

Bachelorarbeit an der Universität Bremen

**Implementierung und Erprobung von Compressed Sensing
für die Elektronenmikroskopie**

von Ilja Rukin

in der Gruppe von Andreas Rosenauer für Elektronenmikroskopie

Studiengang: Physik Bachelor of Science

Prüfer: Prof. Dr. Andreas Rosenauer

Zweitprüfer: Prof. Dr. Thomas Schmidt

vorgelegt am 25.11.2019

Diese Erklärungen sind in jedes Exemplar der Bachelor- bzw. Masterarbeit mit einzubinden.

Name: Ilya Rukin Matr.Nr.: 4348539

Urheberrechtliche Erklärung (§10 (11) Allgemeiner Teil der BPO/MPO v. 27.01.2010 (in der jeweils gültigen Fassung)

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine als die angegebenen Quellen und Hilfsmittel verwendet habe.

Alle Stellen, die ich wörtlich oder sinngemäß aus anderen Werken entnommen habe, habe ich unter Angabe der Quellen als solche kenntlich gemacht.

Die Bachelor-/Masterarbeit darf nach Abgabe nicht mehr verändert werden.

21.10.2019

Datum

Rukin

Unterschrift

Erklärung zur Veröffentlichung von Abschlussarbeiten

Die Abschlussarbeit wird zwei Jahre nach Studienabschluss dem Archiv der Universität Bremen zur dauerhaften Archivierung angeboten.

Archiviert werden:

- 1) Masterarbeiten mit lokalem oder regionalem Bezug sowie pro Studienfach und Studienjahr 10 % aller Abschlussarbeiten
- 2) Bachelorarbeiten des jeweils ersten und letzten Bachelorabschlusses pro Studienfach und Jahr.

Bitte auswählen und ankreuzen:

- Ich bin damit einverstanden, dass meine Abschlussarbeit im Universitätsarchiv für wissenschaftliche Zwecke von Dritten eingesehen werden darf.
- Ich bin damit einverstanden, dass meine Abschlussarbeit nach 30 Jahren (gem. §7 Abs. 2 BremArchivG) im Universitätsarchiv für wissenschaftliche Zwecke von Dritten eingesehen werden darf.
- Ich bin nicht damit einverstanden, dass meine Abschlussarbeit im Universitätsarchiv für wissenschaftliche Zwecke von Dritten eingesehen werden darf.

21.10.2019

Datum

Rukin

Unterschrift

Abstract

The topic of this bachelor thesis application of Compressed Sensing to electron microscopy. First the concept of Compressed Sensing is introduced and different formulations of this problem are shown. A presentation of some common and state of the art solvers follows. Next some transformations for sparse representation of the signal are discussed. Then a test set of signals and images is analyzed and reconstructed with different algorithms to compare their performance. The best algorithms are selected and applied to simulated images as well as simulation of electron intensity on the detector for HAADF scanning electron microscopy. A short overview over some reconstruction guarantees with ℓ_1 -regularization are presented in the appendix.

Inhaltsverzeichnis

1 Einleitung	1
2 Elektronenmikroskopie	2
2.1 Aufbau eines Elektronenmikroskops	2
2.2 Multislice-Simulation	3
3 Theorie zu Compressed Sensing	4
3.1 Einführung in Compressed Sensing	4
3.2 p-Norm	6
3.3 mathematische Formulierung von Compressive Sensing	7
4 Anwendung von Compressed Sensing in der Elektronenmikroskopie	9
5 Algorithmen	10
5.1 Abweichungsquadrat der Samples	10
5.2 Gradientenverfahren	10
5.3 Iterative Hard Thersholding	11
5.4 Iterative Soft Thresholding	11
5.5 Löser SPGL1 für Basis Pursuit Denoising	12
5.6 Löser FPC_AS für Lasso Pursuit	13
5.7 Löser Nesta für Basis Pusuit Denoising	13
5.8 Löser l1eq pd und linprog für Basis Pursuit	14
5.9 Löser lsqlin für Lasso Pursuit	15
5.10 Löser l1qc logbarrier für Basis Pursuit Denoising	15
5.11 Orthogonal Matching Pursuit	15
5.12 Interpolation	16
6 Vergleich unterschiedlicher diskreter Transformationen	17
6.1 Globale Transformationen	17
6.2 Lokale Transformationen (Wavelets)	19
6.3 zweidimensionale Transformation für Löser	21
6.4 Kohärenz zwischen Mess- und Darstellungsmatrix	21
7 Einführung in die Auswertung und Vergleich der Darstellungsbasen	23
7.1 Auswahl der Basis für Bilder	23
7.2 Rekonstruktion in unterschiedlichen Basen	26
8 Rekonstruktion	28
8.1 Rekonstruktion von Signalen (1D)	28
8.2 Rekonstruktion von Bildern (2D)	29
8.3 Rekonstruktion bei unterschiedlicher Anzahl an Samples	31
8.4 Untersuchung zur Skalierung der Laufzeit der Algorithmen	32
9 Rekonstruktion von HAADF-Simulationen	33
9.1 Rekonstruktion von simulierten HAADF-Aufnahmen mit und ohne Rauschen	33
9.2 Rekonstruktion einer Multislice-Simulation und Vergleich der mittleren Intensitäten für unterschiedliche Probendicken	38
10 Fazit	43

1 Einleitung

STEM ist eine auszeichnende Methode zur Untersuchung von mikroskopischen Proben bis hin zu atomaren Strukturen. Dazu wird die Probe mit einem fokussierten Elektronenstrahl gerastert und die Intensität des transmittierten Elektronenstrahls gemessen. Die Elektronen haben oft eine Energie zwischen 30 keV und 1 MeV, so dass dünne und empfindliche Proben bei stark vergrößerten Aufnahmen beschädigt und durch Zersetzung organischer Stoffe kontaminiert werden. Aufnahme der Probe mit einer geringeren Intensität liefert dagegen ein verrausches Bild. Mit Methoden des Compressed Sensing ließe sich ein Bild aus einer unvollständigen und verrauschten Aufnahme wiederherstellen, so dass die Probe einer geringeren Strahlendosis ausgesetzt wäre. Ziel dieser Arbeit ist es an einer simulierten HAADF-Rasterelektronenmikroskop-Aufnahme zu untersuchen, ob die Rekonstruktion mit Compressed Sensing bessere Ergebnisse, als eine verrauschte Aufnahme aufgenommen mit geringerer Intensität, liefert.

