Karsten Fourmont

Schnelle Fourier-Transformation bei nichtäquidistanten Gittern und tomographische Anwendungen

1999

Universität Münster

Für D.F.

Angewandte Mathematik

Schnelle Fourier-Transformation bei nichtäquidistanten Gittern und tomographische Anwendungen

Inaugural-Dissertation
zur Erlangung des Doktorgrades
der Naturwissenschaften im Fachbereich
Mathematik und Informatik
der Mathematisch-Naturwissenschaftlichen Fakultät
der Westfälischen Wilhelms–Universität Münster

vorgelegt von

Karsten Fourmont
aus Düsseldorf

1999

Dekan:	Prof. Dr. N. Schmitz
Erster Gutachter:	Prof. Dr. F. Natterer
Zweiter Gutachter:	Prof. Dr. C. Cryer
Tag der mündlichen Prüfungen:	
Tag der Promotion:	

Inhaltsverzeichnis

Kapitel 1

Überblick

Fourier Methoden spielen in zahlreichen Bereichen der angewandten Mathematik und der Physik eine gewichtige Rolle. Einer der Gründe für die Bedeutung dieser Techniken liegt in dem Umstand begründet, daß die trigonometrischen Funktionen e^{ikx} Eigenfunktionen des Differentialoperators $\frac{\partial}{\partial x}$ sind und sich somit gut zur Modellierung von Lösungen von Differentialgleichungen eignen.
Einen weiteren bedeutenden Schub bekamen die Fourier-Techniken durch die Entwicklung der schnellen Fourier-Transformation (*FFT, fast fourier transform)*, die Mitte der 60er Jahre von J. W. Cooley und J. W. Tukey [7] entwickelt wurde. Diese beruht auf einer einfachen Überlegung. Man kann eine Fourier-Transformation der Länge $N = 2^n$ wie folgt umformen:

$$\begin{aligned}
\hat{f}_k &= \sum_{j=0}^{N-1} e^{2\pi ijk/N} f_j \qquad (1.1)\\
&= \sum_{j=0}^{N/2-1} e^{2\pi ik(2j)/N} f_{2j} + \sum_{j=0}^{N/2-1} e^{2\pi ik(2j+1)/N} f_{2j+1}\\
&= \sum_{j=0}^{N/2-1} e^{2\pi ikj/(N/2)} f_{2j} + \left(e^{2\pi ik/N}\right) \sum_{j=0}^{N/2-1} e^{2\pi ikj/(N/2)} f_{2j+1}.
\end{aligned}$$

Das bedeutet, daß sich eine Fourier-Transformation der Länge N durch zwei Fourier-Transformationen der Länge $N/2$ darstellen läßt: die eine auf den Daten mit geradem Index, die andere auf den Daten mit ungeradem Index. Der entscheidende Vorteil ist nun, daß sich dieser Vorgang rekursiv anwenden

läßt. Auf diese Weise ist es möglich, (1.1) in $\mathcal{O}(N \log N)$ Rechenoperationen auszuwerten, gegenüber $\mathcal{O}(N^2)$ bei direkter Berechnung.
Der Unterschied zwischen $\mathcal{O}(N^2)$ und $\mathcal{O}(N \log N)$ bewirkt bereits bei moderaten Werten von N wie $N = 1024$ eine Beschleunigung um mehr als den Faktor 100, und für große N wie $N = 2^{20}$ macht dies eine FFT-Rechenzeit von unter 2 Sekunden gegenüber etwas über einem Tag bei direkter Berechnung aus.
Diese effiziente Rechner-Implementierung hat dazu geführt, daß Fourier-Methoden in zahlreichen weiteren Bereichen Anwendung gefunden haben, wie z.B. in der Medizintechnik, der Astronomie oder natürlich der Signalverarbeitung.
Für die Aufteilung von (1.1) in zwei Summen ist es notwendig, daß die Stützstellen äquidistant sind, d.h. daß nur Fourier-Moden $e^{2\pi ijk/N}$ mit $j, k \in \mathbb{Z}$ auftreten. Für beliebig verteilte Stützstellen e^{ikx_j} gelingt eine solche Aufteilung nicht, und man gelangt auf diese Weise nicht mehr zu einer schnellen Fourier-Transformation. Für viele Anwendungen stellt die Forderung nach Äquidistanz der Stützstellen eine zu starke Einschränkung dar, vergleiche zum Beispiel auch [25]: „Another significant limitation of all FFT methods is that they require the input data to be sampled at evenly spaced intervals."
Das Problem der effizienten Berechnung der Fourier-Transformation bei nichtäquidistanten Daten hat in den letzten Jahren einige Beachtung gefunden. Zahlreiche Autoren haben zur Lösung konkreter Probleme auf ihrem Gebiet Methoden zur effizienten Berechnung nichtäquidistanter Fourier-Transformationen entwickelt. In dem SIAM REVIEW Artikel von Ware [38] findet sich ein Überblick über die derzeit gebräuchlichen Verfahren.
Im ersten Teil dieser Arbeit wird eine neue einheitliche Theorie entwickelt, mit der sich die unterschiedlichen, bei der Approximation von Fourier-Moden auftretenden Aspekte beschreiben lassen. Besonders hervorzuheben ist hierbei die Beschreibung des Aliasing-Effektes in Abschnitt 2.2.4.
Mit diesen Erkenntnissen wird dann ein verbesserter Algorithmus zur schnellen Fourier-Transformation bei nichtäquidistanten Daten hergeleitet.
Das Ziel ist es, einen Algorithmus zu entwickeln, der robust, einfach und schnell genug ist, um in möglichst vielen Bereichen einsetzt werden zu können. Außerdem werden die Algorithmen vom zumeist betrachteten eindimensionalen Fall auf den mehrdimensionalen Fall erweitert und die entsprechenden Schwierigkeiten einer Implementierung untersucht.

Im Bereich der Computer-Tomographie wurden die Auswirkungen des Fehlens schneller nichtäquidistanter Fourier-Transformationen besonders deutlich. Das Fourier-Slice Theorem (vergleiche z.B. [21]) liefert unmittelbar einen einfachen und naheliegenden Ansatz zur effizienten Inversion der Radon-Transformation und damit zur tomographischen Rekonstruktion. Leider treten hierbei Daten auf einem zweidimensionalen Polargitter auf, die mit einer Standard-FFT nicht behandelt werden können. Der zunächst naheliegende Ansatz, die entsprechenden Daten durch Interpolation zu gewinnen, führt leider nicht zum Erfolg. Trotz zahlreicher Vorschläge zur Behandlung dieses „Interpolations-Problems", vergleiche z.B. Referenzen [1]-[11] in [6], führte diese Problematik dazu, daß in allen derzeit kommerziell gebauten Computertomographen mit der Methode der gefilterten Rückprojektion rekonstruiert wird. Die gefilterte Rückprojektion benötigt zwar einen deutlich höheren Rechenaufwand als die effizienten Fourier-Methoden, nämlich $\mathcal{O}(N^3)$ gegenüber $\mathcal{O}(N^2 \log N)$, in medizinischen Anwendungen ist die Rekonstruktionsqualität aber natürlich das ausschlaggebende Kriterium.
Mit der schnellen nichtäquidistanten Fourier-Transformation aus Kapitel 2 werden wir in Kapitel 3 einen schnellen Algorithmus zur tomographischen Rekonstruktion entwickeln und anhand von numerischen Beispielen die Rekonstruktionsqualität dieses Algorithmus untersuchen.
Der Algorithmus ist bereits für moderate Rekonstruktionsgebiete mehr als 30 mal schneller als die gefilterte Rückprojektion und liefert eine gleichwertige, in Spezialfällen sogar überlegene Rekonstruktionsqualität.
Die Radon-Inversion ist somit ein eindrucksvolles Beispiel dafür, daß die Berechnung von Fourier-Transformationen bei nichtäquidistanten, im Gegensatz zu äquidistanten, Stützstellen nicht nur eine Frage der „Bequemlichkeit" ist, sondern daß sich auf diese Weise den Fourier-Methoden auch neue Bereiche erschließen, die sich hiermit bisher gar nicht oder nicht effizient behandeln ließen.

Ich möchte mich an dieser Stelle bei Herrn Prof. Dr. F. Natterer ganz herzlich für die Anregung zu dieser Arbeit und die immer hilfreichen Hinweise und Diskussionen bedanken.

Kapitel 2

Nichtäquidistante Fourier-Transformation

Die diskrete Fourier-Transformation wird für den eindimensionalen Fall äquidistanter Stützstellen durch die folgende Formel beschrieben:

$$\hat{f}_j = \sum_{k=-N/2}^{N/2-1} f_k e^{2\pi ikj/N} \qquad \text{für } j = -N/2 \dots N/2-1 \;, \tag{2.1}$$

wobei $\{f_k\}_{k=-N/2}^{N/2-1} \in \mathbb{C}^N$ den komplexwertigen Datenvektor und $\{\hat{f}_j\}_{j=-N/2}^{N/2-1} \in \mathbb{C}^N$ die Fouriertransformierte bezeichnet.
Dies kann per FFT schnell in $\mathcal{O}(N \log N)$ Rechenoperationen ausgewertet werden. Gebräuchliche Programmpakete berechnen die FFT zumeist in einer leicht abweichenden Formulierung, bei der die Indizes von 0 bis $N-1$ statt wie hier von $-N/2$ bis $N/2-1$ laufen. Da man sowohl $\hat{f}$ als auch f als N-periodisch auffassen kann, stellt dies allerdings lediglich eine Frage der Notation dar. Falls N ungerade ist, ist die Summe in (2.1) entsprechend als Summe von $-[N/2]$ bis $[N/2]$ aufzufassen.
Dies kann man auf zwei Arten auf den Fall nichtäquidistanter Stützstellen erweitern:
Zum einen kann man sich die Daten f_k auf einem äquidistanten Gitter vorgeben und dann die Fouriertransformierte dieser Daten auf einem nichtäquidistanten Gitter mit freiverteilten Stützstellen $\{x_j\}_{j=0}^{M-1} \subset [-\pi, \pi]$ auswerten. Gesucht ist somit die Fourier-Transformation der Daten f_k an den nicht not-

wendigerweise äquidistanten Stützstellen x_j:

$$\hat{f}_j = \sum_{k=-N/2}^{N/2-1} f_k e^{ikx_j} \qquad \text{für } j = 0 \ldots M-1. \tag{2.2}$$

Man beachte, daß wir hierbei gleich eine weitere, sich aus der FFT ergebene, Einschränkung aufgehoben haben: Die Anzahl der Daten N und die Anzahl der Auswertungspunkte M muß nicht mehr identisch sein.
Der andere Fall ergibt sich, wenn die Eingangsdaten $\{g_j\}_{j=0}^{M-1}$ auf einem nichtäquidistanten Gitter $\{x_j\}_{j=0}^{M-1} \subset [-\pi, \pi]$ vorliegen und anhand dieser Werte die Fouriertransformierte auf einem äquidistanten Gitter berechnet werden soll:

$$\hat{g}_k = \sum_{j=0}^{M-1} g_j e^{ikx_j} \qquad \text{für } k = -N/2 \ldots N/2-1. \tag{2.3}$$

Diesen Fall bezeichnen wir im folgenden als „nichtäquidistante Daten" (*ned-nonequispaced data*), während wir (2.2) als „nichtäquidistantes Ergebnis" (*ner-nonequispaced result*) bezeichnen. Schreibt man die beiden Transformationen in Operatorschreibweise

$$\hat{f} = \mathcal{F}_{ner}(f) \quad \text{für (2.2)} \qquad \text{bzw.} \qquad \hat{g} = \mathcal{F}_{ned}(g) \quad \text{für (2.3)},$$

so sieht man leicht, daß

$$F_{ned} = F_{ner}^T \tag{2.4}$$

gilt, der eine Operator also lediglich der transponierte des anderen ist. Man kann folglich hoffen, beide Probleme mit denselben Methoden behandeln zu können.
Eine direkte Auswertung von (2.2) bzw. (2.3) erfordert $\mathcal{O}(N \cdot M)$ Rechenoperationen. Dies ist erheblich aufwendiger als der äquidistante Fall mit $N = M$ sowie $x_j = 2\pi j/N$, der mit Standard-FFT-Routinen in $\mathcal{O}(N \cdot \log N)$ Rechenoperationen ausgewertet werden kann. Im zwei– und drei–dimensionalen Fall wird das Verhältnis sogar noch ungünstiger, im zweidimensionalen beispielsweise $\mathcal{O}(N^4)$ gegenüber $\mathcal{O}(N^2 \log N)$. Damit wird der Aufwand für die direkte Berechnung dann unter Umständen prohibitiv groß.
Wenn man annimmt, daß die Stützstellen keinerlei Symmetrien unterliegen, erzielt man mit den bei der Standard-FFT so erfolgreichen algebraischen

Methoden zur Einsparung von Rechenoperationen (Cooley-Tukey FFT, Winograd FFT) keine Erfolge. Stattdessen versucht man, mit geeigneten Approximationen eine ausreichende Genauigkeit, nach Möglichkeit im Rahmen der Maschinengenauigkeit, bei gleichzeitiger effizienter Auswertung zu erreichen. Man sucht also nach geeigneten Wegen, sich auf eine äquidistante FFT zurückziehen zu können.

2.1 Bestehende Verfahren

Das grundsätzliche Prinzip zur Berechnung der nichtäquidistanten Fourier-Transformation beruht auf der Approximation der nichtäquidistanten Fouriermoden

$$e^{ikx} \qquad k \in \mathbb{Z}, x \in [-\pi, \pi]$$

durch äquidistante Fouriermoden $e^{2\pi ikj/N}$, $k, j \in \mathbb{Z}$.[1]

Wir wollen in diesem Abschnitt drei bestehende Methoden zur Berechnung der Fourier-Transformation bei nichtäquidistanten Daten beschreiben, nämlich die Verfahren von O'Sullivan [24], von Dutt und Rokhlin [8], sowie das von Beylkin [1]. Eine weitere Übersicht liefert Abschnitt 2.10, in dem wir unser Verfahren mit den bestehenden Methoden vergleichen. Die Vorgehensweise der einzelnen Methoden wird hier jeweils nur kurz skizziert, um die Notwendigkeit einer einheitlichen Theorie darzulegen.

2.1.1 Die Gridding Methode von O'Sullivan

Die Gridding-Methode [24] wurde von O'Sullivan Mitte der Achtziger Jahre entwickelt, siehe auch [27] für eine ausführlichere Beschreibung.
Die Motivation zur Entwicklung dieser Methode lag darin, daß bei der Inversion der Radon-Transformation mittels Fourier-Methoden eine inverse Fourier-Transformation auf einem zweidimensionalen Polargitter durchgeführt werden muß. Standard-FFT-Routinen erfordern jedoch zwingend ein cartesisches Gitter. Genau dieses Problem wird auch Gegenstand von Kapitel 3 sein.

[1]Das bedeutet, wir approximieren Exponentialfunktionen durch Exponentialfunktionen. Dies erscheint zunächst widersinnig, allerdings ist es genau der Unterschied zwischen Äquidistanz und Nichtäquidistanz, der zwischen Effizienz und Ineffizienz entscheidet.

Die Gridding Methode läßt sich wie folgt skizzieren: Gegeben seien die Werte von $\hat{f}$ auf einem beliebigen Gitter. Ziel ist es, f auf einem cartesischen Gitter zu erhalten.
Der Faltungssatz (vergleiche z.B. [37]) liefert

$$f = g \cdot w \iff \hat{f} = \hat{g} * \hat{w}.$$

Man verfährt nun in drei Schritten:

1. Faltung mit einer vorher fest gewählten Fenster-Funktion $\hat{w}$: $\hat{g} = \hat{w} * \hat{f}$

2. inverse Fourier-Transformation: $\hat{g} \longrightarrow g$

3. Skalierung: $f = g/w$

Die Behandlung der nichtäquidistanten Daten erfolgt im ersten Schritt: das Faltungsintegral wird an den cartesischen Gitterpunkten ausgewertet, unter Verwendung der nichtäquidistanten Samples von $\hat{f}$. Durch die glättende Wirkung der Faltung mit $\hat{w}$ ist dieser als „Gridding“ bezeichnete Schritt weniger fehlerträchtig als eine einfache Interpolation.
Zur Berechnung der inversen Fourier-Transformation kann dann eine Standard-FFT-Routine zum Einsatz kommen.
Für das Verfahren benötigt man eine geeignete Fenster-Funktion w, die im Orts- und im Frequenzraum um den Nullpunkt konzentriert ist. O'Sullivan schlägt hierfür die Kaiser-Bessel Funktion vor.
Die Gridding Methode hat hauptsächlich im Bereich der Computer-Tomographie Beachtung gefunden. Als generelles Hilfsmittel zur Berechnung nichtäquidistanter Fourier-Transformationen ist sie bisher nicht bekannt geworden. Möglicherweise liegt dies am Fehlen einer geeigneten Fehlerabschätzung und der eher heuristischen Begründung.

2.1.2 Die schnelle Fourier-Transformation nach Dutt und Rokhlin

Die Methode von Dutt und Rokhlin wurde Anfang der 90er Jahre entwickelt [8].
Sie dient der Berechnung eindimensionaler nichtäquidistanter Fourier-Transformationen und beruht auf der folgenden Approximation:

Satz 2.1.1 (Dutt und Rokhlin)[2] *Sei $p \geq 2$ eine ganze Zahl,$b > 1/2$, c reell und K eine ganze Zahl mit $K \geq 8b\pi$. Dann gilt für jedes $x \in [-\pi/p, \pi/p]$:*

$$\left| e^{icx} - e^{bx^2} \cdot \sum_{k=[c]-K}^{[c]+K} \left(\frac{1}{2\sqrt{b\pi}} e^{-(c-k)^2/4b} \right) e^{ikx} \right| \leq e^{b\pi^2/p^2} \cdot e^{-b\pi^2} \cdot (4b+9).$$

Die Fehlerabschätzung ist nicht scharf, der beobachtbare Fehler liegt um einige Größenordnungen unter dieser Schranke. Der Grund hierfür liegt darin, daß der von uns in Abschnitt 2.2.4 beschriebene Aliasing-Effekt bei der Herleitung des Fehlers nicht berücksichtigt wurde.
Mit dieser Approximation läßt sich eine nichtäquidistante Fourier-Transformation

$$f_j = \sum_{k=-N/2}^{k=N/2-1} \alpha_k \cdot e^{ijx_k} \tag{2.5}$$

für $j = -N/2, \ldots, N/2-1$ wie folgt durchführen:

1. Berechne

$$\tau_l = \sum_k \alpha_k \cdot \frac{1}{2\sqrt{b\pi}} \cdot \exp\left\{ - \left(\frac{\bar{N}}{2\pi} x_k - l \right)^2 / 4b \right\}, \tag{2.6}$$

   für $l = -\bar{N}/2, \ldots, \bar{N}/2-1$ bei $\bar{N} = pN$ fest. Bei der numerischen Auswertung der Summe beschränkt man sich jeweils auf einige wenige Terme mit $x_k \approx l$.

2. Führe eine äquidistante FFT der Länge $\bar{N}$ durch:

$$U_l = \sum_{k=-\bar{N}/2}^{k=\bar{N}/2-1} \tau_k \cdot e^{2\pi ikl/\bar{N}}$$

3. Skaliere das Ergebnis:

$$f_l = e^{b(2\pi l/\bar{N})^2} \cdot U_l$$

[2]Corollar 2.9 aus [8] mit leicht geänderter Notation

2.1.3 Die schnelle Fourier-Transformation nach Beylkin

Die Methode von Beylkin entstand im Zusammenhang mit einem Verfahren zur schnellen Berechnung der Fouriertransformierten von Funktionen mit Singularitäten [1]. Sie liefert aber auch unmittelbar einen Algorithmus zur schnellen nichtäquidistanten Fourier-Transformation, der dem von Dutt und Rokhlin ähnelt.
Die Methode basiert auf der folgenden Approximation:

Satz 2.1.2 (Beylkin)[3] *Sei f eine verallgemeinerte Funktion [im Sinne der Distributionstheorie], $j < 0$. Definiere*

$$f_k = 2^{-j/2} \int_{-\infty}^{\infty} f(x) \beta^{(m)} \left(2^{-j} x - k \right) \, dx,$$

wobei $\beta^{(m)}$ der symmetrische B-Spline der Ordnung m ist.
Dann existiert eine Fehlerkonstante ε, so daß

$$\left| \frac{2^{j/2}}{\sqrt{a^{(m)}(\xi)}} \sum_{k \in \mathbb{Z}} f_k e^{-2\pi i k \xi} - \hat{f}(2^{-j} \xi) \right| \leq \varepsilon$$

gilt. Hierbei ist

$$a^{(m)}(\xi) = \sum_{l} \left| \hat{\beta}^{(m)}(\xi + l) \right|^2 .$$

Die explizite Fehlerabschätzung findet sich in [8].
Eine Methode zur schnellen Berechnung von

$$\hat{g}_k = \sum_{l=0}^{M-1} g_l e^{2\pi i x_l k}$$

wird daraus, wenn man $f(x) = c \sum_{l=0}^{M-1} g_l \delta(x - x_l)$ wählt, so daß $\hat{g}_k = \hat{f}(k)$ gilt.
Die Berechnung erfolgt wiederum in drei Schritten.

[3] Aus [1] Theorem III.1

1. Berechne

$$f_k = \sum_{l=0}^{M-1} g_l \beta^{(m)}(\bar{N} \cdot x_l - k)$$

   mit $\bar{N} > N$.

2. Führe eine FFT durch:

$$\hat{f}_j = \sum_{k=0}^{\bar{N}-1} f_k e^{-2\pi ikj/\bar{N}}$$

3. Skaliere das Ergebnis

$$\hat{g}_j = \frac{1}{\sqrt{a^{(m)}(j/\bar{N})}} \cdot \hat{f}_j.$$

Die drei besprochenen Verfahren weisen deutliche Ähnlichkeiten auf. Die Berechnung erfolgt stets in drei Schritten: zuerst eine Faltung bzw. Interpolation, danach die Berechnung der äquidistanten FFT und schließlich eine Skalierung.
Die mathematischen Beschreibungen der drei Verfahren weichen jedoch stark voneinander ab: O'Sullivan beschränkt sich auf die angegebene heuristische Begründung.
Rokhlin leitet Satz 2.1.1 zur Approximation von Fourier-Moden her, liefert aber keine weitere Motivation für die Wahl der Approximationskoeffizienten bzw. die verwendete Gauß-Glockenkurve $e^{-x^2/4b}$. Geeignete Werte für den Parameter b werden lediglich experimentell ermittelt, auf die Auswirkungen dieses Koeffizienten wird nicht näher eingegangen.
Beylkin leitet sein Verfahren als Methode zur Berechnung von Fourier-Integralen von Funktionen mit Singularitäten her und beschreibt seine Vorgehensweise als ,,Projektion solcher Funktionen auf einen Unterraum einer Multiskalen-Analyse". Dies ist leider für den Fall der nichtäquidistanten Fourier-Transformation wenig erhellend, da hierbei die Skala j einmal fest gewählt wird und dann konstant bleibt.

2.2 Herleitung der Approximation

2.2.1 Resampling

Bei der Vorstellung der bestehenden Methoden im vorherigen Abschnitt haben wir den Fall nichtäquidistanter Eingangsdaten behandelt.
Für die Herleitungen in diesem Abschnitt wollen wir jedoch den Fall äquidistanter Daten und und nichtäquidistanter Auswertungspunkte (*ner*) betrachten. Dieser läßt sich etwas einfacher behandeln, weil sich die auftretenden Effekte dann sehr gut mit dem Vokabular der Signalverarbeitung beschreiben lassen. Die Übertragung auf nichtäquidistante Daten wird Aufgrund der Dualität (2.4) keine Schwierigkeiten bereiten.
Die nichtäquidistanten Stützstellen bezeichnen wir jetzt mit ξ_j bzw. einfach nur mit ξ und skalieren sie statt auf $[-\pi, \pi]$ auf $[-N/2, N/2]$.
Das Problem stellt sich somit als effiziente Berechnung von

$$\hat{f}(\xi) = \sum_{k=-N/2}^{N/2-1} f_k e^{2\pi i k \xi / N} \tag{2.7}$$

für beliebige Werte von $\xi \in [-N/2, N/2]$. Wir wollen somit die N-periodische Funktion f an verschiedenen Stellen ξ_j auswerten.
Die per FFT schnell berechenbaren Werte

$$g_l = \sum_{k=-N/2}^{N/2-1} f_k e^{2\pi i k l / N} \qquad l = -N/2, \ldots, N/2-1 \tag{2.8}$$

entsprechen dann gerade den Werten von f an den (äquidistanten) Punkten $l = -N/2, \ldots, N/2-1$. Da $f(\xi)$ N-periodisch ist liefert die FFT sogar alle Werte $f(l)$ für $l \in \mathbb{Z}$. Es ist somit naheliegend zu versuchen $\hat{f}(\xi)$ für beliebige ξ aus den $\hat{f}(l)$, $l \in \mathbb{Z}$ zu ermitteln.
Dazu entwickeln wir $e^{2\pi i k \xi / N}$ als Funktion von k in eine Fourier-Reihe im Intervall $[-N/2, N/2]$:

$$e^{2\pi i k \xi / N} = \sum_l \sigma_l(\xi) e^{2\pi i k l / N}. \tag{2.9}$$

Man faßt $e^{2\pi i k \xi / N}$ deshalb als Funktion von k auf, damit die Abhängigkeit von ξ sich in den Koeffizienten σ_l und nicht in den Ansatzfunktionen $e^{2\pi i k l / N}$ wiederspiegelt.

Die Fourier-Koeffizienten ergeben sich wie folgt:

$$\begin{aligned}
\sigma_l(\xi) &= \frac{1}{N}\int_{-N/2}^{N/2} e^{2\pi is\xi/N} e^{-2\pi ils/N}\, ds \\
&= \frac{1}{N}\int_{-N/2}^{N/2} e^{2\pi is/N\cdot(\xi-l)}\, ds \\
&= \frac{1}{N}\int_{-N/2}^{N/2} cos\left(2\pi s/N\cdot(\xi-l)\right)\, ds \\
&= \operatorname{sinc}\left(\pi(\xi-l)\right)
\end{aligned}$$

Mit der sinc-Funktion:

$$\operatorname{sinc}(x) = \begin{cases} \frac{\sin x}{x} & : \quad x \neq 0 \\ 1 & : \quad x = 0 \end{cases}$$

Die Fourier-Reihenentwicklung

$$e^{2\pi ik\xi/N} = \sum_{l\in\mathbb{Z}} \operatorname{sinc}\left(\pi(\xi-l)\right) e^{2\pi ikl/N} \tag{2.10}$$

gilt allerdings nur für $k \in]-N/2, N/2[$, nicht für $k = \pm N/2$, da $e^{2\pi ik\xi/N}$ als Funktion von k nicht N-periodisch ist: es gilt $e^{iN\xi/2} \neq e^{-iN\xi/2}$ für $\xi \notin \mathbb{Z}$. Deshalb nehmen wir für den Moment an, daß $f_{-N/2} = 0$ gilt und somit der Fall $|k| = N/2$ nicht ins Gewicht fällt. Dann ergibt sich:

$$\begin{aligned}
\hat{f}(\xi) &= \sum_{k=-N/2}^{N/2-1} f_k e^{2\pi ik\xi/N} \\
&= \sum_{k=-N/2}^{N/2-1} f_k \sum_{l\in\mathbb{Z}} \operatorname{sinc}\left(\pi(\xi-l)\right) e^{2\pi ikl/N} \\
&= \sum_{l\in\mathbb{Z}} \operatorname{sinc}\left(\pi(\xi-l)\right) \sum_{k=-N/2}^{N/2-1} f_k e^{2\pi ikl/N} \\
&= \sum_{l\in\mathbb{Z}} \operatorname{sinc}\left(\pi(\xi-l)\right) \hat{f}(l), \qquad (2.11)
\end{aligned}$$

wobei sich die

$$\hat{f}(l) = \sum_{k=-N/2}^{N/2-1} f_k e^{2\pi ikl/N} \tag{2.12}$$

per FFT berechnen lassen.
Dieses Ergebnis kann man auch auf anderem Wege erhalten: berechnet man die Fouriertransformierte von $\hat{f}$

$$\left(\mathcal{F}\hat{f}\right)(x) = \frac{1}{\sqrt{2\pi}} \sum_{k=-N/2}^{N/2-1} \delta(x - 2\pi k/N), \tag{2.13}$$

so sieht man, daß $\mathcal{F}\hat{f}$ seinen Träger in $[-\pi, \pi]$ hat und somit π-bandbeschränkt ist. Das Shannon-Theorem (vergleiche [14]) liefert dann eine Interpolationsformel zur exakten Berechnung von $\hat{f}(\xi)$ aus den $\hat{f}(l)$:

$$\hat{f}(\xi) = \sum_{l \in \mathbb{Z}} \hat{f}(l) \operatorname{sinc}\left(\pi(\xi - l)\right),$$

was offenbar mit (2.11) identisch ist. Allerdings muß man auch hier wieder $f_{-N/2} = 0$ fordern, da eine Delta-Distribution am Rande des Trägers von $\mathcal{F}\hat{f}$ andernfalls Probleme bereitet. Für eine genauere Betrachtung des Sampling Theorems bei Distributionen vergleiche auch [4] oder [14].
Diese Formel ergibt aber noch keinen schnellen Algorithmus zur Berechnung einer nichtäquidistanten Fourier-Transformation. Dies liegt primär darin begründet, daß die sinc-Funktion für $x \to \infty$ nur wie $1/x$ abfällt; sie ist noch nicht einmal absolut integrierbar. Deshalb muß man bei der numerischen Auswertung der Summe über l zu viele Terme mitnehmen, so daß gegenüber der direkten Berechnung der Fourier-Transformation nichts gewonnen ist[4].

