Genauigkeit liefern. Um eine besser Genauigkeit zu erreichen werden so viele Stützstellen gebraucht, dass die Konstruktionszeit über eine Woche betrug um einen relativen  $\mathcal{L}_2$ -Fehler im Bereich von  $10^{-12}$  bei nur 30 Dimensionen und einer endlichen Ordnung von drei zu erreichen. Bei noch höheren Dimensionen reichte die zur Verfügung stehende Rechenzeit nicht aus um ein solch genaues Gitter zu konstruieren.

Abschließend wird die Konstruktionszeit adaptiver Gitter für unterschiedliche Dimensionen betrachtet und noch einmal das Problem der hohen Anzahl an Unterräumen beleuchtet.

## 6.1 Funktionen mit polynomieller Konvergenzrate

In diesem Abschnitt werden die Eigenschaften der Gitter mit dyadischer und PLUS1-Struktur bei einer Funktion mit polynomielltem Abfall der Fourierkoeffizienten verglichen. Die zu diesem Zweck verwendete Funktion ist

$$f_p(\mathbf{x}) = \bigotimes_{d=1}^D N_p \cdot (\operatorname{sgn}(x_d - \pi) \cdot \sin(x_d))^p \quad (6.2)$$

mit der Signum Funktion

$$\operatorname{sgn}(x) = \begin{cases} 1 & , \text{ für } x > 0 \\ 0 & , \text{ für } x = 0 \\ -1 & , \text{ für } x < 0 \end{cases} \quad (6.3)$$

Der Parameter  $p$  bestimmt die Konvergenzrate. Eine sehr ähnliche Funktion wurde in [GH14] benutzt, um die vorhergesagte Konvergenzrate von verallgemeinerten dünnen Gittern zu bestätigen. Für diese Funktion gilt  $f_p \in \mathcal{H}_{\min}^{\frac{1}{2}+p-\epsilon}$ . Sie ist periodisch auf dem

Gebiet  $[0, 2\pi]^D$  und  $p$  mal stetig differenzierbar. Dies führt zu polynomieller Konvergenz mit der Rate  $\frac{1}{2} + p$ . Dies wurde in Lemma 2.13 hergeleitet. Der Faktor  $N_p$  wird zur Normierung verwendet. Die Funktion hat die Norm  $\|f_p\|_{\mathcal{L}_2} = 1$ .

In den nachfolgenden Plots ist die Funktion für die 4 Parameter  $p = \{1, 2, 3, 4\}$  zu sehen. Auf dem linken unteren und rechten oberen Quadranten hat die Funktion immer positive Werte. Auf den anderen beiden alterniert das Vorzeichen je nachdem ob  $p$  gerade oder ungerade ist. Die Übergänge zwischen den vier Quadranten wird mit steigender Differenzierbarkeit immer glatter. Die Konvergenzrate der Funktion wird durch das Verhalten an den Übergängen bestimmt.

Die Normierung wird unter anderem zu dem Zwecke durchgeführt, dass die Koeffizienten der Basisfunktionen mit steigender Dimension nicht wachsen und erleichtert das Finden einer geeigneten Startkonfiguration für die adaptiven Gitter. Weiterhin wächst die Norm nicht mit steigender Dimension und die benötigte Gittergröße zum Erreichen maschineller Genauigkeit wächst dementsprechend auch nicht überproportional stark an.

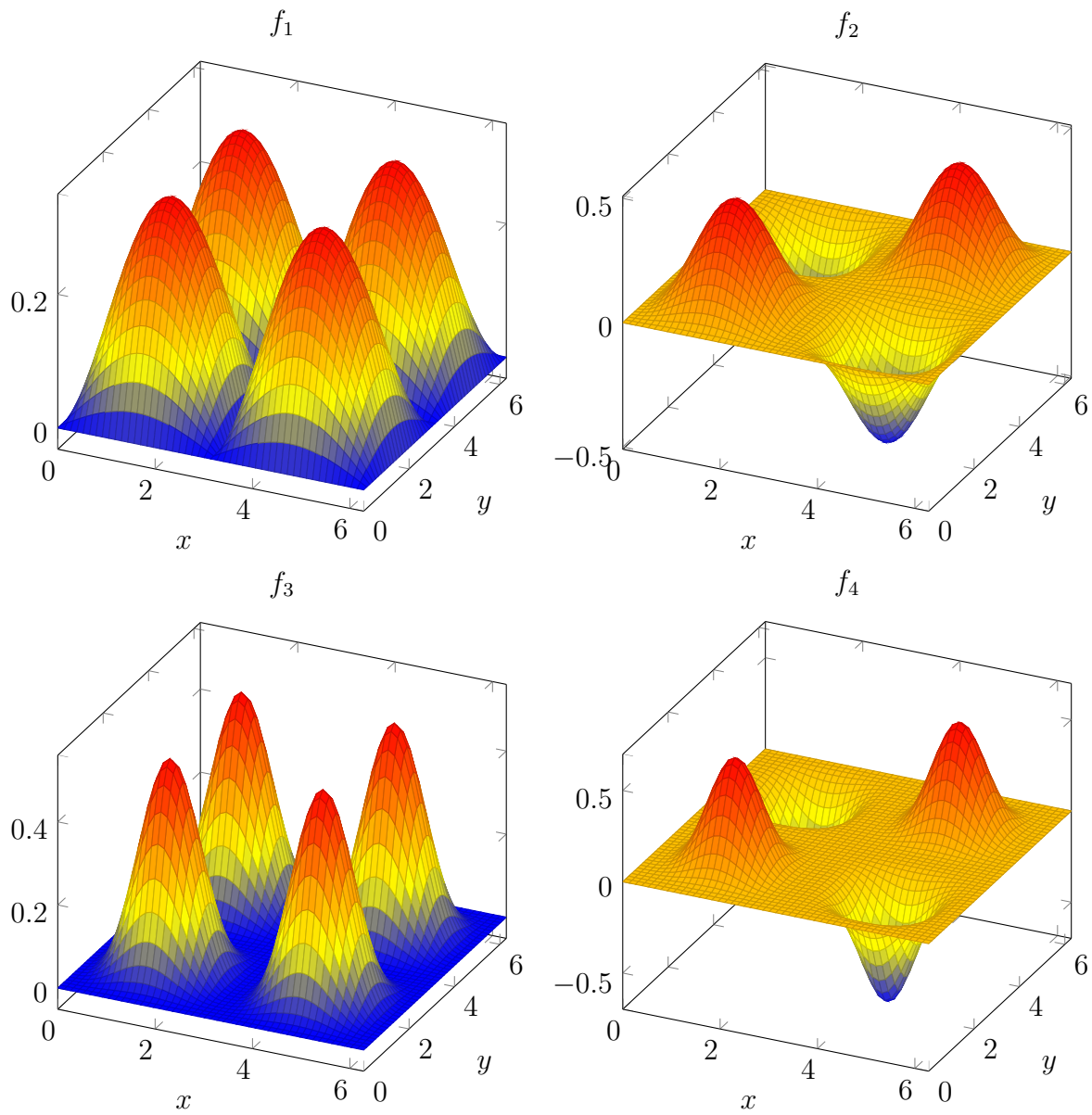


Abbildung 20: Gezeigt ist die Funktion  $f_p(\mathbf{x}) = \bigotimes_{d=1}^D N_p \cdot (\text{sgn}(x_d - \pi) \cdot \sin(x_d)^p)$  für die Parameter  $p = 1, 2, 3, 4$  auf dem Gebiet  $[0, 2\pi]^2$ . Bei dieser Funktion liegt ein polynomieller Abfall der Fourierkoeffizienten vor.



### 6.1.1 Nicht adaptive Gitter

Der erste Vergleich von PLUS1-Gittern mit regulären dünnen Gittern zeigt wie zu erwarten, dass die Konvergenzrate der regulären dünnen Gittern besser ist. Allgemein verbessert sich die Konvergenzrate mit steigender Differenzierbarkeit der Funktion  $f_p$ . Weiterhin ist zu sehen, dass mit größerer Dimensionszahl die Konvergenzrate sinkt. Die Funktion ist so beschaffen, dass erst eine gewisse Menge an Basisfunktionen im Gitter benötigt werden, bevor der relative  $\mathcal{L}_2$ -Fehler kleiner wird. Sobald diese in dem Gitter enthalten ist, tritt die erwartete polynomielle Konvergenzrate zu Tage. Die PLUS1-Gitter erreichen die nötigen Indizes schneller, als die Gitter mit dyadischer Levelstruktur. Dies deutet darauf hin, dass die benötigte Indexmenge, bevor sich eine Fehlerreduktion einstellt, Rautenförmig ist.

Nachfolgend sind die Indexmengen der Basisfunktionen eines nicht adaptiven regulären dünnen Gitter und eines PLUS1-Gitters für zwei und drei Dimensionen zu sehen. Das reguläre dünne Gitter mit Level fünf und Dimension zwei hat 112 Freiheitsgrade, das dünne Gitter mit PLUS1-Struktur hat 66 Freiheitsgrade im Level zehn. Bei dreidimensionalen Gittern steigt die Zahl der Freiheitsgrade auf 272 für ein hyperbolisches Kreuz im Level fünf, bzw. 286 für das PLUS1-Gitter im Level zehn. Die Namensgebenden Formen sind hier deutlich zu erkennen.

Es folgen die Graphen, welche den relativen  $\mathcal{L}_2$ -Fehler gegen die Zahl der Freiheitsgrade auftragen. Gitter mit dyadischer Levelstruktur werden immer mit DG bezeichnet und Gitter mit PLUS1-Struktur entsprechend durch PLUS1. Neben dem relativen  $\mathcal{L}_2$ -Fehler ist immer die Konvergenzrate der letzten fünf Datenpunkte, welche mit linearer Regression berechnet wird, zu sehen. Die in Gleichung (2.66) auf Seite 33 hergeleitete Konvergenzrate für reguläre dünne Gitter von  $p + \frac{1}{2}$ , abzüglich eines logarithmischen Terms für die Zahl der Dimensionen, wird hier in jedem Beispiel in etwa erreicht. Hier zeigt sich außerdem, dass die Konvergenzrate eines PLUS1-Gitters der eines vollen Gitters gleich ist. Die Überlegungen hierzu finden sich am Ende von Kapitel 3.

Da diese Experimente vor allem das zu erwartende Konvergenzverhalten bestätigen sollte, werden keine ausführlichen Untersuchungen von Konstruktionsgeschwindigkeit und Transformationszeit durchgeführt. Die bei einem regulären dünnen Gitter verwendeten Hierarchisierungs, Dehierarchisierungs und Transformationsverfahren sind schneller als die bei einem PLUS1-Gitter verwendeten. Dies bedeutet, dass ein PLUS1-Gitter eine bessere Genauigkeit und Konvergenzrate aufweisen muss, um trotz der aufwendigeren Konstruktion einen Vorteil bei einem Vergleich des Fehlers bezüglich der Laufzeiten zu haben.

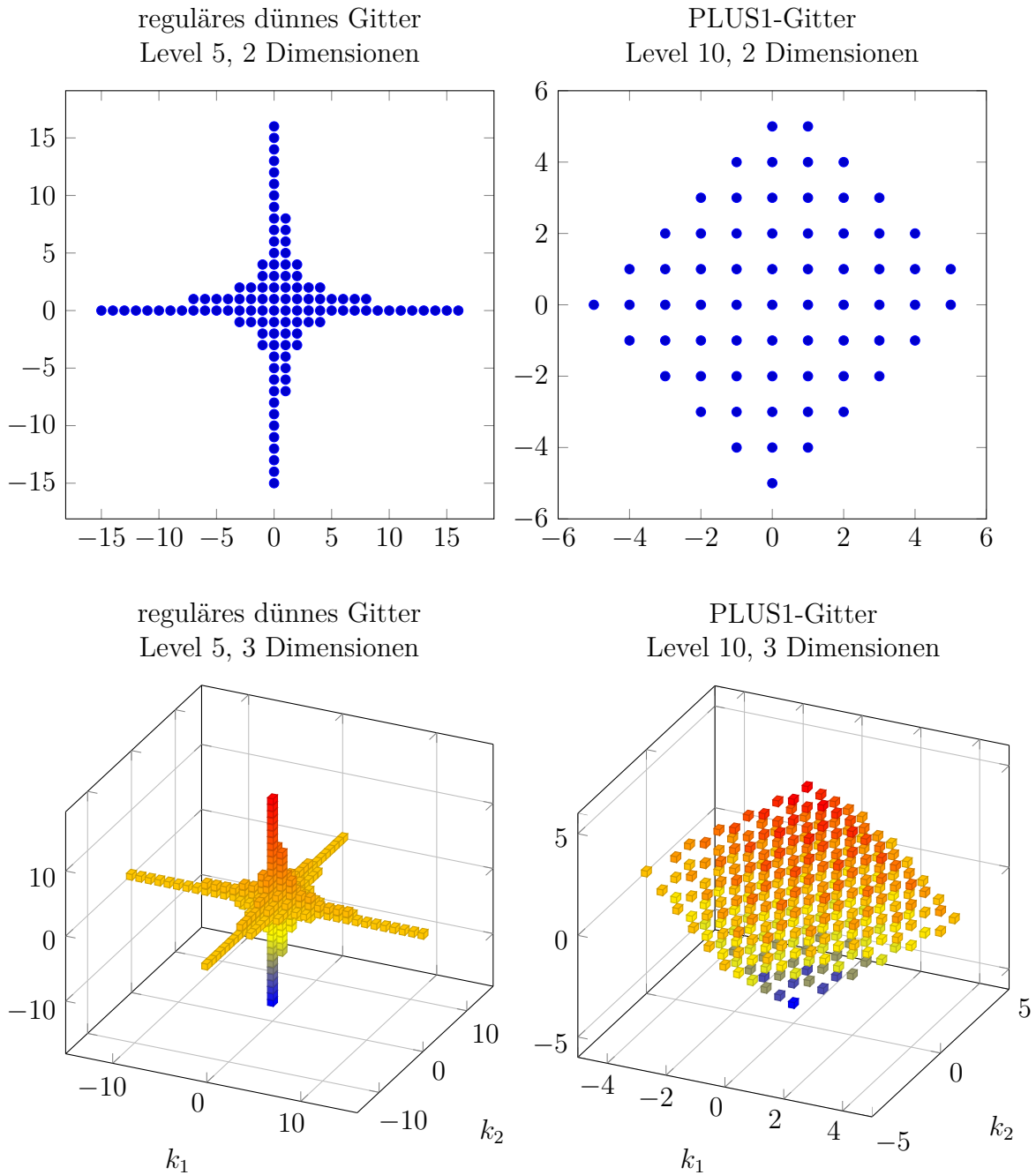


Abbildung 21: Indextmengen der Basisfunktionen nicht adaptiver dünner Gitter. Die linken Grafiken sind Indextmengen bei dyadischer Levelstruktur, die rechten stammen von der PLUS1-Struktur.