Des weiteren werden zu HAADF-Rastertransmissionselektronenmikroskopie-Aufnahmen Multislice-Simulationen (mit Frozen-Lattice) der Intensität am HAADF-Detektor für unterschiedliche Probendicken untersucht. Diese Simulation dient dazu die Dicke der Probe auf einer HAADF-Aufnahme zu bestimmen. Diese Simulation genau durchzuführen dauert gewöhnlich sehr lange, so dass in dieser Arbeit die Genauigkeit einer unvollständigen, mit Compressed Sensing vervollständigten Simulation mit einer vollständigen Simulation verglichen wird. Insbesondere wird die mittlere Intensität der simulierten HAADF-Aufnahme einer Probe bei verschiedenen Probendicken für eine unvollständige Simulation mit einer mit Compressed Sensing rekonstruierten HAADF-Aufnahme verglichen. Daraus wird die Anwendbarkeit von Compressed Sensing für die Verkürzung der Simulationszeit überprüft.

In Kapitel 2 wird eine kurze Einführung in die Elektronenmikroskopie und die Multislice-Simulation gegeben. In Kapitel 3 folgt eine Vorstellung von Compressed Sensing. Dabei wird Compressed Sensing in mehrere mathematische Probleme formuliert. Anschließend werden in Kapitel 4 aktuelle Anwendungen von Compressed Sensing in der Elektronenmikroskopie präsentiert. Kapitel 5 widmet sich den in dieser Arbeit verwendeten Algorithmen zur Lösung des Problems von Compressed Sensing. In Kapitel 6 werden alle verwendeten Transformationen und deren Vor- und Nachteile beschrieben. Außerdem werden erste Anforderungen für eine gute Rekonstruktion von Signalen behandelt. Anschließend wird in Kapitel 7 die Darstellung und Rekonstruktion von Bildern mit unterschiedlichen Transformationen untersucht. Dabei werden die passenden Transformationen zur Darstellung von Bildern für Compressed Sensing ermittelt. Darüber hinaus wird in Kapitel 8 eine Reihe von sehr unterschiedlichen Signalen und Bildern mit den Algorithmen rekonstruiert und dabei die Genauigkeit der Algorithmen untersucht. In Kapitel 9 wird die Rekonstruktion von simulierten HAADF-Aufnahmen behandelt. Daraus werden die besten Algorithmen für die Anwendung in der Elektronenmikroskopie ermittelt. Die Arbeit wird mit einer Zusammenfassung abgeschlossen.

2 Elektronenmikroskopie

2.1 Aufbau eines Elektronenmikroskops

In diesem Kapitel wird das grundlegende Funktionsprinzip von Elektronenmikroskopen vermittelt. Die Auflösung von Lichtmikroskopen ist durch das Abbe-Limit begrenzt. Zum Beispiel ist die Wellenlänge von Elektronen beschleunigt mit 200 kV mit 2 pm deutlich kleiner als die des sichtbaren Lichts (300 – 800 nm). Somit fallen Beugungseffekte geringer aus und es kann eine bessere Auflösung erzielt werden. Darauf aufbauend wurde eine Reihe von verschiedenen Elektronenmikroskopen (EM) entwickelt, die zur Auflösung atomaren Strukturen einen Elektronenstrahl verwenden. Beim EM werden die Elektronen mit magnetischen Feldern abgelenkt, wobei sie durch die Lorentzkraft nicht nur fokussiert werden, sondern auch eine zusätzliche Drehung erfahren und insgesamt eine Spiralbahn beschreiben. Die Spulen, die diese Felder erzeugen, sind fest montiert, so dass der Brennpunkt hier durch Regelung des elektrischen Stromes verändert wird.

EM nutzen eine Elektronenquelle, welche durch thermische Emission Elektronen emittiert. Es kann durch ein zusätzliches elektrisches Feld die Austrittsarbeit der Elektronen gesenkt werden, dann spricht man von einer Feldemissionskathode (FEG). Im weiteren Schritt werden die Elektronen mit einer Anode beschleunigt. Anschließend wird der Elektronenstrahl mit einer Reihe von Kondensorlinsen und Blenden gebündelt und parallelisiert.

Im **Transmissionselektronenmikroskop (TEM)** wird der Elektronenstrahl großflächig und parallel auf die Probe gerichtet und die transmittierten Elektronen mit der Objektivlinse gesammelt. Dabei wird die Probe mit der Objektivlinse, weiteren Zwischenlinsen und einer Projektionslinse auf dem Schirm vergrößert abgebildet und man erhält ein Realbild der Probe. Die Objektivlinse dient der Hauptvergrößerung des Bildes und die Zwischenlinsen vergrößern es weiter.

Ein weiteres Verfahren ist es den Elektronenstrahl mit dem Kondensorlinsensystem auf einen kleinen Punkt zu fokussieren. Dabei ist die Breite des Elektronenpunkts (der sogenannten „Probe“) entscheidend für die maximal erreichbare Auflösung des Mikroskops. Beim **Rastertransmissionselektronenmikroskop (STEM)** wird eine Probe mit dem fokussierten Elektronenstrahl gerastert und die Intensität des transmittierten Elektronenstrahls für die einzelnen Positionen des Elektronenstrahls auf der Probe gemessen. Es gibt eine Reihe verschiedener Detektoren für unterschiedliche Ablenkungswinkel der Elektronen. In dieser Arbeit werden Aufnahmen mit dem „high angle annular dark field detector“, kurz HAADF-Detektor, verwendet. Dieser Detektor registriert verhältnismäßig stark abgelenkte Elektronen. Beispielsweise nimmt er beim Elektronenmikroskop Titan 30-800 die Intensität der Elektronen in einem Winkel von 35 mrad bis 255 mrad auf [1]. Die Messung dieser Elektronen hat den Vorteil, dass die Anzahl der gestreuten Elektronen mit der Ordnungszahl der Atome stark zunimmt, so dass unterschiedliche Atome durch einen hohen Kontrast voneinander unterschieden werden können.