2.2.2 Oversampling

In der Signalverarbeitung ist das Problem, das sich aus dem schlechten Abklingverhalten der sinc-Funktion ergibt, wohlbekannt. Die gebräuchlichste Möglichkeit dem abzuhelfen ist das Oversampling:
Dabei geht man davon aus, daß nicht die Funktionswerte $\hat{f}(l)$ sondern die Werte $\hat{f}(l/p)$ mit $p > 1, l \in \mathbb{Z}$ zur Verfügung stehen.
Diese kann man in unserem Falle mit einer FFT der Länge pN berechnen. Deshalb sollte p so gewählt sein, daß pN per FFT berechnet werden kann,

[4] Trotzdem bietet diese Interpolationsformel einen Ansatz für effiziente Algorithmen. Diese werden in Abschnitt 2.10 kurz vorgestellt.

insbesondere muß $pN \in \mathbb{N}$ gelten. Allerdings sind durchaus auch nicht ganzzahlige Werte für p denkbar wie z.B. $p = 1.5$.
Statt der $e^{2\pi ikl/N}$ wie in (2.9) nimmt man jetzt $e^{2\pi ikl/pN}$ als Ansatzfunktionen und berechnet eine Fourier-Reihenentwicklung auf dem Intervall $[-pN/2, pN/2]$:

$$e^{2\pi ik\xi/N} = \sum_l \sigma_l(\xi) e^{2\pi ikl/pN}. \tag{2.14}$$

Entscheidend ist aber nun, daß man die Approximation für $e^{2\pi ik\xi/N}$ nur für $k \in [-N/2, N/2]$ benötigt. Nimmt man eine beliebige $\mathcal{C}_0^\infty$-Funktion $\phi(k)$ mit

$$\phi(k) \equiv 1 \qquad \text{für } |k| \leq N/2 \tag{2.15}$$

und

$$\operatorname{supp}(\phi) \subset [-pN/2, pN/2] \tag{2.16}$$

an, so kann man statt $e^{2\pi ik\xi/N}$ auch $\phi(k) \cdot e^{2\pi ik\xi/N}$ in eine Fourier-Reihe entwickeln, da die beiden Funktionen für $|k| \leq N/2$ übereinstimmen:

$$\phi(k) \cdot e^{2\pi ik\xi/N} = \sum_l \sigma_l(\xi) e^{2\pi ikl/pN}. \tag{2.17}$$

Die $\sigma_l(\xi)$ sind natürlich andere als in (2.9) bzw. (2.14). Sie berechnen sich nun wie folgt:

$$\begin{aligned}
\sigma_l(\xi) &= \frac{1}{pN} \int_{-pN/2}^{pN/2} \phi(s) e^{2\pi i\xi s/N} e^{-2\pi ils/pN}\, ds \\
&= \frac{1}{pN} \int_{-pN/2}^{pN/2} \phi(s) e^{2\pi is/N\cdot(\xi - l/p)}\, ds \\
&= \frac{1}{pN} \int_{-\infty}^{\infty} \phi(s) e^{2\pi is/N\cdot(\xi - l/p)}\, ds \\
&= c\,\hat{\phi}\left(\frac{2\pi}{N}(\xi - l/p)\right).
\end{aligned} \tag{2.18}$$

mit $c = \sqrt{2\pi}/pN$.
Die Interpolationsformel lautet in diesem Falle:

$$\hat{f}(\xi) = c \sum_{l\in\mathbb{Z}} \hat{\phi}\left(\frac{2\pi}{N}(\xi - l/p)\right) \hat{f}(l/p), \tag{2.19}$$

wobei sich die

$$\hat{f}(l/p) = \sum_{k=-N/2}^{N/2-1} f_k e^{2\pi i k l/pN}$$

wieder per FFT, dieses Mal der Länge pN, berechnen lassen.
Auch der Fall $k = -N/2$ macht jetzt keine Probleme mehr, da wir die Fourier-Reihenentwicklung jetzt auf dem Intervall $[-pN/2, pN/2]$ durchführen und hier $\phi(pN/2) = \phi(-pN/2) = 0$ gilt.
Als Fouriertransformierte einer $\mathcal{C}_0^\infty$-Funktion fällt $\hat{\phi}(x)$ für $|x| \to \infty$ schneller ab als jede Potenz von x. Der Satz von Paley-Wiener [34] besagt, daß, falls F Fouriertransformierte einer $\mathcal{C}_0^\infty$-Funktion ist, es für jedes $N > 0$ ein $C > 0$ gibt, derart daß

$$|F(x)| \leq C_N(p)(1 + |x|)^{-N} \tag{2.20}$$

gilt. Leider kann die Konstante $C_N(p)$ sehr groß werden. Die Anzahl der Terme, die man bei der numerischen Auswertung von (2.19) mitnehmen muß, ist immer noch zu groß, um einen wirklich effizienten Algorithmus zu liefern. In [20] wurde diese Methode zur effizienten Interpolation vorgestellt. Bei der dort getroffenen Wahl von ϕ muß man beispielsweise etwa 40 Terme mitnehmen, bevor der erste Summand $< 10^{-5}$ wird. Eine Methode, die Anzahl der Terme zu reduzieren, liegt darin, den Oversampling Faktor p zu erhöhen. Leider geht dieser als Faktor p^n in die Anzahl der Rechenoperationen ein, wobei n die Anzahl der Dimensionen ist. Bei zwei und mehr Dimensionen sollte man sich also mit einem möglichst kleinen p, vorzugsweise $p \leq 2$, begnügen.
Das Entscheidende Kriterium für die Effizienz ist daher das numerische Abklingverhalten von $\hat{\phi}$.

2.2.3 Modifiziertes Resampling

Wir haben bisher den Weg gewählt, zuerst eine FFT unserer Eingangsdaten zu berechnen, und dann auf die nichtäquidistanten Punkte zu interpolieren. Hierbei haben wir noch keinen Gebrauch von der Möglichkeit gemacht, die Eingangsdaten *vor* Berechnung der FFT zu modifizieren. Diese Modifikation sollte dergestalt sein, daß das numerische Abklingverhalten von $\hat{\phi}(x)$ sich weiter verbessert. Dieses wird vor allem durch die Forderung (2.15), nämlich $\phi(k) \equiv 1$ für $k \in [-N/2, N/2]$, limitiert. Dadurch treten für $N/2 \leq |k| \leq$

$pN/2$ starke Schwankungen der Ableitungen von $\phi(\xi)$ auf, die eine größere Konstante C_N in (2.20) bewirken.
Die entscheidende Verbesserung ergibt sich nun dadurch, daß man die Forderung nach Konstanz von ϕ in $[-N/2, N/2]$ aufgibt und sich auf die Forderung

$$\phi(k) > 0 \qquad \text{für } |k| \in [-N/2, N/2] \tag{2.21}$$

beschränkt.
Die Fourier-Koeffizienten ergeben sich wieder wie in (2.18):

$$\sigma_l(\xi) = c\,\hat{\phi}\left(\frac{2\pi}{N}(\xi - l/p)\right),$$

und damit erhält man die Fourier-Reihe:

$$\phi(k) \cdot e^{2\pi i k\xi/N} = c\sum_l \hat{\phi}\left(\frac{2\pi}{N}(\xi - l/p)\right) e^{2\pi i k l/pN}. \tag{2.22}$$

Natürlich kann man $\hat{f}(\xi)$ jetzt nicht mehr aus geeigneten äquidistanten Werten von $\hat{f}$ gewinnen.
Für $k \in [-N/2, N/2]$ kann man allerdings (2.22) durch $\phi(k) > 0$ dividieren und erhält

$$e^{2\pi i k\xi/N} = \frac{c}{\phi(k)}\sum_l \hat{\phi}\left(\frac{2\pi}{N}(\xi - l/p)\right) e^{2\pi i k l/pN}. \tag{2.23}$$

Setzt man diese Identität für $e^{2\pi i k\xi/N}$ in die Ausgangsgleichung (2.7) ein, so ergibt sich

$$\begin{aligned}
\hat{f}(\xi) &= \sum_{k=-N/2}^{N/2-1} f_k e^{2\pi i k\xi/N} \\
&= \sum_{k=-N/2}^{N/2-1} f_k \frac{c}{\phi(k)} \sum_l \hat{\phi}\left(\frac{2\pi}{N}(\xi - l/p)\right) e^{2\pi i k l/pN} \\
&= \sum_l \hat{\phi}\left(\frac{2\pi}{N}(\xi - l/p)\right) \sum_{k=-N/2}^{N/2-1} f_k \cdot \frac{c}{\phi(k)} e^{2\pi i k l/pN}.
\end{aligned} \tag{2.24}$$

Dies ist das wesentliche Ergebnis, daß nichtäquidistante effiziente Fourier-Transformationen ermöglicht: die Auswirkungen der Interpolation (des

Resamplings) im Fourier-Raum können vorab durch eine Skalierung im Ortsraum kompensiert werden. Zum ersten Mal wurde dies bei der Gridding-Methode von O'Sullivan ausgenutzt. Interpretiert man die Interpolation als Faltung, so entspricht die Vorgehensweise einer Anwendung des Faltungssatzes:

$$f = g/w \iff \hat{g} = \hat{w} * \hat{f}.$$

Wir benötigen also keine FFT der ursprünglichen Eingangsdaten f_k mehr, sondern eine der skalierten Werte $c \cdot f_k/\phi(k)$. Definiert man

$$g_l = \begin{cases} 0 & : \quad l = -pN/2, \ldots, -N/2-1 \\ \frac{c \cdot f_k}{\phi(k)} & : \quad l = -N/2, \ldots, N/2-1 \\ 0 & : \quad l = N/2, \ldots, pN/2-1, \end{cases} \tag{2.25}$$

so kann die Summe über k in (2.24) durch

$$\hat{g}_l \quad = \sum_{k=-N/2}^{N/2-1} \frac{c \cdot f_k}{\phi(k)} e^{2\pi ikl/pN} = \sum_{k=-pN/2}^{pN/2-1} g_k e^{2\pi ikl/pN}$$

per FFT der Länge pN schnell berechnet werden.
Man erhält

$$\begin{aligned} \hat{f}(\xi) &= \sum_{k=-N/2}^{N/2-1} f_k e^{ik\xi} \\ &= \sum_l \hat{\phi}\left(\frac{2\pi}{N}(\xi - l/p)\right) \hat{g}_l. \end{aligned}$$

Ein effizienter Algorithmus wird daraus, wenn man die Summe über l schnell berechnen kann, weil nur wenige Terme nennenswerte Beiträge liefern. Die wesentlichen Anteile von $\hat{\phi}$ sollen also in einem möglichst kleinen Intervall konzentriert sein. Dies ist das grundlegende Prinzip unseres Algorithmus zur schnellen Fourier-Transformation bei nichtäquidistanten Daten.

2.2.4 Aliasing

Nachdem wir von den beiden ursprünglichen Forderungen an ϕ, nämlich $\phi(k) \equiv 1$ für $|k| \leq N/2$ und $\text{supp}(\phi) \subset [-pN/2, pN/2]$, die erste schon modifiziert haben, wollen wir auch die Forderung an den Träger von ϕ abschwächen. Diese diente dazu, das Intervall bei der Berechnung der Fourierkoeffizienten von $[-pN/2, pN/2]$ auf die gesamte reelle Achse auszudehnen, um mit der Fourier-Transformation arbeiten zu können. Dies allein ist noch keine echte Einschränkung, da uns ja bei der Berechnung der Fourier-Koeffizienten ohnehin nur die Werte von $\phi(k)$ für $k \in [-pN/2, pN/2]$ interessieren; welche Werte die Funktion außerhalb annimmt spielt für die Fourier-Koeffizienten keine Rolle.

Wir wollen untersuchen, was passiert, falls man den Träger von ϕ über das Intervall $[-pN/2, pN/2]$ hinaus ausdehnt und trotzdem weiterhin die Fouriertransformierte in der Approximation verwendet.

Dies kann man auch wie folgt interpretieren: Angenommen, ϕ sei so gewählt, daß $\text{supp}(\phi) = [-\alpha pN/2, \alpha pN/2]$ mit $\alpha \geq 1$ gelte.

Man betrachtet nun die pN-Periodifizierung von $\phi(k)e^{2\pi ik\xi/N}$:

$$g(k) := \sum_{l\in\mathbb{Z}} \phi(k+pNl)e^{2\pi i(k+pNl)\xi/N}.$$

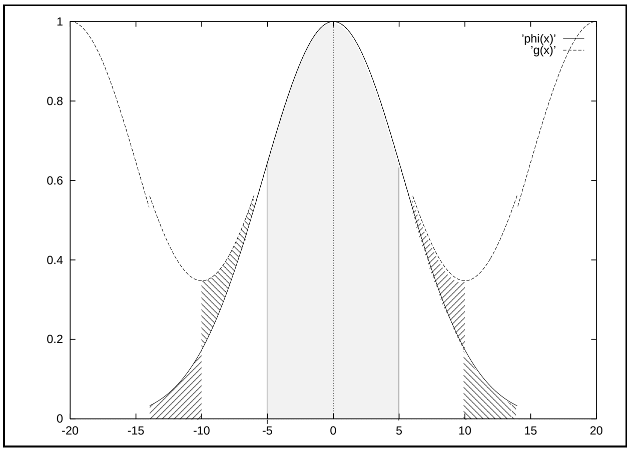


Abbildung 2.1: Aliasing bei $p = 2$, $\alpha = 1.4$, $N = 10$ und $\phi(x) = e^{-0.035x^2}$.

Im Intervall $[-N/2, N/2]$ benötigen wir eine Approximation für

$$e^{ik\xi},$$

deshalb darf g für $k \in [-N/2, N/2]$ keine weitere Abhängigkeit von ξ enthalten, da sonst in der Approximation (2.23) ein Exponentialterm mit einer nichtäquidistanten Frequenz ξ auf der rechten Seite auftauchen würde. Diese Bedingung ist erfüllt, falls

$$\phi(k + pNl) \equiv 0 \qquad \text{für } k \in [-N/2, N/2] \text{ und } l \in \mathbb{Z} \neq 0$$

gilt. Insbesondere soll also $[-N/2, N/2] \pm pN \cap \operatorname{supp}(\phi) = \emptyset$ gelten. Da wir den Träger von ϕ ja mit $[-\alpha pN/2, \alpha pN/2]$ angenommen hatten, bedeutet dies an den Rändern des Intervalls

$$N/2 < -\alpha pN/2 + pN \quad \text{bzw.} \quad -N/2 > \alpha pN/2 - pN.$$

Da der Oversampling-Faktor p größer 1 ist, ergibt sich an α die Forderung

$$\boxed{\alpha < 2 - 1/p.} \tag{2.26}$$

Eine kleine weitere Einschränkung ergibt sich, falls ϕ an den Rändern des Trägers nicht stetig ist, ein Fall, der, wie sich zeigen wird, tatsächlich auftritt. Diese Unstetigkeitsstelle setzt sich durch die pN-Periodifizierung in den Bereich $N/2 < |x| \leq pN/2$ fort. Für die Wahl $\alpha = 2 - 1/p$ liegt die Unstetigkeitsstelle dann genau bei $N/2$, also an der Grenze des Intervalls, in dem wir die Fourier-Approximation benötigen. In der Nähe von Unstetigkeitsstellen verschlechtert sich die Konvergenz der Fourier-Reihe, deshalb sollte α etwas kleiner gewählt werden als (2.26) es erfordert. Für $p = 2$ ergibt beispielsweise, mit der im folgenden Abschnitt vorgestellten Gewichtsfunktion, eine Wahl von $\alpha = 1.49$ gute Ergebnisse.
Abbildung 2.1 verdeutlicht diese Situation für $p = 2$, $\alpha = 1.4$, $N = 10$ und der Funktion $\phi(x) = e^{-0.035x^2}$. Dieses ϕ wurde nur als Beispiel gewählt, in tatsächlich auftretenden Fällen liegt der Sprung an der Unstetigkeitsstelle in der Größenordnung der Maschinengenauigkeit. Die schraffierten Bereiche außerhalb von $[-pN/2, pN/2]$ werden jeweils auf der anderen Seite in das Intervall hineingeschoben. Der grau markierte Bereich, in dem wir die Approximation benötigen, bleibt davon jedoch unbeeinflußt.
Ein α von 1.49 bedeutet, daß man den Träger von ϕ um fast 50% vergrößern kann. Die Funktion wird somit ,,breiter", was wiederum eine im Frequenzraum stärker konzentrierte Fouriertransformierte ermöglicht. Somit benötigt man in der Approximation (2.23) weniger Terme, um eine vorgegebene Genauigkeit zu erzielen. Dieser Effekt ist durchaus beachtlich, bei vorgegebener

Anzahl von Termen verbessert sich die Genauigkeit typischerweise um zwei Dezimalstellen.

Die Fourier-Koeffizienten zur Entwicklung von $g(k)$ in eine Fourier-Reihe im Intervall $[-pN/2, pN/2]$ berechnen sich durch:

$$\begin{aligned}
\sigma_l(\xi) &= \frac{1}{pN}\int_{-pN/2}^{pN/2} g(s)e^{-2\pi i l s/pN}\,ds \\
&= \frac{1}{pN}\int_{-pN/2}^{pN/2} \sum_{l\in\mathbb{Z}} \phi(s+pNl)e^{2\pi i(s+pNl)\xi/N}e^{-2\pi i l(s+pNl)/pN}\,ds \\
&= \frac{1}{pN}\int_{-\infty}^{\infty} \phi(s)e^{2\pi i s\xi/N}e^{-2\pi i l s/pN}\,ds \\
&= \frac{1}{pN}\int_{-\infty}^{\infty} \phi(s)e^{2\pi i s/N\cdot(\xi-l/p)}\,ds \\
&= c\,\hat{\phi}\left(\frac{2\pi}{N}(\xi-l/p)\right)
\end{aligned}$$

Dies ist identisch zu 2.18, die Koeffizienten ergeben sich weiterhin aus der Fouriertransformierten von ϕ.
Diese Technik kann man sich als toleriertes Aliasing vorstellen: Durch Erweiterung des Trägers von ϕ außerhalb von $[-pN/2, pN/2]$ werden Anteile aus dem Bereich $pN/2 \le |t| \le (2-1/p)pN/2$ in den Bereich $N/2 \le |t| \le pN/2$ verschoben (Abbildung 2.1). Da wir die Approximation ja nur im Intervall $[-N/2, N/2]$ benötigen, bereitet diese Form des Aliasing erfreulicherweise keinen Verlust an Genauigkeit.
Damit haben wir alle Techniken zur Verfügung, um einen Algorithmus zur schnellen Fourier-Transformation aufzustellen. Im nächsten Abschnitt werden die Ergebnisse mathematisch ausformuliert, bevor dann eine geeignete Funktion ϕ gesucht wird.

2.3 Approximation der Fourier-Moden

Ab jetzt nehmen wir die Stützstellen wieder in $[-\pi, \pi]$ an. Es ergibt sich somit die Forderung nach einer Approximation von e^{ikx} mit $x \in [-\pi, \pi]$. Der folgende Satz liefert hierzu die Basis.

Satz 2.3.1 *Sei* $N \in \mathbb{N}$, $p > 1$ *mit* $pN \in \mathbb{N}$. *Sei* ϕ *eine gerade, in einer Umgebung von* $[-\pi/p, \pi/p]$ *positive stetig differenzierbare, und ansonsten stückweise stetig differenzierbare Funktion mit Träger* $\operatorname{supp}(\phi) \subset [-\alpha\pi, \alpha\pi]$ *bei* $1 \leq \alpha < 2 - 1/p$.
$\sigma(x, l)$ *sei definiert durch*

$$\sigma(x, l) = \frac{1}{\sqrt{2\pi}} \hat{\phi}\left(\frac{pN}{2\pi}x - l\right). \tag{2.27}$$

Dann gilt für alle $x \in \mathbb{R}$ *und* $k \in [-N/2, N/2]$*:*

$$e^{ikx} = \frac{1}{\phi\left(\frac{2\pi}{pN}k\right)} \sum_l \sigma(x, l) e^{2\pi i k l / pN}. \tag{2.28}$$

Beweis:
Definiere $g(t)$ durch

$$g(t) = \sum_{n \in \mathbb{Z}} \phi(t + 2\pi n) e^{i(t + 2\pi n)y} \tag{2.29}$$

für beliebiges $y \in \mathbb{R}$. g ist dann offenbar 2π-periodisch und es gilt

$$g(t) = \phi(t) e^{ity} \qquad \text{für } t \in [-\pi/p, \pi/p], \tag{2.30}$$

da für diese t $t \pm 2\pi$ nicht mehr im Träger von ϕ liegt. Entwickelt man g im Intervall $[-\pi, \pi]$ in eine Fourierreihe, so ergeben sich die Koeffizienten durch

$$\begin{aligned}
\sigma_l &= \frac{1}{2\pi} \int_{-\pi}^{\pi} g(s) e^{-ils}\, ds \\
&= \frac{1}{2\pi} \int_{-\pi}^{\pi} \sum_{n \in \mathbb{Z}} \phi(s + 2\pi n) e^{i(s + 2\pi n)y} e^{-il(s + 2\pi n)}\, ds \\
&= \frac{1}{\sqrt{2\pi}} \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \phi(s) e^{is(y - l)}\, ds \\
&= \frac{1}{\sqrt{2\pi}} \hat{\phi}(y - l).
\end{aligned}$$

Da $g(t)$ für $|t| \leq \pi/p$ stetig differenzierbar ist, konvergiert die Fourier-Reihe dort punktweise:

$$g(t) = \frac{1}{\sqrt{2\pi}} \sum_l \hat{\phi}(y - l) e^{ilt}.$$

Setzt man nun $t = 2\pi k/pN$ so liefert die Voraussetzung $k \in [-N/2, N/2]$, daß $t \in [-\pi/p, \pi/p]$ gilt. Man kann somit (2.30) anwenden. Substituiert man außerdem noch $y = pNx/2\pi$, so erhält man:

$$\phi(2\pi k/pN) \cdot e^{ikx} = \frac{1}{\sqrt{2\pi}} \sum_l \hat{\phi}\left(\frac{pN}{2\pi}x - l\right) e^{2\pi ikl/pN}.$$

Division durch $\phi(2\pi k/pN) \neq 0$ liefert schließlich die Behauptung.

□

Zur numerischen Auswertung müssen wir die Summe in (2.28) natürlich durch eine endliche Summe ersetzen. Wegen (2.27) liegt es nahe, dazu diejenigen Terme zu verwenden, für die $pNx/2\pi \approx l$ gilt, d.h. wir approximieren jede nichtäquidistante Frequenz x durch die nächstgelegenen äquidistanten Frequenzen $2\pi l/pN$. Nehmen wir eine Approximation aus $2K$ Termen an, so ergibt sich:

$$e^{ikx} \approx \frac{1}{\phi\left(\frac{2\pi}{pN}k\right)} \sum_{[pNx/2\pi]-K+1}^{[pNx/2\pi]+K} \sigma(x,l) e^{2\pi ikl/pN}.$$

Für x mit $pNx/2\pi \notin \mathbb{Z}$ entspricht dies einer Summation über alle Terme l mit $|pNx/2\pi - l| \leq K$. Falls x hingegen mit einer Stützstelle der äquidistanten Berechnung übereinstimmt, d.h. für x mit $pNx/2\pi \in \mathbb{Z}$, wird über alle $l : |pNx/2\pi - l| < K$ und über $l = pNx/2\pi + K$, nicht aber über $l = pNx/2\pi - K$ summiert. Der Unterschied ist allerdings hauptsächlich für die Formulierung der Fehlerabschätzung in Abschnitt 2.5 relevant.
Man kann dann einen Approximationsfehler $\mathrm{err}(\phi, \sigma, p, K)$ wie folgt definieren:

$$\mathrm{err}(\phi, \sigma, K, p) = \max_{k\in[-N/2,N/2], x\in\mathbb{R}} \left| e^{ikx} - \frac{1}{\phi\left(\frac{2\pi}{pN}k\right)} \sum_{[pNx/2\pi]-K+1}^{[pNx/2\pi]+K} \sigma(x,l) e^{2\pi ikl/pN} \right|. \tag{2.31}$$

Hierbei haben wir eine allgemeine Koeffizienten-Funktion $\sigma(x, l)$ angenommen. Dies dient dazu, auch Fälle erfassen zu können, in denen sich die Koeffizienten nicht aus der Fourier-Reihenentwicklung ergeben.

Satz 2.3.2 *Mit den Voraussetzungen von Satz 2.3.1, insbesondere*

$$\sigma(x,l) = \frac{1}{\sqrt{2\pi}}\hat{\phi}\left(\frac{pN}{2\pi}x - l\right).$$

gilt:

$$\mathrm{err}(\phi,\sigma,K,p) \le \max_{k=[-N/2,N/2],x\in\mathbb{R}} \left| \frac{1}{\phi\left(\frac{2\pi}{pN}k\right)} \sum_{l:|pNx/2\pi - l|>K}{}' \sigma(x,l)e^{2\pi ikl/pN} \right|, \quad (2.32)$$

wobei $\sum'$ *bedeutet, daß für* x *mit* $pNx/2\pi \in \mathbb{Z}$ *in der Summe zusätzlich der Summand* $l = pNx/2\pi - K$ *auftritt.*

Beweis:
Satz 2.3.1 liefert unmittelbar:

$$\begin{aligned} e^{ikx} &- \frac{1}{\phi\left(\frac{2\pi}{pN}k\right)} \sum_{[pNx/2\pi]-K+1}^{[pNx/2\pi]+K} \sigma(x,l)e^{2\pi ikl/pN} \\ &= \frac{1}{\phi\left(\frac{2\pi}{pN}k\right)} \sum_{-\infty}^{[pNx/2\pi]-K} \sigma(x,l)e^{2\pi ikl/pN} \\ &\quad + \frac{1}{\phi\left(\frac{2\pi}{pN}k\right)} \sum_{[pNx/2\pi]+K+1}^{\infty} \sigma(x,l)e^{2\pi ikl/pN}. \end{aligned}$$

Daraus folgt die Behauptung. □

Bisher haben wir uns nur für den eindimensionalen Fall interessiert. In einer zweidimensionalen Fourier-Transformation tritt als Exponentialterm $e^{ikx}e^{ily}$ auf. Dies faktorisiert man naheliegenderweise in eine Approximation für e^{ikx} und eine für ^{ily}.

Satz 2.3.3 *Seien* ϕ, N *und* p *wie in Satz 2.3.1. Mit* $l_x = [pNx/2\pi]$ *und* $l_y = [pNy/2\pi]$ *sowie Approximationskoeffizienten* σ *und* τ *gelte*

$$\left| e^{ikx} - \frac{1}{\phi\left(\frac{2\pi}{pN}k\right)} \sum_{l=-K+1}^{K} \sigma(x,l)e^{i(l_x+l)k/pN} \right| \le E_x$$

sowie

$$\left|\frac{1}{\phi\left(\frac{2\pi}{pN}k\right)}e^{iky} - \sum_{l=-K+1}^{K} \tau(y,l)e^{i(l_y+l)k/pN}\right| \leq E_y.$$

mit geeigneten Fehlerschranken E_x und E_y. Dann gilt:

$$\begin{aligned} &\Bigg| e^{ik_1x}e^{ik_2y} - \frac{1}{\phi\left(\frac{2\pi}{pN}k_1\right)\phi\left(\frac{2\pi}{pN}k_2\right)} \\ &\cdot \sum_{l_1=-K+1}^{K}\sum_{l_2=-K+1}^{K} \sigma(x,l_1)\tau(y,l_2)e^{i(l_x+l_1)k/pN}e^{i(l_y+l_2)k/pN}\Bigg| \\ \leq\ & E_x + E_y + E_xE_y \end{aligned}$$

.

Beweis:
Der Beweis ist im wesentlichen eine Anwendung der Dreiecksungleichung. Zur Entlastung der Schreibweise definieren wir

$$S_x = \frac{1}{\phi\left(\frac{2\pi}{pN}k_1\right)}\sum_{l_1=-K+1}^{K} \sigma(x,l_1)e^{i(l_x+l_1)k_1/pN}$$

sowie analog

$$S_y = \frac{1}{\phi\left(\frac{2\pi}{pN}k_2\right)}\sum_{l_2=-K+1}^{K} \sigma(x,l_2)e^{i(l_y+l_2)k_2/pN}$$

und rechnen damit unmittelbar nach:

$$\begin{aligned} & \left|e^{ik_1x}e^{ik_2y} - S_xS_y\right| \\ =\ & \left|e^{ik_1x}e^{ik_2y} - e^{ik_1x}S_y + e^{ik_1x}S_y - S_xS_y\right| \\ \leq\ & \left|e^{ik_1x}\left(e^{ik_2y} - S_y\right)\right| + \left|\left(e^{ik_1x} - S_x\right)S_y\right| \\ \leq\ & \left|e^{ik_1x}\right| E_y + E_x\left|S_y - e^{ik_2y} + e^{ik_2y}\right| \\ \leq\ & E_y + E_x(1+E_y). \end{aligned}$$

□

2.4 Die Wahl der Gewichtsfunktion

Satz 2.3.2 läßt erkennen, wie man ϕ wählen muß, um einen möglichst kleinen Fehler $\text{err}(\phi, \sigma, p, K)$ zu erhalten.
Normiert man ϕ durch $\phi(0) = 1$, so wird der Fehler von zwei Faktoren bestimmt:

1. den Anteilen von $\hat{\phi}(x)$ für $|x| > K$ sowie
2. dem Minimum von $\phi(2\pi k/pN)$ für $k = -N/2, \ldots, N/2 - 1$.