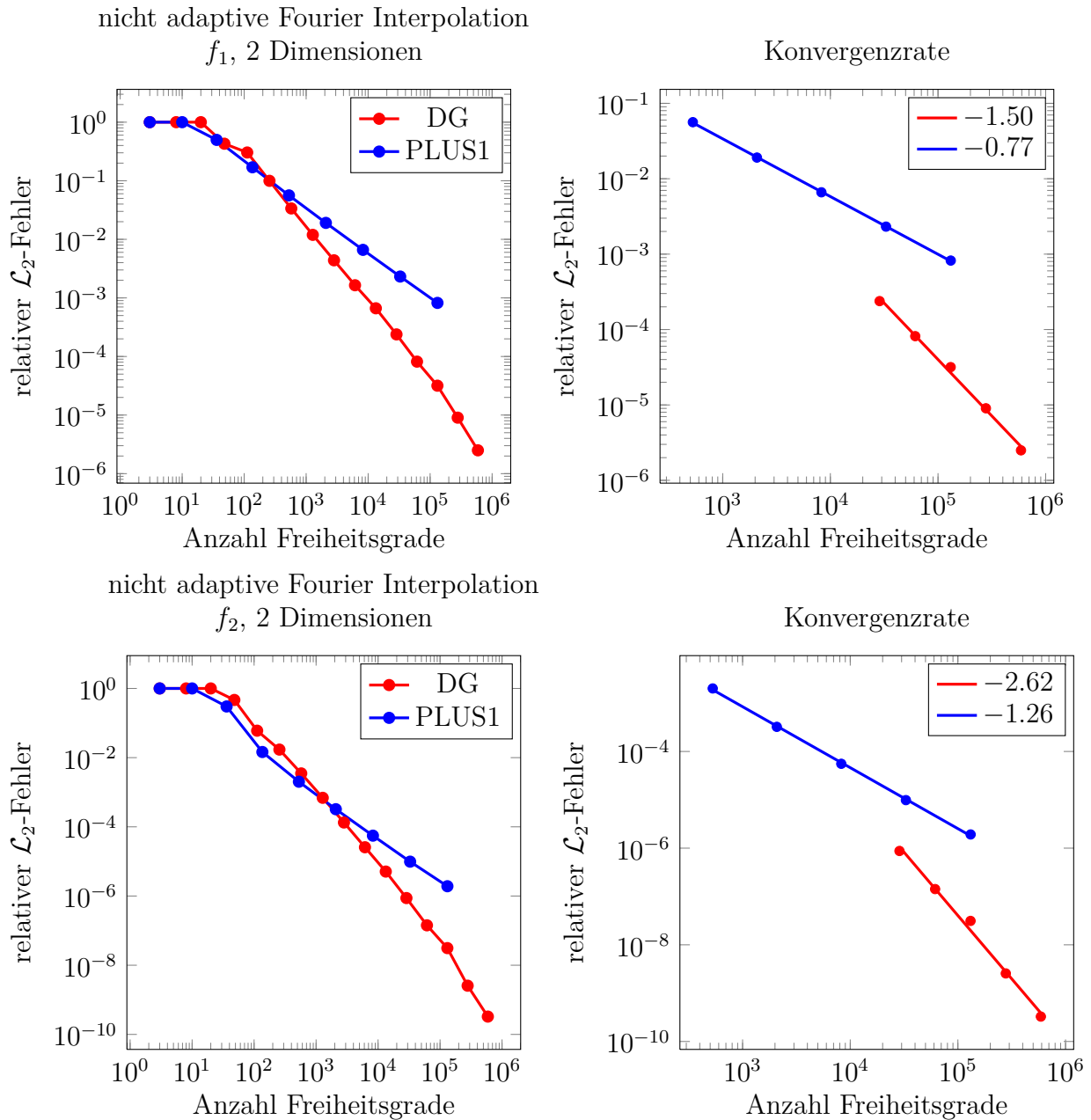


Abbildung 22: Konvergenzverhalten des Interpolanten der Funktionen  $f_p(\mathbf{x}) = \bigotimes_{d=1}^D N_p \cdot (\text{sgn}(x_d - \pi) \cdot \sin(x_d)^p)$  auf einem zweidimensionalen Gitter. Zu sehen ist der relative  $\mathcal{L}_2$ -Fehler sowie die Konvergenzrate in den letzten fünf Datenpunkten für die Parameter  $p = 1$  und  $p = 2$ .

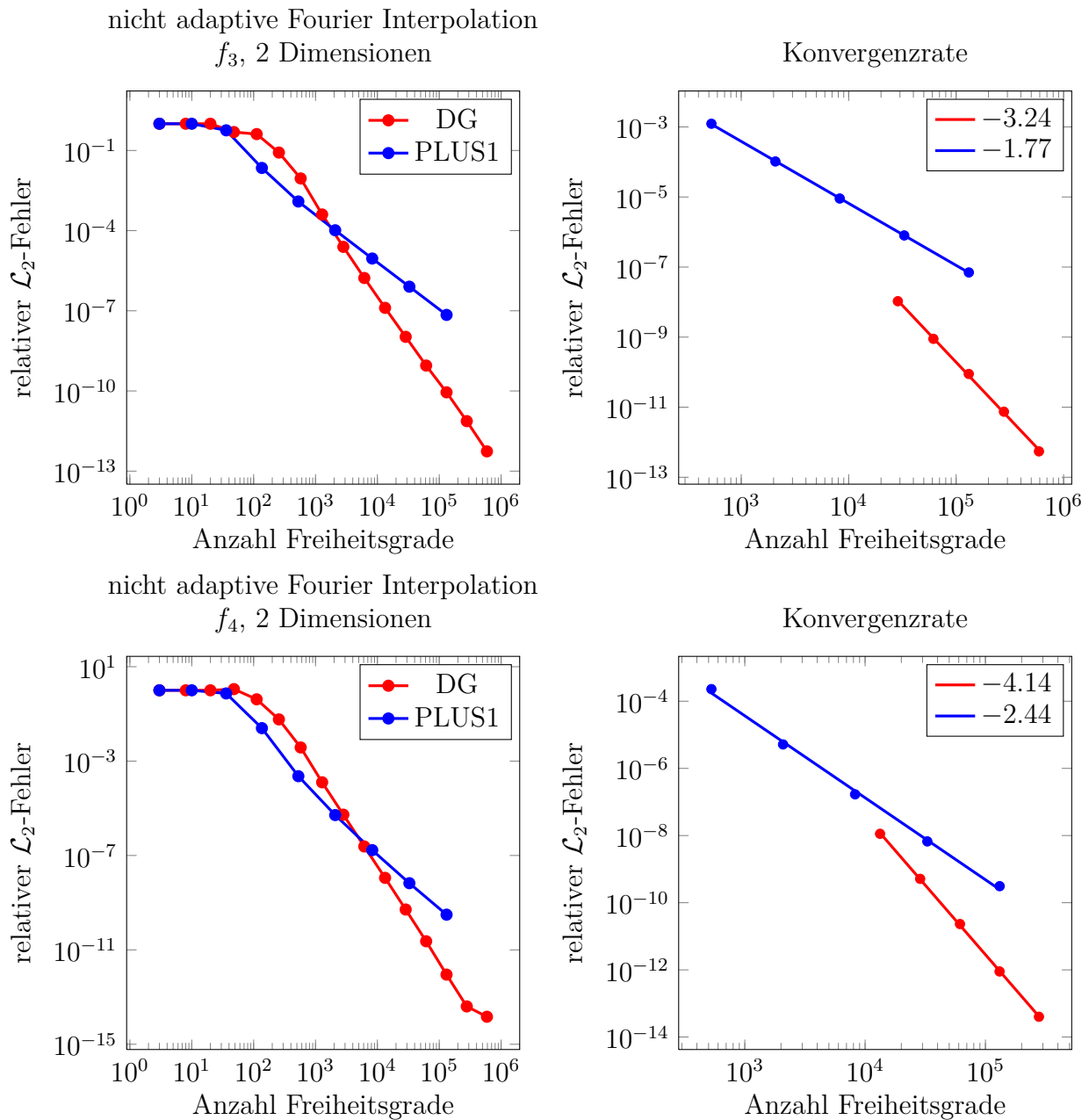


Abbildung 23: Konvergenzverhalten des Interpolanten der Funktionen  $f_p(\mathbf{x}) = \bigotimes_{d=1}^D N_p \cdot (\text{sgn}(x_d - \pi) \cdot \sin(x_d)^p)$  auf einem zweidimensionalen Gitter. Zu sehen ist der relative  $\mathcal{L}_2$ -Fehler sowie die Konvergenzrate in den letzten fünf Datenpunkten für die Parameter  $p = 3$  und  $p = 4$ .

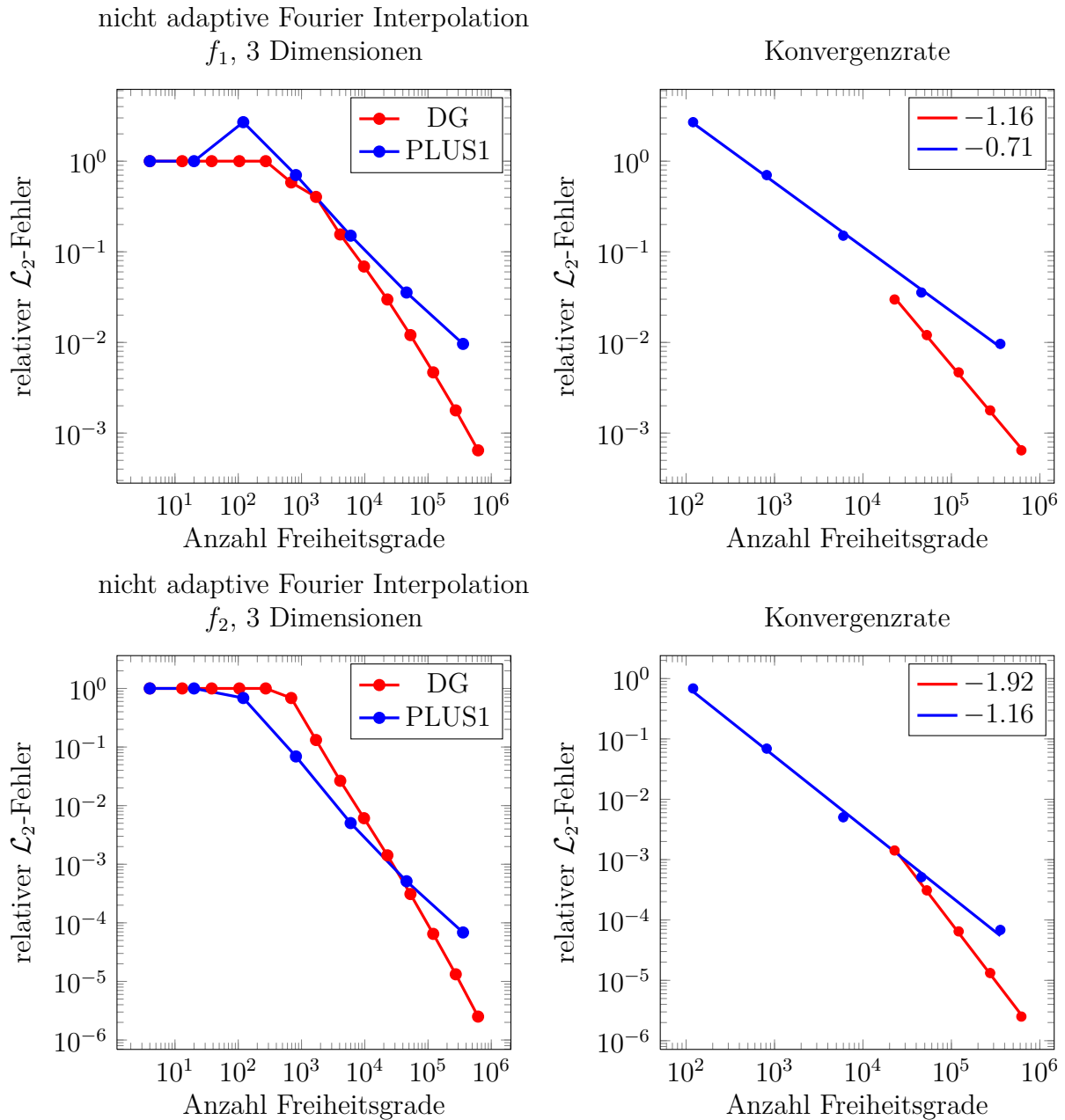


Abbildung 24: Konvergenzverhalten des Interpolanten der Funktionen  $f_p(\mathbf{x}) = \bigotimes_{d=1}^D N_p \cdot (\text{sgn}(x_d - \pi) \cdot \sin(x_d)^p)$  auf einem dreidimensionalen Gitter. Zu sehen ist der relative  $\mathcal{L}_2$ -Fehler sowie die Konvergenzrate in den letzten fünf Datenpunkten für die Parameter  $p = 1$  und  $p = 2$ .

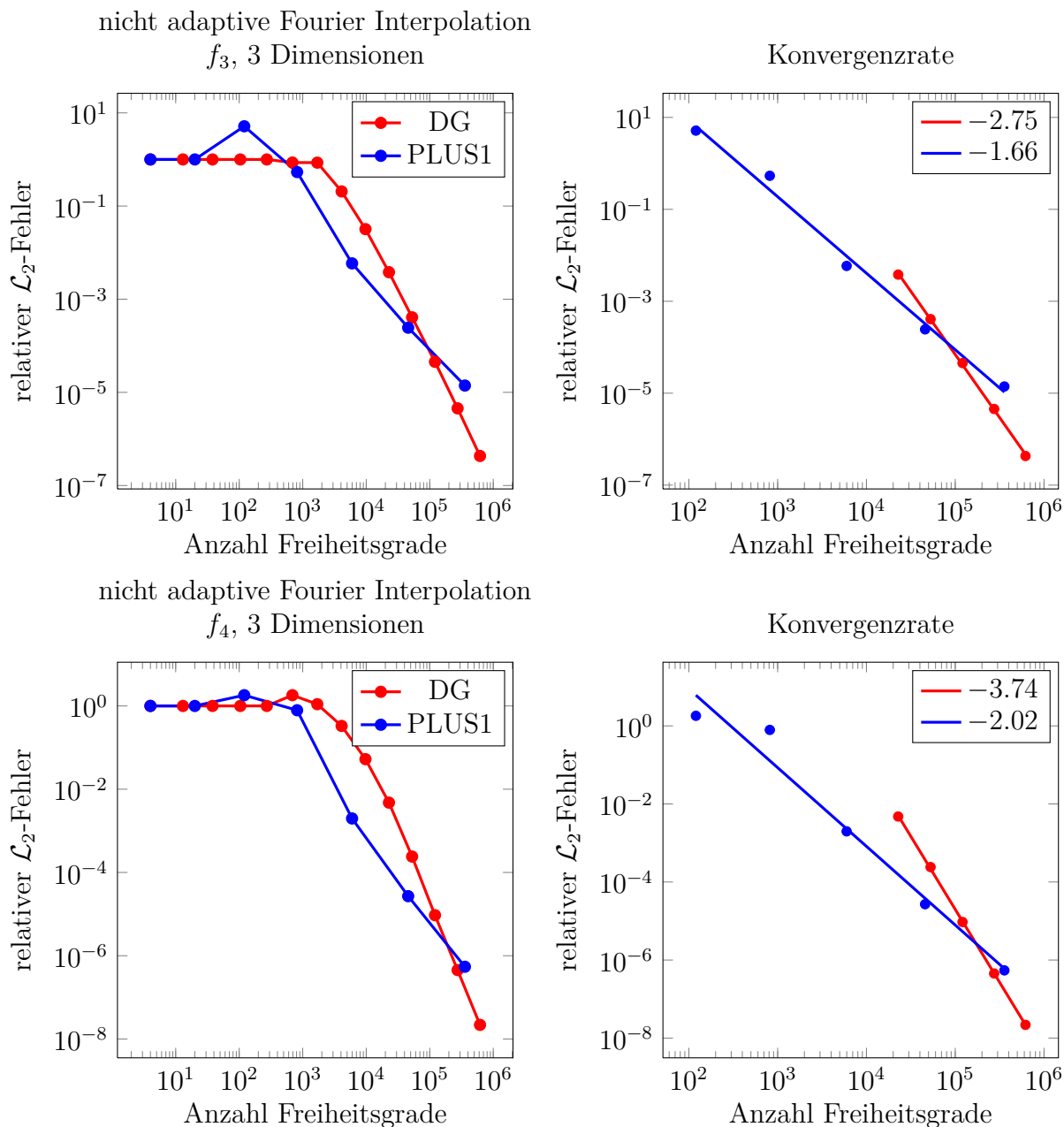


Abbildung 25: Konvergenzverhalten des Interpolanten der Funktionen  $f_p(\mathbf{x}) = \bigotimes_{d=1}^D N_p \cdot (\text{sgn}(x_d - \pi) \cdot \sin(x_d)^p)$  auf einem dreidimensionalen Gitter. Zu sehen ist der relative  $\mathcal{L}_2$ -Fehler sowie die Konvergenzrate in den letzten fünf Datenpunkten für die Parameter  $p = 3$  und  $p = 4$ .