Ein EM, mit dem Rastertransmissionselektronenmikroskopie-Aufnahmen möglich sind, ist in Abbildung 1 dargestellt. Die Auflösung moderner Geräte kann im TEM- und STEM-Betrieb mit Aberrationskorrektur bis zu 50 pm betragen. Dabei werden mit einer Spule, welche einen magnetischen Multipol erzeugt, die Aberrationen der Linsen korrigiert. Wird eine Probe mit starker Vergrößerung aufgenommen, so wird ein kleiner Bereich der Probe mit hoher Elektronenintensität bestrahlt. Dadurch können empfindliche Proben beschädigt werden. Außerdem werden durch diese Intensität organische Verunreinigungen auf der Probe zersetzt, womit die Probe mit diesen Zersetzungprodukten verunreinigt wird [2]. Um diesen Umständen entgegen zu wirken kann die Elektronenintensität beim TEM oder STEM gesenkt werden. Dadurch wird die Aufnahmen stärker verrauscht. Eine weitere Methode, die Elektronenintensität zu senken, ist es eine unvollständige Aufnahme zu machen und den Rest mit Compressed Sensing zu rekonstruieren. Beim TEM wird die Probe auf einmal vollständig aufgenommen, so dass Compressed Sensing hier nicht angewendet werden kann. In einem STEM kann der Elektronenstrahl mit einem Strahlauflaster blockiert werden, so dass die Probe nur an einigen Positionen aufgenommen wird [3]. Aus dieser Information können mit Compressed Sensing die nicht aufgenommenen Datenpunkte rekonstruiert werden. In dieser Arbeit wird speziell die Anwendung von Compressed Sensing für HAADF-Rastertransmissionselektronenmikroskopie-Aufnahmen untersucht.

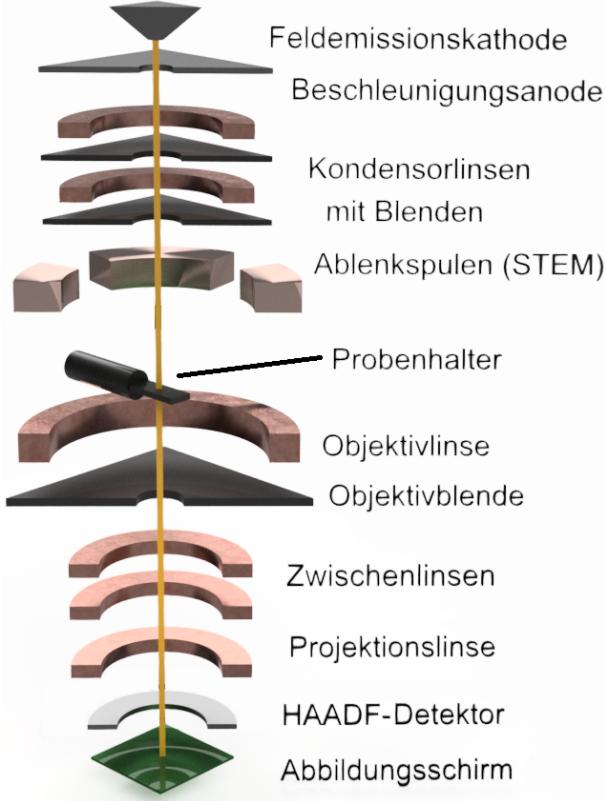


Abb. 1: Prinzipieller Aufbau eines Elektronenmikroskops.

2.2 Multislice-Simulation

Zur Simulation einer elektromikroskopischen Aufnahme einer Probe muss die Dirac-Gleichung für die einfallende Elektronenwelle gelöst werden. Dies ist analytisch ohne weiteres nicht möglich. Eine Alternative ist das sogenannte Multislice-Verfahren. Dabei wird eine Probe in Slices von der Dicke einer Elementarzelle (des Kristallgitters) zerlegt und dann die Wellenfunktion durch jede Slice separat propagiert. Es wird die Welle in der ersten Hälfte der Einheitszelle frei fortgeführt, in dem eine Faltung (implementiert als Multiplikation im Fourieraum) mit dem Fresnelpropagator durchgeführt wird. Dann wird die Wellenfunktion mit dem Phase-Grating multipliziert um das Potential des Slices zu berücksichtigen. Anschließend wird die Welle noch in der zweiten Hälfte der Einheitszelle frei propagiert. Dieses Schema wird wiederholt, bis die Wellenfunktion durch die gesamte Probe propagiert wurde. Anschließend gibt das Betragsquadrat der Wellenfunktion die Intensität des Realbildes bzw. das Betragsquadrat der Fouriertransformation der Wellenfunktion die Intensität des Beugungsbildes wieder. Die Intensität kann auch nach Propagation durch jedes Slice berechnet werden, so dass die Intensität für unterschiedliche Probendicken bekannt wird. Um die thermische Bewegung der Atome zu simulieren kann eine Frozen-Lattice-Simulation durchgeführt werden. Dabei werden viele Multislice-Simulationen mit zufällig ausgelenkten Atomen entsprechend ihrer thermischen Energie durchgeführt und danach die Intensität über diese Simulationen gemittelt.

In dieser Arbeit werden Multislice-Simulationen von HAADF-STEM-Aufnahmen zum Testen von Compressed Sensing verwendet. Tatsächliche Aufnahmen enthalten Rauschen wohingegen eine Simulation ein exaktes Bild liefert. Gerade ein exaktes Ergebnis einer Simulation eignet sich als Vergleichsbild, um die Genauigkeit der Rekonstruktion mit Compressed Sensing zu beurteilen. Außerdem wird die Anwendung von Compressed Sensing für die Vervollständigung einer Multiclice-Simulation überprüft, wozu Teile einer solchen Simulation rekonstruiert werden und mit der vollständigen Simulation verglichen werden. Die hier verwendete Simulationen wurden mit der Software STEMSSIM erzeugt[4], welche auf der Theorie von A. Weickenmeier und H. Kohl aufbaut [5].

3 Theorie zu Compressed Sensing

3.1 Einführung in Compressed Sensing

Misst man eine beliebige Größe, so werden diskrete Datenpunkte erfasst, auch wenn das zugrundeliegende Signal oft kontinuierlich ist. Bei einer STEM-Aufnahme Beispielsweise wird eine Probe an diskreten Stellen gerastert, so dass das Bild der Probe diskrete Datenpunkte (Pixel) enthält.