Die zweite Bedingung ist wenig ergiebig: Die Normierung legt mit $\phi(0) = 1$ eine obere Schranke für das Minimum fest, und die erste Forderung ergibt wie wir sehen werden automatisch Funktionen, die im Intervall $[-\pi/p, \pi/p]$ ausreichend weit von Null entfernt sind. Unter anderem sorgt der in Abschnitt 2.2.4 beschriebene ,,Aliasing-Trick" dafür, daß typischerweise $\phi(x) > 1/3$ für $x \in [-\pi/p, \pi/p]$ gilt.
Das entscheidende Kriterium ist folglich das erste. Wir suchen also eine Funktion mit kompakten Träger, deren Fouriertransformierte möglichst alle ihre Anteile in einem vorgegebenen Intervall $[-K, K]$ um den Nullpunkt konzentriert.
Als Maximierungsaufgabe lautet dies dann wie folgt:

$$\begin{gathered} \text{maximiere} \\ \frac{||\hat{\phi} : [-K, K]||}{||\hat{\phi} : \mathbb{R}||} \\ \text{unter} \\ \phi(0) = 1 \qquad \text{und} \qquad \text{supp}(\phi) \subset [-a, a] \end{gathered} \tag{2.33}$$

$||f : V||$ bedeutet hierbei $\int_V \overline{f(x)} f(x)\, dx$, also die L_2-Norm von f, eingeschränkt auf V.
In der Terminologie der Signalverarbeitung kann man diese Maximierungsaufgabe wie folgt interpretieren: Man sucht eine Funktion, die zeitbeschränkt ist, also kompakten Träger hat, und die gleichzeitig auch noch ,,beinahe" bandbeschränkt ist.[5]

[5] Eine vollständig band- und zeitbeschränkte Funktion ist nach dem Satz von Paley Wiener eine holomorphe Funktion mit kompakten Träger. Das wird offenbar nur von $f(x) \equiv 0$ erfüllt.

Die Lösung von (2.33) ist eine *special zero-order prolate spheroidal function*. Diese Klasse von Funktionen tritt bei der Lösung des Eigenwertproblems einer Differentialgleichung zweiter Ordnung auf. Für eine umfassende Betrachtung dieser Funktionen aus Sicht der Signalverarbeitung siehe [29]. Die *prolate spheroidal functions* haben eine Reihe bemerkenswerter Eigenschaften, einfache Berechenbarkeit gehört aber bedauerlicherweise nicht dazu.
Für Anwendungen in der Signalverarbeitung werden deshalb stattdessen häufig die von J. F. Kaiser in [15] vorgeschlagen Funktionen verwendet, die sogenannten Kaiser-Bessel-Fensterfunktionen. Diese besitzen die für uns wichtigen Eigenschaften der *prolate spheroidal functions*, nämlich kompakten Träger im Ortsraum bei gleichzeitiger Konzentration im Frequenzraum. Zudem sind sie leicht berechenbar.
Die von uns im folgenden verwendete Kaiser-Bessel-Fensterfunktion lautet:

$$\phi_{KB}(x; a, K) = \chi_{[-a,a]}(x) I_0(K\sqrt{a^2 - x^2}).$$

Hierbei bezeichnet $I_0(t)$ die modifizierte Bessel-Funktion nullter Ordnung.
Ihre Fouriertransformierte kann man ebenfalls unmittelbar angeben (vgl. beispielsweise [23]):

$$\hat{\phi}_{KB}(x) = \sqrt{2/\pi}\,\frac{\sinh(a\sqrt{K^2 - x^2})}{\sqrt{K^2 - x^2}} \tag{2.34}$$

Man beachte das unterschiedliche Verhalten für $|x| < K$ und $|x| > K$. Für $|x| < K$ wird das Verhalten durch den Sinus Hyperbolicus dominiert, die Funktion ähnelt e^{-x^2}. Im Nullpunkt wird das Maximum mit

$$\phi_{KB}(0) = \frac{\sqrt{2/\pi}}{2K}\left(e^{aK} + e^{-aK}\right)$$

angenommen. Für $|x| > K$ hingegen wird die Wurzel und damit sowohl der Zähler als auch der Nenner rein imaginär. Die Funktion wird dort daher korrekterweise durch:

$$\hat{\phi}_{KB}(x) = \sqrt{2/\pi}\,\frac{\sin(a\sqrt{x^2 - K^2})}{\sqrt{x^2 - K^2}} \tag{2.35}$$

definiert. Statt des Sinus Hyperbolicus tritt für $|x| > K$ der Sinus auf. Die Funktion verhält sich für große x wie $\sin x/x$, insbesondere gilt $|\hat{\phi}_{KB}(x)| \leq \sqrt{2/\pi}$ für $|x| > K$.

Abbildung 2.2 zeigt $\phi_{KB}(x)$ und $\hat{\phi}_{KB}(x)$ bei einer typischen Wahl von $a = 1.49\pi$ und $K = 5$. Das linke Bild zeigt die Gewichtsfunktion $\phi_{KB}(x)$. Das mittlere und das rechte Bild zeigt die Fouriertransformierte $\hat{\phi}_{KB}(x)$, in der Mitte in derselben Skalierung wie links. Im rechten Bild wird das Verhalten für $|x| > K$ dargestellt. Man beachte, daß die Skalierung dort in etwa um den Faktor 10^9 kleiner ist, als in den anderen beiden Bildern.

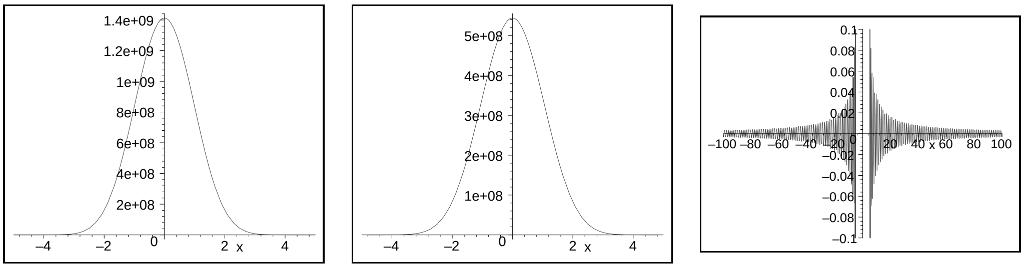


Abbildung 2.2: Kaiser-Bessel-Fensterfunktion ϕ_{KB} und $\hat{\phi}_{KB}$ in zwei Skalierungen

2.5 Fehlerabschätzung

Die Kaiser-Bessel-Gewichtsfunktion erfüllt exakt unsere Voraussetzungen: möglichst große Massenanteile von $\hat{\phi}_{KB}$ im Intervall $[-K, K]$ und nur verschwindend kleine Anteile außerhalb dieses Bereichs. Außerdem gilt $\phi_{KB}(x) > 0$ für $x \in [-a, a]$.
Deshalb wollen wir für die Kaiser-Bessel Funktion eine Fehlerabschätzung entwickeln. Wie Satz 2.3.2 zeigt, benötigen wir hierzu eine Abschätzung für die Summe

$$\sum_{|l|>b} \frac{\sin(a\sqrt{l^2 - b^2})}{\pi\sqrt{l^2 - b^2}} e^{2\pi i k l / pN}. \tag{2.36}$$

Für große l verhält sich der Summand wie $\sin(l)/l$. Leider ist die Summe nicht absolut konvergent, so daß die naheliegende Vereinfachung, sich durch Übergang zu Absolutbeträgen der $e^{2\pi i k l/pN}$ zu entledigen, nicht möglich ist. Stattdessen schätzen wir die Terme gegen $\sin(l)/l$ ab, da für die sinc-Reihe eine Summenformel existiert.

Die folgenden Lemmata sind eher technisch als schön, die wesentliche Aussage dieses Abschnittes ist die Fehlerabschätzung in Satz 2.5.13.
Für den Rest des Abschnitts sei $a > 0 \in \mathbb{R}$ und $b \in \mathbb{N}$. Das erste Lemma liefert eine Abschätzung, falls man $\sqrt{l^2 - b^2}$ in (2.36) durch l ersetzen will:

Lemma 2.5.1 *Für $l > b > 0$ gilt mit*

$$R(l) := l - \sqrt{l^2 - b^2} \tag{2.37}$$

$$R(l) \leq \frac{b^2}{l}$$

denn

$$\begin{aligned} R(l) &= l\left(1 - \sqrt{1 - (b/l)^2}\right) \\ &= l\frac{\left(1 - \sqrt{1 - (b/l)^2}\right)\left(1 + \sqrt{1 - (b/l)^2}\right)}{1 + \sqrt{1 - (b/l)^2}} \\ &= l\frac{(b/l)^2}{1 + \sqrt{1 - (b/l)^2}} \leq \frac{b^2}{l}. \end{aligned}$$

□

Die einfachen Beweise der folgenden beiden Lemmata finden sich in jedem gängigen Analysis-Lehrbuch, beispielsweise in [10].

Lemma 2.5.2 *Sei $f : [c, d] \to \mathbb{R}$ eine stetig differenzierbare Funktion. Für $k \in \mathbb{R}$ gilt dann*

$$\lim_{|k| \to \infty} \int_c^d f(x) \sin kx \, dx = 0.$$

Lemma 2.5.3 *Für $x \in]0, 2\pi]$ gilt*

$$\sum_{k=1}^{\infty} \frac{\sin(kx)}{k} = \frac{\pi - x}{2}.$$

Lemma 2.5.4 *Für alle $x \in \mathbb{R}, b \in \mathbb{N}$ gilt:*

$$\left| \sum_{k=b+1}^{\infty} \frac{\sin kx}{k} \right| \leq \frac{\pi}{2} + \ln b + 1$$

Beweis:
Da der Sinus 2π-periodisch ist, liefert Lemma 2.5.3 für alle $x \in \mathbb{R} \backslash 2\pi\mathbb{Z}$:

$$\sum_{k=1}^{\infty} \frac{\sin(kx)}{k} \leq \pi/2.$$

Für $x \in 2\pi\mathbb{Z}$ verschwindet der Sinus, und die Aussage gilt trivialerweise. Damit hat man

$$\begin{aligned} \left| \sum_{k=b+1}^{\infty} \frac{\sin kx}{k} \right| &\leq \left| \sum_{k=1}^{\infty} \frac{\sin kx}{k} \right| + \left| \sum_{k=1}^{b} \frac{\sin kx}{k} \right| \\ &\leq \frac{\pi}{2} + 1 + \sum_{k=2}^{b} \frac{1}{k} \leq \frac{\pi}{2} + 1 + \int_1^b \frac{1}{k}\, dk \\ &= \frac{\pi}{2} + \ln b + 1. \end{aligned}$$

□

Lemma 2.5.5 *Für $a > 0$, und $R(l)$ wie in (2.37) definiert, gilt:*

$$\sum_{l=b+1}^{\infty} \left| \frac{\sin aR(l)}{l} \right| \leq 1 + \ln ab.$$

Beweis:
Die Abschätzung für $R(l)$ aus Lemma 2.5.1 liefert

$$\begin{aligned} \sum_{l=b+1}^{\infty} \left| \frac{\sin aR(l)}{l} \right| &\leq \sum_{l=b+1}^{\infty} \left| \frac{\max\{1, ab^2/l\}}{l} \right| \\ &\leq \sum_{l=b+1}^{[ab^2]} \frac{1}{l} + \sum_{l=[ab^2]+1}^{\infty} \frac{ab^2}{l^2} \\ &\leq \int_b^{ab^2} \frac{dt}{t} + ab^2 \int_{ab^2}^{\infty} \frac{dt}{t^2} = \ln(ab^2) - \ln(b) + 1 \end{aligned}$$

. □

Lemma 2.5.6 *Mit $a > 0$ und $R(l)$ wie oben gilt:*

$$|\cos aR(l) - 1| \leq \min\left\{2, \frac{a^2b^4}{2l^2}\right\}$$

und

$$\sum_{l=b+1}^{\infty} \left|\frac{\cos aR(l) - 1}{l}\right| \leq 2\ln(ab/2) + 1.$$

Beweis:
Taylorentwicklung von $\cos aR(l)$ um 0 liefert mit Lagrange-Restglied

$$|1 - \cos aR(l)| = \left|1 + \xi_l^2/2 - 1\right| \qquad \text{mit } 0 \leq \xi_l \leq aR(l).$$

Wegen Lemma 2.5.1 gilt $aR(l) \leq ab^2/l$ und damit der erste Teil der Behauptung. Aufsummieren liefert

$$\begin{aligned}
\sum_{l=b+1}^{\infty} \left|\frac{\cos aR(l) - 1}{l}\right| &\leq \sum_{l=b+1}^{\infty} \frac{\min\{2, a^2b^4/2l^2\}}{l} \\
&\leq \sum_{l=b+1}^{[ab^2/2]} \frac{2}{l} + \frac{a^2b^4}{2} \sum_{[ab^2/2]+1}^{\infty} \frac{1}{l^3} \\
&\leq \int_b^{ab^2/2} \frac{2}{t}\, dt + \frac{a^2b^4}{2} \int_{ab^2/2}^{\infty} \frac{dt}{t^3} \\
&= 2(\ln(ab^2/2) - \ln(b)) + \frac{a^2b^4}{4} \frac{4}{a^2b^4}
\end{aligned}$$

und damit den zweiten Teil der Behauptung. □

Corollar 2.5.7 *Für alle $x \in \mathbb{R}$ gilt:*

$$\left|\sum_{l=b+1}^{\infty} \frac{\sin lx}{l} \cos aR(l)\right| \leq \pi/2 + \ln b + 2\ln(ab/2) + 2.$$

Beweis:

$$\begin{aligned}\left|\sum_{l=b+1}^{\infty} \frac{\sin lx}{l} \cos aR(l)\right| &= \left|\sum_{l=b+1}^{\infty} \left\{\frac{\sin lx}{l} + \frac{\sin lx}{l}(\cos(aR(l)) - 1)\right\}\right| \\ &\leq \left|\sum_{l=b+1}^{\infty} \frac{\sin lx}{l}\right| + \sum_{l=b+1}^{\infty} |\cos(aR(l)) - 1|\end{aligned}$$

Schätzt man die erste Summe mit Lemma 2.5.4 und die zweite Summe mit Lemma 2.5.6 ab, so ergibt sich die Behauptung. □

Damit können wir $\sqrt{l^2 - b^2}$ im Zähler abschätzen.

Lemma 2.5.8 *Sei*

$$A_\pm := \sin(a\sqrt{l^2 - b^2} \pm tl). \tag{2.38}$$

Dann gilt

$$\left|\sum_{l=b+1} \frac{A_\pm}{l}\right| \leq \pi/2 + \ln(ab^2) + 2\ln(ab/2) + 3$$

Beweis:
Wegen der Definition von $R(l)$ gilt

$$\begin{aligned}A_\pm &= \sin(a\sqrt{l^2 - b^2} \pm tl) = \sin(l(a \pm t) - al + a\sqrt{l^2 - b^2}) \\ &= \sin(l(a \pm t) - aR(l)) \\ &= \sin(l(a \pm t))\cos(aR(l)) - \cos(l(a \pm t))\sin(aR(l))\end{aligned}$$

und damit

$$\begin{aligned}\left|\sum_{l=b+1}^{\infty} \frac{A_\pm}{l}\right| &\leq \left|\sum_{l=b+1} \frac{\sin(l(a \pm t))\cos(aR(l))}{l}\right| \\ &\quad + \sum_{l=b+1} \left|\frac{\cos(l(a \pm t))\sin(aR(l))}{l}\right| \\ &\leq \pi/2 + \ln b + 2\ln(ab/2) + 2 \ + \ 1 + \ln ab.\end{aligned}$$

Hierbei wurde die erste Summe mit Corollar 2.5.7 und die zweite Summe mit Lemma 2.5.5 abgeschätzt. □

Jetzt benötigen wir noch eine Abschätzung für $\sqrt{l^2 - b^2}$ im Nenner.

Lemma 2.5.9 *Für $b \in \mathbb{N}$ und*

$$c_b := \frac{1}{\sqrt{1 - b^2/(b+1)^2}(1 + \sqrt{1 - b^2/(b+1)^2})} \tag{2.39}$$

gilt:

$$\sum_{l=b+1}^{\infty} \left| \frac{1}{\sqrt{l^2 - b^2}} - \frac{1}{l} \right| \leq c_b/2.$$

Für $b \in \mathbb{N} < 20$ ist $c_b < 5/2$.

Beweis:
Für $l \geq b+1$ gilt:

$$\begin{aligned}
\frac{1}{\sqrt{l^2 - b^2}} - \frac{1}{l} &= \frac{1}{l\sqrt{1 - (b/l)^2}} - \frac{1}{l} = \frac{1 - \sqrt{1 - (b/l)^2}}{l\sqrt{1 - (b/l)^2}} \\
&= \left(1 - \sqrt{1 - (b/l)^2}\right) \left(\frac{1 + \sqrt{1 - (b/l)^2})}{l\sqrt{1 - (b/l)^2}(1 + \sqrt{1 - (b/l)^2}} \right) \\
&= \frac{b^2}{l^3} \frac{1}{\sqrt{1 - (b/l)^2}(1 + \sqrt{1 - (b/l)^2})} \\
&\leq \frac{c_b b^2}{l^3}
\end{aligned}$$

und damit

$$\sum_{l=b+1}^{\infty} \left| \frac{1}{\sqrt{l^2 - b^2}} - \frac{1}{l} \right| \leq c_b b^2 \int_b^{\infty} \frac{dt}{t^3} = c_b/2.$$

□

Lemma 2.5.10 *Für $a > 0, b \in \mathbb{N}$, $t \in \mathbb{R}$ und $A_\pm$ wie in (2.38) gilt:*

$$\begin{aligned}
&\left| \sum_{l=b+1}^{\infty} \frac{\sin(a\sqrt{l^2 - b^2})}{l} \cdot \cos tl \right| \\
&\leq \pi/2 + \ln(ab^2) + 2\ln(ab/2) + 3.
\end{aligned}$$

Beweis:
Wegen $2\sin\alpha\cos\beta = \sin(\alpha-\beta)+\sin(\alpha+\beta)$ gilt

$$\begin{aligned}
&\left|\sum_{l=b+1}^{\infty}\frac{\sin(a\sqrt{l^2-b^2})\cos(tl)}{l}\right| \\
= & \left|\sum_{l=b+1}^{\infty}\frac{\sin(a\sqrt{l^2-b^2}-tl)+\sin(a\sqrt{l^2-b^2}+tl)}{2l}\right| \\
\leq & \left|\sum_{l=b+1}^{\infty}\frac{A_-}{2l}\right| + \left|\sum_{l=b+1}^{\infty}\frac{A_+}{2l}\right| \\
\leq & \ \pi/2+\ln(ab^2)+2\ln(ab/2)+3,
\end{aligned}$$

gemäß Lemma 2.5.8. □

Damit ergibt sich:

Satz 2.5.11 *Für $a>0, b\in\mathbb{N}, t\in\mathbb{R}$ und c_b wie in (2.39) gilt:*

$$\sum_{|l|>b}\frac{\sin(a\sqrt{l^2-b^2})}{\sqrt{l^2-b^2}}\cdot e^{itl} \leq \pi+2\ln(ab^2)+4\ln(ab/2)+6+c_b$$

Beweis:

$$\begin{aligned}
&\left|\sum_{|l|>b}^{\infty}\frac{\sin(a\sqrt{l^2-b^2})}{\sqrt{l^2-b^2}}\cdot e^{itl}\right| \\
= & \left|\sum_{l=b+1}^{\infty}\frac{\sin(a\sqrt{l^2-b^2})}{\sqrt{l^2-b^2}}\cdot\left(e^{itl}+e^{-itl}\right)\right| \\
= & \ 2\left|\sum_{l=b+1}^{\infty}\frac{\sin(a\sqrt{l^2-b^2})}{\sqrt{l^2-b^2}}\cdot cos(tl)\right| \\
= & \ 2\left|\sum_{l=b+1}^{\infty}\frac{\sin(a\sqrt{l^2-b^2})}{l}\cdot\cos(tl)+\sin(a\sqrt{l^2-b^2})\cos(tl)\left(\frac{1}{\sqrt{l^2-b^2}}-\frac{1}{l}\right)\right| \\
\leq & \ 2\left|\sum_{l=b+1}^{\infty}\frac{\sin(a\sqrt{l^2-b^2})}{l}\cdot cos(tl)\right|+2\sum_{l=b+1}^{\infty}\left|\frac{1}{\sqrt{l^2-b^2}}-\frac{1}{l}\right| \qquad (2.40) \\
\leq & \ 2\left(\pi/2+\ln(ab^2)+2\ln(ab/2)+3\ +\ c_b/2\right).
\end{aligned}$$

Hierbei haben wir in (2.40) für die erste Summe Lemma 2.5.10 und für die zweite Summe Lemma 2.5.9 angewendet. $\square$

Uns interessiert der Fall $a = \alpha\pi$ mit $\alpha < 2 - 1/p < 2$ und $b = K < 15$. Setzt man diese Konstanten in die Abschätzung aus Satz 2.5.11 ein, so ergibt sich insgesamt:

Corollar 2.5.12 *Für* $1 < b \leq 15$ *und* $a \leq 2\pi$ *gilt:*

$$\left| \sum_{|l|>b} \frac{\sin(a\sqrt{l^2 - b^2})}{\sqrt{l^2 - b^2}} \cdot e^{itl} \right| < 29.5.$$

Der genaue Wert dieser Konstante ist nicht allzu relevant, da die Größenordnung des Fehler durch das Verhalten von ϕ_{KB} bestimmt wird, wie der folgende Satz mit der endgültigen Fehlerabschätzung zeigt:

Satz 2.5.13 *Sei* $p > 1, 1 \leq \alpha < 2 - 1/p, N \in \mathbb{N}$. *Sei*

$$\phi_{KB}(x) = \chi_{[-\alpha\pi,\alpha\pi]}(x) I_0(K\sqrt{(\alpha\pi)^2 - x^2})$$

und

$$\sigma(x, l) = \frac{1}{\sqrt{2\pi}} \hat{\phi}_{KB} \left(\frac{pN}{2\pi} x - l \right).$$

Dann gilt

$$\mathrm{err}(\phi, \sigma, p, K) \leq \frac{36}{\pi I_0(K\pi\sqrt{\alpha^2 - p^{-2}})},$$

also

$$\left| e^{ikx} - \frac{1}{\phi\left(\frac{2\pi}{pN}k\right)} \sum_{[pNx/2\pi]-K+1}^{[pNx/2\pi]+K} \sigma(x, l) e^{2\pi ikl/pN} \right| \leq \frac{36}{\pi I_0(K\pi\sqrt{\alpha^2 - p^{-2}})}. \quad (2.41)$$

Beweis:
Die Definition von σ ergibt

$$\sigma(x, l) = \frac{\sinh(\alpha\pi\sqrt{K^2 - (pNx/2\pi - l)^2})}{\pi\sqrt{K^2 - (pNx/2\pi - l)^2}} \quad (2.42)$$

Setzt man die Summen-Abschätzung aus Corollar 2.5.12 in Satz 2.3.2 ein, so erhält man

$$\begin{aligned}
& \operatorname{err}(\phi_{KB}, \sigma, K, p) \\
= & \max_{k=[-N/2,N/2], x \in \mathbb{R}} \left| \frac{1}{\phi_{KB}\left(\frac{2\pi}{pN}k\right)} \sum_{l:|pNx/2\pi - l|>K}{}' \sigma(x,l) e^{2\pi i k l/pN} \right| \\
\leq & \max_{k=[-N/2,N/2]} \frac{1}{\pi \phi_{KB}\left(\frac{2\pi}{pN}k\right)} \left|29.5 + \hat{\phi}_{KB}(K)\right| \qquad (2.43) \\
= & \frac{1}{\pi \phi_{KB}(\pi/p)} |29.5 + \alpha\pi| \\
\leq & \frac{36}{\pi \phi_{KB}(\pi/p)}
\end{aligned}$$

Die Definition von ϕ_{KB} ergibt schließlich die Behauptung. Der zusätzliche Term $\hat{\phi}_{KB}(K)$ in (2.43) tritt aufgrund der Definition von $\sum'$ in (2.32) auf. □

Abbildung 2.3 zeigt einige Werte dieser Fehlerabschätzung für $p = 1.5$, $p = 2$ sowie $p = 4$. Es sind die Fehlerabschätzungen aus (2.41) sowie der numerisch ermittelte tatsächliche Fehler, jeweils in Abhängigkeit von K, aufgetragen. Außerdem ist die IEEE Maschinengenauigkeit bei einfach und doppelt genauer Arithmetik markiert.
Man sieht deutlich das exponentielle Abfallen des Fehlers mit zunehmendem K, für $p = 2$ beispielsweise bewirkt eine Erhöhung von K um eins zwei zusätzliche Dezimalstellen Genauigkeit.
Der tatsächliche Fehler stimmt recht gut mit der Fehlerabschätzung überein, ab 10^{-12} ergibt sich jedoch aufgrund der beschränkten Maschinengenauigkeit keine wesentliche Verbesserung mehr. Dies liegt unter anderem auch an der beschränkten Genauigkeit, mit der die Bessel-Funktionen berechnet wurden.
Bemerkenswert ist, daß der Einfluß der Oversampling-Rate p geringer ist, als der der Interpolations-Länge K. Deshalb ist es nicht erforderlich, einen Oversampling-Faktor größer als $p = 2$ zu verwenden, auch mit einem Wert von $p = 1.5$ läßt sich eine zumeist ausreichende Genauigkeit erzielen. Man kann in der Praxis p daher so wählen, daß sich für $\bar{N} = pN$ mit $1.5 \leq p \leq 2$ ein Wert ergibt, für den eine effiziente FFT-Routine existiert.

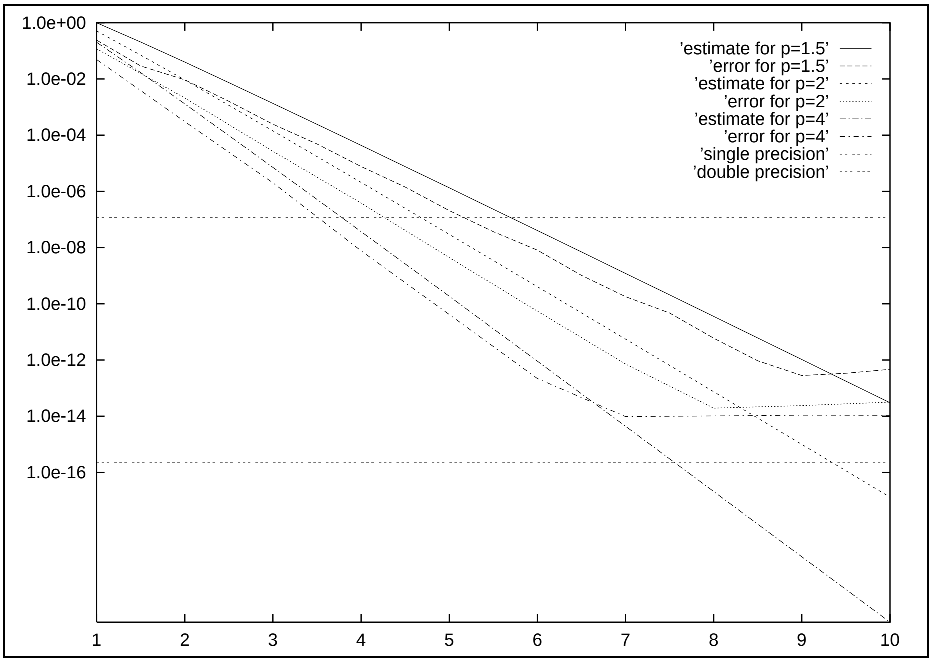


Abbildung 2.3: Approximationsfehler und Fehlerabschätzung bei Oversampling $p = 1.5; 2; 4$ in Abhängigkeit von K.

2.6 Diskrete Betrachtungsweise

Betrachten wir nochmals die Definition des Approximationsfehlers:

$$\mathrm{err}(\phi, \sigma, K, p) = \max_{k \in [-N/2, N/2], x \in \mathbb{R}} \left| e^{ikx} - \frac{1}{\phi\left(\frac{2\pi}{pN} k\right)} \sum_{[pNx/2\pi]-K+1}^{[pNx/2\pi]+K} \sigma(x, l) e^{2\pi i k l / pN} \right|$$

Dies kann man auch wie folgt interpretieren: Seien die Gewichtsfunktion ϕ und die Konstanten K und p vorgegeben. Zu jeder nichtäquidistanten Stützstelle x_0 sind dann diejenigen $2K$ Koeffizienten $\sigma_l(x_0)$ gesucht, so daß der Approximationsfehler

$$\left| e^{ikx_0} - \frac{1}{\phi\left(\frac{2\pi}{pN} k\right)} \sum_{[pNx_0/2\pi]-K+1}^{[pNx_0/2\pi]+K} \sigma(x_0, l) e^{2\pi i k l / pN} \right|$$

für $k = -N/2, \ldots, N/2-1$ möglichst klein wird. Dies kann man als überbestimmtes Gleichungssystem mit N Gleichungen ($k = -N/2, \ldots, N/2-1$) für $2K$ Unbekannte (die $\sigma(x_0, l)$) auffassen. Das Gleichungssystem ist massiv überbestimmt (typischerweise gilt $N \gg 100$ und $K < 15$), aber man kann mit der Methode der kleinsten Quadrate natürlich eine Lösung finden, die

$$\sum_{k=-N/2}^{N/2-1} \left| e^{ikx_0} - \frac{1}{\phi\left(\frac{2\pi}{pN}k\right)} \sum_{l=[pNx_0/2\pi]-K+1}^{[pNx_0/2\pi]+K} \sigma(x_0, l) e^{2\pi ikl/pN} \right|^2$$

minimiert.
Dazu definiert man einen Vektor b durch

$$b_k = e^{ikx_0} \qquad \text{für } k = -N/2, \ldots, N/2-1$$

und eine Matrix C durch

$$C_{kl} = \frac{1}{\phi\left(\frac{2\pi}{pN}k\right)} \cdot e^{2\pi ik([pNx_0/2\pi]+l)/pN}$$

für $k = -N/2, \ldots, N/2-1$ und $l = -K+1, \ldots, K$.
Die kleinste Quadrate Lösung ergibt sich dann als Lösung von

$$C^T C\sigma = C^T b.$$

Auf diese Weise kann man zu jeder beliebigen Gewichtsfunktion ϕ mit $\phi\left(\frac{2\pi}{pN}k\right) \neq 0$ die Koeffizienten ermitteln. Hierzu ist allerdings für jede einzelne Stützstelle ein lineares Gleichungssystem zu lösen, was jedesmal $\mathcal{O}(NK^2)$ Rechenoperationen erfordert, wenn man die kleinste Quadrate Lösung effizient über eine QR-Zerlegung berechnet.
In numerischen Experimenten hat sich herausgestellt, daß man auf diese Weise zu der Kaiser-Bessel-Gewichtsfunktion Koeffizienten finden kann, die zu einem etwa um den Faktor 5 kleineren Approximationsfehler führen, als bei Verwendung der Fouriertransformierten.
Wenn man zu jeder beliebigen Gewichtsfunktion ϕ mit $\phi\left(\frac{2\pi}{pN}k\right) \neq 0$ die Koeffizienten ermitteln kann, liegt es nahe, nach Alternativen zur Kaiser-Bessel-Gewichtsfunktion zu suchen.
Nimmt man allgemein ein gerades Polynom vom Grad $\leq 2P$ mit $\phi(0) = 1$ an:

$$\phi_a(t) = \sum_{k=0}^{P} a_k t^{2k},$$

mit $a_0 = 1$, so kann man zu jedem dieser Polynome die Koeffizienten σ per kleinste-Quadrate Lösung ermitteln und damit dann numerisch den Approximationsfehler (2.31) bestimmen. Somit erhält man ein P-dimensionales Optimierungsproblem:
Gesucht sind Koeffizienten $a_1, \ldots, a_P$, so daß

$$\sum_{j=1}^{M} \left| e^{ikx_j} - \frac{1}{\phi_a\left(\frac{2\pi}{pN}k\right)} \sum_{[pNx_j/2\pi]-K+1}^{[pNx_j/2\pi]+K} \sigma_a(x_j, l) e^{2\pi ikl/pN} \right|$$

minimal wird.
Diese Optimierungsaufgabe wurde mit einem simulated-annealing Algorithmus behandelt. Die Gewichtsfunktion $\phi_a^{opt}(x)$, die man auf diese Weise erhält, ermöglicht einen um ein bis zwei Zehnerpotenzen kleineren Approximationsfehler als die Kaiser-Bessel-Gewichtsfunktion mit analytischen Koeffizienten aus der Fourier-Transformation.
Dies bedeutet, daß man auf diese Weise zwar eine Verbesserung der Approximation erzielt, allerdings nur in einer Größenordnung, die man auch durch Erhöhung von K um eins erreicht.
Kleine Geschwindigkeits-Vorteile bei der Auswertung stehen somit großen Nachteilen bei der Initialisierung entgegen, so daß man in Normalfall der analytischen Variante den Vorzug geben dürfte. Der Vergleich zwischen diskreter und analytischer Berechnung zeigt in jedem Fall, daß die Kaiser-Bessel-Gewichtsfunktion bereits sehr dicht am Optimum liegt.