### 6.1.2 Adaptive Gitter

Zur Untersuchung adaptiv konstruierter Gitter bei polynomieller Konvergenz wird die Funktion  $f_0$  interpoliert. Diese wurde gewählt, da ihre Fourierkoeffizienten langsam abfallen und so feinere Abstufungen bei der Konstruktion der Gitter erhalten werden. An dieser Stelle wird vor allem gezeigt, dass der adaptive Algorithmus funktioniert und die zu erwartenden Indexmengen generiert. Weiterhin ist zu beobachten, dass die Konvergenzrate adaptiv konstruierter Gitter bei beiden Levelstrukturen vergleichbar ist.

Abbildung 26 zeigt die Indexmenge adaptiver zweidimensionaler Gittern mit höchstens 32 Stützstellen entlang einer Dimension. Dies bedeutet Level 5 bei dyadischer und Level 31 bei PLUS1-Struktur. Bei beiden Levelstrukturen wird ein hyperbolisches Kreuz erzeugt, was auf Grund des Abfallverhaltens der Funktion zu erwarten ist.

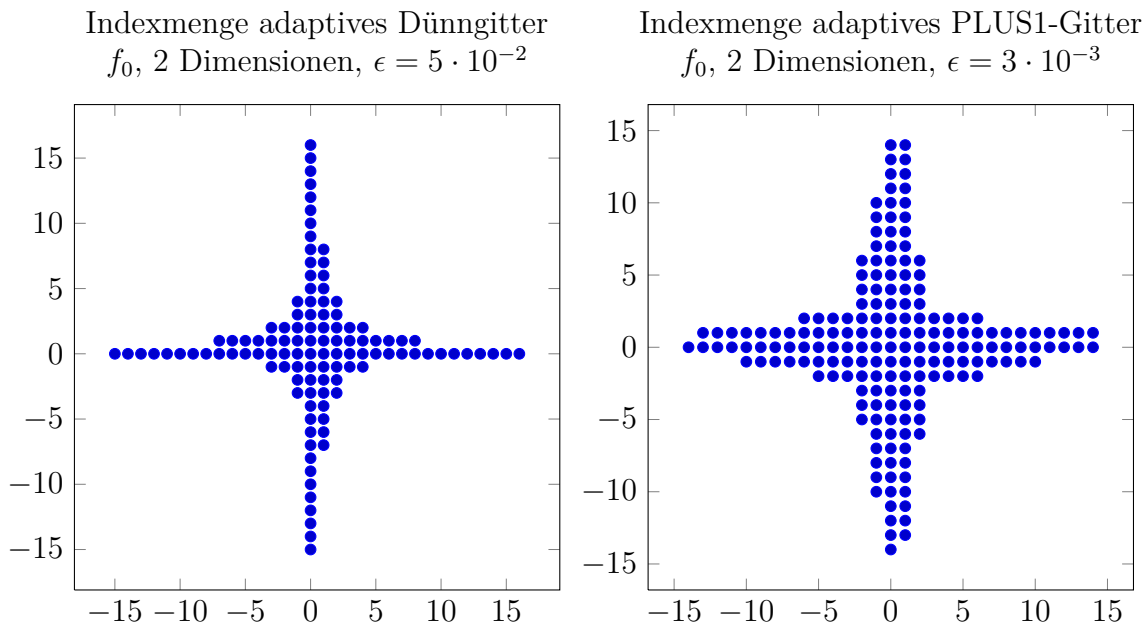


Abbildung 26: Indexmengen der Basisfunktionen bei einem adaptiven Gitter mit dyadischer und PLUS1-Levelstruktur. Es wurde die Funktion  $f_0(\mathbf{x}) = \bigotimes_{d=1}^2 N_0 \cdot (\text{sgn}(x_d - \pi))$  interpoliert

In Abbildung 27 zeigt sich, dass die Konvergenzrate und der Verlauf des relativen  $\mathcal{L}_2$ -Fehlers beider Gitter nahezu identisch sind. Auf Grund des adaptiven Verfahrens und dem polynomiellem Abfallverhalten der Fourierkoeffizienten wird dies auch erwartet.

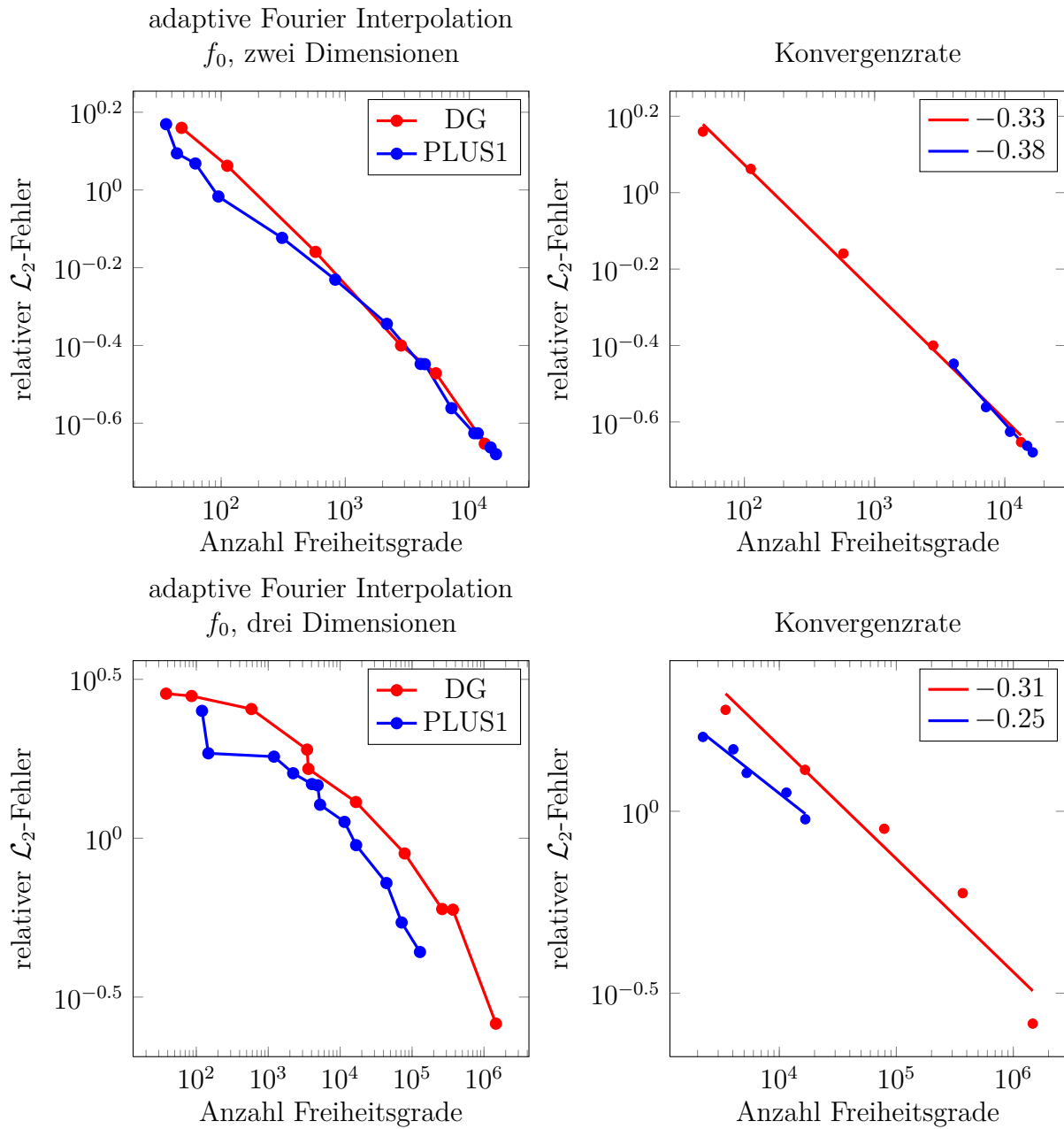


Abbildung 27: Konvergenzverhalten adaptiver Gitter mit dyadischer und PLUS1-Levelstruktur zur Interpolation der Funktion  $f_0(\mathbf{x}) = \bigotimes_{d=1}^2 N_0 \cdot (\text{sgn}(x_d - \pi))$ . Es werden zwei- und dreidimensionale Gitter konstruiert und die Konvergenzrate in den letzten fünf Datenpunkten mit linearer Regression bestimmt.



## 6.2 Funktionen mit exponentieller Konvergenzrate

Für die Experimente mit eine Funktion, bei welcher die Fourierkoeffizienten exponentiell Abfallen, wird die folgende verwendet:

$$f_{D,O}(\mathbf{x}) = \frac{1}{D} \sum_{d=0}^{D-1} e^{\frac{1}{D} \sum_{f=0}^{O-1} \cos(2\pi * x_{d+f \bmod D})} \quad (6.4)$$

Diese Funktion ist eine Summe von Tensorprodukten analytischer periodischer Funktionen auf dem Intervall  $[0, 1]$  und unterliegt somit einem exponentiellem Abfall der Fourierkoeffizienten. Mit den beiden Parametern  $D$  und  $O$  kann die Dimension und endliche Ordnung der Funktion festgelegt werden. Der Wertebereich der Funktion ist  $[\frac{1}{e}, e]$ , was sich ergibt, wenn der Wertebereich des Exponenten untersucht wird. Dies ist bei steigender Dimensionszahl wichtig, da bei der Berechnung des  $\mathcal{L}_2$ -Fehlers anderweitig Probleme mit dem Volumen des Gebietes auftreten. Die Normierung des Exponenten auf den Wertebereich  $[-1, 1]$  sowie der ganzen Funktion auf den Bereich  $[\frac{1}{e}, e]$  sorgt dafür, dass die Koeffizienten der Basisfunktionen mit steigender Dimension nicht wachsen. Außerdem nimmt die Funktion auf dem Punkt  $\mathbf{0}$  einen Funktionswert ungleich 0 an, was für die Konstruktion adaptiver Gittern von Vorteil ist. Ein Startgitter welches nur den Punkt  $\mathbf{0}$  beinhaltet kann als Ausgangskonfiguration in Betracht gezogen werden.

In Abbildung 28 sind die Funktionen  $f_{2,1}$  und  $f_{2,2}$  zu sehen. Diese unterscheiden sich optisch nur durch eine Streckung um den Faktor zwei. Der Unterschied beider Funktionen liegt in den Basisfunktionen, welche zur Interpolation benötigt werden.  $f_{2,1}$  benötigt eine Summe aus eindimensionalen Basisfunktionen.  $f_{2,2}$  hingegen wird durch Basisfunktionen, welche ein Produkt von je zwei eindimensionalen Basisfunktionen sind, interpoliert. Dies hat zur Folge, dass die Indexmengen sehr unterschiedlich sind.

### 6.2.1 Nicht adaptive Gitter

Es werden wieder die gleichen Untersuchungen wie im Abschnitt zuvor mit nicht adaptiven Gittern vorgenommen. Die Funktion  $f_{D,D}$  wird mit zwei bis fünf Dimensionen durch nicht adaptiven Gittern interpoliert. Für die regulären dünnen Gitter mit dyadischer Levelstruktur werden Gitter mit maximalem Level zwölf verwendet. Die PLUS1-Gitter werden mit bis zu 25 Level konstruiert. Zuerst wird das Konvergenzverhalten betrachtet. In Abbildung 31 ist der relative  $\mathcal{L}_2$ -Fehler in Abhängigkeit der Zahl der Freiheitsgrade des Gitterraumes zu sehen. Hier zeigt sich eine deutlich bessere Konvergenz bei der Verwendung von PLUS1-Gittern.

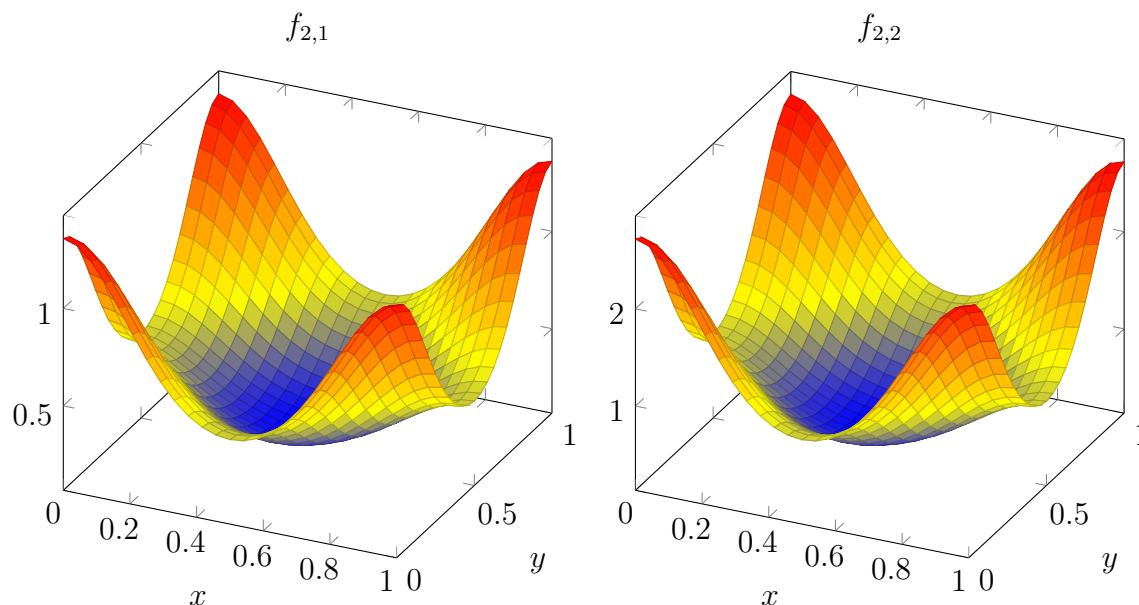


Abbildung 28: Gezeigt ist die Funktion  $f_{2,0}$  mit endlicher Ordnung 1 und 2. Auf den Grafiken sehen beide bis auf eine Streckung um den Faktor 2 identisch aus, der Unterschied liegt in der benötigten Indexmenge zur Interpolation.

Ein besserer  $\mathcal{L}_2$ -Fehler in Abhängigkeit der Zahl der Freiheitsgrade ist schon sehr gut, da die Zeit für eine Funktionsauswertung vor allem von der Zahl der Freiheitsgrade abhängt. Der Interpolant ist für eine Funktion nur einmal zu konstruieren, und kann danach sehr schnell ausgewertet werden. In Abbildung 29 ist die durchschnittliche Zeit für eine Funktionsauswertung zu sehen. Diese wurden auf zwei Arten bestimmt. Einmal wird eine einzelne Auswertung 100 mal iteriert und die durchschnittliche Zeit gemessen. Bei der zweiten Variante wird die Unterraumstruktur nur einmal durchlaufen und jede Basisfunktion an 100 Stellen ausgewertet. Dies ist von Vorteil, falls im Vorfeld mehrere Auswertungspunkte bekannt sind.

Hier zeigt sich, dass es schneller ist, viele Auswertungen gleichzeitig durchzuführen, da nur einmal die Datenstruktur, die die Unterräume verwaltet, durchlaufen wird. Außerdem ist die Zeit für eine Auswertung fast ausschließlich von der Zahl der Freiheitsgrade abhängig. Die roten und blauen Graphen sind die durchschnittlichen Zeiten für eine Auswertung, wenn 100 einzelne Auswertung in Schleife vorgenommen werden. Der blaue Graph ist von einem Gitter mit PLUS1-Struktur. Ein solches Gitter hat somit eine, um mehrere Größenordnungen höhere Zahl an Unterräumen, als ein dünnes Gitter mit dyadischer Struktur. Trotzdem unterscheiden sich die Zeiten nur sehr gering, und wenn man nur einmal durch alle Unterräume läuft, fast gar nicht mehr, was im goldenen und grünen Graphen zu sehen ist.