Beim Compressed Sensing (CS) wird ein diskretes Signal \mathbf{f} aus einer unvollständigen Messung dessen rekonstruiert. Wir definieren ein Signal \mathbf{f} , welches exakt und ohne jegliches Rauschen ist. Dies ist das grundwahre Signal, dass im Idealfall durch die Rekonstruktion wiederhergestellt wird. Die uns bekannte Information über das Signals bekommen wir durch Messung dessen. Die Messung kann als Multiplikation des Signals, formuliert als Vektor \mathbf{f} , mit einer Matrix Φ geschrieben werden:

$$\mathbf{y}_n = \sum_t \Phi_{n,t} \mathbf{f}_t + \mathbf{e}_n \quad (1)$$

Bei der Messung kommen noch Ungenauigkeiten hinzu, welche durch hinzu addiertes Rauschen \mathbf{e} berücksichtigt werden. Das Rauschen ist im Voraus unbekannt. Das Ergebnis der Messung sind Datenpunkte \mathbf{y} , welche im weiteren als Samples bezeichnet werden. Bei unvollständiger Messung des Signals ist die Messmatrix Φ rechteckig und die Zahl der Samples y kleiner als die Zahl der Werte im Signal \mathbf{f} . Somit kann Φ nicht invertiert werden, um das Signal \mathbf{f} aus den Samples zu berechnen. Das Rauschen \mathbf{e} erschwert diese unlösbare Aufgabe noch weiter. Beim CS geht man dieses Problem an, in dem man das Signal \mathbf{f} mit Koeffizienten \mathbf{x} in einer Darstellungsbasis Ψ darstellt. In Matrixschreibweise werden die Koeffizienten \mathbf{x} mit der Matrix $\bar{\Psi}$ multipliziert, um das Signal \mathbf{y} zu erhalten:

$$\mathbf{f}_t = \sum_k \bar{\Psi}_{t,k} \mathbf{x}_k \quad (2)$$

Dabei ist $\bar{\Psi}_{t,k}$ die Rücktransformationsmatrix. Die Hintransformation vom Signal zu den Koeffizienten wird mit der Matrix Ψ durchgeführt:

$$\mathbf{x}_k = \sum_t \Psi_{k,t} \mathbf{f}_t \quad (3)$$

Die beiden Matrixmultiplikationen, um Koeffizienten in Samples umzurechnen, können mit einer Gesamtmatrix \mathbf{A} zusammengefasst werden:

$$\mathbf{y} = \Phi \bar{\Psi} \mathbf{x} + \mathbf{e} = \mathbf{Ax} + \mathbf{e} \quad (4)$$

Diese Gleichung wird zu

$$\mathbf{y} \approx \mathbf{Ax} \quad (5)$$

vereinfacht, weil das Rauschen unbekannt ist und nicht direkt berücksichtigt werden kann. Es diente hier nur der Veranschaulichung, dass \mathbf{f} das grundwahre Signal ist und die Samples eine gewisse Ungenauigkeit enthalten. Ziel ist es die uns bekannten Samples \mathbf{y} mit möglichst wenigen Koeffizienten darzustellen. Diese Bedingung wird im weiteren als Sparsity und die dünn besetzten Koeffizienten der Lösung als sparse bezeichnet. Dadurch bekommen wir eine eindeutige Lösung der Gleichung 5. Rauschen wird bei der Rekonstruktion berücksichtigt, in dem die Samples nicht exakt, sondern mit einer geringen Ungenauigkeit angepasst werden. Am Anfang werden die Koeffizienten mit einem Algorithmus bestimmt, um anschließend aus diesen durch eine Transformation mit der Darstellungsmatrix das gesuchte Signal wiederherzustellen. Für eine Formulierung von Compressed Sensing kann folgende Publikation von Emmanuel J. Candès und Michael B. Wakin nachgeschlagen werden [6].

Für eine STEM-Aufnahme ist das Signal ein exaktes und unverrauscht Bild der Probe. Weil es so ein Bild in der Praxis nicht gibt, wird das Ergebnis einer Simulation dafür verwendet. Nimmt man die Probe vollständig auf, in dem die Probe mit dem Elektronenstrahl vollständig gerastert wird, so ist die Messmatrix eine Einheitsmatrix und das wahre Signal kann mit der invertierten Messmatrix wiederhergestellt werden. Zudem ist die inverse der Einheitsmatrix eine Einheitsmatrix, weil das Signal und die Samples in unserem Fall ununterscheidbar sind. (Dies gilt für den einfachen Fall eines eindimensionalen Signals. Bilder erfordern vierdimensionale Matrizen oder einen Algorithmus zur Transformation.)

Bei anderen Anwendungen kann die Messmatrix völlig anders aufgebaut sein, so dass sich Signale und Samples deutlich unterscheiden. Zum Beispiel wird bei der bildgebenden Kernspintomographie die Präzession der Kernspins einer Probe in einem inhomogenen Magnetfeld gemessen. Dabei hängt die Frequenz der Präzession von der Stärke des Magnetfelds ab. Die gemessene Präzession über die gesamte Probe kann dann mit einer Fouriertransformation in ihre Frequenzen zerlegt werden, um dann die Amplituden der Präzession unterschiedlicher Frequenz anhand der Frequenzen den zugehörigen Magnetfeldstärken zuzuordnen, welche wiederum Orten in der Probe entsprechen. Die Amplitude der Präzession in einem Bereich der Probe entspricht der Kernspindichte in diesem Bereich. Die Probe wird somit im Frequenzraum aufgenommen, so dass die Messmatrix eine Fouriermatrix ist. Ein weiteres Beispiel sind experimentelle Ein-Pixel-Kameras, welche aus einem einzigen Lichtsensor und einer veränderbaren Blende bestehen. Es werden bei der Aufnahme eines Bildes mehrere Intensitäten mit dem Sensor für unterschiedliche Blenden gemessen. Die Messmatrix besteht in diesem Fall aus Nullen und Einsen entsprechend ob die Blende an der jeweiligen Position Licht durchlässt oder nicht.