2.7 Detaillierte Beschreibung der Algorithmen

Damit haben wir jetzt das Rüstzeug beisammen, um die Algorithmen auszuformulieren. Wir gehen von M in $[-\pi, \pi]$ freiverteilten Stützstellen x_i aus mit denen wir eine Fouriertransformierte der Länge N berechnen wollen. Dazu wählen wir eine FFT-Länge $\bar{N} \in \mathbb{N}$:

$$\bar{N} = pN, \qquad p > 1, \text{ typischerweise } 1.5 \leq p \leq 2, \; \bar{N} \text{ FFT-freundlich} \quad (2.44)$$

Der zweite Parameter, der die Genauigkeit der Approximation bestimmt, ist $K \in \mathbb{N}$. Dieser Parameter ist in Abhängigkeit von p zu wählen, beispielsweise durch

$$
\begin{aligned}
p = 2 &\quad : \quad \text{wähle } K = 3 \text{ bzw. } 6 \text{ für einfache bzw. doppelte Genauigkeit} \\
p = 1.5 &\quad : \quad \text{wähle } K = 4 \text{ für einfache und } K = 7 \text{ für doppelte Genauigkeit}
\end{aligned}
\tag{2.45}
$$

Einen Überblick über den jeweiligen Fehler je nach Wahl von p und K liefert Tabelle 2.6 in Abschnitt 2.9.

2.7.1 Eindimensional, nichtäquidistantes Ergebnis

Ziel ist die Berechnung von:

$$\hat{f}_j = \sum_{k=-N/2}^{N/2-1} f_k e^{ikx_j} \qquad \text{für } j = 0 \dots M-1,\ x_j \in [-\pi, \pi] \tag{2.46}$$

Die benötigten Werte der Gewichtsfunktion speichern wir in einem Vektor $w \in \mathbb{R}^N$:

$$w_i = \phi(2\pi i/\bar{N}) \qquad i = -N/2 \dots N/2-1 \tag{2.47}$$

Man definiert:

$$\mu_l = \left[x_l \cdot \frac{\bar{N}}{2\pi} \right] \qquad \text{für } l = 0, \dots, M-1 \tag{2.48}$$

Dies entspricht der nächstgelegenen äquidistanten Stützstelle, skaliert auf $[-\bar{N}/2, \bar{N}/2]$.
Die Koeffizienten $\sigma(x_j, l)$ ergeben eine Koeffizientenmatrix $\sigma \in \mathbb{R}^{M \times 2K}$, definiert durch

$$\sigma_{jk} = \frac{1}{\sqrt{2\pi}} \hat{\phi} \left(\frac{pN}{2\pi} x_j - (\mu_j + l) \right), \tag{2.49}$$

für $j = 0, \dots, M-1$ und $l = -K+1, \dots, K$. Jede Zeile von σ liefert damit die Approximationskoeffizienten bezüglich einer Stützstelle x_j. Alternativ kann man wie in Abschnitt 2.6 erläutert auch die kleinste Quadrate Lösung verwenden.
Für den Fehler gilt:

$$\left| e^{ikx_j} - \frac{1}{w_k} \sum_{l=-K+1}^{K} \sigma_{jl} e^{2\pi k(\mu_j + l)/\bar{N}} \right| \leq \operatorname{err}(\phi, \sigma, p, K), \tag{2.50}$$

bzw. die Abschätzung aus Satz 2.5.13. Setzt man diese Approximation für e^{ikx_j} in (2.46) ein, so ergibt sich:

$$\begin{aligned}
\tilde{f}_j &= \sum_{k=-N/2}^{k=N/2-1} \underbrace{\alpha_k \cdot \frac{1}{w_k}}_{:=u_k} \cdot \sum_{l=-K+1}^{K} \sigma_{jl} \cdot e^{i(\mu_j+l)2\pi k/\bar{N}} \\
&= \sum_{l=-K+1}^{K} \sigma_{jl} \underbrace{\sum_{k=-N/2}^{k=N/2-1} u_k \cdot e^{i(\mu_j+l)2\pi k/\bar{N}}}_{:=U_{\mu_j+l}} \qquad (2.51) \\
&= \sum_{l=-K+1}^{K} \sigma_{jl} U_{\mu_j+l}
\end{aligned}$$

Die eigentliche Berechnung erfolgt also in drei Schritten:

Eindimensionale Fourier-Transformation bei nichtäquidistantem Ergebnisvektor:
Gegeben seien die Eingangsdaten α und Parameter w gemäß (2.47), σ gemäß 2.49, μ gemäß (2.48), $\bar{N}$ gemäß (2.44) und K gemäß (2.45). Berechne

$$\hat{f}_j = \sum_{k=-N/2}^{N/2-1} \alpha_k e^{ikx_j} \qquad \text{für } j = 0 \ldots M-1 \tag{2.52}$$

durch

1. Skalierung der Eingangsdaten

   $$u_k = \alpha_k / w_k \qquad \text{für } k = -N/2 \ldots N/2 - 1$$

   (Rechenaufwand: $2N$ reelle Multiplikationen)

2. Berechnung von

   $$U_j = \sum_{k=-N/2}^{N/2-1} u_k \cdot e^{2\pi i k j/\bar{N}}$$

   per FFT. (Rechenaufwand $\mathcal{O}(\bar{N} \log \bar{N})$ Rechenoperationen)

3. Interpolation auf das nichtäquidistante Gitter:

   $$\tilde{f}_j = \sum_{l=-K+1}^{K} \sigma_{jl} U_{\nu_j + l} \tag{2.53}$$

   (Rechenaufwand: $4KM$ reelle Multiplikationen und ebensoviele Additionen).

Als Fehlerabschätzung ergibt sich:

$$\begin{aligned}
\left| f_j - \tilde{f}_j \right| &= \left| \sum_{k=-N/2}^{N/2-1} \alpha_k \left(e^{ikx_j} - \frac{1}{w_k} \sum_{k=-K+1}^{K} \sigma_{jk} e^{2\pi i k (\mu_j + l)/\bar{N}} \right) \right| \\
&\leq \operatorname{err}(\phi, \sigma, p, K) \sum_{k=-N/2}^{N/2-1} |\alpha_k|.
\end{aligned}$$

2.7.2 Zweidimensional, nichtäquidistantes Ergebnis

Die Verallgemeinerung auf den mehrdimensionalen Fall bereitet keine größeren Schwierigkeiten:
Ziel ist jetzt die Berechnung von

$$\hat{f}_j = \sum_{k=-N/2}^{N/2-1} \sum_{l=-N/2}^{N/2-1} f_{lk} e^{ikx_j} e^{ily_j} \qquad \text{für } j = 0 \ldots M-1. \tag{2.54}$$

x_j und y_j sind jetzt die $x-$ bzw. $y-$ Koordinaten der nichtäquidistanten Stützstellen. Diese werden nur mit *einem* Index j versehen, obwohl sie natürlich im Allgemeinen eine zweidimensionale Fläche aufspannen, typischerweise gilt $M = N^2$.
Die zweidimensionale DFT läßt sich als Tensorprodukt oder Hintereinanderausführung zweier eindimensionaler DFTs schreiben:

$$\begin{aligned} \hat{f}_j &= \sum_{k=-N/2}^{N/2-1} \left(\sum_{l=-N/2}^{N/2-1} f_{lk} e^{ikx_j} \right) e^{ily_j} \\ &= \sum_{l=-N/2}^{N/2-1} \left(\sum_{k=-N/2}^{N/2-1} f_{lk} e^{ily_j} \right) e^{ikx_j}. \end{aligned}$$

Gewichtsvektor w, Stützstellen μ und Koeffizienten σ kann man aus den obigen Definitionen (2.47), (2.48) und (2.49) übernehmen. Zusätzlich benötigen wir jetzt aber auch Stützstellen und Koeffizienten für die y-Koordinaten. Dazu definieren wir analog zu (2.48):

$$\nu_l = \left[y_l \cdot \frac{\bar{N}}{2\pi} \right] \qquad \text{für } l = 0, \ldots, M-1 \tag{2.55}$$

und analog zu (2.49) Koeffizienten $\tau \in \mathbb{R}^{M \times 2K}$:

$$\tau_{jl} = \frac{1}{\sqrt{2\pi}} \hat{\phi} \left(\frac{pN}{2\pi} y_j - (\nu_j + l) \right). \tag{2.56}$$

Auch hier kann natürlich alternativ wieder die kleinste Quadrate Lösung verwendet werden.

Satz 2.3.3 liefert für den zweidimensionalen Fall:

$$\begin{aligned}
& \left| e^{ikx_j} e^{ily_j} - \frac{1}{w_k \cdot w_l} \sum_{l_1=-K+1}^{K} \sum_{l_2=-K+1}^{K} \sigma_{jl_1} \cdot \tau_{jl_2} e^{2\pi k(\mu_j + l_1)/\bar{N}} e^{2\pi l(\nu_j + l_2)/\bar{N}} \right| \\
\leq \; & 2\mathrm{err} + \mathrm{err}^2,
\end{aligned}$$

mit $\mathrm{err} = \max\left\{\mathrm{err}(\phi, \sigma, p, K), \mathrm{err}(\phi, \tau, p, K)\right\}$. Setzt man diese Approximation für $e^{ikx_j} e^{ily_j}$ in (2.54) ein, so ergibt sich

$$\begin{aligned}
\tilde{f}_j &= \sum_{k,l=-N/2}^{N/2-1} \frac{\alpha_{lk}}{\underbrace{w_l \cdot w_k}_{:=u_{lk}}} \sum_{l_1=-K+1}^{K} \sum_{l_2=-K+1}^{K} \sigma_{jl_1} \cdot \tau_{jl_2} e^{2\pi k(\mu_j^1 + l_1)/\bar{N}} e^{2\pi l(\mu_j^2 + l_2)/\bar{N}} \\
&= \sum_{l_1,l_2=-K+1}^{K} \sigma_{jl_1} \cdot \tau_{jl_2} \underbrace{\sum_{k,l=-N/2}^{N/2-1} u_{lk} \cdot e^{2\pi ik(\mu_j^1 + l_1)/\bar{N}} e^{2\pi il(\mu_j^2 + l_2)/\bar{N}}}_{:=U_{\mu_j^1 + l_1, \mu_j^2 + l_2}} \\
&= \sum_{l_1,l_2=-K+1}^{K} \sigma_{jl_1} \cdot \tau_{jl_2} U_{\mu_j^1 + l_1, \mu_j^2 + l_2}.
\end{aligned}$$

Die unübersichtliche Index-Anhäufung täuscht ein wenig darüber hinweg, daß es sich hierbei um eine völlig kanonische Erweiterung des eindimensionalen Falles handelt.

Die Berechnung erfolgt weiterhin in drei Schritten:

Zweidimensionale Fourier-Transformation bei nichtäquidistantem Ergebnisvektor:
Gegeben seien die Eingangsdaten α und Parameter w gemäß (2.47), σ und τ gemäß (2.49) und (2.56), μ und ν gemäß (2.48) und (2.55) , $\bar{N}$ gemäß (2.44) und K gemäß (2.45). Berechne

$$\hat{f}_j = \sum_{k=-N/2}^{N/2-1} \sum_{l=-N/2}^{N/2-1} \alpha_{kl} e^{ikx_j} e^{ily_j} \qquad \text{für } j = 0 \ldots M-1 \tag{2.57}$$

durch

1. Skalierung der Eingangsdaten

$$u_{kl} = \frac{\alpha_{kl}}{w_k \cdot w_l} \qquad \text{für } k,l = -N/2 \ldots N/2-1$$

(Rechenaufwand: $3N^2$ reelle Multiplikationen)

2. Berechnung von

$$U_{mn} = \sum_{k,l=-N/2}^{N/2-1} u_{kl} \cdot e^{2\pi ikm/\bar{N}} e^{2\pi iln/\bar{N}}$$

per FFT. (Rechenaufwand $\mathcal{O}((\bar{N})^2 \log \bar{N})$ Rechenoperationen)

3. Interpolation auf das nichtäquidistante Gitter:

$$\tilde{f}_j = \sum_{l_1=-L}^{K} \sigma_{jl_1} \sum_{l_2=-K+1}^{K} \tau_{jl_2} U_{\mu_j+l_1,\nu_j+l_2} \tag{2.58}$$

(Rechenaufwand: $8K^2M + 4KM$ reelle Multiplikationen und ebensoviele Additionen).

Wenn man die beiden Matrizen σ_{j,l_1} und $\tau_{j,l2}$ durch

$$T_{j,l_1,l_2} = \sigma_{j,l1} \tau_{j,l2}$$

in einem einzigen Tensor zusammenfaßt, lassen sich bei der Berechnung von (2.58) $4KM$ Additionen und Multiplikationen einsparen. Der Tensor

benötigt dafür aber auch $M \cdot K^2$ Einträge, was bei typischen Werten von $M = 512^2, K = 5$ bereits 200MByte ausmacht. Der geringe Zeitgewinn dürfte deshalb in dem meisten Anwendungen den immensen Speicherverbrauch und die längere Initialisierungsdauer nicht aufwiegen.
In jedem Falle ergibt sich als Fehlerabschätzung:

$$\left|f_j - \tilde{f}_j\right| \leq \left(2\mathrm{err} + \mathrm{err}^2\right) \cdot \sum_{k,l=-N/2}^{N/2-1} |\alpha|_{kl}.$$

2.7.3 Eindimensional, nichtäquidistante Daten

Wie aus (2.4) ersichtlich, ergeben sich die Operatoren für nichtäquidistante Daten durch Transposition aus denen für nichtäquidistante Frequenzen.
Die Algorithmen ähneln sich deshalb — im Wesentlichen ist die Reihenfolge von Skalierung, FFT und Interpolation umgekehrt.
Ziel ist nun die Berechnung von:

$$\hat{f}_j = \sum_{k=0}^{M-1} f_k e^{ix_k j} \qquad \text{für } j = -N/2 \ldots N/2-1 \tag{2.59}$$

Die Definitionen für w,μ und σ können wir übernehmen, der Approximationsfehler lautet:

$$\left| e^{ijx_k} - \frac{1}{w_j} \sum_{l=-K+1}^{K} \sigma_{kl} e^{2\pi j(\mu_k + l)/\bar{N}} \right| < \mathrm{err}.$$

Setzt man diese Approximation für e^{ijx_k} in (2.59) ein, so ergibt sich:

$$\begin{aligned} \tilde{f}_j &= \sum_{k=0}^{M-1} f_k \frac{1}{w_j} \sum_{l=-K+1}^{K} \sigma_{kl} e^{2\pi ij(\mu_k + l)/\bar{N}} \\ &= \frac{1}{w_j} \sum_{k=0}^{M-1} \sum_{l=-K+1}^{K} f_k \sigma_{kl} e^{2\pi j(\mu_k + l)/\bar{N}}. \end{aligned}$$

Hier ergeben sich die ersten Unterschiede zum Fall der äquidistanten Daten bei nichtäquidistantem Ergebnis.

Auch hier will man analog zu (2.51) alle Terme mit gleichem $(\mu_k + l)$ zusammenfassen, um eine FFT-Routine anwenden zu können. Es soll also

$$u_{\mu_k+l} = \sum_{l=-K+1}^{K} f_k \sigma_{kl}$$

gelten. Hierzu führt man einen Index $m := (\mu_k + l)$ ein und definiert für $m = -\bar{N}/2 \ldots \bar{N}/2 - 1$:

$$u_m := \sum_{k=-pN/2}^{pN/2-1} \sigma_{k,m-\mu_k} \alpha_k, \tag{2.60}$$

wobei man für $m - \mu_k \notin [-K, K]$ $\sigma_{k,m-\mu_k} = 0$ setzt. In diesem Schritt kann man sehr gut die formale Ähnlichkeit mit einer Faltung erkennen.

Damit hat man dann:

$$\begin{aligned} \tilde{f}_j &= \frac{1}{w_j} \underbrace{\sum_{k=-\bar{N}/2}^{\bar{N}/2-1} u_k e^{2\pi ijk/\bar{N}}}_{:=U_j} \\ &= \frac{U_j}{w_j} \end{aligned}$$

Hierbei wird

$$U_j = \sum_{k=-\bar{N}/2}^{\bar{N}/2-1} u_k e^{2\pi ijk/\bar{N}}$$

per FFT berechnet.

Die Berechnung erfolgt also wieder in drei Schritten:

Eindimensionale Fourier-Transformation bei nichtäquidistantem Eingangsdaten:
Gegeben seien die Eingangsdaten α und Parameter w gemäß (2.47), σ gemäß 2.49, μ gemäß (2.48), $\bar{N}$ gemäß (2.44) und K gemäß (2.45). Berechne

$$\hat{f}_j = \sum_{k=0}^{M-1} \alpha_k e^{ix_k j} \qquad \text{für } j = -N/2 \ldots N/2-1 \tag{2.61}$$

durch

1. Auswertung von (2.60):
   $u := 0$
   für $k = 0 \ldots M-1$:
   für $l = -K+1 \ldots K$:
   $$u_{\mu_k+j} = u_{\mu_k+j} + \sigma_{kl} \cdot \alpha_k \tag{2.62}$$
   (Rechenaufwand: $4MK$ reelle Multiplikationen und ebensoviele Additionen.)

2. Berechnung von
   $$U_j = \sum_{k=-\bar{N}/2}^{\bar{N}/2-1} u_k \cdot e^{2\pi ikj/\bar{N}}$$
   per FFT. (Rechenaufwand $\mathcal{O}(\bar{N} \log \bar{N})$ Rechenoperationen)

3. Skalierung des Ergebnisses
   $$\tilde{f}_j = U_j / w_j \qquad \text{für } j = -N/2 \ldots N/2-1$$
   (Rechenaufwand: $2N$ reelle Multiplikationen)

Als Fehlerabschätzung ergibt sich wieder:

$$\begin{aligned}
\left|f_j - \tilde{f}_j\right| &= \left| \sum_{k=-N/2}^{N/2-1} \alpha_k \left(e^{ikx_j} - \frac{1}{w_k} \sum_{k=-K+1}^{K} \sigma_{jk} e^{2\pi ik(\mu_j+l)/\bar{N}} \right) \right| \\
&\leq \text{err} \sum_{k=-N/2}^{N/2-1} |\alpha_k|.
\end{aligned}$$

2.7.4 Zweidimensional, nichtäquidistante Daten

Der zweidimensionale Fall ergibt sich wieder als direkte Erweiterung des Eindimensionalen. Deshalb hier nur die zur Berechnung notwendigen Schritte:

Zweidimensionale Fourier-Transformation bei nichtäquidistantem Eingangsdaten:
Gegeben seien die Eingangsdaten α und Parameter w gemäß (2.47), σ und τ gemäß 2.49 und 2.56, μ und ν gemäß (2.48) und (2.55) , $\bar{N}$ gemäß (2.44) und K gemäß (2.45). Berechne

$$\hat{f}_j = \sum_{k=-N/2}^{N/2-1} \sum_{l=-N/2}^{N/2-1} \alpha_{lk} e^{ix_k j} e^{iy_l j} \qquad \text{für } j = 0 \ldots M-1 \tag{2.63}$$

durch

1. Berechne
   $u := 0$
   für $l = 0 \ldots M-1$:
   für $j = -K+1 \ldots K$:
   für $k = -K+1 \ldots K$:
   $u_{\mu_l+j,\nu_l+k} = u_{\mu_l+j,\nu_l+k} + \sigma_{lj}\tau_{lk} \cdot \alpha_k$
   (Rechenaufwand: $8K^2M$ reelle Additionen und $12K^2M$ reelle Multiplikationen)

2. Berechnung von

   $$U_{mn} = \sum_{k,l=-\bar{N}/2}^{\bar{N}/2-1} u_{kl} \cdot e^{2\pi imk/\bar{N}} e^{2\pi inl/\bar{N}}$$

   per FFT. (Rechenaufwand $\mathcal{O}((\bar{N})^2 \log(\bar{N})^2)$ Rechenoperationen)

3. Skalierung des Ergebnisses durch

   $$\tilde{f}_{jk} = \frac{U_{jk}}{w_j w_k} \qquad \text{für } j,k = -N/2 \ldots N/2-1$$

   (Rechenaufwand: $3N^2$ reelle Multiplikationen)

Die Fehlerabschätzung lautet:

$$\left|f_{jk} - \tilde{f}_{jk}\right| \leq \left(2\text{err} + \text{err}^2\right) \cdot \sum_{l=0}^{M-1} |\alpha_l|.$$

2.7.5 Symmetrische FFT

Wir haben die FFT bisher immer in der folgenden symmetrischen Formulierung verwendet:

$$f_j = \sum_{k=-N/2}^{N/2-1} \alpha_k \cdot e^{\pm 2\pi ijk/N} \tag{2.64}$$

für $j = -N/2, \ldots, N/2 - 1$.
Das Vorzeichen im Exponenten bewirkt eine Fourier-Transformation bzw. eine inverse Fourier-Transformation. In den meisten FFT-Implementierungen anzutreffen ist jedoch die folgende Notation:

$$\tilde{f}_j = \sum_{k=0}^{N-1} \tilde{\alpha}_k \cdot e^{\pm 2\pi ijk/N} \tag{2.65}$$

für $j = 0, \ldots, N-1$. Dieses beiden Formulierungen sind jedoch völlig äquivalent, da man sowohl die Eingangsdaten α_k als auch die Fouriertransformierte f_k als N-periodisch auffassen kann.
Sind α bzw. f auf einem Rechner in einem Array gespeichert, so bedeutet dies nur, daß man vor und nach Berechnung von (2.65) die erste mit der zweiten Hälfte des Arrays vertauschen muß, um (2.64) mittels einer Standard-FFT-Routine zu berechnen. Entsprechendes gilt für den zweidimensionalen Fall, hierbei sind jeweils halbe Zeilen zu vertauschen, und zwar nach dem in Tabelle 2.1 skizzierten Schema.

2.8 Betrachtung des Rechenaufwandes

In diesem Kapitel wollen wir den Rechenaufwand für die Algorithmen anhand der Anzahl Rechenoperationen (flops) ermitteln. Ein flop bezeichnet hierbei eine (reelle) Addition oder Multiplikation. Folglich benötigt eine komplexe Addition 2 und eine komplexe Multiplikation 6 flops. Eine FFT der Länge N erfordert dann bei Verwendung einer Cooley-Tukey FFT Routine mit

1	2
3	4
5	6
2	1
4	3
6	6

Tabelle 2.1: Vertauschungsschema bei zweidimensionaler symmetrischer FFT der Länge $N = 6$. Es sind jeweils die Halb-Zeilen mit gleichem Index zu vertauschen

Radix-2 Faktorisierung $5N \log_2 N$ flops und bei einer Radix-4 Faktorisierung $4.25N \log_2 N$ flops (vergleiche hierzu auch [36]).
Zur Berechnung der nichtäquidistanten Fourier–Transformation ist gemäß Abschnitt 2.7.1 eine FFT der Länge pN, eine Skalierung sowie das Resampling durchzuführen. Insgesamt ergibt sich hierfür ein Rechenaufwand von

$$\mathrm{flops}_{1D}(N, p, K) = 4.25pN \log_2(pN) + (8K + 2)N, \tag{2.66}$$

wenn man $M = N$ setzt.
Setzt man dies in Relation zu der Anzahl der Rechenoperationen für eine normale (d.h. äquidistante) FFT, so ergibt sich ein Faktor von

$$\begin{aligned} N_1(N, p, K) &= \frac{\mathrm{flops}_{1D}(N, p, K)}{4.25N \log_2 N} \\ &= \frac{4.25p \log_2(pN) + (8K + 2)}{4.25 \log_2 N} \\ &= p + \frac{4.25 \log_2 p + (8K + 2)}{4.25 \log_2 N}. \end{aligned}$$

Abbildung 2.4 zeigt einige Werte von N_1 für verschiedene K. Die linke Abbildung zeigt die Verhältnisse für einen Oversampling-Faktor für $p = 2$, die rechte für $p = 1.5$.
Bei den numerischen Experimenten in Abschnitt (2.9) ist dieser Faktor aus den gemessenen Laufzeitwerten berechnet worden. Vergleicht man die

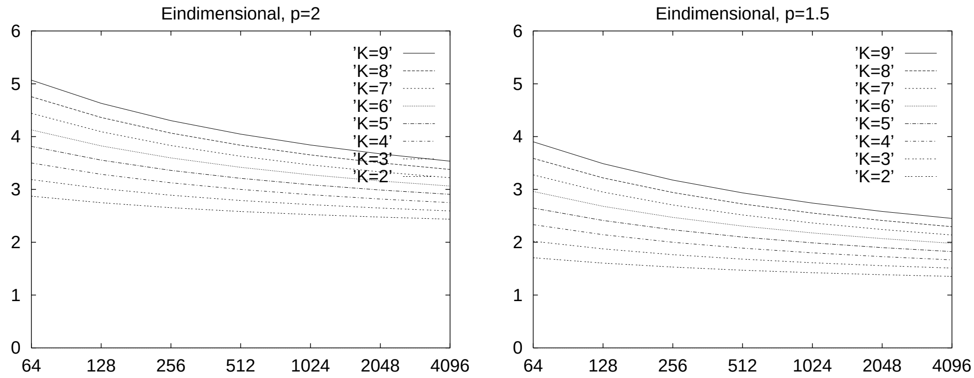


Abbildung 2.4: Faktor nichtäquidistante/äquidistante eindimensionale FFT

Einträge in der Spalte ,,FFT-Faktor" der Tabellen 2.2 und 2.3 so erkennt man eine recht gute Übereinstimmung mit der analytischen Berechnung.

Eine analoge Betrachtung kann man natürlich auch für den zweidimensionalen Fall machen: Zur Berechnung einer zweidimensionalen FFT ist eine eindimensionale FFT für alle Spalten und für alle Zeilen durchzuführen. Somit ergibt sich für eine zweidimensionale Radix-4 FFT der Länge $N \times N$ ein Aufwand von $8.5N^2 \log_2 N$ Rechenoperationen.
Zur Berechnung der zweidimensionalen nichtäquidistanten Fourier–Transformation ist gemäß Abschnitt 2.7.4 eine zweidimensionale Fourier–Transformation der Länge pN, eine Skalierung sowie das Resampling durchzuführen. Damit ergibt sich dann insgesamt ein Rechenaufwand von

$$\text{flops}_{2D}(N, p, K) = 9p^2N^2 \log_2(pN) + 20K^2N^2 \qquad (2.67)$$

Rechenoperationen, wenn man die Anzahl der nichtäquidistanten Werte mit $M = N^2$ annimmt. Dies gilt für den Fall nichtäquidistanter Ergebnisse. Bei nichtäquidistanten Daten fallen etwas weniger Rechenoperationen an, statt $20K^2N$ erscheint in (2.67) $16K^2 + 8K$.
Setzt man dies wieder in Relation zu der Anzahl der Rechenoperationen für eine normale (d.h. äquidistante) zweidimensionale FFT, so ergibt sich ein Faktor von

$$N_2(N, p, K) \quad = \quad \frac{8.5p^2N^2 \log_2(pN) + (20K^2 + 3)N^2}{8.5N^2 \log_2 N}$$

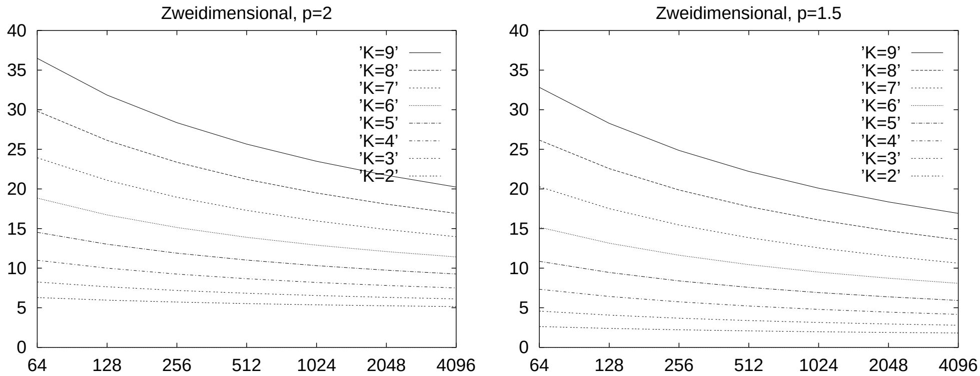


Abbildung 2.5: Faktor nichtäquidistante/äquidistante zweidimensionale FFT

$$= \quad p^2 + \frac{8.5p^2 \log_2 p + (20K^2 + 3)}{8.5 \log_2 N}.$$

Abbildung 2.5 zeigt wieder einige Werte von N_2 für verschiedene K und $p = 2$ bzw. 1.5.
Die vergleichbaren Werte der numerischen Untersuchungen finden sich in den Tabellen 2.4 und 2.5.