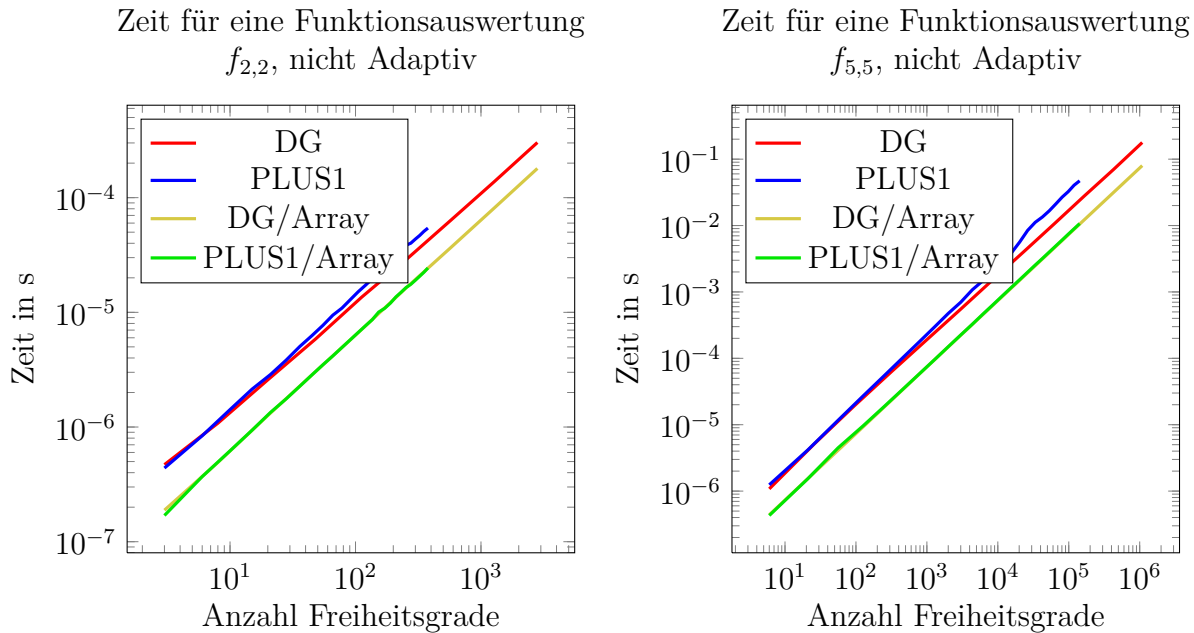


Abbildung 29: Durchschnittliche Zeit für eine Funktionsauswertung. Einmal wird die durchschnittliche Zeit für eine einzelne Auswertung, welche 100 mal iteriert wird, gemessen. Die zweite Möglichkeit ist es 100 Auswertungspunkte als Array zu übergeben und die Datenstrukturen nur einmal abzulaufen. Beide Möglichkeiten werden hier mit regulären dünnen Gittern und PLUS1-Gittern durchgeführt.

Hier wird auch die Frage betrachtet, ob ein PLUS1-Gitter auch eine bessere Konvergenzrate aufweist, wenn der relative  $\mathcal{L}_2$ -Fehler gegen die Konstruktionszeit aufgetragen wird. Es wird wieder die Summe der Zeit zum Aufbau der Unterräume und die Zeit für die Berechnung der Basiskoeffizienten gemessen.

Es zeigt sich, dass das PLUS1-Gitter auch im Verhältnis des relativen  $\mathcal{L}_2$ -Fehlers zur Konstruktionszeit besser als das Dünngitter ist. Hier bilden zweidimensionale Gitter eine Ausnahme, da hier die Zahl der Gitterpunkte klein ist. Dies zeigt die Überlegenheit der beliebigen Levelstruktur bei Funktionen mit exponentieller Konvergenz sehr deutlich.

Abschließend wird in Abbildung 30 die Zeit der einzelnen Arbeitsschritte bei der Konstruktion der Gitter aufgeschlüsselt. Dies ist die Konstruktion der Unterräume und die Berechnung der Basiskoeffizienten. Hier ist der Mehraufwand zu sehen, den ein PLUS1-Gitter gegenüber einem dünnen Gitter für jeden Schritt benötigt. Wie immer bezeichnet DG ein Gitter mit dyadischer Levelstruktur und PLUS1 ein Gitter mit einer Basisfunktion pro Unterraum. Hierin begründet sich der Mehraufwand bei der Konstruktion der Datenstrukturen.

Mit FT wird die Berechnung der Basiskoeffizienten bezeichnet, welche bei einem dyadischen Gitter die schnelle Fouriertransformation, und bei einem Gitter mit beliebiger Levelstruktur eine Matrix-Vektor Multiplikation ist.

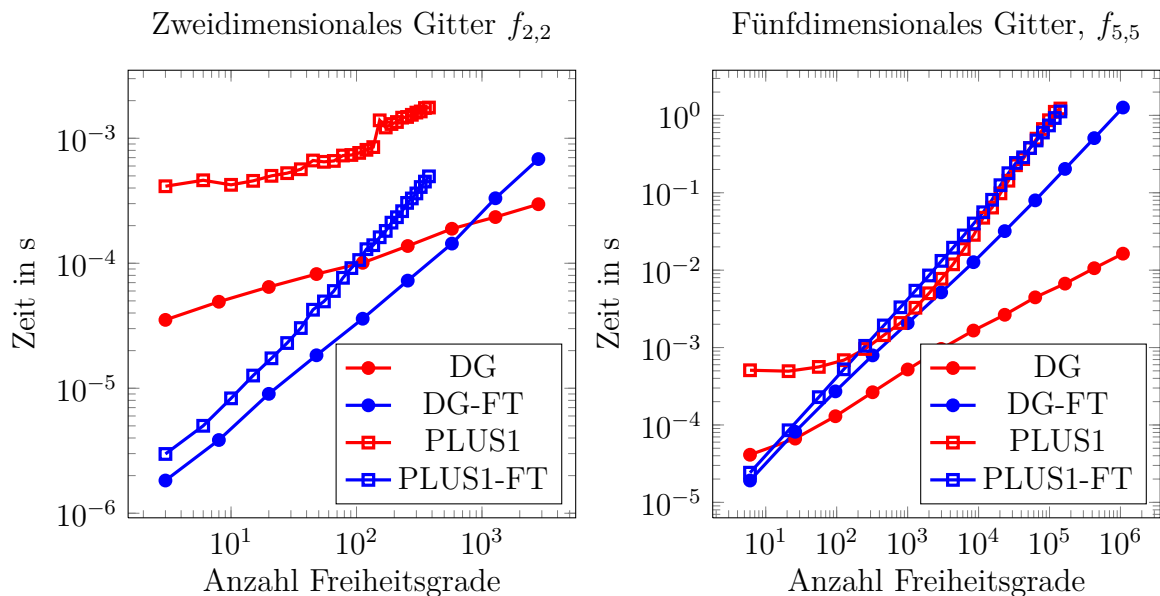


Abbildung 30: Detaillierte Konstruktionszeiten von Gittern zur Interpolation der Funktionen  $f_{2,2}$  und  $f_{5,5}$ . Es werden die Zeiten der einzelnen Arbeitsschritte bei der Konstruktion eines Gitters aufgeschlüsselt. Zu sehen ist die Zeit für die Konstruktion der Unterräume der Gitter mit dyadischer Levelstruktur welche mit DG bezeichnet wird. Weiterhin ist die Zeit zum Aufbau der Unterräume bei einem PLUS1-Gitter zu sehen. Außerdem ist die Zeit zur Berechnung der Basiskoeffizienten aufgetragen, welche mit DG-FT und PLUS1-FT bezeichnet werden.

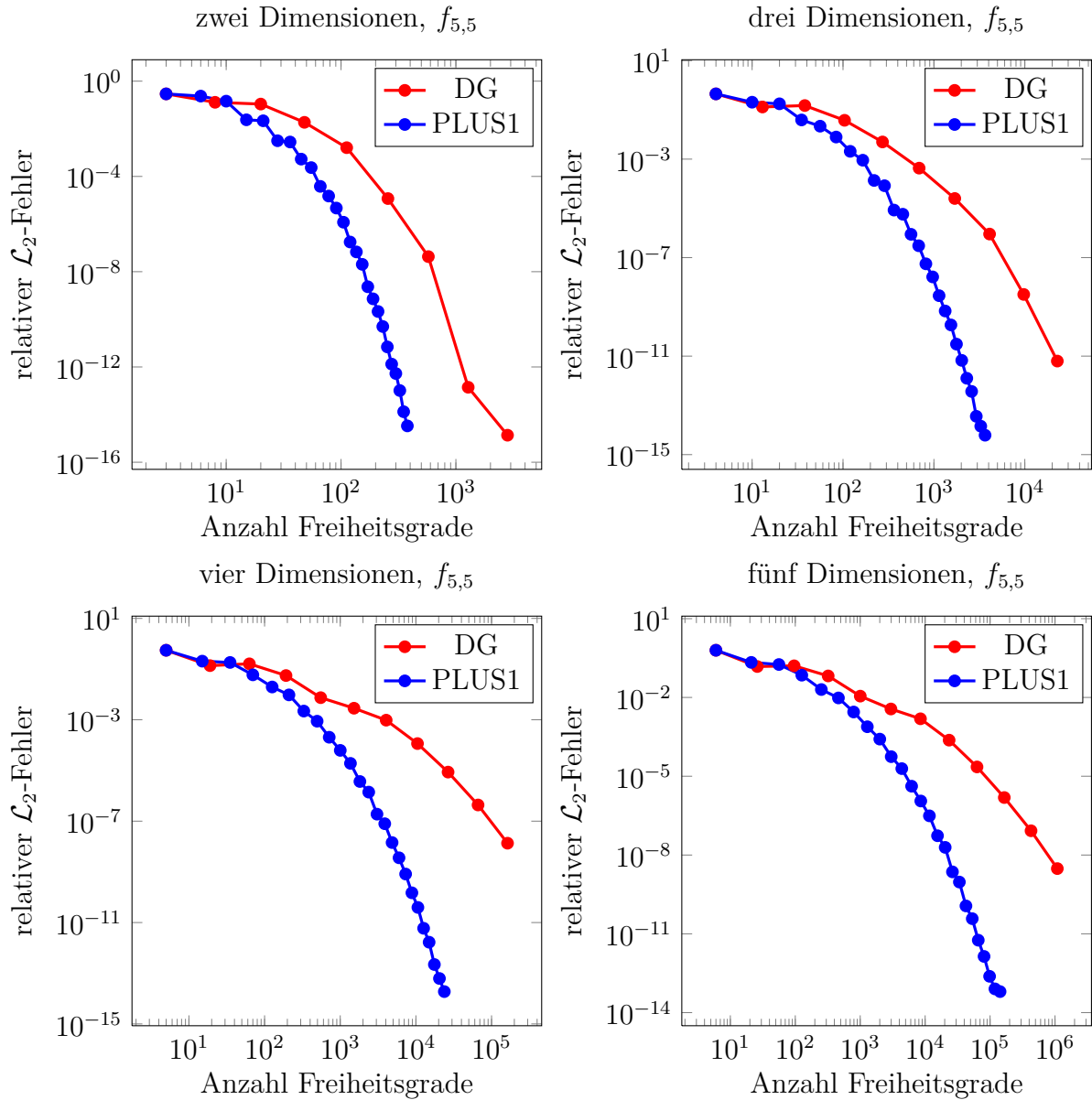


Abbildung 31: Konvergenzverhalten bei der Interpolation einer Funktion mit exponentiellem Abfall der Fourierkoeffizienten. Es wurde reguläre dünne Gitter und PLUS1-Gitter mit zwei bis fünf Dimensionen konstruiert. Das maximale Level bei der Verwendung einer dyadischen Levelstruktur ist 12, bei Verwendung einer PLUS1-Struktur 25.

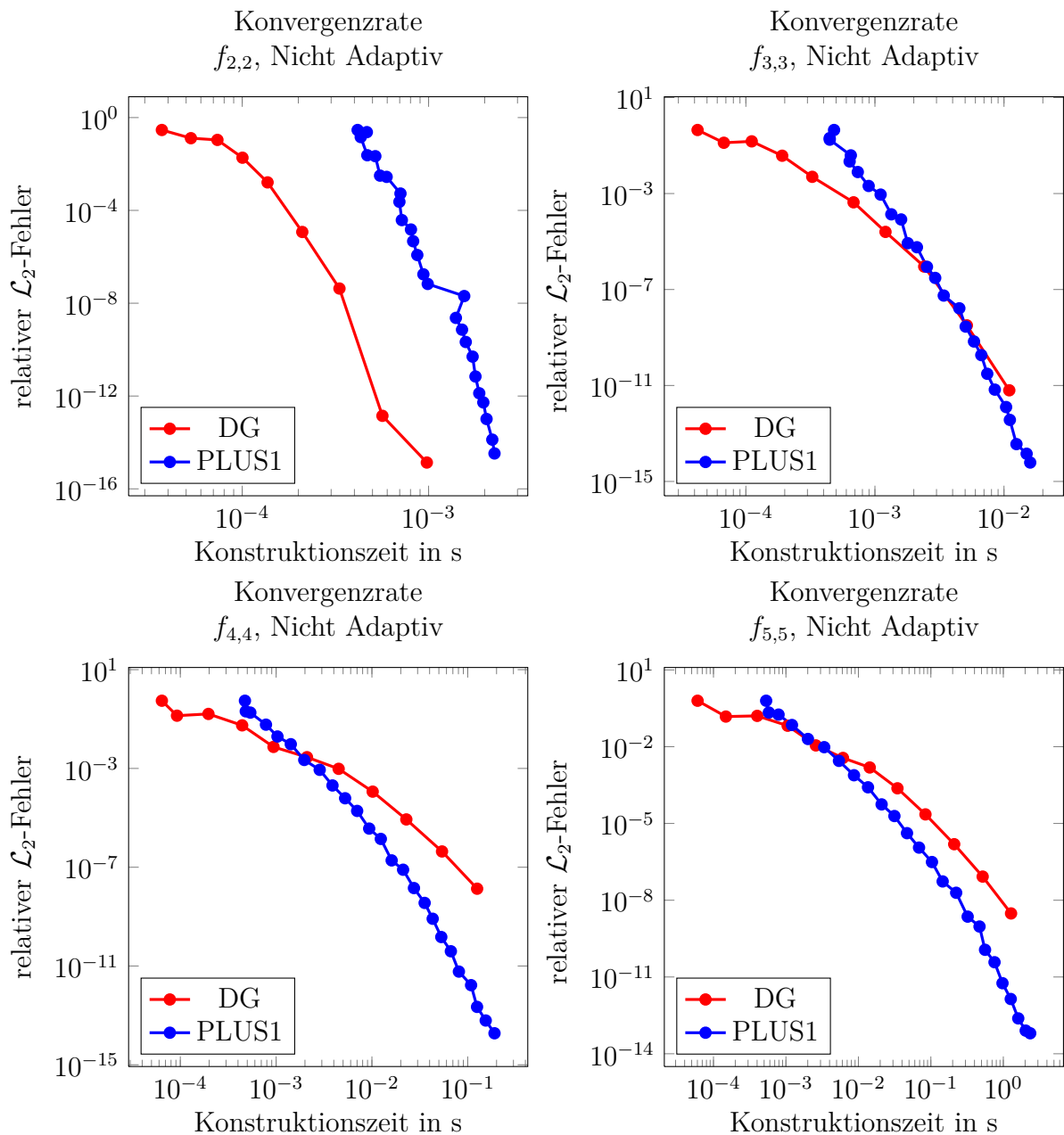


Abbildung 32: Konvergenzrate in Abhängigkeit der Konstruktionszeit. Es ist der relative  $\mathcal{L}_2$ -Fehler gegen die benötigte Zeit zur Konstruktion der Unterräume und Berechnung der Basisfunktionen aufgetragen. Zu sehen ist, dass dyadische Gitter, außer im zweidimensionalen Fall, auch hier schlechter als PLUS1-Gitter sind.