Jetzt soll die Probe im STEM nicht vollständig aufgenommen werden, um die Elektronenintensität, dem ein Bereich der Probe ausgesetzt wird, zu reduzieren. Dazu wird die Probe an einer begrenzten Anzahl an Orten mit dem Elektronenstrahl gemessen. Jede solche Messung entspricht der Multiplikation des Signals mit der zugehörigen Zeile einer Einheitsmatrix. Die Messmatrix wird aus solchen Zeilen zusammengesetzt, wobei sie rechteckig und nicht invertierbar ist. Durch Darstellung des Signals in einer Basis und der Lösung der Gleichung 5 mit einer sparsen Besetzung der Koeffizienten kann das Signal eindeutig wiederhergestellt werden. Eindeutigkeit bedeutet allerdings nicht, dass auch das tatsächlich zugrundeliegende Signal rekonstruiert wurde. Dafür muss das Signal in der Darstellungsbasis mit sehr wenigen Koeffizienten darstellbar sein. Eine sparse Besetzung der Koeffizienten ist in guter Näherung nicht abwegig anzunehmen. Dieses Prinzip wird auch bei der Datenkompression von Audio- und Bilddateien (im MP3 bzw. JPEG-Format) verwendet. Es beruht darauf, dass viele Signale in einer anderen Basis näherungsweise mit wenigen Koeffizienten dargestellt werden können. Demnach sollte die Darstellungsmatrix passend zum Signal gewählt werden, so dass das Signal möglichst sparse in dieser Basis dargestellt werden kann. Dazu wurden die diskrete Kosinustransformation (DCT), diskrete Fouriertransformation (DFT), eine reduzierte Fouriertransformation (CDFT), die Haar-Wavelets (HAAR) und die Daubechies-4-Wavelets (DB4) als Darstellungsmatrix in dieser Arbeit verwendet. In Abbildung 2 ist ein Signal und der Betrag dessen Koeffizienten in der Fourierbasis dargestellt. Das Signal hat eine breite Ausdehnung im Ortsraum, kann allerdings mit wenigen Koeffizienten in der Fourierbasis dargestellt werden. Sogar das detailreiche Bild Huhn enthält in der Fourierbasis wenige große Koeffizienten (siehe Abbildung 3), so dass es in erster Näherung mit wenigen Koeffizienten dargestellt werden kann. Das Bild Huhn selbst ist in Abbildung 4 zu sehen. Es ist somit nicht ganz abwegig anzunehmen, dass ein Signal, zumindest näherungsweise, sparse in einer anderen Basis darstellbar ist.

Zur Veranschaulichung von CS folgt hier ein Beispiel. Es wurden aus dem Graustufenbild Huhn (Abbildung 4) 10 Prozent Samples (Abbildung 5) zufällig gemessen und mit CS mit dem Löser SPGL1 (Abbildung 6) rekonstruiert. Die Angabe der Samples in Prozent entspricht dabei dem Verhältnis der Zahl der Samples verglichen mit der Zahl der Pixel im Originalbild. Auf der Rekonstruktion ist ein Huhn erkennbar. Es ist allerdings auch viel Rauschen vorhanden, weil das Bild Huhn nicht exakt mit wenigen Koeffizienten darstellbar ist.

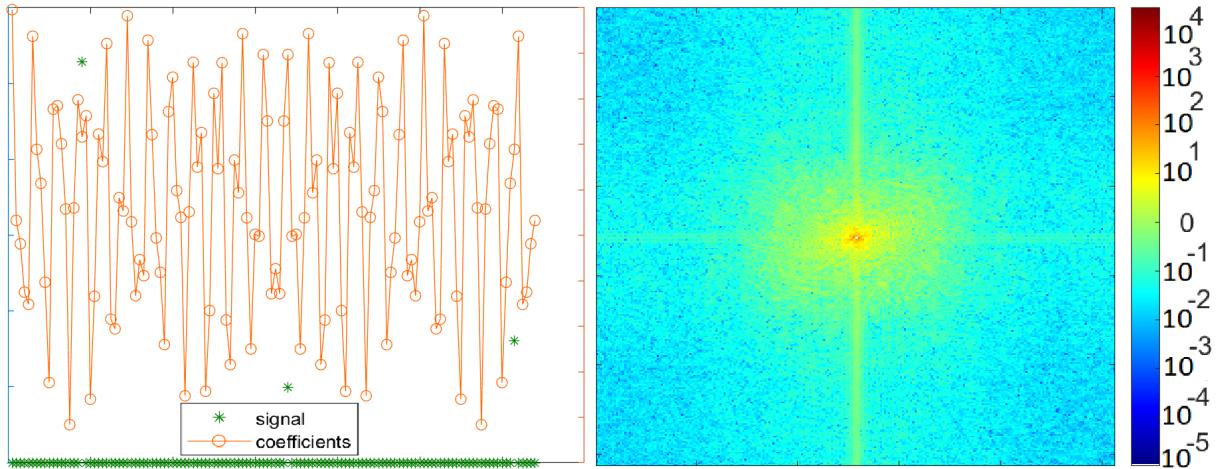


Abb. 2: Signal und Betrag der Fourierkoeffizienten eines Signals: einige Signale können mit wenigen Koeffizienten dargestellt werden. In diesem Beispiel sind es exakt drei Koeffizienten.

Abb. 3: Betrag der Fouriertransformation vom Bild Huhn: viele Koeffizienten sind sehr klein.

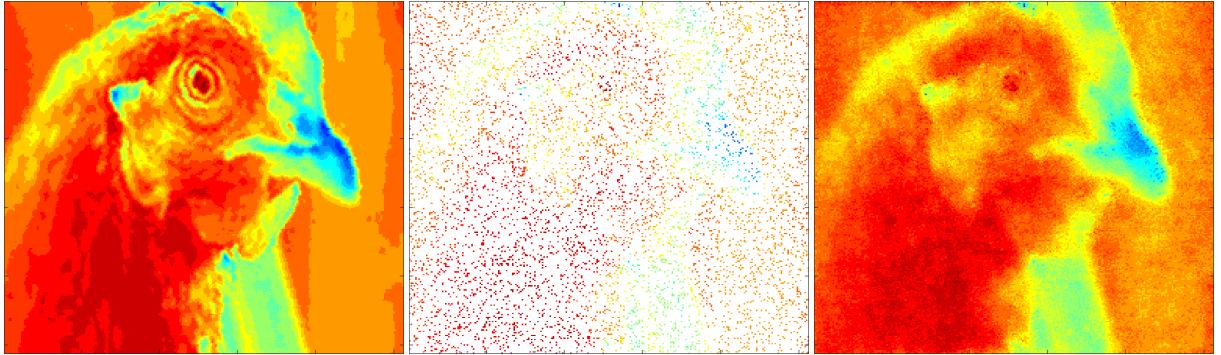


Abb. 4: Beispielbild Huhn: ein Graustufenbild mit vielen Details.

Abb. 5: Beispielsweise 10% Samples aus dem Bild Huhn entnommen.

Abb. 6: Mit CS aus den Samples rekonstruiertes Bild: Huhn erkennbar, aber viel Rauschen vorhanden.