2.9 Numerische Experimente

Für die numerischen Untersuchungen wurden die Algorithmen in C implementiert und auf einer 300MHz Sun UltraSPARC-II getestet. Die Compilierung erfolgte mit dem Sun C-Compiler und der Option „-fast", d.h. voller Optimierung. Für die Untersuchung der Genauigkeit wurden die beiden folgenden Fehler-Normen verwendet:

$$E_\infty = \frac{\max_j |\bar{y}_j - y_j|}{\max_j |\bar{y}_j|}$$

und

$$E_2 = \frac{\sum_j |\bar{y}_j - y_j|^2}{\sum_j |\bar{y}_j|^2}.$$

Hierbei bezeichnet y die mit unserem Algorithmus berechnete Fouriertransformierte, $\bar{y}$ die Werte aus direkter Berechnung. Es ist zu beachten, daß auch $\bar{y}$ mit einem Maschinenfehler behaftet ist. Somit sind die Angaben im Bereich der Maschinengenauigkeit eher als Richtwerte denn als echte Meßwerte zu verstehen; zur Beurteilung der Genauigkeit des Algorithmus reicht dies aber völlig aus.
Als nichtäquidistante Stützstellen wurden gleichverteilte Zufallswerte aus dem Intervall $[-\pi, \pi]$ bzw. im zweidimensionalen aus $[-\pi, \pi] \times [-\pi, \pi]$ angenommen. Die Eingangsdaten wurden mit gleichverteilten Zufallswerten aus dem komplexen Einheitsquadrat gefüllt:

$$0 \leq \mathrm{Re}(\alpha_j) \leq 1, \qquad 0 \leq \mathrm{Im}(\alpha_j) \leq 1.$$

Die (Pseudo-)Zufallszahlen wurden mit der entsprechenden IMSL-Routine erzeugt. Diese Parameter ähneln denen in dem Übersichtsartikel [38] von Ware, so daß man die Ergebnisse mit den dort ermittelten vergleichen kann. Der einzige Unterschied liegt darin, daß Ware die komplexen Eingangsdaten alle mit Norm eins erzeugt. Dies dient der Konsistenz mit den von ihm zusätzlich betrachteten Spezialfällen glatter Funktionen; auf die hier auftretenden Approximations-Fehler hat der Unterschied keine wesentlichen Auswirkungen.
Zur Berechnung der FFT wurden die Routinen der IMSL-Bibliothek verwendet. Diese haben den Vorteil, daß sie sowohl für einfache als auch für doppelte Genauigkeit zur Verfügung stehen und somit in beiden Fällen als Referenz dienen können.
Tabelle 2.6 zeigt die Genauigkeit in Abhängigkeit von der Interpolationslänge K. Es wurde jeweils eine Fourier-Transformation der Länge $N = M = 1024$ berechnet. Die oberen Hälfte der Tabellen zeigt jeweils die Ergebnisse bei nichtäquidistantem Ergebnis, die untere Hälfte bei nichtäquidistanten Daten. Man kann sehr gut erkennen, daß die Genauigkeit mit zunehmenden K exponentiell wächst, eine Erhöhung von K um eins verbessert die Genauigkeit um zwei Dezimalstellen. Im folgenden werden wir für einfache Genauigkeit $K = 4$ und für doppelte Genauigkeit $K = 7$ wählen, jeweils bei einem Oversampling von $p = 1.5$. Damit erhalten wir Fehlerordnungen, die denen in [38] entsprechen, wo die Grenze für einfache Genauigkeit bei $E_\infty < 10^{-5}$ und für doppelte Genauigkeit bei $E_\infty < 10^{-10}$ gezogen wird.
Der rechte Teil der Tabelle zeigt die Laufzeit des Algorithmus an. ,,Eval" bezeichnet die Zeit zur Berechnung der jeweiligen schnellen nichtäquidistan-

ten Fourier-Transformation, ,,Faktor” gibt an, wieviel mal schneller dies im Vergleich zur direkten Berechnung ist. Man erkennt deutlich, daß diese Beschleunigung sich in einer Größenordnung bewegt, die für den Aufwand zur Herleitung des Algorithmus entschädigt.
Zu Vergleichszwecken ist in der Tabelle auch die Laufzeit einer FFT-Routine derselben Länge angegeben, wie man sie bei äquidistanter Daten verwenden könnte. ,,FFT-Faktor” ist der entsprechend der Quotient aus der Laufzeit der schnellen nichtäquidistanten Fourier-Transformation und dieser FFT-Laufzeit. Diese Größe wird üblicherweise zur Bewertung der Effizient eines nichtäquidistanten Fourier-Transformations-Algorithmus herangezogen. Je kleiner sie ist, desto effizienter ist der Algorithmus. Sie entspricht den in Abschnitt 2.8 ermittelten Faktoren N_1 bzw. im zweidimensionalen N_2.

Tabellen 2.2 und 2.3 zeigen dann das Verhalten bei Fourier-Transformationen unterschiedlicher Länge, einmal bei einfacher Genauigkeit und einmal bei doppelter Genauigkeit. Man sieht, daß sowohl der Approximations-Fehler, als auch der FFT-Faktor nur leicht von der Länge der Fourier-Transformation abhängt und im wesentlichen konstant bleiben. Der genaue Wert des Faktors wird durch eine Reihe von Maschinen-Parametern, insbesondere der genauen Speicherarchitektur des jeweiligen Rechners bestimmt. Weiterhin muß man berücksichtigen, daß für die FFT-Routinen eine kommerzielle Bibliothek mit entsprechend optimierter Implementierung zum Einsatz kam.
Der Algorithmus ist bei nichtäquidistanten Daten etwas langsamer als bei nichtäquidistanten Frequenzen, obwohl die Anzahl an Rechenoperationen praktisch gleich ist. Dies liegt daran, daß der Interpolationsschritt in (2.53) jeweils auf aufeinanderfolgende Speicherstellen zugreifen kann, was im Falle nichtäquidistanter Eingangsdaten in (2.62) nicht möglich ist.

Tabellen 2.4 und 2.5 wiederholen diese Untersuchungen für den zweidimensionalen Fall. Hierbei wurde eine Fourier-Transformation der Länge $N \times N$ bei $M = N^2$ nichtäquidistanten Stützstellen berechnet.
Hier wird der Geschwindigkeitsvorteil besonders drastisch deutlich: eine schnelle nichtäquidistante zweidimensionale Fourier-Transformation der Länge 1024 benötigt weniger als 40 Sekunden (bei doppelter Genauigkeit). Direkte Berechnung hingegen erfordert rund vier Wochen.
In den Beispielen ist immer $M = N$ bzw. im zweidimensionalen $M = N^2$ gewählt worden, N als Zweier-Potenz. Beides ist aber keinesfalls zwingend. Die Anzahl der nichtäquidistanten Punkte M kann beliebig variiert werden,

NE	N	K	p	E_∞	E_2	**Eval msecs**	**Faktor**	FFT msecs	FFT-Faktor
ner	128	4	1.5	6.81e-07	1.17e-06	0.113	64	0.050	2.2
ner	256	4	1.5	2.43e-06	1.73e-06	0.210	124	0.071	3.0
ner	512	4	1.5	1.34e-06	1.33e-06	0.425	251	0.129	3.3
ner	1024	4	1.5	6.02e-07	1.03e-06	0.838	504	0.272	3.1
ner	2048	4	1.5	5.15e-07	8.73e-07	1.693	956	0.561	3.0
ner	4096	4	1.5	3.49e-07	8.12e-07	3.798	1822	1.278	3.0
ned	128	4	1.5	4.35e-07	8.04e-07	0.127	54	0.050	2.5
ned	256	4	1.5	4.20e-07	7.09e-07	0.241	108	0.071	3.4
ned	512	4	1.5	2.81e-07	6.31e-07	0.480	225	0.129	3.7
ned	1024	4	1.5	1.92e-07	6.55e-07	0.953	440	0.273	3.5
ned	2048	4	1.5	1.25e-07	6.61e-07	2.000	837	0.601	3.3
ned	4096	4	1.5	4.45e-07	7.64e-07	4.222	1628	1.228	3.4

Tabelle 2.2: **Eindimensionale nichtäquidistante Fouriertransformation bei einfacher Genauigkeit**

die Länge der FFT bleibt in diesem Falle gleich, die für den Resampling-Schritt benötigten Rechenoperationen sind proportional zu M.

Für beliebiges N kann man eine geeignete FFT-Länge $\bar{N}$ wählen. Dadurch ändert sich der Oversampling-Faktor $p = \bar{N}/N$, der die Genauigkeit des Algorithmus beeinflußt. Tabelle 2.6 zeigt die Genauigkeit für $p = 1.5$ und $p = 2$. In der Praxis sollte man $\bar{N}$ als FFT-freundliche (d.h. bei den meisten FFT-Algorithmen ein Produkt kleiner Primzahlen, z.B. 2^k oder $3 \cdot 2^k$) Zahl so wählen, daß in etwa

$$1.5 \leq p = \frac{\bar{N}}{N} \leq 2$$

gilt. Bei kleineren Werten von p nimmt die Genauigkeit weiter ab, größere Werte von p verlangsamen den Algorithmus unnötig.

NE	N	K	p	E_∞	E_2	**Eval msecs**	**Faktor**	FFT msecs	FFT-Faktor
ner	128	7	1.5	4.50e-11	3.20e-11	0.152	47	0.051	3.0
ner	256	7	1.5	2.46e-11	2.36e-11	0.297	103	0.075	3.9
ner	512	7	1.5	7.78e-12	1.43e-11	0.564	211	0.161	3.5
ner	1024	7	1.5	3.37e-12	1.34e-11	1.133	400	0.351	3.2
ner	2048	7	1.5	4.33e-12	1.17e-11	2.348	783	0.698	3.4
ner	4096	7	1.5	3.11e-12	1.28e-11	5.036	1445	1.402	3.6
ned	128	7	1.5	6.96e-12	1.15e-11	0.192	37	0.050	3.8
ned	256	7	1.5	6.14e-12	1.13e-11	0.383	79	0.075	5.1
ned	512	7	1.5	3.85e-12	8.97e-12	0.746	159	0.161	4.6
ned	1024	7	1.5	5.52e-12	1.05e-11	1.494	301	0.349	4.3
ned	2048	7	1.5	4.13e-12	1.03e-11	3.098	600	0.693	4.5
ned	4096	7	1.5	8.34e-12	1.27e-11	6.493	1121	1.404	4.6

Tabelle 2.3: **Eindimensionale nichtäquidistante Fourier–Transformation bei doppelter Genauigkeit**

2.10 Vergleich mit anderen Algorithmen

Eine Übersicht über schnelle nichtäquidistante Fourier–Transformationen findet sich in [38].
Dort werden unter anderem die folgenden Algorithmen zur schnellen nichtäquidistanten Fourier–Transformation vorgestellt:

1. Die nichtäquidistante Fourier–Transformation von Dutt und Rokhlin [8]. Dieser Algorithmus entspricht dem hier vorgestellten, wen man als Gewichtsfunktion die Gauß-Glockenkurve

   $$\phi(x) = e^{-bx^2}$$

   wählt. Diese Gewichtsfunktion hat keinen kompakten Träger. Daraus entsteht ein zusätzlicher Fehler, der sich aus den Anteilen von ϕ außerhalb des Intervalls $[-\alpha\pi, \alpha\pi]$ der Fourier-Reihenentwicklung ergibt und somit ungefähr proportional zum Funktionswert $\phi(\alpha\pi)$ ist. Sinnvollerweise wählt man den Parameter b daher so, daß dieser Fehler von der

NE	N	K	p	E_∞	E_2	**Eval secs**	**Faktor**	FFT secs	FFT-Faktor
ner	64	4	1.5	3.45e-07	9.91e-07	0.029	1161	0.01	4.1
ner	128	4	1.5	3.29e-07	7.86e-07	0.125	4311	0.03	4.6
ner	256	4	1.5	5.74e-07	3.17e-06	0.567	14751	0.11	5.1
ner	512	4	1.5	5.05e-06	8.91e-06	2.947	50361	0.51	5.4
ner	1024	4	1.5	1.69e-07	4.04e-06	13.663	173800	2.51	5.5
ned	64	4	1.5	1.54e-07	1.36e-06	0.034	991	0.01	4.9
ned	128	4	1.5	1.83e-07	2.14e-06	0.141	3822	0.03	5.4
ned	256	4	1.5	1.88e-07	3.85e-06	0.643	13008	0.11	5.7
ned	512	4	1.5	2.07e-07	7.45e-06	3.157	47011	0.51	6.2
ned	1024	4	1.5	2.43e-07	8.21e-06	14.806	160383	2.51	5.9

Tabelle 2.4: **Zweidimensionale nichtäquidistante Fouriertransformation bei einfacher Genauigkeit**

selben Größenordnung ist wie der Fehler, der sich aus dem Abschneiden der Fourierreihen-Approximation ergibt.

Dutt und Rokhlin berücksichtigen bei Ihrer Herleitung nicht den von uns in Abschnitt 2.2.4 beschriebenen Aliasing-Effekt, und gehen somit von $\alpha = 1$ aus. Allerdings ermitteln sie den Parameter b experimentell und geben bei einem Oversampling von $p = 2$ eine Wahl von $b = 0.6$ für einfache und $b = 1.56$ für doppelte Genauigkeit an. Nach ihrer eigenen Herleitung ist dies eine schlechte Wahl, da an den Grenzen des Intervalls die Funktion ϕ mit $\exp(-0.6\pi^2) \approx 0.003$ bzw. $\exp(-1.56\pi^2) \approx 0.2 \cdot 10^{-6}$ noch einige Zehnerpotenzen von der jeweiligen Maschinengenauigkeit von 10^{-7} bzw. 10^{-16} entfernt ist. Diese Werte gehen auch in die Fehlerabschätzung ein und liefern somit viel zu pessimistische Ergebnisse.

Berücksichtigt man aber die Erkenntnisse aus Abschnitt 2.2.4, so kann man für $p = 2$ einen Wert von $\alpha = 1.5$ zulassen. Die entsprechenden Werte für ϕ an den Rändern des des Intervalls $[-\alpha\pi, \alpha\pi]$ sind dann $\exp(-0.6(\alpha\pi)^2) \approx 1.6 \cdot 10^{-6}$ bzw. $\exp(-1.56(\alpha\pi)^2) \approx 9 \cdot 10^{-16}$ was erheblich besser zu einer Approximation paßt, die etwas unterhalb

NE	N	K	p	E_∞	E_2	**Eval** **sec**	**Faktor**	FFT sec	FFT-Faktor
ner	64	7	1.5	5.98e-12	1.54e-11	0.066	577	0.01	9.4
ner	128	7	1.5	4.52e-12	1.69e-11	0.293	2230	0.03	9.5
ner	256	7	1.5	1.47e-11	2.35e-11	1.418	7068	0.14	10.4
ner	512	7	1.5	7.92e-11	3.12e-11	7.907	22262	0.76	10.4
ner	1024	7	1.5	4.50e-12	7.49e-12	34.332	84082	3.09	11.2
ned	64	7	1.5	3.07e-12	1.43e-11	0.077	586	0.01	9.6
ned	128	7	1.5	1.46e-12	1.40e-11	0.324	1960	0.03	10.5
ned	256	7	1.5	9.62e-13	1.40e-11	1.533	6517	0.14	11.1
ned	512	7	1.5	6.85e-13	1.39e-11	8.512	20977	0.76	11.2
ned	1024	7	1.5	2.94e-12	1.45e-11	36.539	77497	3.09	11.9

Tabelle 2.5: **Zweidimensionale nichtäquidistante Fouriertransformation bei doppelter Genauigkeit**

der Maschinengenauigkeit liegen soll. Die hier vorgestellte Theorie liefert also den fehlenden Teil zur Beschreibung der Auswirkung des Parameters b. Das Fehlen einer solchen Beschreibung war auch in dem Übersichtsartikel von Ware in [38] angemerkt worden.

Um dieselbe Genauigkeit wie der hier vorgestellte Algorithmus zu erreichen, benötigt die von Dutt und Rokhlin getroffene Wahl, für ϕ eine Gauß–Glockenkurve zu verwenden, eine Interpolations-Länge von $K = 6$ bei einfacher und $K = 12$ bei doppelter Genauigkeit und einem Oversampling-Faktor von $p = 2$ — also jeweils die doppelte Länge des hier vorgestellten Verfahrens.

e^{-x^2} als Gewichtsfunktion zu verwenden ist eigentlich sehr naheliegend, wenn man eine im Orts- und Frequenzraum konzentrierte Funktion sucht. Daher erstaunt es ein wenig, daß dies hier nicht die beste Wahl ist. Allerdings sind unsere Anforderungen an das Verhalten der Funktion im Orts- und im Frequenzraum nicht vollständig symmetrisch, so daß die Kaiser-Bessel Funktion bessere Ergebnisse erzielt. Sie hat außerdem den Vorteil, daß sich das numerische Verhalten ihrer Fou-

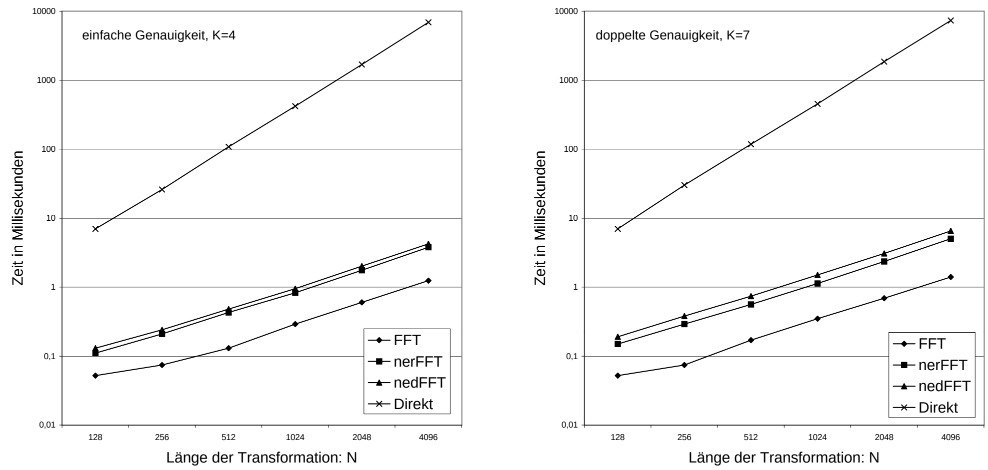


Abbildung 2.6: Eindimensionale Fourier–Transformation bei einfacher und doppelter Genauigkeit

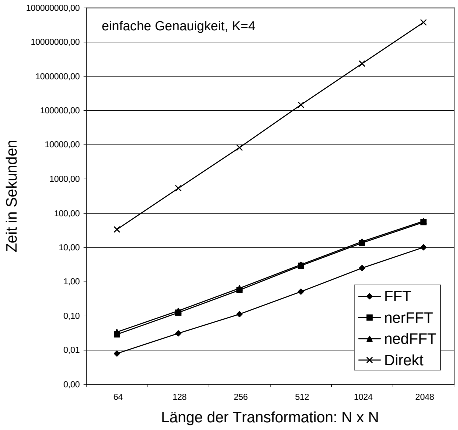
riertransformierte $\hat{\phi}_{KB}$ im Intervall $[-K, K]$ stark von dem außerhalb dieses Intervalls unterscheidet und somit genau an eine Interpolation der Länge $2K$ angepaßt ist.

2. Die Methode von Beylkin verwendet B-Splines $\beta^{(m)}$ der Ordnung m als Approximations-Koeffizienten. Diese haben einen kompakten Träger der Breite m, so daß $m = K/2$ der Interpolations-Länge entspricht. Als Gewichtsfunktion tritt $\sqrt{a^{(m)}(\xi)}$ mit

$$a^{(m)}(\xi) = \sum_{l} \left| \hat{\beta}^{(m)}(\xi + l) \right|^2$$

auf.

Diese Wahl von Koeffizienten und Gewichtsfunktion erfordert bei einem Oversampling-Faktor von $p = 2$ eine Interpolations-Länge $K = 8$ bei einfacher und $K = 13$ bei doppelter Genauigkeit, verglichen mit $K = 3$ und $K = 6$ bei unserem Algorithmus.



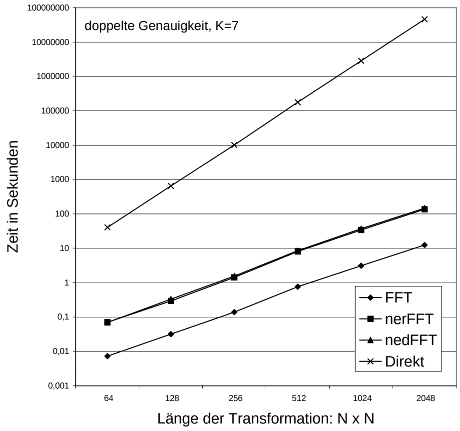


Abbildung 2.7: Zweidimensionale Fourier–Transformation bei einfacher und doppelter Genauigkeit

Für die Beylkin-Methode existiert eine kommerzielle Implementierung (`www.fmah.com`), die den ein- und zweidimensionalen Fall abdeckt.

3. Die Gridding-Methode von O'Sullivan. O'Sullivan verwendet als Gewichtsfunktion die Kaiser-Bessel Funktion, die sich auch in unseren Betrachtungen als beste Wahl herausgestellt hat. Allerdings verzichtet er auf weitergehende mathematische Betrachtungen und eine detaillierte Fehleranalyse, da sein Ziel nicht die nichtäquidistante Fourier-Transformation sondern die Inversion der Radon-Transformation in der Computer-Tomographie ist.

4. Lokale Interpolation. Zusätzlich zu den hier detailliert vorgestellten Verfahren existieren zahlreiche Methoden die auf einer Interpolation der Funktion
$$\hat{f}(x) = \sum_{k=-N/2}^{N/2-1} f_k e^{\pm 2\pi i k x/N}$$
in einer Umgebung der per FFT ermittelbaren äquidistanten Funktionswerte bei $\bar{x} = 2\pi j/pN$ aufbauen, um so die Funktionswerte der in

der Nähe liegenden Punkte $x = \bar{x} + s$ mit kleinem s zu erhalten. Für eine ausführlichere Betrachtung dieser Verfahren sei wieder auf [38] verwiesen.

Das einfachste Beispiel für eine Interpolations-Methode ist die kubische Spline-Interpolation, bei der ein interpolierender Spline durch die äquidistanten Gitterpunkte gelegt wird. Dieses Verfahren wird in zahlreichen Anwendungen verwendet, ist aber den speziell auf die nichtäquidistante Fourier–Transformation zugeschnittenen Verfahren deutlich unterlegen.

Die verschiedenen Interpolationsverfahren funktionieren immer dann besonders gut wenn die nichtäquidistanten Stützstellen in unmittelbarer Nähe der äquidistanten Punkte liegen oder wenn $f(x)$ eine glatte Funktion darstellt. Dann erreichen sie, zumindest im eindimensionalen Fall, annähernd das Laufzeitverhalten des hier vorgestellten Algorithmus. Allerdings muß man dann spezielle Annahmen über die Stützstellen oder den Eingangsvektor machen. Im mehrdimensionalen Fall spielt zudem die durch K parametrisierte Länge der Interpolation eine stärkere Rolle als im eindimensionalen Fall, was einen Vorteil für die hier vorgestellte Methode darstellt.

5. Eine bemerkenswerte Variante der Interpolationsverfahren stellt das zweite von Dutt und Rokhlin entwickelte Verfahren [9] zur schnellen Fourier–Transformation bei nichtäquidistanten Daten dar. Hier wird eine explizite Resampling-Formel für den eindimensionalen Fall hergeleitet, die es ermöglicht, beliebig zwischen äquidistanten und nichtäquidistanten Daten zu konvertieren. Direkte Auswertung dieser Resampling-Formel erfordert $\mathcal{O}(N^2)$ Rechenoperationen, allerdings kann man mittels einer modifizierten schnellen Multipolentwicklung (FMM, fast multipole method [5]) die Formel schnell auswerten und somit zu einem Algorithmus gelangen, der von der Effizienz her nicht ganz an die konkurrierenden Verfahren heranreicht, aber dafür einige andere Vorzüge hat:

   So kann man auf diese Weise ohne zusätzlichen Aufwand das inverse

Problem lösen, bei dem für

$$\hat{g}_k = \sum_{j=0}^{M-1} g_j e^{ikx_j} \qquad \text{für } k = -N/2 \dots N/2 - 1$$

zu gegebenem $\hat{g}$ die Eingangsdaten g gesucht sind. Anders als im äquidistanten Fall ist die diskrete nichtäquidistante Fourier–Transformation nämlich nicht mehr zwangsläufig unitär, und die Kondition kann bei ungünstiger Verteilung der Stützstellen unerfreulich schlecht werden.

Leider läßt sich diese Methode nicht ohne weiteres auf den mehrdimensionalen Fall übertragen. Zum einen fällt es hier erheblich schwerer eine direkte Formel für das Resampling anzugeben, zum anderen nimmt die Komplexität der Multipolentwicklung weiter zu, so daß es selbst bei bekannter Resampling-Formel nicht leicht fallen dürfte, einen effizienten Algorithmus zu erhalten.

2.11 Mögliche Erweiterungen

Bisher wurde nur der Fall der diskreten Fourier–Transformation bei komplexwertigen Daten betrachtet. Das Verfahren läßt sich aber ohne weiteres auch auf reelle Daten und auf Sinus- bzw. Cosinus-Transformationen anwenden. Bei Verwendung entsprechend angepaßten Routinen ergibt sich dann in etwa eine Halbierung der Laufzeit. Im nächsten Kapitel über die schnelle Radon-Inversion wird eine solche Fourier–Transformation reellwertiger Meßwerte auftreten. Die hierfür notwendigen Änderungen stellen in erster Linie ein Problem der Implementierung dar, für das grundsätzliche mathematische Prinzip der Approximation von Fourier-Moden ergeben sich keine Änderungen. Auch die Erweiterung auf mehr als zwei Dimensionen ergibt sich unmittelbar.

NE	N	K	p	E_∞	E_2	**Eval msec**	**Faktor**	FFT msec	FFT-Faktor
ner	1024	2	2.0	2.86e-04	4.45e-04	1.035	435.6	0.3	3.1
ner	1024	3	2.0	2.39e-06	4.89e-06	1.112	410.9	0.3	3.3
ner	1024	4	2.0	2.54e-08	5.36e-08	1.204	370.4	0.3	3.6
ner	1024	5	2.0	2.07e-10	5.46e-10	1.228	377.9	0.3	3.7
ner	1024	6	2.0	4.99e-12	7.00e-12	1.367	335.7	0.3	4.0
ner	1024	7	2.0	6.42e-14	1.02e-13	1.416	321.4	0.3	4.0
ner	1024	2	1.5	5.54e-04	1.37e-03	0.828	546.9	0.3	2.4
ner	1024	3	1.5	3.27e-05	4.70e-05	0.902	486.8	0.3	2.6
ner	1024	4	1.5	6.01e-07	1.00e-06	0.982	462.4	0.3	2.8
ner	1024	5	1.5	1.04e-08	2.44e-08	0.993	457.0	0.3	2.8
ner	1024	6	1.5	3.29e-10	5.40e-10	1.112	403.7	0.3	3.2
ner	1024	7	1.5	3.37e-12	1.34e-11	1.136	411.8	0.3	3.4
ned	1024	2	2.0	6.32e-05	3.16e-04	1.138	1728.1	0.3	3.4
ned	1024	3	2.0	5.80e-07	3.58e-06	1.250	1591.7	0.3	3.6
ned	1024	4	2.0	7.71e-09	4.17e-08	1.370	1437.7	0.3	4.1
ned	1024	5	2.0	1.09e-10	4.85e-10	1.487	1329.1	0.3	4.4
ned	1024	6	2.0	1.77e-12	5.31e-12	1.600	1230.4	0.3	4.7
ned	1024	7	2.0	1.94e-14	7.06e-14	1.714	1146.7	0.3	5.1
ned	1024	2	1.5	2.13e-04	1.07e-03	0.929	2132.8	0.3	2.6
ned	1024	3	1.5	4.85e-06	2.31e-05	1.041	1903.2	0.3	2.9
ned	1024	4	1.5	1.84e-07	5.77e-07	1.155	1717.2	0.3	3.2
ned	1024	5	1.5	4.78e-09	1.47e-08	1.267	1681.0	0.3	3.4
ned	1024	6	1.5	1.19e-10	3.67e-10	1.388	1436.7	0.3	3.9
ned	1024	7	1.5	5.52e-12	1.05e-11	1.496	1313.9	0.3	4.1

Tabelle 2.6: **Genauigkeit und Geschwindigkeit der eindimensionalen nichtäquidistanten Fouriertransformation**

Kapitel 3

Schnelle Fourier-Rekonstruktion

3.1 Einleitung, Problemstellung

In diesem Kapitel wollen wir die in Kapitel 2 entwickelten Verfahren anwenden, um zu einem schnellen Algorithmus zur Invertierung der Radon-Transformation zu gelangen.
Das Interesse an einer effizienten numerischen Invertierung rührt daher, daß mit der Radon-Transformation der Meßvorgang in der Computer-Tomographie (CT) modelliert werden kann.
Die Rekonstruktion eines Bildes aus den Meßdaten entspricht dann einer Invertierung der Radon-Transformation.
Wir werden in diesem Kapitel nur diejenigen Aspekte der Computer-Tomographie erwähnen, die zur Herleitung und Bewertung des Algorithmus unbedingt notwendig sind. Für eine umfassende mathematische Darstellung sei auf [21] verwiesen.