### 6.2.2 Adaptive Gitter

Jetzt werden adaptive Gitter zur Interpolation von analytischen Funktionen untersucht. Es wird erneut die Funktion  $f_{F,O}$  für die Experimente verwendet. Es werden Gitter mit zwei bis fünf Dimensionen und Funktionen mit gewichten endlicher Ordnung der Größe zwei bis fünf benutzt um die Genauigkeit und Geschwindigkeit von Gittern mit dyadischer und PLUS1-Levelstruktur zu vergleichen.

Zuerst werden die Indexmengen der Basisfunktionen von einem adaptiven Gitter mit dyadischer Struktur, und von einem mit einer PLUS1-Struktur gezeigt. Zur Konstruktion der Indexmengen wurde die Funktion  $f_{3,2}$  adaptiv interpoliert. Wie erwartet wird ein Kreuz von zweidimensionalen Ebene als Indexmenge generiert.

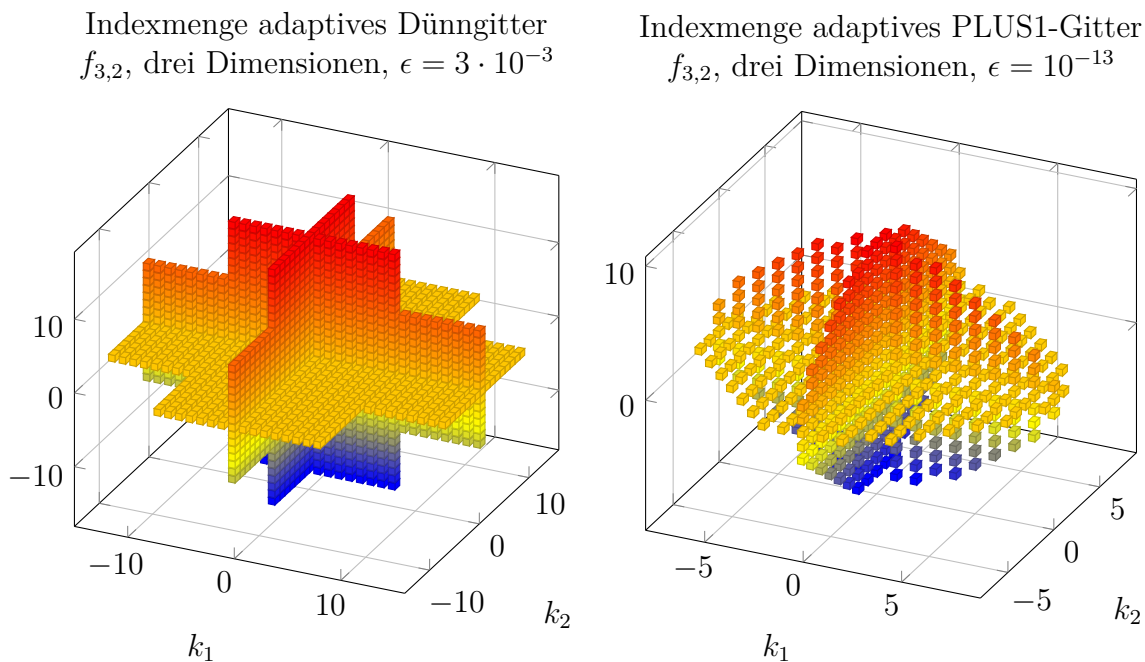


Abbildung 33: Vergleich der Indexmengen eines dreidimensionalen adaptiven Gitters mit dyadischer Levelstruktur links und eines dreidimensionalen adaptiven PLUS1-Gitters rechts. Es wurde die die Funktion  $f_{3,2}$  verwendet, so dass ein Kreuz von Ebenen entsteht.

Um den Vorteil von adaptiven Gittern gegenüber nicht adaptiven Gittern zu sehen, wird zuerst ein Vergleich von beiden Gittertypen in Abbildung 34 gezeigt. Einmal wird ein Gitter mit dyadischer und einmal mit PLUS1-Struktur zur Interpolation der Funktion  $f_{5,3}$  verwendet. Dieses Gitter hat Basisfunktionen der endlichen Ordnung 3. Der Vorteil des adaptiven Algorithmus liegt darin, dass nicht jede 3er Kombination von Koordinatenachsen im Exponenten der Funktion  $f_{5,3}$  auftaucht. Zur Interpolation werden also nur bestimmte dreidimensionale Produkte von Basisfunktionen benötigt. Bei einem nicht adaptiven Gitter mit endlicher Ordnung 3 werden alle möglichen Kombinationen gebildet,

während ein adaptives Gitter die richtigen Kombinationen findet. Die Genauigkeit der adaptiv konstruierten Gitter ist deshalb um mehrere Größenordnungen besser, als die der nicht adaptiven Gitter.

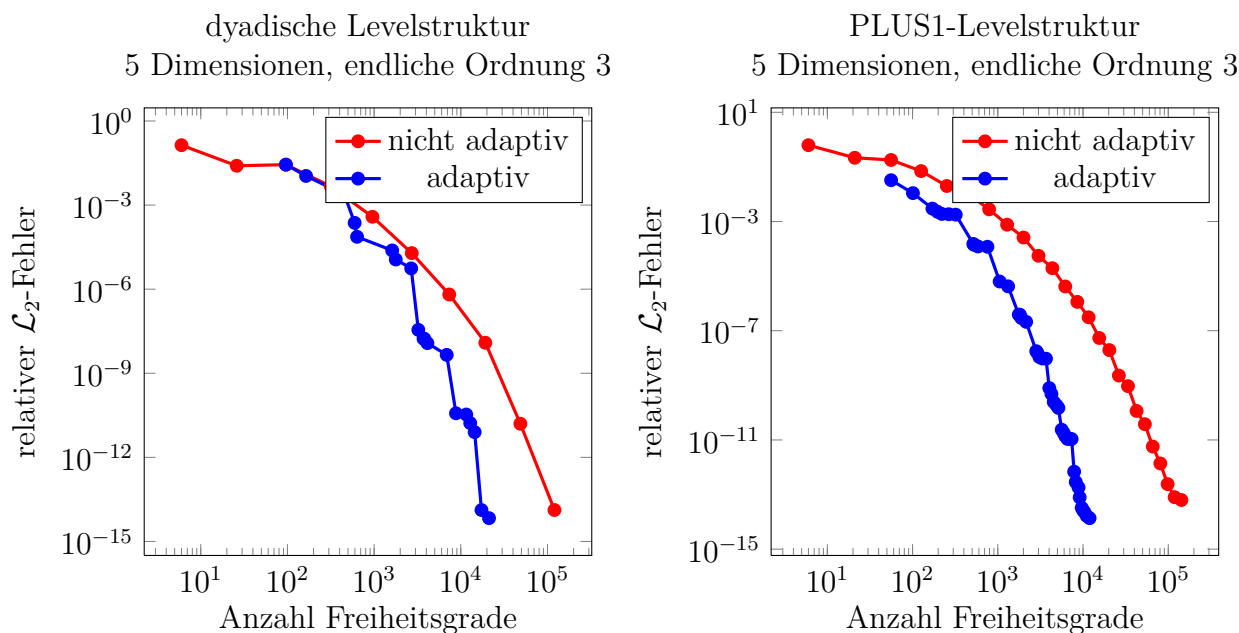


Abbildung 34: Vergleich der Interpolationsgenauigkeit von adaptiven und nicht adaptiven Gittern mit dyadischer und PLUS1-Struktur. Es wird die Funktion  $f_{5,3}$  interpoliert und es zeigt sich dass adaptive Gitter eine wesentlich bessere Konvergenz aufweisen.

Die nachfolgenden Abbildungen sind nach der Größe der Gewichte endlicher Ordnung sortiert, die bei der Funktion  $f_{D,O}$  vorliegen. Zuerst sind in Abbildung 35 vier Gitter mit Dimension zwei bis fünf für Gewichte der Ordnung zwei zu sehen. Die Konvergenzrate dieser Gitter ist sehr ähnlich, was auch zu erwarten ist. Lediglich die Zahl der benötigten Stützstellen um eine bestimmte Genauigkeit zu erreichen nimmt mit steigender Dimensionszahl leicht zu. Dies liegt daran, dass pro Dimension eine weiteres Produkt von zwei Basisfunktionen benötigt wird, um die Funktion nachzubilden.

Da ein ähnliches Verhalten auch bei den Funktionen mit Ordnung drei und vier auftritt, wird hier nur ein Vergleich von vier- und fünf dimensionalen Gittern gezeigt. Wieder ist die Konvergenz von adaptiven Gittern besser als die von nicht adaptiven und die Zahl der benötigten Freiheitsgrade um eine bestimmte Genauigkeit zu erreichen steigt mit zunehmender Dimension etwas an.



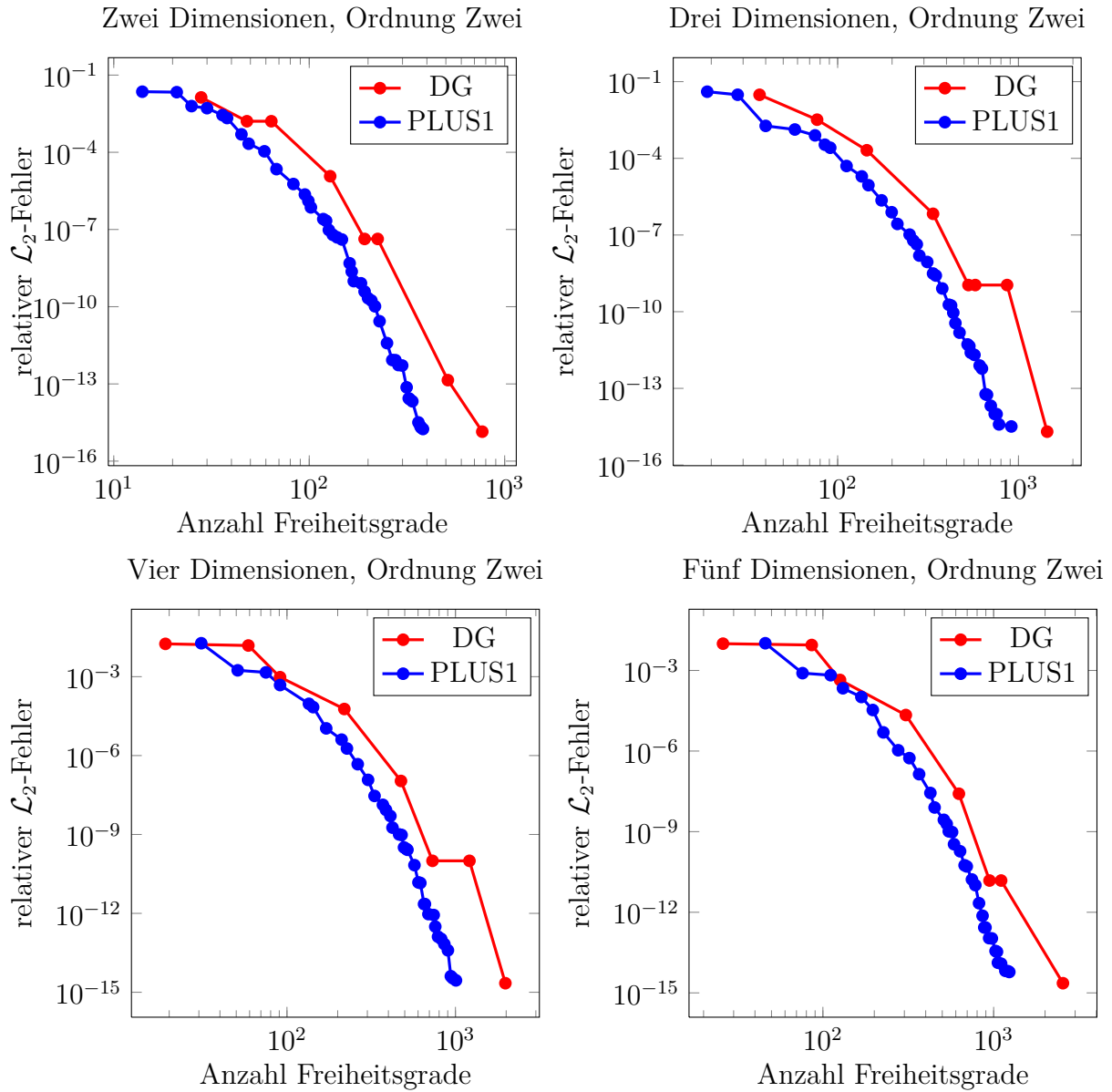


Abbildung 35: Konvergenzverhalten von adaptiven Gittern mit dyadischer und PLUS1-Struktur bei der Interpolation der Funktion  $f_{D,2}$  mit den Parametern  $D = 2, 3, 4, 5$ . Es wird der relative  $\mathcal{L}_2$ -Fehler gegen die Zahl der Freiheitsgrade aufgetragen. Auch ist die erreichte Genauigkeit der PLUS1-Gitter besser als die der regulären dünnen Gitter.

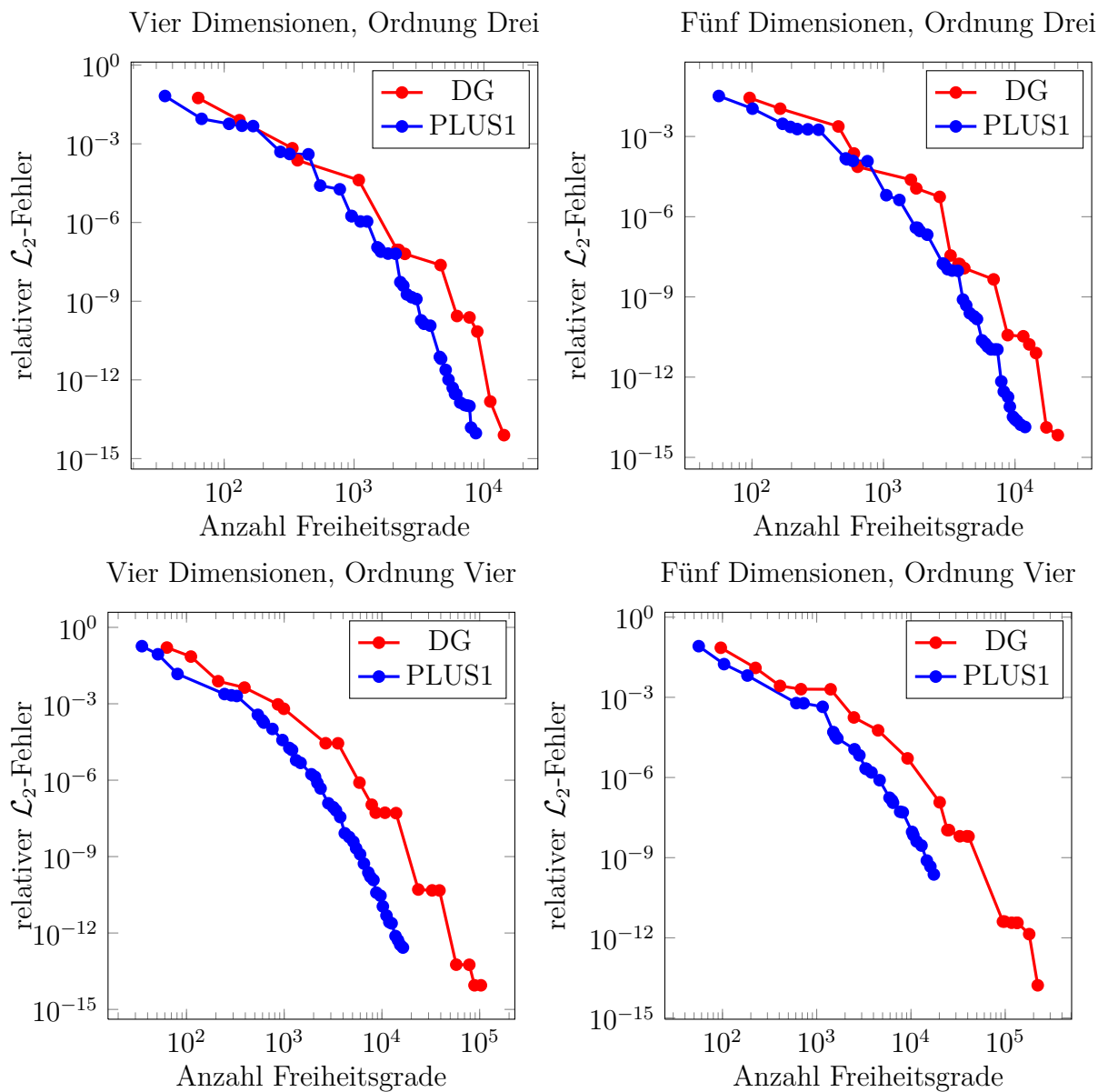


Abbildung 36: Konvergenzverhalten von adaptiven Gittern mit dyadischer und PLUS1-Struktur bei der Interpolation der Funktion  $f_{D,O}$  mit den Parametern  $D = 4, 5$  und  $O = 3, 4$ . Es wird der relative  $\mathcal{L}_2$ -Fehler gegen die Zahl der Freiheitsgrade aufgetragen. Auch ist die erreichte Genauigkeit der PLUS1-Gitter besser als die der regulären dünnen Gitter.