3.2 p-Norm

Bevor man das Problem des CS mathematisch richtig formuliert muss quantifiziert werden, was eine sparse Besetzung von Koeffizienten bedeutet. Dazu definieren wir die p-Norm eines Vektors \mathbf{x} als

$$\|\mathbf{x}\|_p = \left(\sum_i (\mathbf{x}_i)^p \right)^{\frac{1}{p}} . \quad (6)$$

Für CS sind die Normen ℓ_0 , ℓ_1 , ℓ_2 und ℓ_∞ wichtig. Die ℓ_0 -Norm ist eine Quasinorm, weil sie nicht alle Bedingungen für eine Norm erfüllt. Für $p \rightarrow 0$ geht Gleichung 6 gegen unendlich. Für den Sonderfall der ℓ_0 -Norm lässt man deswegen die Potenz $1/p$ in Gleichung 6 weg, so dass $\|\mathbf{x}\|_0$ der Anzahl der Elemente ungleich Null im Vektor \mathbf{x} entspricht.

Daraus ausgehend kann die dünne Besetzung eines Vektors durch die ℓ_0 -Norm quantifiziert werden. Ein Vektor mit kleiner gleich s Elementen ungleich Null wird s -sparse bezeichnet.

Die drei anderen wichtigen Normen sind in ausgeschriebener und vereinfachter Form:

$$||\mathbf{x}||_1 = \sum_i |\mathbf{x}_i| \quad (7)$$

$$||\mathbf{x}||_2 = \sqrt{\sum_i (\mathbf{x}_i)^2} \quad (8)$$

$$||\mathbf{x}||_\infty = \max_i(|\mathbf{x}_i|) \quad (9)$$

3.3 mathematische Formulierung von Compressive Sensing

Als nächstes wird das Optimierungsprobleme fürs CS formuliert. Zur Rekonstruktion des Signals \mathbf{f} müssen die Koeffizienten \mathbf{x} gefunden werden, die Gleichung $\mathbf{Ax} = \mathbf{y}$ erfüllen und zugleich möglichst sparse besetzt sind. Das ist das **Ausgangsproblem**:

$$\min ||\mathbf{x}||_0 \quad \text{mit} \quad \mathbf{Ax} = \mathbf{y} \quad (10)$$

Weil die ℓ_0 -Norm eines Vektors unstetig ist, lässt sich dieses Problem nicht effizient lösen. Es müssten dazu alle Kombinationen, unterschiedliche Koeffizienten gleich und ungleich Null zu setzen, durchgerechnet werden und dann die Lösung des Problems mit der spärtesten Besetzung gewählt werden. Die ℓ_0 -Norm kann durch die ℓ_1 -Norm angenähert werden und hat in speziellen Fällen sogar ein identisches Ergebnis. Das resultierende Problem wird als **Basis Pursuit** bezeichnet:

$$\min ||\mathbf{x}||_1 \quad \text{mit} \quad \mathbf{Ax} = \mathbf{y} \quad (11)$$

Die ℓ_1 -Norm eines Vektors hat den Vorteil, dass sie stetig (aber nicht differenzierbar) ist. Die Gleichheit des Ausgangsproblems zum Basis Pursuit kann geometrisch für den einfachen Fall von zwei Koeffizienten \mathbf{x} und einer linearen Gleichung $\mathbf{Ax} = \mathbf{y}$ gezeigt werden. Für diesen Fall liegen die Lösungen auf einer Geraden. Für eine eindeutige Lösung werden die Lösungen durch minimale p -Normen der Lösung eingegrenzt. In Abbildung 7 ist ein Beispiel dafür gezeigt. Die Lösungen mit kleinster ℓ_0 und ℓ_1 -Norm sind sparse und identisch, wohingegen die Lösung mit der minimalen ℓ_2 -Norm dies nicht ist.

Die ℓ_0 und ℓ_1 -Normen sind für das Problem äquivalent, wenn die (grüne) geometrische Figur der minimalen ℓ_1 -Norm die Lösung mit einer Ecke an einer Koordinatenachse (dann ist nur ein Koeffizient ungleich Null) oder mit einer Kante an einer Hyperebene aufgespannt durch möglichst wenige Koordinatenachsen (dann sind mehrere Koeffizienten ungleich Null) trifft.

Die Äquivalenz der ℓ_0 zur ℓ_1 -Norm der Koeffizienten kann für eine Matrix \mathbf{A} mit der „Null Space Property“ oder der „Restricted Isometry Property“ bewiesen werden. Eine weitere Möglichkeit ist es die Eindeutigkeit durch eine geringe Kohärenz der Gesamtmatrix \mathbf{A} zu zeigen (genaueres dazu im Anhang zur „Äquivalenz der Rekonstruktion mit ℓ_0 und ℓ_1 -Norm“).

Weitere Erweiterungen des Basis Pursuit berücksichtigen Ungenauigkeiten der Samples aufgrund von Rauschen. Eine Variante ist das **Basis Pursuit Denoising**:

$$\min ||\mathbf{x}||_1 \quad \text{mit} \quad ||\mathbf{Ax} - \mathbf{y}||_2 \leq \sigma \quad (12)$$

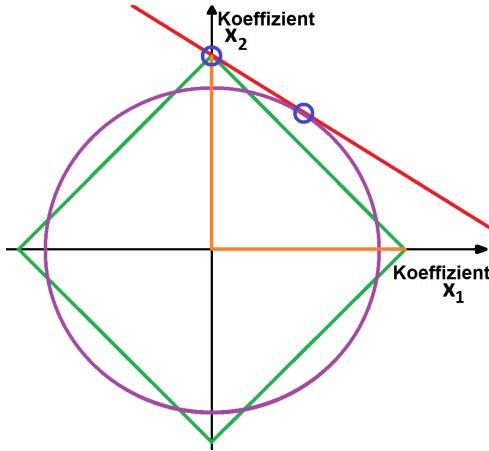


Abb. 7: Beispiel für ein Gleichungssystem $\mathbf{Ax} = \mathbf{y}$ mit zwei Koeffizienten x_1 und x_2 und einer zufälligen linearen Gleichung. Mögliche Lösungen dieses Gleichungssystems für die Koeffizienten werden mit der roten Geraden dargestellt. Die Lösungen werden durch eine minimale ℓ_0 -Norm (gelb), ℓ_1 -Norm (grün) und ℓ_2 -Norm (violett) eingegrenzt, so dass eindeutige Lösungen (blaue Kreise) erhalten werden. In diesem Fall ist die Lösung mit minimaler ℓ_0 -Norm und ℓ_1 -Norm identisch und die Lösung mit minimaler ℓ_2 -Norm unterscheidet sich davon.