3.2 Physikalischer Hintergrund

Dieser Abschnitt gibt eine einfache, für unsere Zwecke aber vollständig ausreichende Beschreibung der physikalischen Hintergründe der Computer-Tomographie bzw. des von uns hierfür verwendeten mathematischen Modells. Abbildung 3.1 zeigt das grundlegende Prinzip. Von einer Röntgenquelle

(Source) wird ein Röntgenstrahl ausgesendet, der das zu messende Objekt durchquert und dabei teilweise absorbiert wird. Auf der anderen Seite wird die Intensität des austretenden Röntgenstrahls mit einem Detektor gemessen.
Da Röntgenstrahlen sehr energiereich sind, durchqueren sie den Körper auf einer geraden Bahn, Streueffekte können weitgehend vernachlässigt werden.
Die Intensität am Detektor ergibt sich dann aus

$$I = I_0 \exp\left\{- \int_L f(x)\, dx\right\},$$

wobei I_0 die Intensität an der Quelle, L die (gerade) Bahn des Röntgenstrahls und $f(x)$ den Absorptionskoeffizienten des betreffenden Gewebes an der Stelle x bezeichnet. Wir skalieren das Problem so, daß $f(x) = 0$ für $|x| \geq 1$ gilt.
Die Abbildung, die einer Funktion im $\mathbb{R}^2$ die Menge ihrer Linienintegrale zuordnet, ist die Radon-Transformation:

$$\mathcal{R}f(s,\varphi) = \int_{x\cdot\theta=s} f(x)\, dx \qquad \text{mit } \theta = \begin{pmatrix} \cos\varphi \\ sin\varphi \end{pmatrix} \tag{3.1}$$

Der Funktionswert $\mathcal{R}f(s,\varphi)$ beschreibt den Meßwert eines Röntgenstrahls, der das Objekt im Abstand s vom Ursprung und Winkel φ zur x-Achse durchquert hat.
Die Rekonstruktion eines Bildes entspricht mathematisch also einer Inversion der Radon-Transformation. Die Standard-Vorgehensweise zur numerischen Invertierung der Radon-Transformation ist die gefilterte Rückprojektion (FBP, filterted backprojection). Diese liefert eine hervorragende Rekonstruktionsqualität, allerdings auf Kosten eines Rechenaufwandes, der bei $\mathcal{O}(N^3)$ liegt.
Abbildung 3.6 zeigt den Datensatz einer mit einem Computertomographen durchgeführten Messung, d.h. eine Radon-Transformierte, und das daraus rekonstruierte Bild. Jede Zeile der Radon-Transformierten, in der Medizin als Sinogramm bezeichnet, entspricht den Messungen bezüglich eines Winkels. Die oberste Zeile entspricht den genau von oben einfallenden Strahlen, während die mittlere Zeile Messungen mit waagerecht von der Seite kommenden Strahlen entspricht.

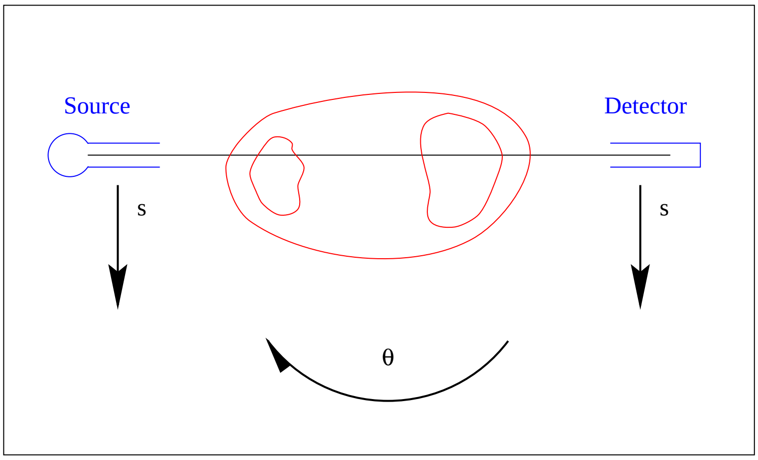


Abbildung 3.1: Schematische Darstellung der Computer-Tomographie

3.3 Fourier-Rekonstruktion

In diesem Text wird ein Rekonstruktionsansatz untersucht, der auf dem Fourier-Slice-Theorem beruht. Dieses stellt einen Zusammenhang zwischen der Radon- und der Fourier-Transformation her. Für $f \in \mathcal{S}$ gilt

$$\hat{f}(s,\varphi) = (2\pi)^{-1/2}\widehat{\mathcal{R}f}(s,\varphi). \tag{3.2}$$

Hierbei ist auf der linken Seite die zweidimensionale Fouriertransformation, auf der rechten Seite hingegen die eindimensionale Fouriertransformation nach dem ersten Parameter gemeint. Für einen Beweis des Satzes siehe z.B. [21].

(3.2) wird oft auch als Projektionssatz bezeichnet: Die Fourier-Transformierte der Messungen bezüglich eines Winkels liefert einen eindimensionalen Schnitt durch die Fourier-Transformierte von f.

Dies liefert unmittelbar einen Ansatz zur Invertierung der Radon-Transformation: man nimmt die Messungen bezüglich eines Winkels und berechnet hierfür die eindimensionale Fouriertransformation. Nachdem man dies für alle Winkel φ gemacht hat, liefert eine zweidimensionale inverse Fouriertransformation die ursprüngliche Funktion f.

In der Praxis wird $\mathcal{R}f(s,\varphi)$ natürlich nur an endlich vielen Punkten (s_j,φ_k) gemessen. Die Verteilung dieser Meßwerte bezeichnet man auch als Meßgeometrie.
Wir nehmen zur Herleitung unseres Algorithmus die folgende naheliegende Verteilung an. Es seien die Meßwerte

$$g_{jk} = \mathcal{R}f(j\pi/p\,,\,k/q) \qquad \text{für } j = 0,\dots,p-1 \text{ und } k = -q,\dots,q \tag{3.3}$$

gegeben, d.h. wir betrachten Messungen aus p Richtungen und pro Richtung $2q+1$ Messungen. Dies bezeichnet man auch als Parallel-Geometrie, eine Anordnung die auch in den ersten kommerziell gebauten CT-Scannern zur Anwendung kam.
Moderne Scanner verwenden hingegen die sogenannte Fan-Beam Geometrie. Hierbei sendet die Quelle einen fächerförmigen Strahl (fan) aus, der auf der anderen Seite von mehreren nebeneinander liegenden Detektoren gleichzeitig gemessen wird. Auf diese Weise kann der Meßvorgang beschleunigt werden, was die Strahlenbelastung für den Patienten verringert. Die mathematischen Unterschiede zur Parallelgeometrie sind relativ gering, wir werden sie kurz in Abschnitt 3.6 behandeln.
Die Probleme bei auf (3.2) basierenden Rekonstruktionsmethoden entstehen nun dadurch, daß man die Fourier-Transformationen per FFT berechnen will, um so zu einem effizienten Algorithmus zu gelangen. Führt man die eindimensionalen Fourier-Transformationen auf der rechten Seite von (3.2) durch, so erhält man die Daten $\hat{f}(s,\varphi)$ auf einem Polargitter. Um die zweidimensionale inverse FFT durchführen zu können, benötigt man die Daten jedoch auf einem cartesischen Gitter. Abbildung 3.2 verdeutlicht diese Situation.
Dies scheint kein größeres Problem zu sein. Durch einfache Interpolation kann man natürlich die Daten auf dem kartesischen Gitter aus denen auf dem Polargitter gewinnen. Leider führt dies nicht zum Erfolg. Interpolationen im Fourier-Raum erzeugen im rekonstruierten Bild deutlich sichtbare Artefakte, Artefakte, die sich auch durch eine Erhöhung der Interpolations-Ordnung nicht beseitigen lassen.
Besonders bei Anwendungen im medizinischen Bereich sind diese Artefakte nicht tolerabel. Man muß also nach Methoden suchen, die Rekonstruktionen hoher Qualität erzeugen, und dabei trotzdem effizienter als die gefilterte Rückprojektion mit ihren $\mathcal{O}(N^3)$ Rechenoperationen sind.

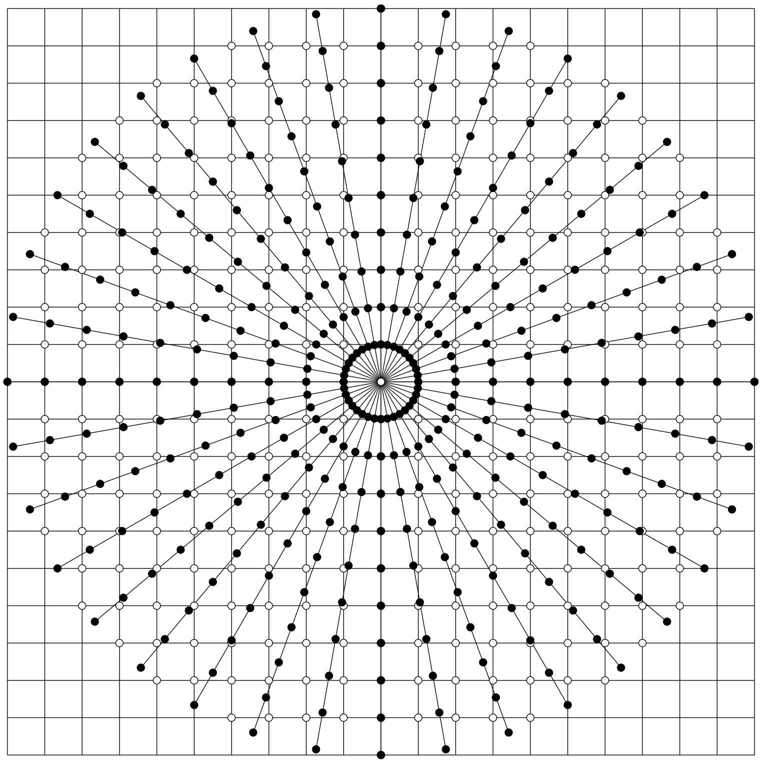


Abbildung 3.2: Polargitter und cartesisches Gitter

Im Laufe der Jahre sind zahlreiche Algorithmen vorgestellt worden, die sich mit diesem „Interpolationsproblem" bei der Radon-Inversion mit Fourier-Methoden beschäftigen [19, 24, 18, 33]. Besonders hervorzuheben ist hierbei wiederum die von O'Sullivan in [24] entwickelte Methode.

Diese beruht darauf, das Polargitter in Abbildung 3.2 als Stützstellengitter einer nichtäquidistanten Fouriertransformation aufzufassen.

Diese kann mit den zweidimensionalen Algorithmen aus dem vorherigen Kapitel effizient berechnet werden.

Unabhängig von O'Sullivan entwickelte Kaveh ein ähnliches Verfahren: die „unified fourier reconstruction (UFR)". Hierbei [17, 18] wird als Gewichtsfunktionen eine Gauß–Glockenkurve e^{-bx^2} verwendet. Übertragen auf den Fall der nichtäquidistanten Fouriertransformation entspricht die Methode von Kaveh daher der Methode von Dutt und Rokhlin. Eine ausführliche Betrachtung findet sich in [28], dort wird auch die Rekonstruktionsqualität des Verfahrens untersucht.

Die Gridding Methode liefert einen schnellen Algorithmus zur tomographischen Rekonstruktion in $\mathcal{O}(N^2 \log(N))$ Rechenoperationen.
Unsere Vorgehensweise ähnelt dem Gridding-Algorithmus. Allerdings nutzt sie zusätzlich die Erkenntnisse aus [19] aus, wo gezeigt wird, daß die Probleme bei der Interpolation im Fourier-Raum primär durch Interpolation in radialer Richtung entstehen, während die Interpolation in Winkelrichtung vergleichsweise unproblematisch ist.
Die zentrale Idee der in diesem Kapitel vorgestellten Methode zur schnellen Fourier Rekonstruktion liegt nun darin, in Winkelrichtung lediglich linear zu interpolieren und in radialer Richtung mit dem im vorigen Kapitel entwickelten Verfahren zur nichtäquidistanten Fouriertransformation den (nahezu) exakten Wert zu berechnen.
Auf diese Weise gelangt man zu einem Algorithmus, der die nichtäquidistante Fouriertransformation nur für den eindimensionalen Teil benötigt, während für die zweidimensionale inverse Fouriertransformation eine Standard FFT-Routine verwendet werden kann. Dies ergibt zum einen ein gegenüber der Gridding Methode nochmals deutlich verbessertes Laufzeitverhalten und zum anderen kann man die eindimensionalen Berechnungen sehr gut parallel ausführen, falls man eine weitere Beschleunigung benötigt.

3.4 Schnelle Fourier-Rekonstruktion

Im folgenden bezeichne $g(s,\varphi) = \mathcal{R}f(s,\varphi)$ die Radon-Transformierte und $\hat{g}(r,\varphi)$ die eindimensionale Fourier-Transformierte von g nach dem ersten Parameter.
Die Berechnung von f aus g erfolgt in drei Schritten:

1. Eindimensionale nichtäquidistante Fourier-Transformationen zur Berechnung von $\hat{g}$.

2. Interpolation in Winkelrichtung auf das cartesische Gitter.

3. Filterung

4. zweidimensionale inverse FFT.

Die Filterung im dritten Schritt wird einfach durch Multiplikation der Funktion $\hat{f}$ mit einer geeigneten radialsymmetrischen Filterfunktion $F(r)$ realisiert, für die $F(0) = 1$ und $F(|r| \geq 1) = 0$ gilt. Eine Filterung bei der Inversion der Radon-Transformation ist allgemein üblich, für Details sei auf [21] verwiesen. Für die Beschreibung unseres Algorithmus ist die Filterung allerdings ohne Belang, wir nehmen daher F einfach als fest vorgegebene Funktion an.
Durch die Messungen gegeben sind die Werte

$$g_{jk} = g(j\pi/p\,,\, k/q) \qquad \text{für } j = 0, \dots, p-1 \text{ und } k = -q, \dots, q.$$

Wir wollen für die zweidimensionale inverse Fouriertransformation im dritten Schritt eine Standard-FFT-Routine verwenden. Hierzu benötigen wir folglich die Daten auf einem cartesischen Gitter.
Allerdings kann man die Anzahl der cartesischen Gitterpunkte, für die wir eine Interpolation benötigen, durch einige elementare Überlegungen vorab etwas reduzieren.
Die durch Interpolation gesuchten Daten auf dem cartesischen Gitter entsprechen der Fourier-Transformierten von f. Da f reellwertig ist, gilt $\hat{f}(x,y) = \overline{\hat{f}(-x,-y)}$, wie man leicht der Definition der Fouriertransformation entnimmt. Für den diskreten Fall liest sich das als

$$\begin{aligned} \hat{f}_{jk} &= \sum_{l=-N/2}^{N/2-1} \sum_{m=-N/2}^{N/2-1} f_{lm} e^{2\pi i j l/N} \cdot e^{2\pi i k m/N} \\ &= \sum_{l=-N/2}^{N/2-1} \sum_{m=-N/2}^{N/2-1} f_{lm} \overline{e^{-2\pi i j l/N}} \cdot \overline{e^{-2\pi i k m/N}} \\ &= \overline{\hat{f}_{-j,-k}}, \end{aligned}$$

da $f_{lm} = \overline{f_{lm}}$ gilt. Dies bedeutet, daß die Daten auf der oberen Halbebene ausreichen, um f zu rekonstruieren. Es existieren FFT-Implementierungen, die diese Symmetrie unterstützen und aus den komplexen Werten von $\hat{f}$ auf der oberen Halbebene f berechnen. Daraus ergibt sich in etwa eine Halbierung des Laufzeit- und Speicher-Verbrauchs der FFT Routine.

Weiterhin werden durch die Filterung alle Werte von $\hat{f}(x)$ für $||x|| \geq 1$ unterdrückt und müssen daher auch nicht berechnet werden.
Man benötigt also die Funktionswerte $\hat{f}(x_1, x_2)$ auf den Punkten des cartesischen Gitters $x = (x_1, x_2) \in \frac{1}{N}\mathbb{Z}^2$ mit $||x|| \leq 1$ und $x_2 \geq 0$.

Um die Werte an diesem Punkten durch Interpolation in Winkelrichtung zu erhalten, benötigen wir die Daten auf den benachbarten beiden Strahlen; Abbildung 3.3 verdeutlicht dies.

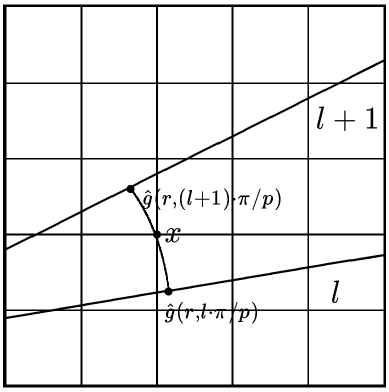


Abbildung 3.3: Interpolation eines Punktes auf dem cartesischen Gitter

Seien l_x und $(l_x + 1)$ die zu x nächstgelegenen Strahlen. Wir benötigen dann die Werte von $\hat{g}(r, l_x \cdot \pi/p)$ und $\hat{g}(r, (l_x + 1) \cdot \pi/p)$. Die beiden benötigten Werte von $\hat{g}$ lassen sich mit $r = ||x||$ wie folgt berechnen:

$$\begin{aligned} \hat{g}(r, l_x \cdot \pi/p) &= \sum_{k=-q}^{q} g_{l_x k} e^{i\pi rk} \\ \hat{g}(r, (l_x + 1) \cdot \pi/p) &= \sum_{k=-q}^{q} g_{l_x+1,k} e^{i\pi rk}. \end{aligned}$$

Dies bedeutet, daß wir die Fourier-Transformation der äquidistant gesampelten Funktion $g(\cdot, l_x \cdot \pi/p)$ bzw. $g(\cdot, (l_x+1) \cdot \pi/p)$ an einer beliebigen (nichtäquidistanten) Stützstelle r auswerten.
Führt man diese Prozedur für alle benötigten Punkte des cartesischen Gitters durch, so ergibt sich für jeden Strahl l mit $0 \leq l < p$ eine Menge von

Auswertungspunkten $\{x_j^l\}_{j=1}^{M_l}$. Diese Liste der Auswertungspunkte muß vorab erstellt werden, damit wir im ersten Schritt die eindimensionalen Fourier-Transformationen direkt an den entsprechenden Punkten x_j^l auswerten können.
Kleinere Schwierigkeiten ergeben sich nur für den Fall, das es sich bei den beiden nächstgelegenen Strahlen l_x und $l_x + 1$ um die Strahlen $p - 1$ und p handelt. Die Messungen werden ja nur für $l = 0, \ldots, p - 1$ durchgeführt. Allerdings liefert die Symmetrie der Radon-Transformation

$$\mathcal{R}f(s, \varphi + \pi) = \mathcal{R}f(-s, \varphi)$$

sofort, daß die Messungen $g(s, l\pi/p)$ für $l = 0$ denen von $g(-s, l\pi/p)$ für $l = p$ entsprechen, was geometrisch ja auch unmittelbar einsichtig ist. Wir können deshalb im folgenden annehmen, daß die Messungen für die Winkel $l \cdot \pi/p$, $l = 0, \ldots, p$ vorliegen und man somit für den Fall $l_x + 1 = p$ keine weiteren Vorkehrungen treffen muß.
Insgesamt ergibt sich das in Abbildung 3.4 dargestellte Interpolationsschema. Mit kleinen Kreisen sind diejenigen Punkte des cartesischen Gitters markiert, auf denen wir die $\hat{f}$ benötigen, um eine zweidimensionale Standard–FFT–Routine anwenden zu können. Diese Punkte werden linear aus den Daten auf den beiden benachbarten Strahlen interpoliert, und zwar aus den Werten an den durch die ausgefüllten Punkte markierten Stellen.

3.5 Reskalierung

Die numerischen Untersuchungen werden zeigen, daß insbesondere bei simulierten Daten in bestimmten Fällen Aliasing-Effekte auftreten. Das bedeutet, die Meßdaten sind nicht ausreichend bandbeschränkt und dadurch entstehen im rekonstruieren Bild Artefakte, d.h. niederfrequente Fehler. Dies kann man einfach dadurch umgehen, daß man das Rekonstruktionsgebiet künstlich vergrößert und auf diesem vergrößerten Gebiet rekonstruiert. Bezeichnet man den Faktor, um den das Gebiet vergrößert wurde, mit m, so entspricht dies der folgenden Vorgehensweise:

1. Gegen seien Meßdaten

$$g_{jk} = \mathcal{R}f(j\pi/p\,,\, k/q) \qquad \text{für } j = 0, \ldots, p-1 \text{ und } k = -q, \ldots, q$$

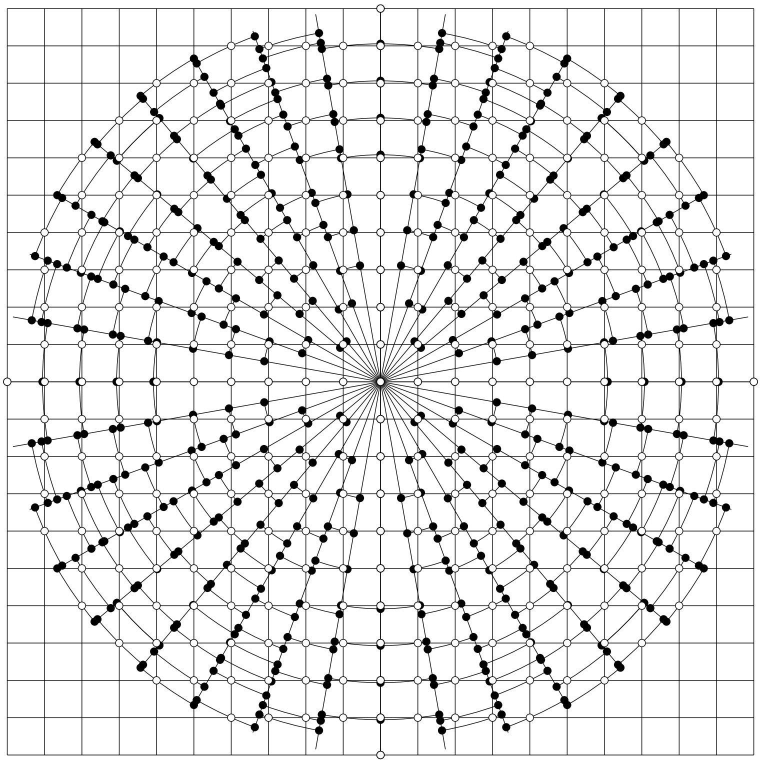


Abbildung 3.4: Interpolationsschema der schnellen Fourier-Rekonstruktion

Erzeuge daraus Meßdaten $\bar{g}_{jk}$ für ein vergrößertes Gebiet durch

$$\bar{g}_{jk} = \begin{cases} g_{jk} & \text{für } k = -q, \ldots, q \\ 0 & \text{für } q < |k| \leq mk \end{cases}$$

für $j = 0, \ldots, p-1$.

2. Rekonstruiere $\bar{f}$ anhand von $\bar{g}_{jk}$ auf dem vergrößerten Gebiet $mN \times mN$.

3. Die Einschränkung von $\bar{f}$ auf die in der Mitte gelegenen $N \times N$ Werte ist die gesuchte Rekonstruktion f.

Diese Vorgehensweise bewirkt, daß im Frequenzraum mehr Frequenzen berücksichtigt werden und unterdrückt so bei geeigneter Wahl von m (typischerweise 1.5 oder 2) die Aliasing-Effekte.
In der praktischen Anwendung ist diese Vorgehensweise vermutlich nicht notwendig, da die Probleme sich nur aus der Simulation der Meßdaten ergeben, wie die numerischen Experimente zeigen werden.

3.6 Rebinning

Moderne CT-Scanner liefern die Daten in der sogenannten Fächer-Geometrie (fan beam geometry). Um unseren Algorithmus anwenden zu können, benötigen wir die Daten jedoch in der Parallelgeometrie. Dieser Abschnitt liefert eine kurze Beschreibung der für diese Transformation notwendigen Schritte, wir folgen hierbei wiederum der Darstellung in [21]:
Sei $A = r(\cos\beta, \sin\beta)$ ein Punkt der Ebene (die Röntgenquelle) mit Abstand r vom Ursprung. Bezeichnet man die Gerade, die durch A läuft und einen Winkel α mit der Gerade $\overline{0A}$ bildet mit $L(\beta, \alpha)$, so besteht zwischen den Koordinaten (β, α) der Fächergeometrie und (r, φ) der Parallelgeometrie der folgende Zusammenhang:

$$s = r\sin\alpha \qquad \text{und} \qquad \varphi = \beta + \alpha - \pi/2,$$

also

$$\alpha = \arcsin(s/r) \qquad \text{und} \qquad \beta = \varphi - \arcsin(s/r) + \pi/2.$$

Wenn $\mathbf{D}(\beta, \alpha)$ die Meßdaten bezüglich der Fächergeometrie bezeichnet, so erhalten wir daraus die von uns benötigten Daten der Parallelgeometrie durch

$$g_{jk} = \mathcal{R}f(j\pi/p\,,\, k/q) = \mathbf{D}\left(j\pi/p - \arcsin(k/qr) + \pi/2, \arcsin(k/qr)\right) \quad (3.4)$$

für $j = 0, \ldots, p-1$ und $k = -q, \ldots, q$. Da in der Praxis natürlich auch $\mathbf{D}(\beta, \alpha)$ nur für diskrete Werte $\mathbf{D}(\beta_l, \alpha_m)$ gegeben ist, erfordert dies eine entsprechende Interpolation. Für gerade p und $\alpha_m = m \cdot \pi/p$ kommt man allerdings für das erste Argument von $\mathbf{D}$ mit einer einfachen Umordnung aus (vergleiche auch [16], Abschnitt 3.4.3). Die Interpolation im zweiten Argument ist unproblematisch, einfache lineare Interpolation liefert bereits zufriedenstellende Resultate. Berechnet man die entsprechenden Approximationskoeffizienten vorab, so kann dieser üblicherweise „Rebinning" genannte

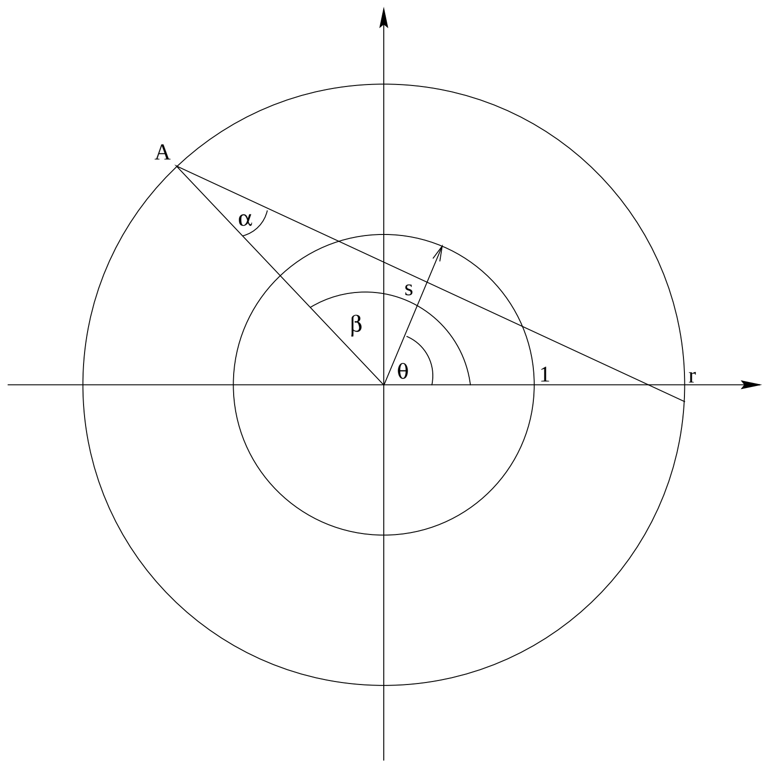


Abbildung 3.5: Parameter der Fan-Beam-Geometrie und der Parallel-Geometrie

Interpolations- bzw. Umordnungsvorgang somit in $3pq$ Rechenoperationen durchgeführt werden, was das Laufzeitverhalten des gesamten Rekonstruktionsalgorithmus nur unmerklich verschlechtert.

Die Rekonstruktionen in Abbildung 3.12 sind aus Meßdaten entstanden, die zuvor auf diese Weise bearbeitet wurden. Gegenüber einer gefilterten Rückprojektion, die direkt auf Fan-Beam Daten aufbaut war kein Unterschied in der Rekonstruktion festzustellen. Das Rebinning stellt also kein größeres Problem dar, wir werden deshalb im folgenden von Parallelgeometrie-Daten ausgehen.

3.7 Detaillierte Beschreibung der Algorithmen

Der Algorithmus zur schnellen Fourier-Rekonstruktion besteht aus zwei Stufen. Zuerst die nur von der Geometrie abhängigen Vorberechnungen:

Vorberechnungen:

Vorgegeben seien die Parameter p für die Anzahl der Richtungen, q für $2q+1$ Messungen pro Richtung, N für eine Rekonstruktionsgröße $N \times N$ und eine geeignete Filterfunktion F mit $F(0) = 1$ und $F(1) = 0$.