Zum Abschluss dieses Abschnittes wird in Abbildung 37 ein direkter Vergleich von adaptiven und nicht adaptiven Gittern zur Interpolation der Funktion  $f_{5,5}$  gezeigt. Hier ist zu sehen, dass bei dyadischer Levelstruktur adaptive Gitter eine bessere Genauigkeit aufweisen als nicht adaptive. Bei Gittern mit einer PLUS1-Levelstruktur hingegen gibt es kaum einen Unterschied zwischen adaptiv und nicht adaptiv konstruierten Gittern. Dies liegt einfach daran, dass bei glatten Funktionen eine rautenförmige Indexmenge der Fourierbasis zur Interpolation, optimal ist.

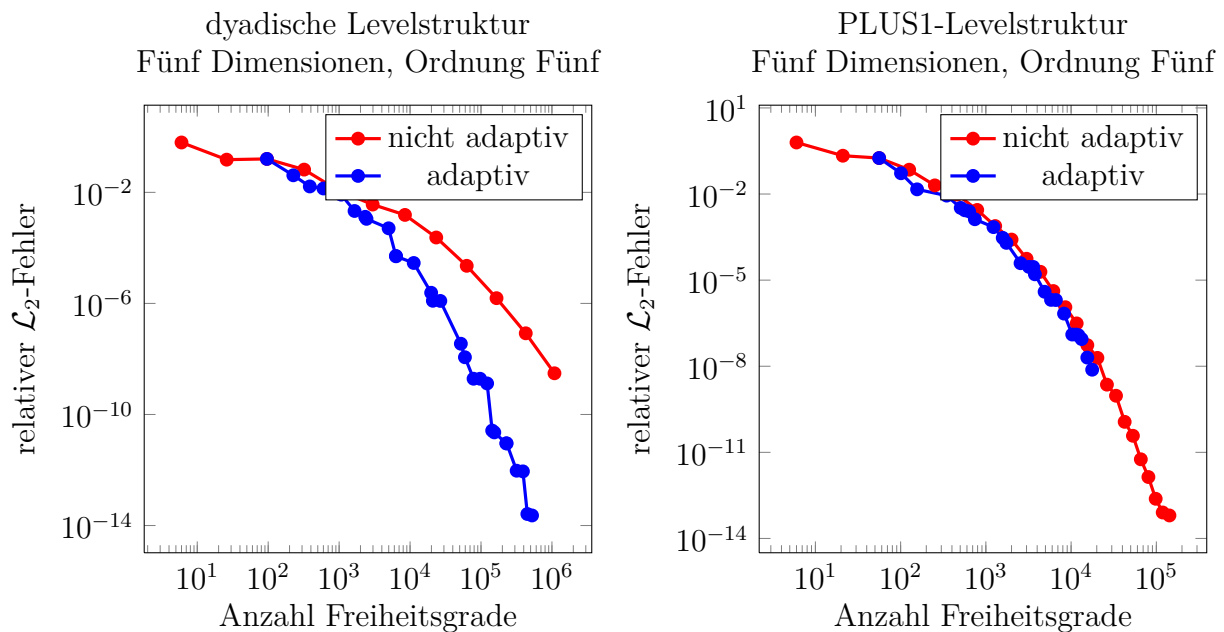


Abbildung 37: Vergleich der Konvergenz von adaptiven und nicht adaptiven dünnen Gittern für dyadische und PLUS1-Struktur. Es wird die Funktion  $f_{5,5}$  interpoliert. Bei der PLUS1-Struktur ist kein signifikanter Unterschied im Verlauf des Fehlers festzustellen. Bei dyadischer Levelstruktur ist hingegen ein deutlicher Unterschied zu sehen.

### 6.3 Hochdimensionale analytische Funktionen mit Gewichten geringer endlicher Ordnung

In der Praxis treten viele hochdimensionale Probleme auf, bei denen die einzelnen Koordinaten nur eine geringe Korrelation aufweisen oder sogar völlig getrennt betrachtet werden können. Beispielsweise treten bei der Berechnung von Pfadintegralen in der Finanzmathematik Funktionen mit geringer endlicher Ordnung aber vielen Dimensionen auf [GG03].

Der letzte Abschnitt dieses Kapitels enthält Konvergenzanalysen bei Funktionen mit endlicher Ordnung eins bis drei für bis zu 500 Dimensionen. Zuerst werden nicht adaptiv konstruierte Gitter betrachtet. Wenn im Vorfeld von einer Funktion bekannt ist, dass die Koordinatenachsen separierbar sind und weiß wie stark die größte auftretende Korrelation zwischen den Koordinaten ist, kann auch ein nicht adaptives Gitter für die Interpolation solcher Funktionen benutzt werden. Der Vorteil von nicht adaptiven Gittern liegt klar in der Konstruktionszeit.

Im vorherigen Abschnitt war zu sehen, dass ein nicht adaptiv konstruiertes PLUS1-Gitter nicht unbedingt schlechter als ein adaptiv konstruiertes ist. Wenn von einer glatten Funktion die endliche Ordnung bekannt ist, und jede mögliche Kombination an Basisfunktionen bis zu der Ordnung benötigt wird um die Funktion zu interpolieren, sollte ein nicht adaptives Gitter zur Interpolation verwendet werden.

Es wird wieder die Funktion  $f_{D,O}$  für die Experimente benutzt. Bei dieser sind nur Achsen gekoppelt, deren Indizes aufeinander folgen. Ein nicht adaptiv konstruiertes Gitter hat also zu viele Punkte, selbst wenn die endliche Ordnung des Gitters der Ordnung der Funktion angepasst wird.

Zuerst wurden die Funktionen  $f_{100,1}$  und  $f_{500,1}$  interpoliert. Hier ist der treppenförmige Verlauf des relativen  $\mathcal{L}_2$ -Fehlers bei dem Gitter mit PLUS1-Levelstruktur zu sehen. Dies liegt, wie in Kapitel 3 erläutert, an der Wahl der Stützstellen. Die nächsten beiden Graphen zeigen den relativen  $\mathcal{L}_2$ -Fehler für die Interpolation der Funktionen  $f_{50,2}$  und  $f_{100,2}$ . Beide Fehler fallen ab einer gewissen Zahl der Freiheitsgrade sehr schnell ab. Der Fehler des PLUS1-Gitters ist allerdings bei gleicher Zahl an Freiheitsgraden um etwa zwei Größenordnungen kleiner.

Abschließend werden adaptive Gitter für die Funktion  $f_{D,O}$  mit Ordnung 2 und 3 und bis zu 100 Dimensionen untersucht. Bei dyadischer Levelstruktur war es bei Funktionen mit der Ordnung 3 nur bei 30 Dimensionen möglich innerhalb der zur Verfügung stehenden Rechenzeit ein Gitter zu konstruieren, welches Maschinengenauigkeit erreicht. Der adaptive Algorithmus braucht bei hohen Dimensionen sehr lange um die Gitter zu konstruieren. Es sollten noch wesentliche Verbesserungen erreicht werden können.

Bei Funktionen der Ordnung zwei ist zu sehen, dass beide Gitter einen sehr ähnlichen Verlauf haben, was auch daran liegt, dass die gezeigten Gitter sehr klein sind und damit bei dyadischer Levelstruktur nur wenig überflüssige Basisfunktionen zur Interpolation verwendet werden. Bei Funktionen mit Ordnung drei hingegen sind die Gitter mit PLUS1-Struktur den Gittern mit dyadischer Levelstruktur auch schon bei kleinen Gittern überlegen. Weiterhin ist das Erreichen sehr hoher Genauigkeiten mit Rechenzeiten unter einer Woche nur bei PLUS1-Gittern möglich. Zur Interpolation der Funktion  $f_{30,3}$  benötigte das Gitter mit dyadischer Levelstruktur  $2^{21}$  Gitterpunkte und wurden fast 11 Tage zur Konstruktion benötigt. Das Gitter mit nur einer Basisfunktion pro Unterraum kommt mit weniger als  $2^{17}$  Gitterpunkte aus und die Konstruktionszeit lag bei 16 Stunden. Dies ist ein deutlicher Erfolg dieser Diplomarbeit.

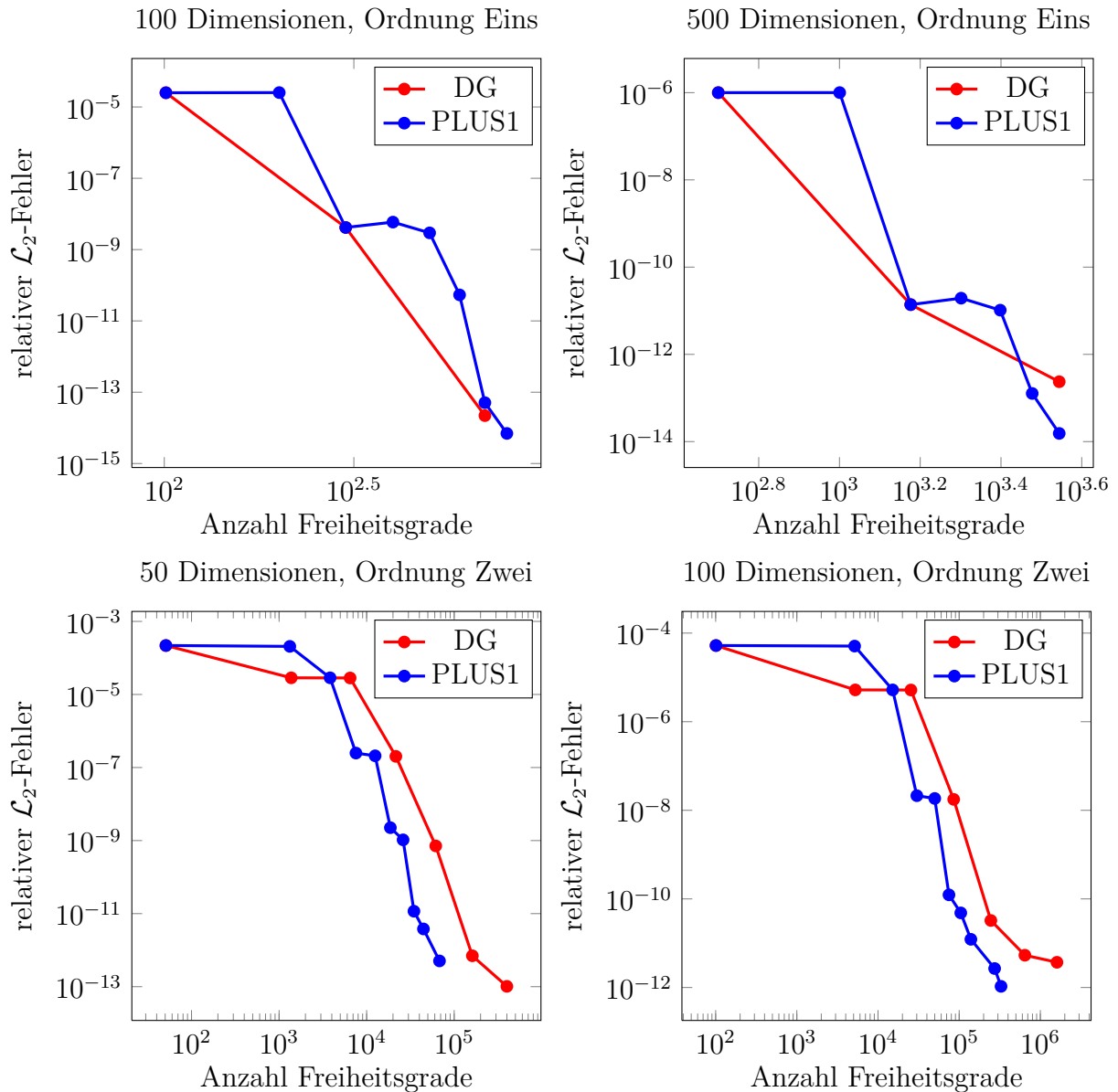


Abbildung 38: Konvergenzverhalten hochdimensionaler analytischer Funktionen mit geringer Ordnung. Zur Interpolation werden nicht adaptive dyadische Gitter und PLUS1-Gitter miteinander verglichen. Es wird die Funktion  $f_{D,O}$  mit den Parametern  $O = 1, 2$  und bis zu 500 Dimensionen interpoliert. Bei endlicher Ordnung Eins fällt die Treppenstruktur wieder auf und bei einer Ordnung von Zwei ist die Überlegenheit der PLUS1-Gitter zu sehen.