Dieses Verfahren passt die Samples mit einer Ungenauigkeit, vorgegeben durch den Parameter σ , an. Somit fällt die ℓ_1 -Norm kleiner aus, als bei exakter Anpassung. Wenn ein sparses Bild mit leichtem Rauschen aufgenommen wurde und dieses nur ungenau angepasst wird, so werden mit diesem Verfahren nur die signifikanten Koeffizienten gefunden und leichtes Rauschen wird nicht mit angepasst. Eine weitere Variante ist das **Lasso Pursuit**:

$$\min ||\mathbf{Ax} - \mathbf{y}||_2 \quad \text{mit} \quad ||\mathbf{x}||_1 \leq \tau \quad (13)$$

Dieses Verfahren verwendet den Parameter τ um zwischen Anpassungsgenauigkeit der Samples und Sparsity der Koeffizienten abzuwegen. Dieser Parameter kann im Voraus schlecht abgeschätzt werden. Verglichen dazu ist die Nebenbedingung $||\mathbf{Ax} - \mathbf{y}||_2 \leq \sigma$ des Basis Pursuit Denoising einfacher im Voraus anzugeben, aber erheblich schwieriger in einen Algorithmus zu implementieren. Im Allgemeinen muss man bei Berücksichtigung von Rauschen ein Kompromiss zwischen Anpassungsgenauigkeit und Sparsity treffen. Ist die Sparsity zu gering, so können die vorgegebenen Punkte mit einer kleinen Anzahl an Koeffizienten nicht genau angepasst werden. Ist die Anpassungsgenauigkeit zu hoch, so werden die Samples durch viele Koeffizienten überangepasst, so dass die Rekonstruktion auch hier erheblich vom Bild abweicht. Das liegt daran, dass das Bild nicht mehr sparse repräsentiert wird, das Rauschen verstärkt mit angepasst wird und die Lösung gegebenenfalls nicht mehr eindeutig ist.

4 Anwendung von Compressed Sensing in der Elektronenmikroskopie

Mit CS können unvollständige Aufnahmen von Proben mit einem STEM rekonstruiert werden, so dass die Probe bei der Aufnahme einer kleineren Strahlungsdosis ausgesetzt wird. Die rekonstruierten Aufnahmen sind allerdings mit abnehmender Zahl an Samples stärker verzerrt.[7] Dadurch wird gleichzeitig die Aufnahmezeit verringert, so dass mehr Bilder einer Probe in der gleichen Zeit erstellt werden können. Dies wurde für die Ptychographie verwendet, weil hier für jede Elektronenstrahlposition ein zweidimensionales Beugungsbild aufgenommen werden muss.[8] Aus den aufgenommenen Beugungsbildern wird dann in der Ptychographie nicht nur die Intensität gemessen sondern auch die Phase der Elektronenwelle rekonstruiert.

In einem TEM kann mit CS die Aufnahmerate der CCD-Kamera vergrößert werden, indem eine Blende während der Aufnahme eines Frames vor dem CCD-Sensor durch Piezoelemente verschoben wird. So werden verschiedene Pixel eines Frames zu unterschiedlichen Zeiten beleuchtet, so dass aus den zu unterschiedlichen Zeiten aufgenommenen Pixeln mit CS jeweils ein Bild rekonstruiert werden kann. Daraus kann eine Videosequenz mit mehr Frames erstellt werden, als es der ursprüngliche Detektor ermöglicht.[9]

Eine weitere Anwendung für CS ist die Tomographie. Dabei wird die Probe dreidimensional aus zweidimensionalen Bildern rekonstruiert. Ohne CS wird dazu gewöhnlich die gefilterte Rückprojektion oder die Kaczmarz-Methode verwendet. Mit CS können in der Tomographie gute Ergebnisse erzielt werden, auch wenn die Probe nicht aus allen Winkeln ($< 180^\circ$) aufgenommen werden kann [10] oder die Aufnahmen unvollständig sind.[11] Bei atomaren Aufnahmen kann die begrenzte (sparse) Anzahl an Atomen genutzt werden, indem diese durch Gaußpunkte modelliert werden, um mit CS die Atompositionen in der Probe zu rekonstruieren.[12]

5 Algorithmen

Dieser Abschnitt liefert ein Überblick über die verwendeten Algorithmen für CS. Davon wurden OMP und IHT selbst implementiert. Die Löser linprog und lsqlin sind allgemeine Löser aus der Optimization Toolbox in Matlab, so dass CS für diese Löser in eine passende Form umformuliert werden musste. Die restlichen Algorithmen wurden von ihren Autoren speziell für CS entwickelt, so dass sie meist nur die jeweiligen Vektoren und Matrizen \mathbf{A} , $\bar{\mathbf{A}}$, \mathbf{y} und einen Anfangswert der Lösung \mathbf{x}_0 benötigen.

Die verwendeten Algorithmen sind Orthogonal Matching Pursuit (OMP), Iterative Hard Thresholding (IHT), die SPGL1 Löser für Basis Pursuit (spg bp), Basis Pursuit Denoising (spg bpdn) und Lasso Pursuit (spg lasso), die ℓ_1 -Magic Löser für Basis Pursuit (l1eq pd) und für Basis Pursuit Denoising (l1qc logbarrier), die Löser NESTA und FPC_AS und Matlab-Löser linprog und lsqlin zur Lösung linearer bzw. quadratischer Optimierungsprobleme. Bevor die Funktionsweise der Algorithmen erklärt werden kann, werden wichtige Konzepte der konvexen Optimierung erläutert.

Hinweis: Alle Vektoren wurden mit kleingeschriebenen fettgedruckten Buchstaben \mathbf{a} und alle Matrizen mit großgeschriebenen fettgedruckten Buchstaben \mathbf{A} bezeichnet.