1. Für jeden Punkt $x = (x_1, x_2)$ des cartesischen Gitters $\frac{1}{N}\mathbb{Z}^2$ mit $x_1 \geq 0$ und $r = ||x|| \leq 1$:
   (a) Finde die beiden zu x nächstgelegenen Strahlen l_x und (l_x+1) bzw denjenigen Strahl l_x, der direkt durch x läuft.
   (b) Füge $\pi \cdot ||x||$ zur Liste $\{x^{l_x}\}$ bzw. $\{x^{l_x+1}\}$ der Auswertungspunkte für die betreffenden Strahlen hinzu.
   (c) Berechne die Koeffizienten a_x und b_x für die lineare Interpolation gemäß:
   $$a_x = F(r) \cdot |\varphi - l_x\pi/p| \cdot p/\pi$$
   $$b_x = F(r) \cdot |\varphi - (l_x+1)\pi/p| \cdot p/\pi.$$
   Hierbei ist $\varphi = \arctan(x_2/x_1)$.

2. Vorberechnung der Daten für die nichtäquidistanten Fourier-Transformationen. Initialisiere für jeden Strahl $l = 0, \ldots, p$ die Koeffizienten zur schnellen Berechnung von
   $$\hat{g}_{jl} = \sum_{k=-q}^{q} g_{kl} e^{ikx_j^l}$$
   gemäß den Vorgaben aus Abschnitt 2.7.1. Dazu ist jeweils ein Stützstellenvektor μ^l und eine Koeffizientenmatrix σ^l zu erzeugen. Der Gewichtsvektor w ist für alle Strahlen l identisch und muß nur einmal angelegt werden.

Für den eigentlichen Algorithmus sind dann nur noch die folgenden Schritte durchzuführen:

schnelle Fourier-Rekonstruktion

1. **eindimensionale nichtäquidistante Fouriertransformationen**
   für jeden Strahl $l = 0, \ldots, p$: berechne die eindimensionale nichtäquidistante Fouriertransformation

   $$\hat{g}_{jl} = \sum_{k=-q}^{q} g_{kl} e^{ikx_j^l}$$

   für $j = 1, \ldots, M_l$ mit der Methode aus Kapitel 2.

2. **lineare Interpolation vom Polargitter auf das Cartesische Gitter**
   für jeden Punkt des cartesischen Gitters $x = (x_1, x_2) \in \frac{1}{N}\mathbb{Z}^2$ mit $||x|| \leq 1$ und $x_2 \geq 0$: Berechne $\hat{f}(x)$ durch lineare Interpolation:

   $$f(x) = a_x \hat{g}_{l_x, j_x^1} + b_x \hat{g}_{l_x+1, j_x^2}$$

   j_x^1 und j_x^2 sind die Indizes der zugehörigen nichtäquidistanten Stützstellen. Die Werte von $\hat{g}$ entsprechen $\hat{g}(r\pi, l_x\pi/p)$ und $\hat{g}(r\pi, (l_x + 1)\pi/p)$.

3. **inverse FFT**
   Führe eine inverse Fourier-Transformation anhand der Funktionswerte $\hat{f}(x, y)$ für $y \geq 0$ durch. Das Ergebnis ist eine reellwertige Funktion f, das rekonstruierte Bild.

3.8 Betrachtung des Rechenaufwandes

Wir wollen in diesem Kapitel eine ungefähre Abschätzung des Rechenaufwandes angeben.
Alle nur von der Geometrie und dem verwendeten Filter abhängigen Daten seien vorberechnet. Außerdem nehmen wir der Einfachheit halber an, daß wir statt $2q + 1$ nur $2q$ Messungen pro Richtung durchführen und somit $2pq$ Eingangsdaten gegeben sind.

Zuerst werden die p eindimensionalen nichtäquidistanten Fourier–Transformationen gemäß der im vorherigen Kapitel, insbesondere in Abschnitt 2.7.1 vorgestellten Methode berechnet. Diese gliedert sich in drei Teile: Als erstes werden die (reellwertigen) Eingangsdaten skaliert, was $p \cdot 2q$ Rechenoperationen erfordert.
Danach ist für jede Richtung eine eindimensionale FFT erforderlich. Die Eingangsdaten der FFT sind reell, was zu einem in etwa halbierten Rechenaufwand gegenüber der komplexen FFT führt. Wir nehmen eine eindimensionale FFT der Länge n daher mit $2.5n \log n$ Rechenoperationen an. Wir benötigen p FFTs, jeweils der Länge $4q$, wenn wir für den nichtäquidistanten Algorithmus ein Oversampling von 2 annehmen. Somit ergibt sich ein Aufwand von $10pq \log(4q)$ Rechenoperationen.
Für jeden Auswertungspunkt ist anschließend ein eindimensionaler Gridding-Schritt der Länge K durchzuführen. Pro cartesischem Gitterpunkt sind das $2 \cdot 8K$ Rechenoperationen: $8K$ Rechenoperationen ergeben sich laut (2.66) in Kapitel 2 pro Auswertung an einer nichtäquidistanten Stützstelle, und pro cartesischem Punkt interpolieren wir linear aus zwei Werten $\hat{g}$. Diese lineare Interpolation ist von der Form:

$$f_{jk} = a_{jk} g_1 + b_{jk} g_2,$$

wobei g_1 und g_2 die entsprechenden Werte von $\hat{g}$ sind und a und b die Koeffizienten der linearen Interpolation darstellen. Diese werden vorberechnet. Man kann in sie auch die Filterwerte integrieren, so daß die Filterung keine weiteren Rechenoperationen benötigt. a und b sind reellwertig, während g_1 und g_2 komplex sind, daher benötigt diese Interpolation 6 Rechenoperationen (flops).
Wollen wir auf f einem $N \times N$ Gitter rekonstruieren, benötigen wir die Werte von $\hat{f}$ auf den Gitterpunkten in einem Halbkreis. Dies entspricht in etwa $\pi N^2/8$ Punkten. Somit ergibt sich für die Interpolations in Winkel- und und das Resampling in radialer Richtung ein Aufwand von zusammen etwa

$$\frac{\pi N^2}{8} \cdot (6 + 16K)$$

Rechenoperationen. Die abschließende zweidimensionale FFT rekonstruiert eine reellwertige Funktion, weshalb sie nur rund den halben Rechenaufwand einer komplexen FFT benötigt. Wir nehmen sie deshalb mit $5N^2 \log_2(N)$ an.

Der gesamte Rechenaufwand lautet somit:

$$\text{flops}_{FFR}(p,q,N) \approx 2pq + 10pq\log_2(4q) + \pi N^2(6+16K)/8 + 5N^2\log_2(N)$$

FFR steht hierbei für *fast fourier reconstruction.* Um diese Formel weiter zu vereinfachen, kann man für p und q typische Werte annehmen. Mathematische Betrachtungen der erreichbaren Auflösung (z.B. in [21]) legen es nahe, zur Rekonstruktion eines Bildes der Größe $N \times N$ die Parameter p und q durch

$$q = N/2 \qquad \text{und} \qquad p = \pi q = \pi N/2$$

festzulegen. In der Praxis begnügt man sich aus technischen Gründen häufig mit einer kleineren Anzahl p von Richtungen; wir setzen $p = 3q = 3N/2$ an, um eine einfachere Formel zu erhalten. Außerdem wählen wir $K = 3$, was eine eindimensionale nichtäquidistante Fouriertransformation im Rahmen der einfachen Maschinengenauigkeit ergibt und somit für unsere Zwecke mehr als ausreichend ist.
Mit $\pi(6+16K)/8 \approx 22$ berechnet sich der Rechenaufwand dann zu:

$$\begin{aligned}\text{flops}_{FFR}(N) &\approx 3N^2/2 + 15N^2\log_2(2N)/2 + 22N^2 + 5N^2\log_2 N\\ &\approx 3N^2/2 + 15N^2\log_2(2)/2 + N^2\log_2 N\,(15/2+5) + 22N^2\\ &= 31N^2 + 12.5N^2\log_2 N \qquad (3.5)\end{aligned}$$

Dies kann man mit der Anzahl der mindestens für eine Radon-Inversion benötigten Rechenoperationen vergleichen. Diese ergibt sich bei direkter Anwendung des Fourier-Slice-Theorems, d.h. man führt für jede Richtung eine FFT der Länge $2q$ durch und berechnet dann ohne weitere Interpolation eine zweidimensionale inverse FFT der Länge $N \times N$. Der Rechenaufwand hierfür beträgt dann

$$\text{flops}_{min}(N) = p(5\cdot 2q\log_2(2q) + 5N^2\log(N) = 12.5N^2\log(N)$$

Man sieht, daß unser Algorithmus zur schnellen Fourier-Rekonstruktion nicht wesentlich über dieser minimal benötigten Anzahl Rechenoperationen liegt, insbesondere ist der mit $N^2\log(N)$ wachsende Anteil identisch.
Der Algorithmus zur schnellen Fourier-Rekonstruktion bewegt sich damit in der Größenordnung zweidimensionaler FFTs. Wir werden bei den numerischen Experimenten sehen, daß dies ein ein großer Fortschritt gegenüber der gefilterten Rückprojektion mit $\mathcal{O}(N^3)$ Rechenoperationen ist.

Tabelle 3.1 zeigt die Verteilung des Rechenaufwandes auf die einzelnen Schritte exemplarisch für $N = 256$. Neben dem prozentualen Anteil an Rechenoperationen, sind auch die bei tatsächlichen Laufzeitmessungen ermittelten Verhältnisse eingetragen.
Man erkennt, daß der Interpolationsschritt auf das zweidimensionale Gitter in der Praxis etwas länger dauert, als es die Anzahl an Rechenoperationen vermuten ließe; dies liegt daran das hierbei auf ständig wechselnde Speicherbereiche zugegriffen wird.
Mehr als drei Viertel der Rechenoperationen entfallen auf FFT-Routinen, das Laufzeitverhalten wird also primär durch die Geschwindigkeit der FFT bestimmt. Die restliche Laufzeit entfällt größtenteils auf das eindimensionale Resampling. Hierbei handelt es sich um eine vergleichsweise einfache Routine, die man zudem gut parallel ausführen kann, da die Daten für jede Richtung $\varphi = l \cdot \pi/p$ getrennt betrachtet werden.
Dies ist auch ein Vorteil gegenüber der Gridding-Methode von O'Sullivan. Der dazu nötige zweidimensionale Resampling- bzw. Gridding-Schritt ist zum einen deutlich rechen-aufwendiger als das von uns bevorzugte eindimensionale Verfahren, zum anderen läßt sich dieser Schritt auch nicht so gut parallelisieren oder mit Hardware-Bauteilen durchführen.

Abschnitt	$\approx flops$	flops	Laufzeit
Skalieren der Eingangsdaten	$2pq$	1%	3%
p eindimensionale FFTs	$10pq \log_2 4q$	50%	50%
eindimensionales Resampling	$2\pi K N^2$	15%	12%
Interpolation polar nach cartesisch	$3\pi N^2/4$	2%	11%
zweidimensionale inverse FFT	$5N^2 \log_2 N$	32%	24%
Gesamt	$\mathcal{O}(N^2 \log_2 N)$	100%	100%

Tabelle 3.1: Verteilung des Rechenaufwandes ($N = 256$, $p = 400$, $q = 128$)

3.9 Implementierung und Laufzeitverhalten

Für die numerischen Untersuchungen wurde der Algorithmus in C implementiert und wiederum auf einer 300MHz Sun UltraSPARC-II getestet. Die

Compilierung erfolgte auch hier mit dem Sun C-Compiler und der Option „-fast", d.h. voller Optimierung. Als FFT-Routine wurde die FFTW-Bibliothek verwendet, da diese spezielle Routinen für Fourier-Transformationen reeller Daten bereitstellt. Die FFTW-Bibliothek ist über Netlib (`www.netlib.org`) frei verfügbar.
Die Implementierung berechnet sämtliche Daten, die nur von der Geometrie abhängen, im voraus. Die dazu benötigte Zeit wurde bei den Laufzeitmessungen natürlich nicht berücksichtigt.
Um die Laufzeit mit der gefilterten Rückprojektion zu vergleichen, wurde eine Standard-Routine zur gefilterten Rückprojektion verwendet. Effiziente Implementierungen der gefilterten Rückprojektion verwenden für den Filterungs-Schritt FFT-Methoden. Unsere Rückprojektions–Implementierung tut dies nicht. Deshalb wurden bei den Laufzeitmessungen nur die Anteile für die reine Rückprojektion gemessen. Eine vollständige gefilterte Rückprojektion benötigt also länger als in der Tabelle mit den Laufzeitmessungen angegeben. Die Rückprojektions-Routine wurde effizient implementiert und natürlich mit dem selben Compiler, identischen Compileroptionen und auf dem selben Rechner bearbeitet.
Damit ergibt sich das in Tabelle 3.2 angegebene Laufzeitverhalten. Die einzelnen Spalten haben die folgenden Bedeutungen: p und q bezeichnen wie üblich die Anzahl der Richtungen bzw. die Messungen je Richtung (2q+1 Messungen pro Richtung). $N \times N$ ist die Größe des Gitters, auf dem rekonstruiert wird. „Size" bezeichnet den Reskalierungs–Faktor aus Abschnitt 3.5. Bei einer Reskalierung mit dem Faktor 2 wurde also tatsächlich ein Bild mit einem um den Faktor 2 vergrößerten q und N rekonstruiert.
BP bezeichnet die Zeit, die eine reine Rückprojektion (ohne Filterung) benötigt. *FFR* bezeichnet die Zeit für eine schnelle Fourier-Rekonstruktion (Fast Fourier Reconstruction) mit unserem Algorithmus, *% FFT* gibt an, wieviele Anteile davon auf FFT-Routinen entfallen. *FFT* ist die nur zu Vergleichszwecken angegebene Zeit für eine zweidimensionale komplexe FFT der Länge $N \times N$. Alle Zeitangaben sind in Millisekunden.
Das wesentliche Ergebnis findet sich in der Spalte *Faktor BP/FFR*. Diese Spalte gibt an, wieviel mal schneller der Algorithmus gegenüber der Rückprojektion und damit gegenüber der gefilterten Rückprojektion ist. Der Unterschied in der Komplexität zwischen $\mathcal{O}(N^2 \log N)$ und $\mathcal{O}(N^3)$ macht sich wie man sieht deutlich bemerkbar.
Der *FFR*-Algorithmus benötigt bei einem Rekonstruktionsgebiet der Größe

256×256 und $p = 400$ für die Rekonstruktion 60ms, ermöglicht also 16 Rekonstruktionen pro Sekunde. Die gefilterte Rückprojektion hingegen benötigt für eine einzige Rekonstruktion bereits über 5 Sekunden.

p	q	N	Size	BP	*FFR*	FFT	% FFT	**Faktor FB/FFR**
200	64	128	1	652	12.1	4	55%	54
200	128	256	1	2665	44.4	19	45%	61
400	128	256	1	5260	57.6	18.4	52%	92
180	256	512	1	9627	205.6	144.8	44%	47
400	256	512	1	21235	225.6	128.8	50%	94
800	256	512	1	43094	287.2	127.8	55%	150
200	64	128	1.5	652	27.4	4	57%	24
200	64	128	2	652	44.9	3.9	45%	15
400	128	256	1.5	5289	154.8	18.4	58%	34
400	128	256	2	5247	221.8	18.2	50%	24
800	256	512	1.5	43093	679.2	121.2	60%	63
800	256	512	2	42913	1133	125.4	59%	38

Tabelle 3.2: Laufzeitverhalten für den FFR-Algorithmus

3.10 Genauigkeit und Vergleich mit der gefilterten Rückprojektion

Was die Geschwindigkeit angeht übertrumpft der Algorithmus die gefilterte Rückprojektion also um mehr als nur eine Größenordnung.
Der kritische Punkt bei auf Fourier-Methoden aufbauenden Algorithmen ist allerdings immer die Rekonstruktionsqualität.
Um diese zu überprüfen und mit der gefilterten Rückprojektion zu vergleichen, wurden verschiedene Phantome erstellt. Dann wurde die zugehörige Radon-Transformierte ermittelt, um den Meßvorgang zu simulieren.
Als erstes wurde das in der Computer-Tomographie häufig verwendete Shepp-Logan-Phantom bemüht. Dies besteht aus 11 einander überlagernden Ellipsen, die entsprechenden Daten sind in Tabelle 3.3 aufgelistet. Die Absorptionskoeffizienten sind in sogenannten Hounsfield-Einheiten angegeben, einer

in der Computer-Tomographie gebräuchlichen Skalierung.
Um etwas realistischere Meßwerte zu erhalten, wurden die Meßwerte nicht einfach gemäß (3.3) berechnet, sondern durch die folgende Formel gemittelt:

$$g_{jk} = \frac{1}{(20+1)^2} \sum_{l=-20}^{20} \sum_{m=-20}^{20} \mathcal{R}f\left((j+l/N)/q, (k+m/N)\pi/q\right) \tag{3.6}$$

Die Funktionswerte von Rf wurden hierbei analytisch aus der Superposition der in Tabelle 3.3 angegebenen Ellipsen berechnet.
Somit setzt sich ein Meßwert aus vielen leicht versetzen Strahlen zusammen, was der Realität besser entspricht. Verzichtet man auf diese Mittelung, entstehen in der Rekonstruktion Aliasing-Artefakte aufgrund nicht ausreichender Bandbeschränktheit, die in realen Messungen nicht auftreten.
Bei der Rekonstruktion wurde für die gefilterte Rückprojektion ein Shepp-Logan-Filter und für die Fourier-Rekonstruktion ein sinc^3 Filter gewählt. Die unterschiedliche Filter-Wahl begründet sich dadurch, das bei der gefilterten Rückprojektion während der Rückprojektion interpoliert wird. Die Auswirkungen dieser Interpolation lassen sich durch einen zusätzlichen sinc^2-Filter bei der Fourier-Inversion beschreiben, so daß man für die Fourier-Rekonstruktion insgesamt einen sinc^3 Filter erhält. Dies entspricht auch der Wahl der Filter in [27].

Abbildung 3.7 zeigt das durch dieses Phantom beschriebene Bild und zwei Rekonstruktionen: einmal mittels gefilterter Rückprojektion und einmal mittels schneller Fourier-Rekonstruktion. Wie man sieht liefern beide Algorithmen eine sehr gute Rekonstruktionsqualität. Wie die Querschnitte zeigen, liegen die Fehler im Bereich von unter einer halben Hounsfield-Einheit. Dies entspricht weniger als einem halben Promille des Wertebereichs von f.
Für die *FFR*-Rekonstruktion wurde für diese Rekonstruktion ein Reskalierungs-Faktor von 1.5 gewählt. Die nächste Abbildung zeigt, wieso diese Reskalierung notwendig ist.
Das linke Bild in Abbildung 3.8 zeigt eine Rekonstruktion ohne Reskalierung. Obwohl die Artefakte nur eine halbe Hounsfield-Einheit ausmachen, sind sie schwach als regelmäßiges Muster zu erkennen. Diese Artefakte entstehen allerdings keineswegs als Interpolationsfehler. Rekonstruiert man das Bild mit vollständiger Sinc-Interpolation (z.B. mit der von Stark in [31] entwickelten Formel) in $\mathcal{O}(N^4)$ Rechenoperationen, so erhält man dieselben Artefakte und

Mittelpunkt		Achse 1	Achse 2	Rotation	Absorption
0.00	0.00	0.94	0.71	1.5708	980
0.00	0.00	0.92	0.69	1.5708	1000
0.00	-0.0184	0.874	0.6624	1.5708	-980
0.22	0.00	0.31	0.11	1.2566	-20
-0.22	0.00	0.41	0.16	1.885	-20
0.00	0.35	0.25	0.21	1.5708	10
0.00	0.01	0.046	0.046	0	10
0.00	-0.01	0.046	0.046	0	10
-0.08	-0.605	0.046	0.023	0	10
0.00	0.605	0.023	0.023	0	10
0.06	-0.605	0.046	0.023	1.5708	10

Tabelle 3.3: Parameter für das Shepp-Logan Phantom

denselben Rekonstruktionsfehler. Es handelt sich hierbei also um ein echtes Aliasing: die höheren Frequenzen erzeugen einen niederfrequenten Fehler.
In diesem Falle hilft allerdings wie bereits erwähnt die in Abschnitt 3.5 beschriebene Reskalierung.
Man erweitert die Anzahl der Messungen pro Winkel (den Parameter q) und füllt die zusätzlichen Werte durch Nullen auf. Dann rekonstruiert man das somit vergrößerte Rekonstruktionsgebiet und wählt anschließend den Ausschnitt, der das ursprüngliche Rekonstruktionsgebiet enthält. Dies führt zwar zu einer Verschlechterung des Laufzeitverhaltens, ungefähr um den Faktor 2 bei 1.5facher Reskalierung und um den Faktor 4 bei 2facher Reskalierung, beseitigt die Artefakte aber zuverlässig, wie Abbildung 3.8 zeigt.
Allerdings ist fraglich, ob dies bei realen Messungen überhaupt notwendig ist. Die Artefakte rühren nämlich daher, daß das Phantom in Winkelrichtung eine zu große Bandbreite besitzt, bedingt durch die große Flankensteilheit beim Übergang zwischen ,,Luft" und ,,Schädelknochen". In Abbildung 3.9 wurde das Shepp-Logan-Phantom deshalb nochmals rekonstruiert, diesmal mit einem Kreis statt einer Ellipse als Schädelform. Die Querschnitte belegen, daß die schnelle Fourier-Rekonstruktion in diesem Falle auch ohne Reskalierung Ergebnisse liefert, die die Qualität der gefilterten Rückprojektion erreichen

oder sogar übertreffen.

Interessanterweise ließen sich die Aliasing-Effekte weder durch eine Erhöhung der Anzahl von Richtungen p noch durch eine irgendwie geartete Filterung der Eingangsdaten entfernen; nur der Reskalierung gelingt dies.

Bild 3.10 zeigt ein anderes Phantom. Die darin enthaltenen Rechtecke erzeugen gerade Kanten mit hartem Kontrast. Diese bewirken bei der Rekonstruktion mittels gefilterter Rückprojektion Streifen-Artefakte als Verlängerung der Kanten. Die Artefakte sind im Ausdruck möglicherweise nicht zu erkennen, zeigen sich aber im unteren Querschnitt. Erfreulicherweise treten diese Artefakte beim *FFR*-Algorithmus nicht auf.

Zusätzlich kann man an diesem Phantom die erreichte Auflösung untersuchen, da die Rechtecke eine unterschiedliche Breite haben. Der mittlere Querschnitt zeigt, welche dieser Rechtecke noch mit dem vollen Kontrast wiedergegeben werden. Hierbei ergeben sich Vorteile für die gefilterte Rückprojektion. Allerdings liegt der Grund hierfür in der unterschiedliche Wirkung der Filter bei Rückprojektion bzw. Fourier-Rekonstruktion. Bei anderer Filter-Wahl (z.B. cos-Filter) ergab sich ein teilweise umgekehrtes Verhalten. Im Sinne der Abtasttheorie können Details, die größer als die Breite des vierten Rechtecks sind, vollständig rekonstruiert werden. Dies gelingt beiden Algorithmen.

Das Phantom in Abbildung 3.11 zeigt einen weiteren Problemfall, ein Objekt mit großem Kontrast am Rande des Rekonstruktionsgebietes. Das rot dargestellte Gewebe liegt bei 1000 Hounsfield Einheiten, der weiße Ring und der schwarze Punkt bei 0 bzw. 2000. Die mittlere Reihe zeigt dieselben Rekonstruktionen nochmals, diesmal aber mit einem anderen Farbkeil, um auch im Ausdruck die Struktur der entstehenden Artefakte deutlich zu machen.

Der Punkt mit dem hohen Kontrast erzeugt bei der gefilterten Rückprojektion kreisförmige Artefakte. Die Fourier-Rekonstruktion ist bei Verwendung der Reskalierung um den Faktor 2 nahezu Artefakt-frei.

Abbildung 3.12 zeigt schließlich die Rekonstruktion eines realen Datensatzes. Gefilterte Rückprojektion und schnelle Fourier-Rekonstruktion liefern hier praktisch gleiche Ergebnisse. Die Unterschiede entstehen hauptsächlich durch die Filterwirkung der Interpolation bei Anwendung der gefilterten Rückprojektion.

3.11 Mögliche Erweiterungen

In diesem Kapitel wurde ein Algorithmus zur schnellen numerischen Inversion der Radon-Transformation entwickelt, der dem derzeitigen Standard-Verfahren (der gefilterten Rückprojektion) sowohl in Fragen der Rekonstruktionsqualität als auch der Geschwindigkeit überlegen ist.
Die Unterschiede in der Komplexität von $\mathcal{O}(N^2 \log N)$ bei der schnellen Fourier Rekonstruktion gegenüber $\mathcal{O}(N^3)$ bei der gefilterten Rückprojektion ermöglichen bereits bei moderaten Rekonstruktionsgrößen Beschleunigungsfaktoren größer als 30.
Zu noch größerer Bedeutung gelangt dieser Geschwindigkeitsvorteil bei der Erweiterung auf den dreidimensionalen Fall. Hierbei wird die (dreidimensionale) Röntgen-Transformation gemessen. Die Röntgen-Transformation ordnet einer Funktion die Menge ihrer Linienintegrale zu, was im zweidimensionalen daher bis auf Fragen der Notation mit der Radon-Transformation übereinstimmt. Zur Inversion der dreidimensionalen Röntgen-Transformation kann man die Formel von Grangeat [12] anwenden. Diese Formel transformiert das Problem im Wesentlichen auf die Inversion der Ableitung einer dreidimensionalen Radon-Transformation. Für die Radon-Transformation gilt aber auch im dreidimensionalen das Fourier-Slice Theorem: die eindimensionale Fouriertransformation der Messungen bezüglich einer Richtung liefert einen Schnitt durch die jetzt dreidimensionale Fourier-Transformierte von f. Unser Algorithmus zur schnellen Fourier-Rekonstruktion läßt sich somit direkt auf den dreidimensionalen Fall erweitern. In [26] wurde die Gridding–Methode bereits auf diese Weise für die Inversion der Röntgen-Transformation verwendet.