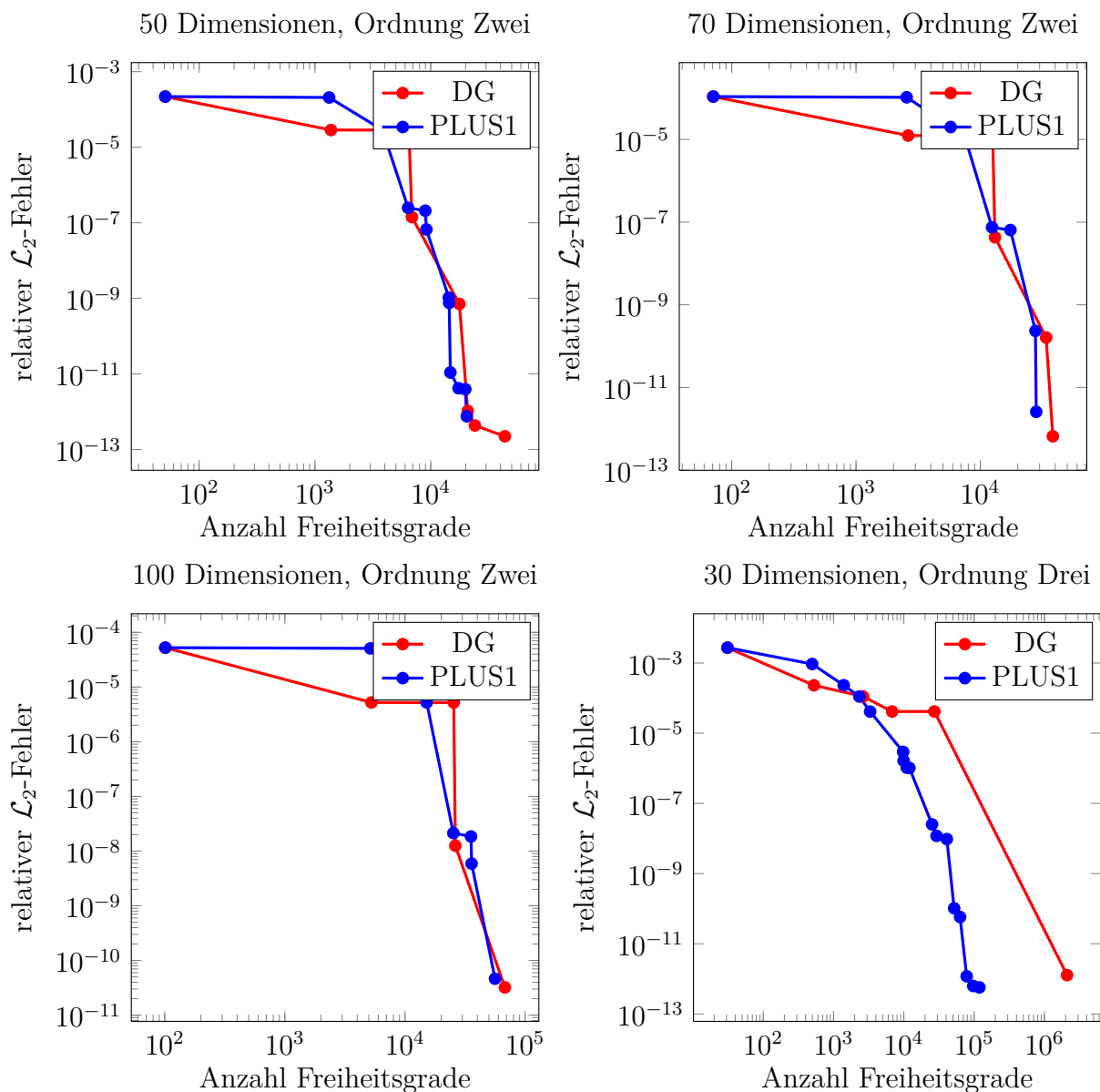


Abbildung 39: Konvergenzverhalten hochdimensionaler analytischer Funktionen mit geringer endlicher Ordnung. Zur Interpolation werden adaptive dyadische Gitter und PLUS1-Gitter miteinander verglichen. Es wird die Funktion  $f_{D,O}$  mit den Parametern  $O = 2, 3$  und bis zu 100 Dimensionen interpoliert. Bei der Funktion  $f_{30,3}$  zeigt sich der Vorteil der beliebigen Levelstruktur sehr deutlich. Es werden mit wesentlich weniger Gitterpunkten deutlich bessere Ergebnisse erzielt.

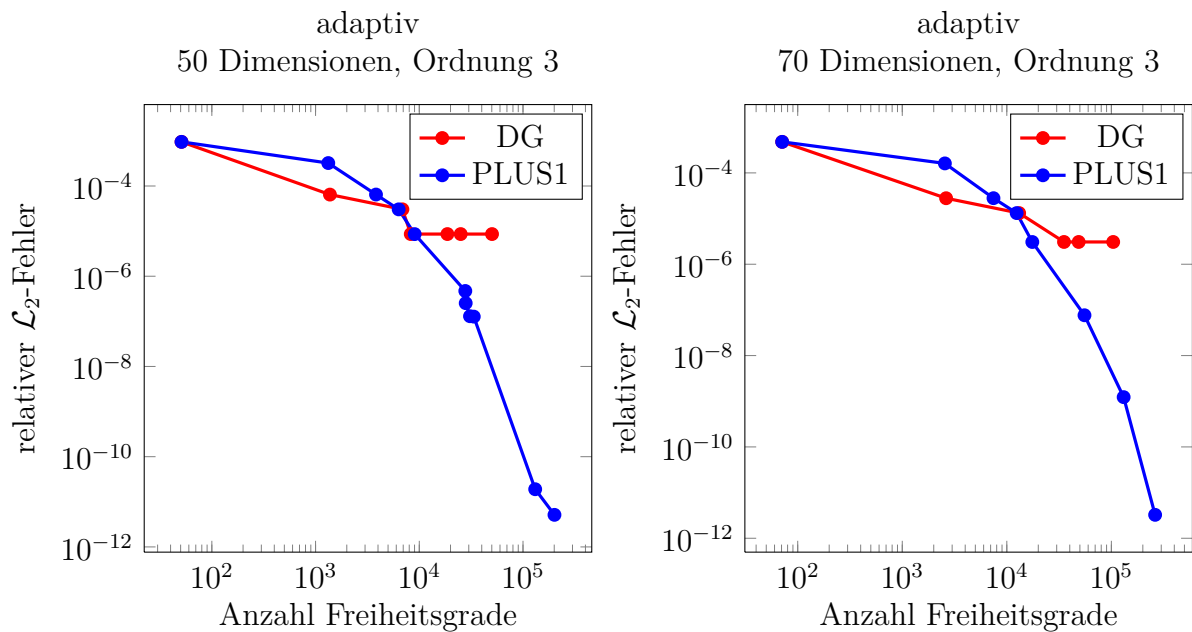


Abbildung 40: Konvergenzverhalten hochdimensionaler analytischer Funktionen mit geringer Ordnung. Zur Interpolation werden adaptive dyadische Gitter und PLUS1-Gitter miteinander verglichen. Es wird die Funktion  $f_{D,3}$  mit den bis zu 70 Dimensionen interpoliert. Auch hier zeigt sich der Vorteil der PLUS1-Gitter deutlich. Innerhalb der zur Verfügung stehenden Rechenzeit konnten keine dyadischen Gitter mit einer Genauigkeit im Bereich der maschinell bedingten Grenze erstellt werden.

## 6.4 Konstruktionszeit adaptiver Gitter

Die letzten Untersuchungen gelten der Konstruktionszeit adaptiver PLUS1-Gitter. Diese ist sehr hoch, da der Algorithmus für dyadische Gitterstrukturen mit vielen Basisfunktionen pro Unterraum entwickelt wurde. Es werden die Konstruktionszeiten für vier Gittertypen gezeigt. Als Funktion wurde wieder  $f_{D,O}$  verwendet und es wurden Gitter mit 2, 5, 30 und 100 Dimensionen konstruiert. Die Gitter mit zwei und fünf Dimensionen haben eine Ordnung, welcher der Dimension der Funktion entspricht. Bei 30 Dimensionen wurde die endliche Ordnung drei verwendet, und bei 100 Dimensionen hatte die untersuchte Funktion die Ordnung zwei.

Es ist zu sehen, dass die Konstruktionszeit bei beiden Gittertypen für hohe Dimensionen sehr hoch ist. Weiterhin ist der Einfluss der Anzahl der Unterräume zu sehen. Bei Gittern mit geringer Dimension ist die Zahl der Unterräume auch schon bei wenig Freiheitsgraden sehr unterschiedlich. Bei Gittern hoher Dimension laufen die Graphen erst bei höherer Stützstellenzahl auseinander. Dies hat den Grund, dass erst bei steigendem Level der Unterräume des Gitters die Zahl der Basisfunktionen pro Unterraum ansteigt.

Hier zeigt sich wieder, dass der adaptive Algorithmus verbessert werden muss. Im nächsten Kapitel werden im Ausblick einige Ideen hierzu gegeben.



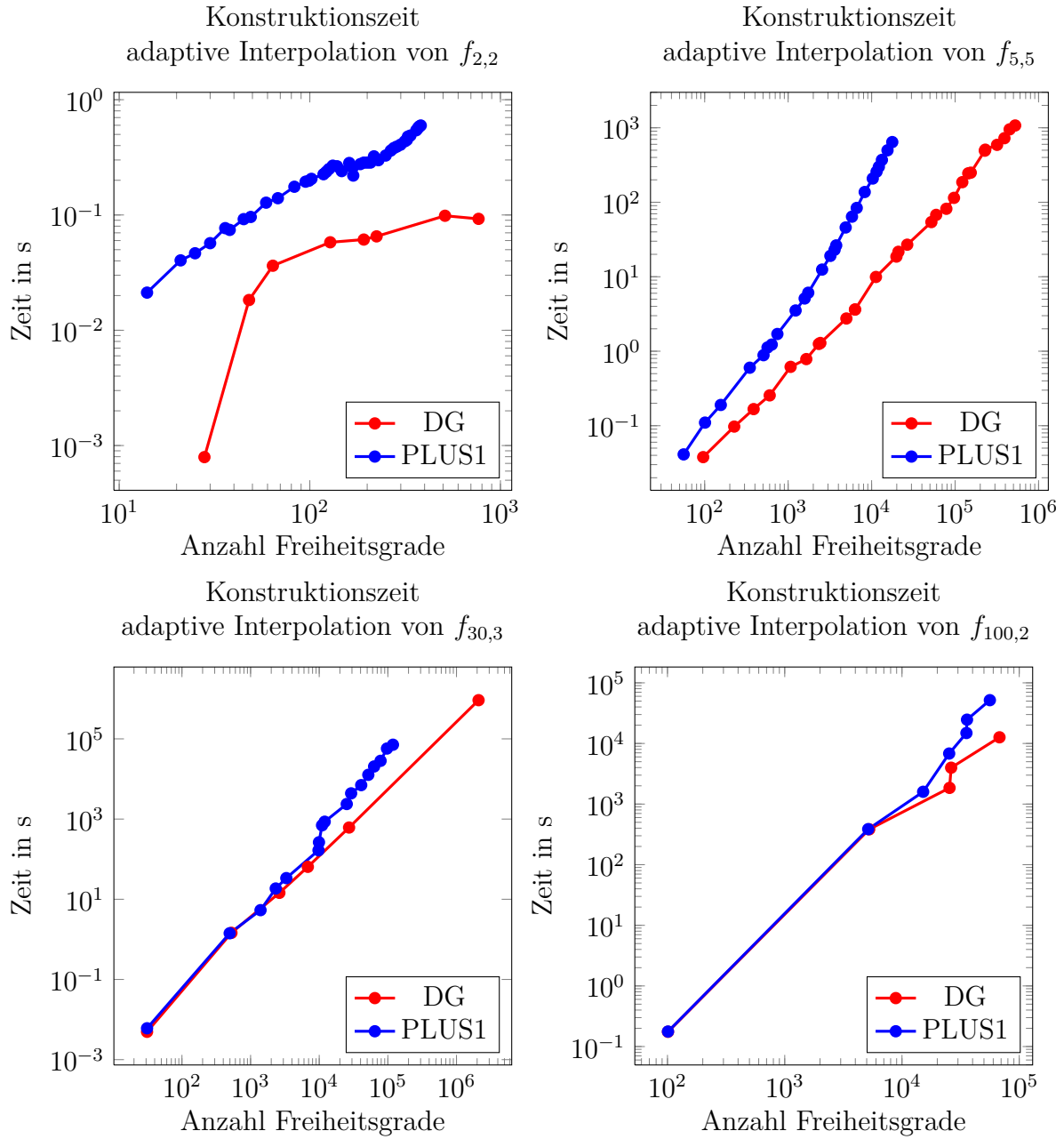


Abbildung 41: Vergleich der Konstruktionszeiten von adaptiven Gittern mit dyadischer und PLUS1-Levelstruktur. Hier zeigen sich die Schwächen des adaptiven Verfahrens bei hoher Anzahl von Unterräumen sehr deutlich.



# 7 Schlussbemerkungen

## 7.1 Zusammenfassung

In dieser Arbeit wurde eine weitere Verallgemeinerung dünner Gitter eingeführt und implementiert. Bisher war es möglich Gitter zu konstruieren, bei denen die Auswahl der Unterräume nur der Bedingung genügen musste, dass die zugehörige Indexmenge monoton ist. Basierend auf den theoretischen Grundlagen verallgemeinerter dünner Gitter und der Hyperbolic Cross Fast Fourier Transform Bibliothek, welche von Dr. Jan Hamaekers programmiert und am Institut für Numerische Simulation weiterentwickelt wurde, ist in dieser Arbeit eine Klasse dünner Gitter implementiert worden, bei denen die Levelstruktur und Position der Stützstellen der eindimensionalen Interpolationsregeln frei wählbar ist.

Diese verallgemeinerten dünnen Gitter mit beliebiger Levelstruktur zeichnen sich dadurch aus, dass als Indexmenge der Basisfunktionen jede beliebige monotone Menge in Frage kommt. Die verbleibenden Einschränkungen sind durch die Verschachtelung der Stützstellen und die Tensorproduktkonstruktion des Gitters gegeben.

Zuerst wurde der vorhandene Code überprüft und ein Vergleich mit alternativen Bibliotheken erstellt. Hierbei wurde sicher gestellt, dass das Verhalten des bisherigen Programmes sich von anderen Programmen nicht merklich unterscheidet. Es ist überprüft worden dass die verwendeten Algorithmen funktionieren und sowohl die Laufzeit, als auch Genauigkeit der erstellten Gitter vergleichbar und in den einigen Fällen sogar besser als die anderer Programme sind.

Der erste Schritt zur Weiterentwicklung der dünnen Gitter war es, den verwendeten Algorithmus zu untersuchen, die Teile zu überarbeiten und neu zu entwickeln, die sich für beliebige Levelstrukturen nicht geeignet haben. Hierbei wurde festgestellt, dass die Operationen zur Berechnung der Basisfunktionen anders gelöst werden mussten. Die eindimensionalen Transformationsregeln waren bisher schnelle Fouriertransformationen. Die Beliebigkeit der Levelstruktur warf das Problem auf, dass die Zahl der Stützstellen nicht mehr Zweierpotenzen sind. Und die Einschränkung dass die Stützstellenmengen der Level verschachtelt sind führt zwangsläufig zu nicht äquidistanten Stützstellen. Dies ist der Grund, weshalb sowohl die schnelle, als auch die diskrete Fouriertransformation nicht mehr benutzt werden konnten.

Es wurde der Ansatz gewählt die Basiskoeffizienten durch das Lösen linearer Gleichungssysteme zu berechnen. Die resultierende Matrix ist eine Vandermonde Matrix mit

Exponentialfunktionen. Für diese ist die Unisolvenz bekannt und da die auftretenden Gleichungssysteme klein sind ist die Kondition dieser Matrizen kein Problem.

Als nächstes musste die Konstruktion einer hierarchischen Basis überdacht werden. Bei einer dyadischen Levelstruktur ergibt sich eine hierarchische trigonometrische Basis als Differenz je zweier Exponentialfunktionen. Diese Möglichkeit besteht bei beliebiger Levelstruktur nicht und auch hier wurde wieder der Ansatz gewählt die Basisfunktionen über ein lineares Gleichungssystem zu berechnen. Bei diesem Verfahren traten die gleichen Matrizen auf, die schon bei der Berechnung der Basiskoeffizienten verwendet wurden. Auch hier erweist sich die Unisolvenz der Matrix und die Größe der auftretenden Gleichungssysteme als Vorteil.

Um Rechenzeit zu sparen wurde eine Möglichkeit implementiert, die Matrizen, die in den linearen Gleichungssystemen zur Berechnung der Basiskoeffizienten auftreten, im Vorfeld zu invertieren und abzuspeichern. Genau so wurde auch die Hierarchisierungs- und Dehierarchisierungsmatrix im Vorfeld berechnet und abgespeichert. Auf diese Weise entspricht jede Operation einer Matrix Vektor Multiplikation.

Da sich die im Vorfeld berechneten und Invertierten Matrizen nur aus der Levelstruktur ergeben, ist das Abspeichern und die Wiederverwendung ein sinnvoller Schritt um Speicheraufwand gegen Rechenzeit zu tauschen. Wieder zeigte sich die geringe Größe der Gleichungssysteme von Vorteil, da der Speicheraufwand quadratisch mit der Anzahl der eindimensionalen Stützstellen wächst.

Schließlich stand ein Program zur Verfügung, dass mit geringem Mehraufwand Gitter mit beliebiger Levelstruktur konstruieren kann. Für die weiteren Untersuchungen wurde die allgemeinste Levelstruktur verwendet. Diese benutzt bei jedem Level nur eine Stützstelle mehr als im vorherigen Level. Hierdurch ist es möglich Gitter zu konstruieren, die jede beliebige monotone Menge als Indexmenge der Basisfunktionen verwenden können. Weiter kann der Beitrag einer jeden Basisfunktion einzeln untersucht und in das Gitter aufgenommen werden. So ist es möglich ein, bezüglich der Anzahl an Freiheitsgraden, sehr effizientes Gitter zu konstruieren.

Es zeigte sich, dass die allgemeinere Levelstruktur ein Vorteil bei der adaptiven Interpolation hochdimensionaler Funktionen geringer Ordnung ist. Hierdurch war es möglich Genauigkeiten zu erreichen, die dyadische Gitter in der zur Verfügung gestandenen Rechenzeit nicht erreichten. Das gesetzte Ziel ein verallgemeinertes dünnes Gitter mit beliebiger Levelstruktur zu konstruieren und das Konvergenzverhalten sowie die Laufzeiten zu untersuchen wurde erreicht. Auch die Interpolation sehr hochdimensionaler Funktionen mit einem adaptiven Algorithmus konnten im Rahmen dieser Arbeit schließlich behandelt werden.

## 7.2 Ausblick

Verallgemeinerte dünne Gitter mit beliebiger Levelstruktur sind bezüglich der Anzahl der Stützstellen optimale Gitter. Genauer gesagt bedeutet dies, dass jede beliebige monotone Indexmenge für die Basisfunktionen verwendet werden kann. Weiterhin kann ein adaptiver Algorithmus mit dieser Gitterstruktur den Beitrag jeder einzelnen Basisfunktion untersuchen und entscheiden ob diese dem Gitter hinzugefügt wird. Die noch bestehenden Einschränkungen des Gitters sind durch die Verschachtelung der Stützstellenmengen gegeben. Außerdem wird das Gitter mit einem Tensorproduktansatz konstruiert, was die Position der Gitterpunkte als Produkte der eindimensionalen Stützstellen vorgibt. Im Rahmen dieser Einschränkungen können die einzelnen Basisfunktionen und Stützstellen optimal ausgewählt werden.

Während der Arbeit zeigten sich Probleme bei dem verwendeten adaptiven Algorithmus. Dieser wurde optimiert für Unterräume, deren Anzahl der Freiheitsgrade exponentiell vom Index abhängen. Die in dieser Arbeit verwendete PLUS1-Levelstruktur führt zu Unterräumen mit nur einer Basisfunktion und somit zu einer wesentlich höheren Anzahl von Unterräumen im Gitter. Dies hatte wiederum sehr langen Konstruktionszeiten zur Folge.

Erste Lösungsmöglichkeiten für dieses Problem wurden erdacht. Jedoch war es im Rahmen dieser Arbeit nicht möglich diese fehlerfrei zu implementieren und zu untersuchen. Zuerst sollte das Verfahren so erweitert werden, dass die Gewichte endlicher Ordnung selbstständig bestimmt werden. Weiterhin könnte die Zahl der Dimensionen nicht fest vorgegeben, sondern sukzessive erhöht werden, bis eine gewünschte Genauigkeit erreicht wurde. Dies ermöglicht die Handhabung von Funktionen, bei denen über die Bedeutung der Dimensionen wenig bekannt und deren Ordnung unbekannt ist.

Eine weitere Schwäche ist, dass für jeden hinzugefügten Unterraum alle Basiskoeffizienten neu berechnet werden. Dies führt zu einer sehr hohen Zahl von Transformationen. Es wäre wesentlich effizienter zuerst alle Basisfunktionen aus der aktiven Menge hinzuzufügen, danach alle Nachbarn auf Monotonie zu untersuchen und schließlich die Koeffizienten neu zu berechnen. Dies würde die Zahl der Berechnungen während der Konstruktion adaptiver Gitter stark senken.

Mit einem solchen adaptiven Algorithmus sollten signifikante Verbesserungen der Laufzeit erreicht werden. Gleichzeitig wäre dieser in der Lage wesentlich mehr Arten von Funktionen zu behandeln, da sowohl die maximale Dimension als auch die maximale endliche Ordnung der Basisfunktionen selbstständig gefunden wird.

Im Rahmen dieser Arbeit wurden erste Aussagen bezüglich der Best Approximation und der Laufzeit eines PLUS1-Gitters gegeben. Jedoch fehlen entsprechende Untersuchungen für die Interpolationsgenauigkeit und die Konvergenzrate. Das Problem hier liegt darin, den Fehler, der durch die Berechnung der Basiskoeffizienten über ein lineares Gleichungssystem entsteht, zu bestimmen. Bisherige Aussagen über verallgemeinerte dünne Gitter

mit dyadischer Levelstruktur können nicht ohne weiteres übertragen werden.

Es fehlen auch Aussagen über die Genauigkeit der anderen in der HCFFT implementierten Basisfunktionen. Die nötigen Testprogramme existieren schon, jedoch hätten diese Untersuchungen den Rahmen der Arbeit überstiegen. Auch ist es denkbar weitere Basisfunktionen wie beispielsweise Gauss-Hermite Polynome zu integrieren. Der Code wurde so geschrieben, dass weitere Basisfunktionen problemlos integriert werden können. Es müssen lediglich die Funktionen selbst und die zugehörigen Stützstellen einprogrammiert werden.

Hierbei ist eine Erweiterung, die es ermöglicht entlang jeder Dimension unterschiedliche Basisfunktionen oder Levelstrukturen zu verwenden, denkbar. So könnten Funktionen mit unterschiedlichem Abfallverhalten der Fourierkoeffizienten entlang der Dimensionen effizienter behandelt werden. Weiterhin können Funktionen die nicht entlang jeder Richtung periodisch sind mit dieser Art von Gittern interpoliert werden.

Schließlich sollten auch andere Algorithmen für die Berechnung der Basisfunktionen ausprobiert werden. Es gibt Bibliotheken wie die NFFT [KKP08] oder NHCFFT [DKP10], die Fouriertransformationen auf nicht äquidistanten Stützstellen berechnen. Es ist nicht bekannt, ob diese eine bessere Stabilität oder Geschwindigkeit aufweisen, als die in dieser Arbeit verwendete Matrix-Vektor Multiplikation und Matrix Invertierung zur Berechnung der Basisfunktionen der eindimensionalen Interpolationsoperatoren.

Die Position der Stützstellen bedarf selbstverständlich auch weiterer Untersuchungen. Die in dieser Arbeit verwendeten Stützstellen wurden gewählt, da diese in den eindimensionalen Testfällen bei analytischen Funktionen gute Ergebnisse zeigten. Die Untersuchung weiterer Mengen hätte den Rahmen dieser Arbeit überstiegen. Es ist auch unklar, ob nicht verschachtelte Stützstellenmengen möglich sind. Mit diesen wäre die Verwendung diskreter Fouriertransformationen möglich. Von diesen sind die Fehler, die sich bei der Berechnung der Fourierkoeffizienten ergeben bekannt, und bisherige Konvergenzaussagen könnten auf PLUS1-Gitter übertragen werden.

Es zeigt sich dass mit der Abhandlung eines Problem es viele neue Fragen aufgeworfen werden, die auch in Zukunft noch einiger Untersuchungen bedürfen. Dies ist die erste Implementierung einer trigonometrischen Interpolation auf verallgemeinerten dünnen Gittern mit beliebiger Levelstruktur. Die ersten Ergebnisse sind vielversprechend und zeigen, dass weitere Untersuchungen auf diesem Gebiet viele interessante Ergebnisse bringen werden.

# Literaturverzeichnis

- [BG04] Bungartz, Hans Joachim und Michael Griebel: *Sparse grids*. Acta Numerica, 13:1–123, 2004.
- [BNTT11] Bäck, Joakim, Fabio Nobile, Lorenzo Tamellini und Raul Tempone: *Stochastic spectral Galerkin and collocation methods for PDEs with random coefficients: a numerical comparison*. In: *Spectral and High Order Methods for Partial Differential Equations*, Seiten 43–62. Springer, 2011.
- [CCS14] Chkifa, Abdellah, Albert Cohen und Christoph Schwab: *High-Dimensional Adaptive Sparse Polynomial Interpolation and Applications to Parametric PDEs*. Foundations of Computational Mathematics, 14(4):601–633, 2014, ISSN 1615-3375.
- [CHQZ10] Canuto, C., M.Y. Hussaini, A. Querteroni und T.A. Zang: *Spectral Methods - Fundamentals in Single Domains*. Springer, 2010.
- [CKN06] Cools, Ronald, Frances Y. Kuo und Dirk Nuyens: *Constructing embedded lattice rules for multivariate integration*. SIAM J. Sci. Comp, 28:2272256, 2006.
- [Dav75] Davis, P.J.: *Interpolation and approximation*. Dover books on advanced mathematics. Dover Publications, 1975, ISBN 9780486624952. <http://books.google.de/books?id=228PAQAAMAAJ>.
- [DKP10] Döhler, Michael, Stefan Kunis und Daniel Potts: *Nonequispaced Hyperbolic Cross Fast Fourier Transform*. SIAM J. Numerical Analysis, 47(6):4415–4428, 2010.
- [ES81] Efron, B. und C. Stein: *The Jackknife Estimate of Variance*. The Annals of Statistics, 9(3):pp. 586–596, 1981, ISSN 00905364. <http://www.jstor.org/stable/2240822>.
- [Gar13] Garcke, J.: *Sparse Grids in a Nutshell*. In: Garcke, J. und M. Griebel (Herausgeber): *Sparse grids and applications*, Band 88 der Reihe *Lecture Notes in Computational Science and Engineering*, Seiten 57–80. Springer, 2013. extended version with python code [http://garcke.ins.uni-bonn.de/research/pub/sparse\\_grids\\_nutshell\\_code.pdf](http://garcke.ins.uni-bonn.de/research/pub/sparse_grids_nutshell_code.pdf).
- [GG98] Gerstner, Thomas und Michael Griebel: *Numerical integration using sparse grids*. Numerical Algorithms, 18(3-4):209–232, 1998, ISSN 1017-1398.

- [GG03] Gerstner, T. und M. Griebel: *Dimension-Adaptive Tensor-Product Quadrature*. Computing, 71(1):65–87, 2003.
- [GH14] Griebel, M. und J. Hamaekers: *Fast Discrete Fourier Transform on Generalized Sparse Grids*. In: *Sparse grids and Applications*, Band 97 der Reihe *Lecture Notes in Computational Science and Engineering*, Seiten 75–108. Springer, 2014. INS Preprint No. 1305.
- [GK00] Griebel, M. und S. Knapek: *Optimized tensor-product approximation spaces*. Constructive Approximation, 16(4):525–540, 2000.
- [GK09] Griebel, M. und S. Knapek: *Optimized General Sparse Grid Approximation Spaces for Operator Equations*. Mathematics of Computations, 78(268):2223–2257, Oktober 2009. Also available as SFB611 preprint No 402.
- [GOS99] Griebel, M., P. Oswald und T. Schiekofers: *Sparse Grids for Boundary Integral Equations*. Numer. Mathematik, 83(2):279–312, 1999. also as SFB 256 report 554, Universität Bonn.
- [Gou09] Gough, Brian: *GNU Scientific Library Reference Manual - Third Edition*. Network Theory Ltd., 3rd Auflage, 2009, ISBN 0954612078, 9780954612078.
- [Gri91] Griebel, Michael: *A parallelizable and vectorizable multi-level algorithm on sparse grids*. In: Hackbusch, W. (Herausgeber): *Parallel Algorithms for partial differential equations*, Band 31, Seiten 94–100, Braunschweig, 1991. Vieweg.
- [GSZ92] Griebel, Michael, Michael Schneider und Christoph Zenger: *A Combination Technique For The Solution Of Sparse Grid Problems*, 1992.
- [Hal92] Hallatschek, K.: *Fouriertransformation of dünnen Gittern mit hierarchischen Basen*. Numerische Mathematik, Springer Verlag, 63:83–97, 1992.
- [KK11] Kämmerer, Lutz und Stefan Kunis: *On the stability of the hyperbolic cross discrete Fourier transform*. Numer. Math., 117:581–600, 2011.
- [KKP08] Keiner, Jens, Stefan Kunis und Daniel Potts: *Using NFFT 3 – a software library for various nonequispaced fast Fourier transforms*, 2008.
- [KKP12] Kämmerer, Lutz, Stefan Kunis und Daniel Potts: *Interpolation lattices for hyperbolic cross trigonometric polynomials*. J. Complexity, 28(1):76–92, 2012. <http://dblp.uni-trier.de/db/journals/jc/jc28.html#KammererKP12>.
- [Kli07] Klimke, Andreas: *Sparse Grid Interpolation Toolbox – User’s Guide*. Technischer Bericht IANS report 2007/017, University of Stuttgart, 2007.
- [Kna00] Knapek, S.: *Approximation und Kompression mit Tensorprodukt-Multiskalenräumen*. Dissertation, Universität Bonn, April 2000.



- [Kop09] Kopriva, David A.: *Implementing Spectral Methods for Partial Differential Equations: Algorithms for Scientists and Engineers*. Springer Publishing Company, Incorporated, 1st Auflage, 2009, ISBN 9048122600, 9789048122608.
- [Kup97] Kupka, F. .: *Sparse grid spectral methods for the numerical solution of partial differential equations with periodic boundary conditions*. Dissertation, 1997.
- [Kup99] Kupka, F. G.: *Sparse Grid Spectral Methods and some Results from Approximation Theory*, 1999.
- [KW05] Klimke, Andreas und Barbara Wohlmuth: *Algorithm 847: spinterp: Piecewise Multilinear Hierarchical Sparse Grid Interpolation in MATLAB*. ACM Transactions on Mathematical Software, 31(4), 2005.
- [NTT14] Nobile, F., L. Tamellini und R. Tempone: *Convergence of quasi-optimal sparse grid approximation of Hilbert-valued functions: application to random elliptic PDEs*. 2014.
- [Pas80] Pasciak, Joseph E.: *Spectral and pseudospectral methods for advection equations*. Math. Comp., 35:1081–1092, 1980.
- [Sha48] Shannon, C. E.: *A mathematical theory of communication*. Bell system technical journal, 27, 1948.
- [Smo63] Smolyak, S.: *Quadrature and interpolation formulas for tensor products of certain classes of functions*. Soviet Mathematics, Doklady, 4:240–243, 1963.
- [SWW04] Sloan, Ian H., Xiaoqun Wang und Henryk Woźniakowski: *Finite-order Weights Imply Tractability of Multivariate Integration*. J. Complex., 20(1):46–74, Februar 2004, ISSN 0885-064X. <http://dx.doi.org/10.1016/j.jco.2003.11.003>.
- [Tem93] Temlyakov, V. N.: *Approximation of periodic functions*. Journal of Approximation Theory, Nova Science, New York, Seite 419 pp., 1993.
- [WW04] Wasilkowski, G. W. und H. Woźniakowski: *Finite-order Weights Imply Tractability of Linear Multivariate Problems*. J. Approx. Theory, 130(1):57–77, September 2004, ISSN 0021-9045. <http://dx.doi.org/10.1016/j.jat.2004.06.011>.
- [Zen91] Zenger, Christoph: *Sparse Grids*. In: Hackbusch, Wolfgang (Herausgeber): *Parallel Algorithms for Partial Differential Equations*, Band 31 der Reihe *Notes on Numerical Fluid Mechanics*, Seiten 241–251. Vieweg, 1991.