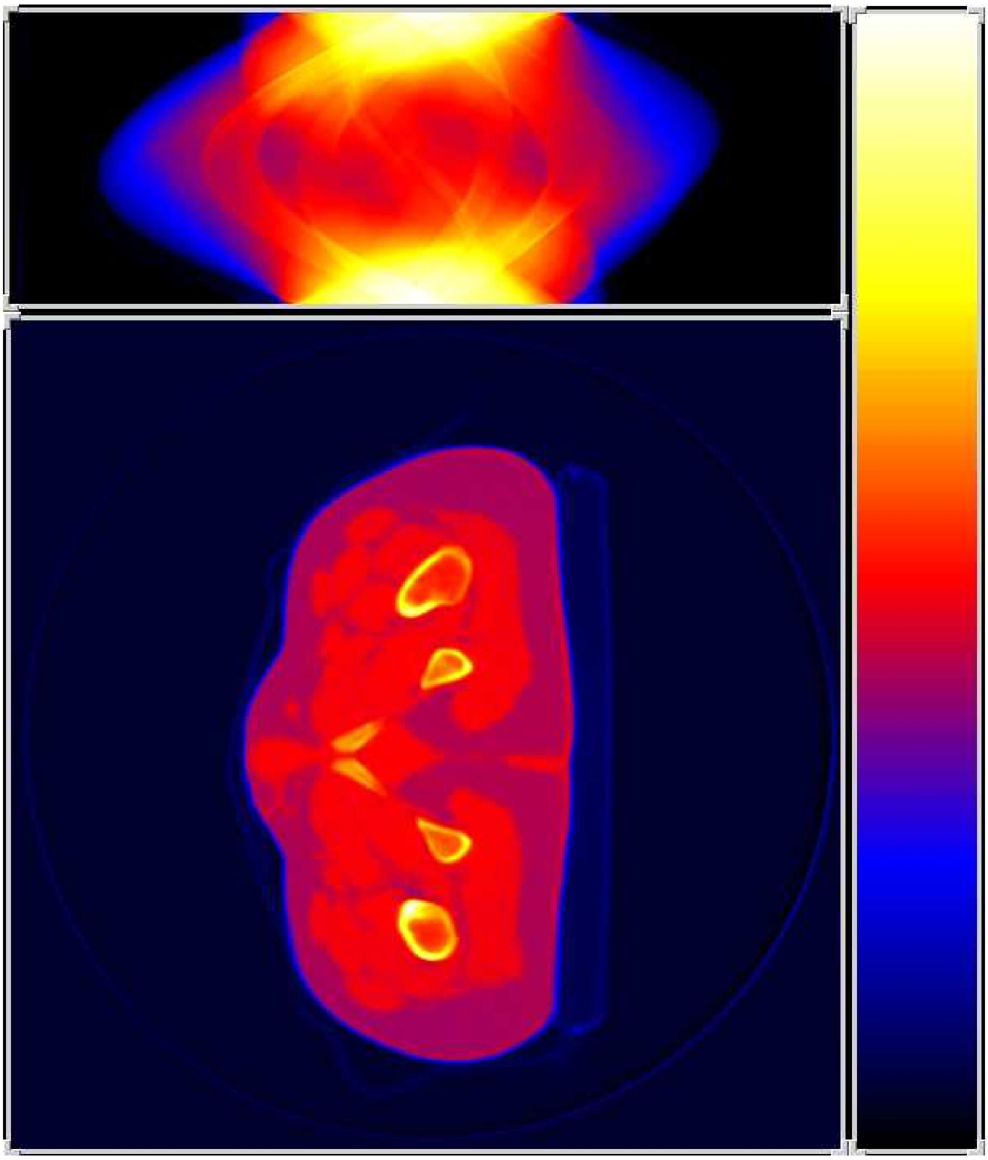
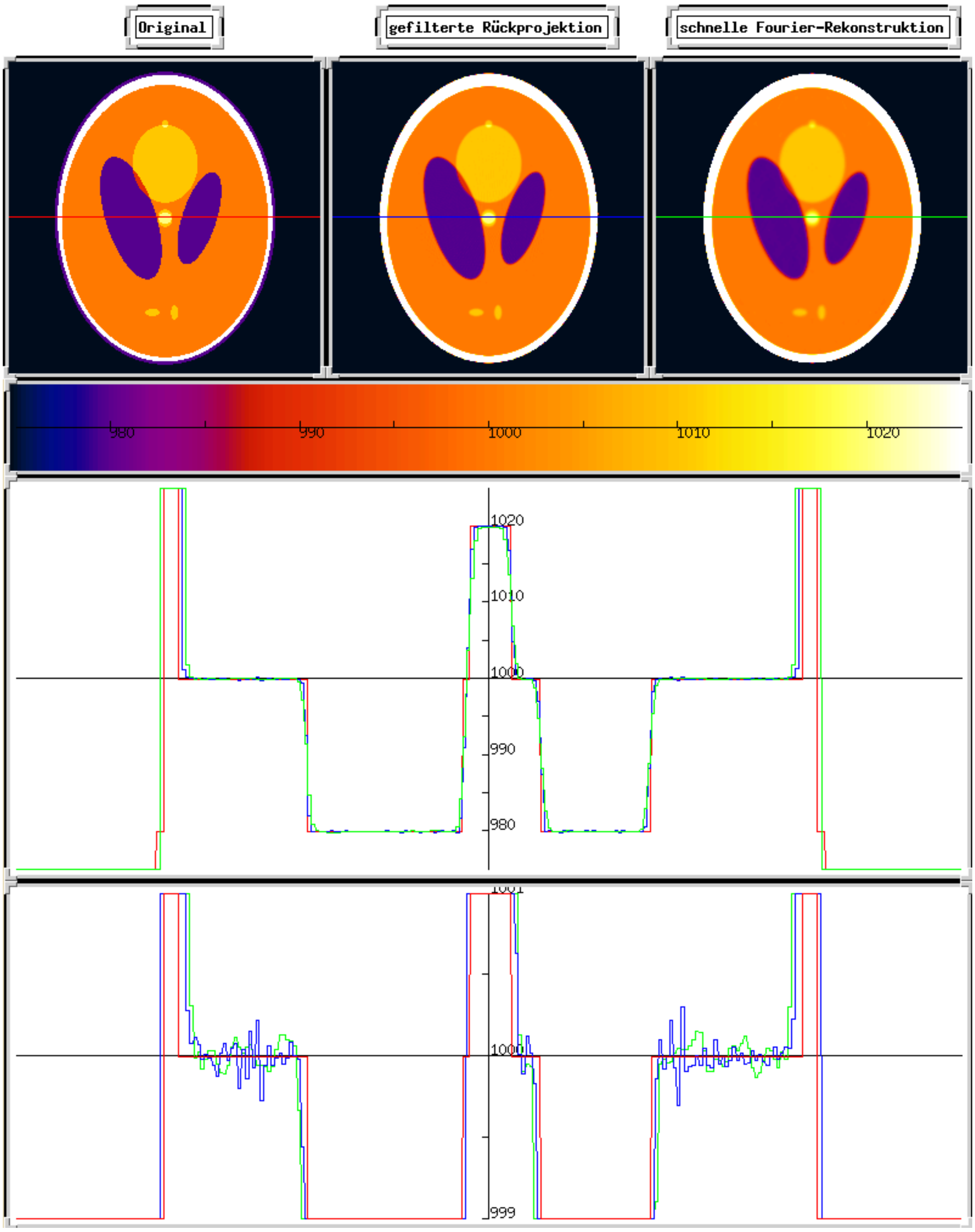
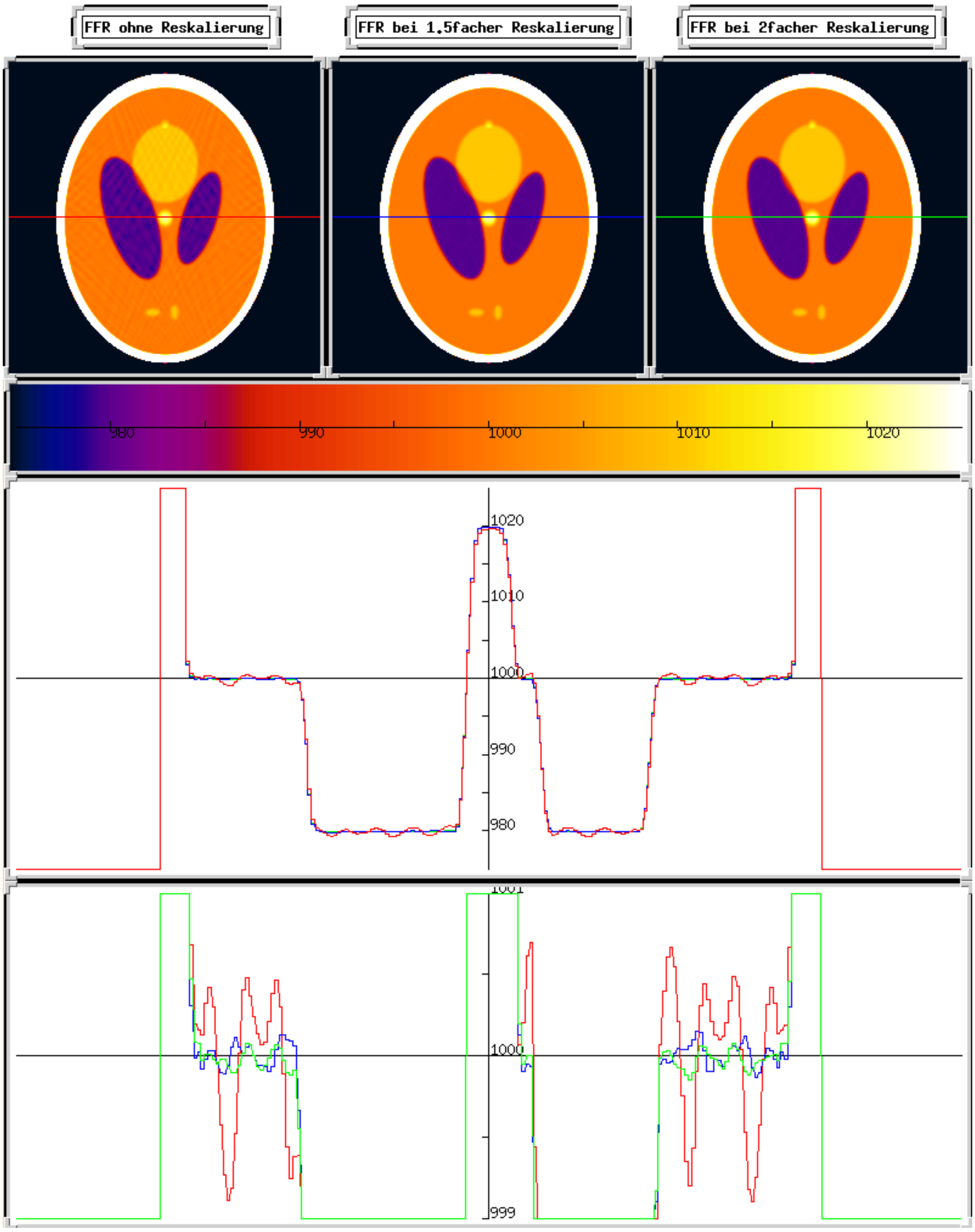


Abbildung 3.6: Radon-Transformation mit Rekonstruktion



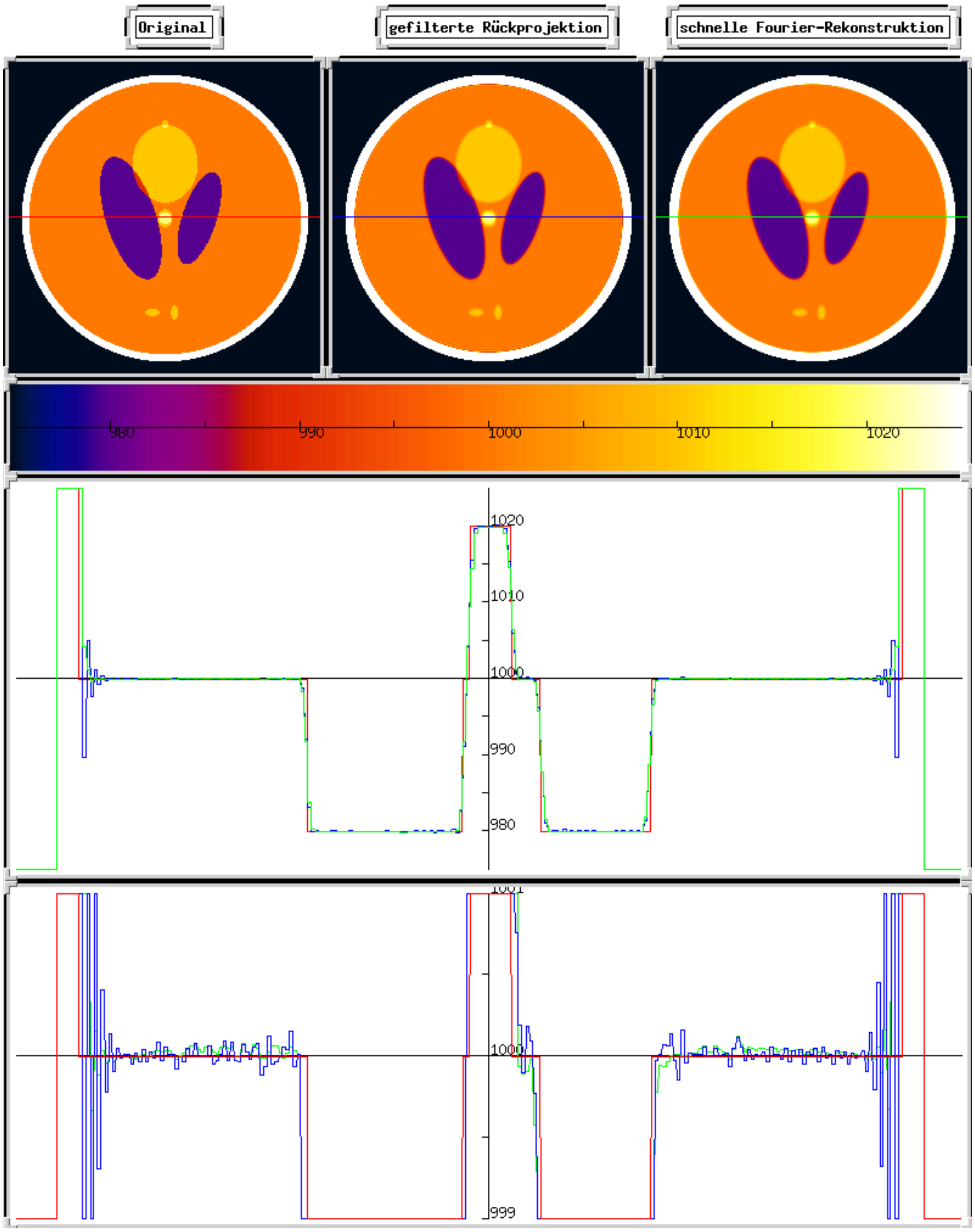
Parameter $p = 400$, $q = 128$, 256×256-Rekonstruktion	
Links:	Original
Mitte:	gefilterte Rückprojektion mit Shepp-Logan Filter
Rechts:	schnelle Fourier-Rekonstruktion mit sinc^3 Filter, 1.5fache Reskalierung

Abbildung 3.7: Original, gefilterte Rückprojektion und schnelle Fourier Rekonstruktion



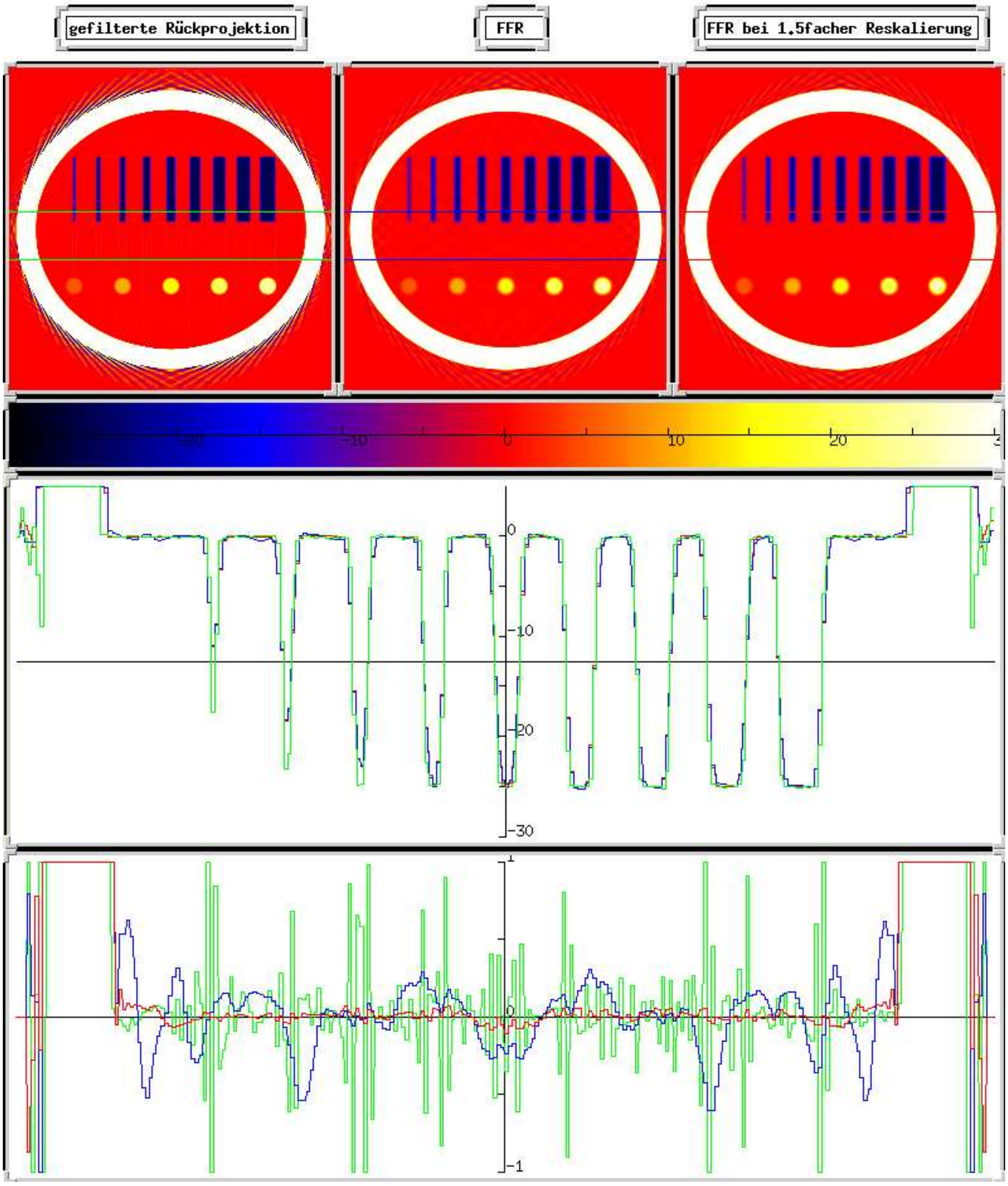
Parameter $p = 400$, $q = 128$, 256×256-Rekonstruktion

Abbildung 3.8: Auswirkung der Reskalierung



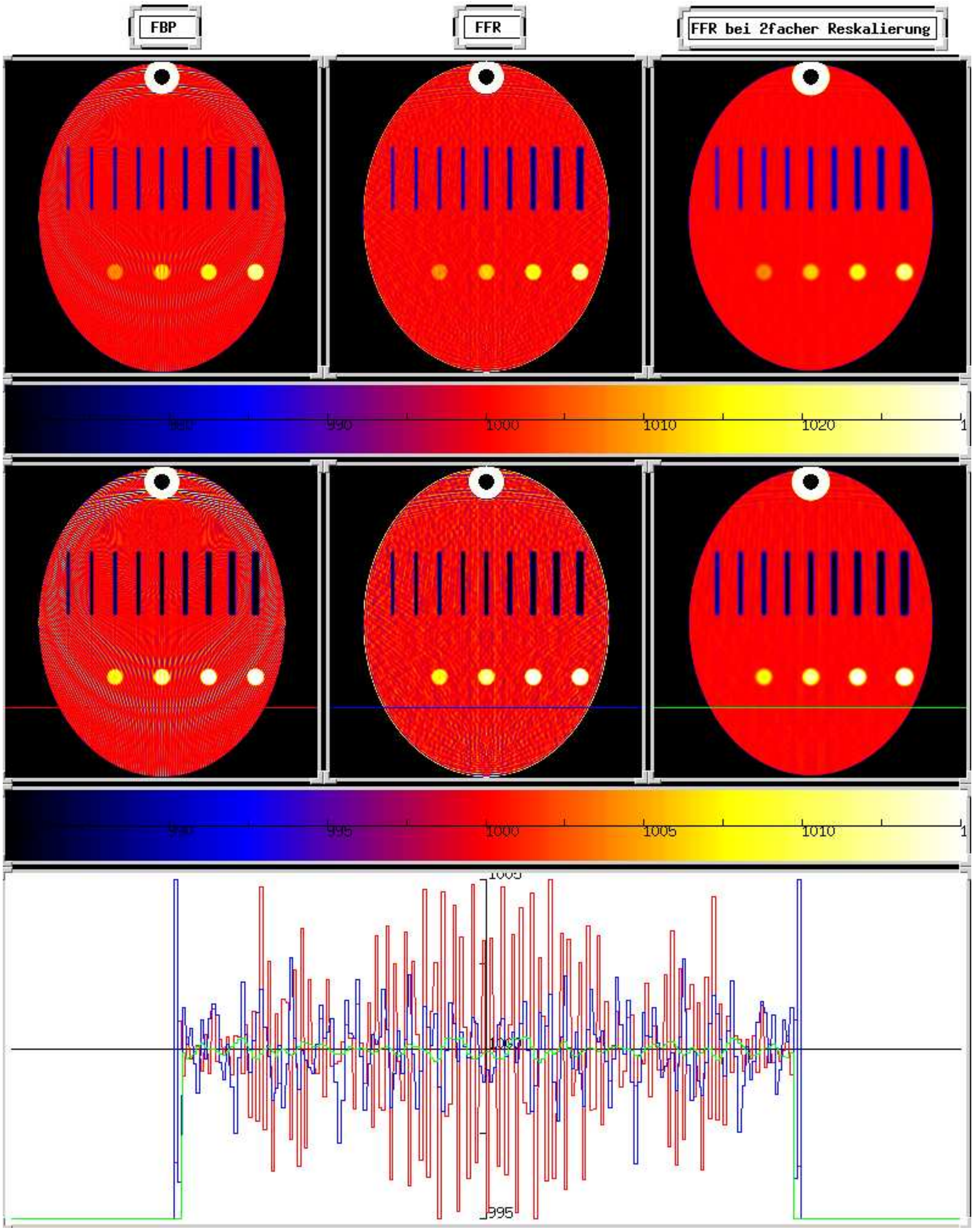
Parameter $p = 400$, $q = 128$, 256×256-Rekonstruktion	
Links:	Original
Mitte:	gefilterte Rückprojektion mit Shepp–Logan Filter
Rechts:	schelle Fourier-Rekonstruktion mit sinc^3 Filter

Abbildung 3.9: Rekonstruktion mit kreisförmigen Rand



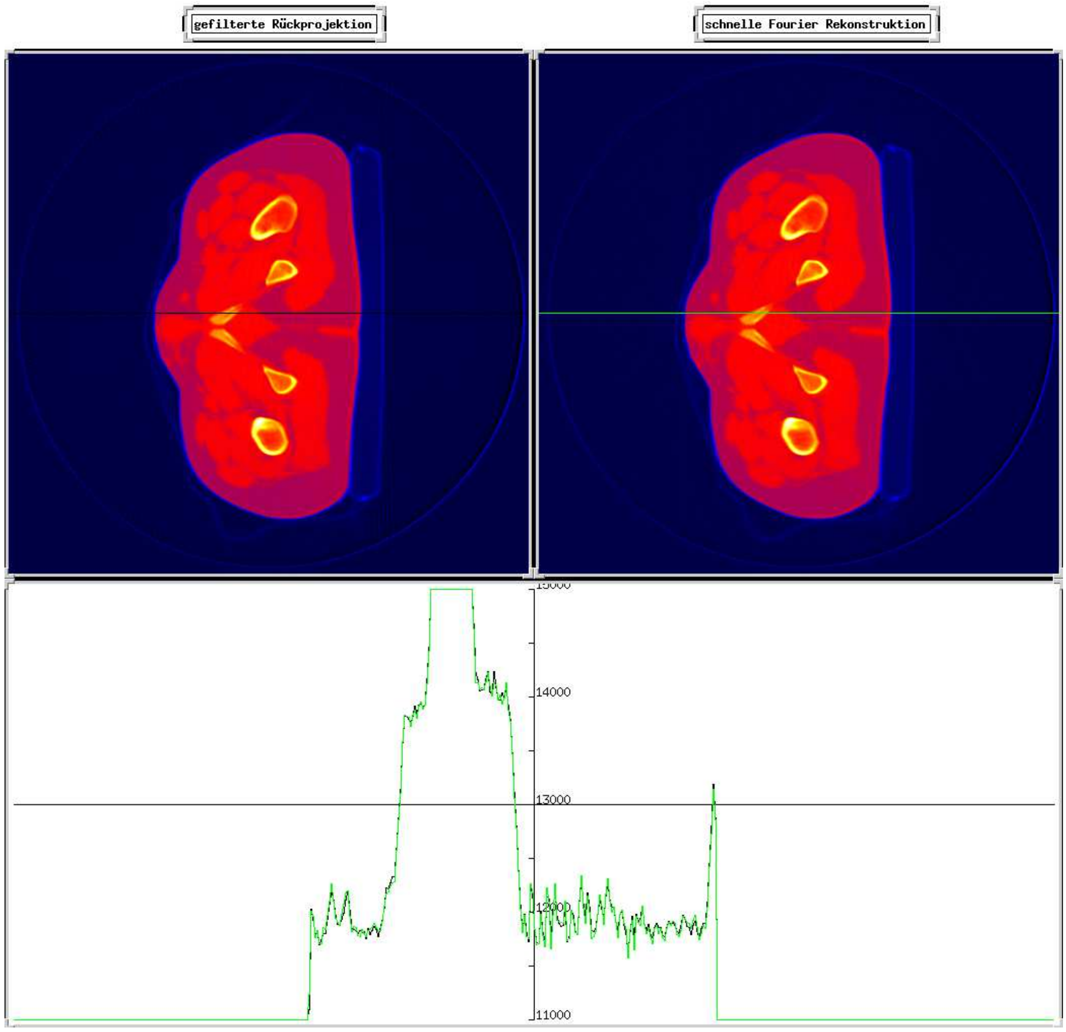
Parameter $p = 400$, $q = 128$, 256×256-Rekonstruktion

Abbildung 3.10: Phantom mit Rechtecken



Parameter $p = 400$, $q = 128$, 256×256-Rekonstruktion	
Links:	gefilterte Rückprojektion
Mitte:	schnelle Fourier-Rekonstruktion
Rechts:	FFR mit 2facher Reskalierung

Abbildung 3.11: Rekonstruktion mit kontrastreichem Punkt am oberen Rand



Parameter $p = 180$, $q = 256$, 512×512-Rekonstruktion	
Links:	gefilterte Rückprojektion mit Shepp-Logan Filter
Rechts:	schelle Fourier-Rekonstruktion (keine Reskalierung) mit sinc^3 Filter

Abbildung 3.12: Rekonstruktion tatsächlich gemessener Daten

Literaturverzeichnis

[1] G. Beylkin. On the fast Fourier transform of functions with singularities. *Appl. Comput. Harmon. Anal.*, 2(4):363–381, 1995.

[2] John P. Boyd. A fast algorithm for Chebyshev, Fourier, and sinc interpolation onto an irregular grid. *J. Comput. Phys.*, 103(2):243–257, 1992.

[3] John P. Boyd. Multipole expansions and pseudospectral cardinal functions: a new generalization of the fast Fourier transform. *J. Comput. Phys.*, 103(1):184–186, 1992.

[4] L. L. Campbell. Sampling theorem for the Fourier transform of a distribution with bounded support. *SIAM J. Appl. Math.*, 16:626–636, 1968.

[5] J. Carrier, L. Greengard, and V. Rokhlin. A fast adaptive multipole algorithm for particle simulations. *SIAM J. Sci. Statist. Comput.*, 9(4):669–686, 1988.

[6] H. Choi and David C. Munson, Jr. Direct-fourier reconstruction in tomography and synthetic aperture radar. *Int. J. of Imaging System and Technology*, 9(1):1–13, 1998.

[7] James W. Cooley and John W. Tukey. An algorithm for the machine calculation of complex Fourier series. *Math. Comp.*, 19:297–301, 1965.

[8] A. Dutt and V. Rokhlin. Fast Fourier transforms for nonequispaced data. *SIAM J. Sci. Comput.*, 14(6):1368–1393, 1993.

[9] A. Dutt and V. Rokhlin. Fast Fourier transforms for nonequispaced data. II. *Appl. Comput. Harmon. Anal.*, 2(1):85–100, 1995.

[10] O Forster. *Analysis I.* Vieweg, Braunschweig, 1983.

[11] Karsten Fourmont. schnelle fourier-transformation bei nicht äquidistanten daten. Dipolomarbeit, Universität Münster, 1997.

[12] Pierre Grangeat. Mathematical framework of cone beam 3D reconstruction via the first derivative of the Radon transform. In *Mathematical methods in tomography (Oberwolfach, 1990)*, pages 66–97. Springer, Berlin, 1991.

[13] Gabor T. Herman. *Image reconstruction from projections.* Academic Press Inc., New York, 1980. The fundamentals of computerized tomography, Computer Science and Applied Mathematics.

[14] A.J. Jerri. The shannon sampling theorem – its various extension and applications: A tutorial review. *Proc. IEEE*, 65(11):1565–1596, 1977.

[15] J.F. Kaiser. Digital filters. In F. Kuo and J.F. Kaiser, editors, *System Analysis by Digital Computer*, pages 218–285. Wiley, New York, 1966.

[16] Avinash C. Kak and Malcolm Slaney. *Principles of computerized tomographic imaging.* IEEE Press, New York, 1988.

[17] M. Kaveh and M. Soumekh. Computer-assisted diffraction tomography. In Henry Stark, editor, *Image recovery: theory and application*, pages 369–413. Academic Press Inc., Orlando, Fla., 1987.

[18] M. Kaveh, M. Soumekh, and J.F. Greenleaf. Signal processing for diffraction tomography. *IEEE Transactions on Sonics and Ultrasonics*, SU-31(4), 1984.

[19] Frank Natterer. Fourier reconstruction in tomography. *Numer. Math.*, 47(3):343–353, 1985.

[20] Frank Natterer. Efficient evaluation of oversampled functions. *J. Comput. Appl. Math.*, 14(3):303–309, 1986.

[21] Frank Natterer. *The mathematics of computerized tomography.* B. G. Teubner, Stuttgart, 1986.

[22] Henri J. Nussbaumer. *Fast Fourier transform and convolution algorithms*. Springer-Verlag, Berlin, 1981.

[23] F. Oberhettinger. *Tables of Fourier transforms and Fourier transforms of distributions*. Springer-Verlag, Berlin, 1990. Translated and revised from the German.

[24] J.D. O'Sullivan. A fast sinc function gridding algorithm for fourier inversion in computer tomography. *IEEE Trans. Med. Imag.*, MI-4(4):200–207, 1985.

[25] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical recipes in C*. Cambridge University Press, Cambridge, second edition, 1992. The art of scientific computing.

[26] S. Schaller, T. Flohr, and P. Steffen. An efficient fourier method for 3-d radon inversion in exact cone-beam ct reconstruction. *IEEE Trans. Med. Imag.*, MI-17(2):200–207, 1998.

[27] H. Schomberg and J. Timmer. The gridding method for image reconstruction by fourier transformation. *IEEE Trans. Med. Imag.*, MI-14(3):596–607, 1995.

[28] J. Schulte. Fourierrekonstruktion in der computer-tomographie. Dipolomarbeit, Universität Münster, 1994.

[29] David Slepian. Some comments on Fourier analysis, uncertainty and modeling. *SIAM Rev.*, 25(3):379–393, 1983.

[30] Eugene Sorets. Fast Fourier transforms of piecewise constant functions. *J. Comput. Phys.*, 116(2):369–379, 1995.

[31] Henry Stark. Sampling theorems in polar coordinates. *J. Opt. Soc. Amer.*, 69(11):1519–1525, 1979.

[32] Henry Stark, editor. *Image recovery: theory and application*. Academic Press Inc., Orlando, Fla., 1987.

[33] Henry Stark, John W. Woods, Indraneel Paul, and Rajesh Hingorani. Direct Fourier reconstruction in computer tomography. *IEEE Trans. Acoust. Speech Signal Process.*, 29(2):237–245, 1981.

[34] Robert S. Strichartz. *A guide to distribution theory and Fourier transforms.* CRC Press, Boca Raton, FL, 1994.

[35] Endre Süli and Antony Ware. A spectral method of characteristics for hyperbolic problems. *SIAM J. Numer. Anal.*, 28(2):423–445, 1991.

[36] Charles Van Loan. *Computational frameworks for the fast Fourier transform.* Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1992.

[37] James S. Walker. *Fourier analysis.* Oxford University Press, New York, 1988.

[38] Antony F. Ware. Fast approximate Fourier transforms for irregularly spaced data. *SIAM Rev.*, 40(4):838–856 (electronic), 1998.