5.1 Abweichungsquadrat der Samples

Beim CS wird die ℓ_1 -Norm der Koeffizienten $\|\mathbf{x}\|_1$ minimiert. Gleichzeitig wird das Ergebnis der Transformation \mathbf{Ax} an die Samples \mathbf{y} angepasst. Der Fehler der Anpassung wird oft als Abweichungsquadrat der Samples

$$sqr(\mathbf{x}) = \frac{1}{2} \left\| \mathbf{Ax} - \mathbf{y} \right\|_2^2 \quad (14)$$

$$= \frac{1}{2} \sum_n \left(\sum_{k,t} \Phi_{n,t} \Psi_{t,k}^H \mathbf{x}_k - \mathbf{y}_n \right)^2 \quad (15)$$

angegeben. Das Abweichungsquadrat der Samples wird zum Beispiel bei der Anpassung der Samples durch ein Gradientenabstiegsverfahren minimiert.

5.2 Gradientenverfahren

Der Gradientenabstieg ist ein Verfahren, um ein lokales Minimum einer Funktion $opt(\mathbf{x})$ zu finden. Beim CS wird in erster Linie entweder $\|\mathbf{x}\|_1$ oder $sqr(\mathbf{x})$ minimiert. Dabei sind beide Funktionen konvex, so dass das lokale Minimum dem globalen Minimum entspricht.

Beim Gradientenabstieg wird eine Fehlerfunktion $opt(\mathbf{x})$ minimiert. Der Gradient dieser Funktion an einer Stelle $\mathbf{x}_{[n]}$ zeigt die Richtung der höchsten Zunahme der Fehlerfunktion an. Ausgehend von einem Punkt $\mathbf{x}_{[n]}$ kann in entgegengesetzte Richtung zum Gradienten ein Punkt $\mathbf{x}_{[n+1]}$ gefunden werden, in dem die Fehlerfunktion einen kleineren Wert ergibt. Dabei muss die Schrittweite μ von $\mathbf{x}_{[n]}$ zu $\mathbf{x}_{[n+1]}$ sehr genau gewählt werden. Bei sehr kleiner Schrittweite wird das Minimum nur langsam erreicht und bei großer Schrittweite kann das Minimum übersprungen werden. Zusammengefasst wird für den Gradientenabstieg folgende Rechenvorschrift bis zur Konvergenz wiederholt:

$$\mathbf{x}_{[n+1]} = \mathbf{x}_{[n]} - \mu \nabla opt(\mathbf{x}_{[n]}) \quad (16)$$

Für die Anpassung der Samples kann das Abweichungsquadrat der Samples als Fehlerfunktion des Gradientenabstiegs gewählt werden. Die Ableitung von $sqr(\mathbf{x})$ nach \mathbf{x}_i ergibt:

$$\frac{\partial}{\partial \mathbf{x}_i} sqr(\mathbf{x}) = \sum_n \Psi_{k,t} \bar{\Phi}_{t,n} \Big|_{k=i} \cdot \left(\sum_{k,t} \Phi_{n,t} \bar{\Psi}_{t,k} \mathbf{x}_k - \mathbf{y}_n \right) \quad (17)$$

Somit beträgt der Gradient

$$\nabla sqr(\mathbf{x}) = \bar{\mathbf{A}} \cdot (\mathbf{Ax} - \mathbf{y}) \quad (18)$$

und

$$\mathbf{x}_{[n+1]} = \mathbf{x}_{[n]} - \mu \bar{\mathbf{A}} \cdot (\mathbf{A}\mathbf{x}_{[n]} - \mathbf{y}) \quad (19)$$

ist ein Iterationsschritt des Gradientenabstiegs für eine Anpassung der Samples.

Eine Erweiterung des Gradientenabstiegs ist das Newtonverfahren zur mathematischen Optimierung:

$$\mathbf{x}_{[n+1]} = \mathbf{x}_{[n]} - \mu \frac{\nabla \text{opt}(\mathbf{x}_{[n]})}{\nabla^2 \text{opt}(\mathbf{x}_{[n]})} \quad (20)$$

Dies folgt (für $\mu = 1$) daraus, dass der Gradient $\nabla \text{opt}(\mathbf{x}_{[n]})$ durch dessen Ableitung am Ort $\mathbf{x}_{[n]}$ linear interpoliert wird, so dass die Nullstelle der Interpolation $\mathbf{x}_{[n+1]}$ näher an der tatsächlichen Nullstelle des Gradienten liegt.

Analog dazu gibt es das Newtonverfahren für die iterative Berechnung von Nullstellen einer Funktion $f(\mathbf{x})$:

$$\mathbf{x}_{[n+1]} = \mathbf{x}_{[n]} - \mu \frac{f(\mathbf{x}_{[n]})}{\nabla f(\mathbf{x}_{[n]})} \quad (21)$$

5.3 Iterative Hard Thersholding

Beim Iterative Hard Thersholding[13] (IHT) wird eine sparse Lösung \mathbf{x} zur Darstellung der Samples \mathbf{y} gesucht. Dazu wird in jeder Iteration zuerst ein Schritt $\mathbf{x}_{[n+1]}$ durch Gradientenabstieg berechnet und anschließenden der sogenannte der Hard Thresholding Operator H darauf angewandt. Dieser erhält die größten s Koeffizienten und setzt den Rest auf Null. Zusammengefasst wird der nächste Schritt der Iteration so berechnet:

$$\mathbf{x}_{[n+1]} = H \left\{ \mathbf{x}_{[n]} - \mu \bar{\mathbf{A}} \cdot (\mathbf{A}\mathbf{x}_{[n]} - \mathbf{y}) \right\} \quad (22)$$

Diese Schritte werden bis zur Konvergenz wiederholt. Damit der Algorithmus schnell konvergiert, wird die Schrittweite μ des Gradientenabstiegs bei jeder Iteration angepasst, die Zahl s der Koeffizienten \mathbf{x} ungleich Null schrittweise erhöht und der Algorithmus bei überschreiten einer festgelegten Genauigkeit der Anpassung der Samples beendet.

5.4 Iterative Soft Thresholding

Das Iterative Soft Thresholding beruht auf dem Proximal Gradientenabstiegsverfahren. In unserem Fall wird dieses Verfahren zur Lösung des folgenden, dem Lasso Pursuit äquivalenten Problem, verwendet:

$$\min \left(\|\mathbf{Ax} - \mathbf{y}\|_2^2 + \lambda \|\mathbf{x}\|_1 \right) \quad (23)$$

Beim Proximal Gradientenabstiegsverfahren wird eine Summe aus einer differenzierbaren Funktion $g(\mathbf{x}) = \text{sqr}(\mathbf{x})$ und einer nicht differenzierbaren Funktion $h(\mathbf{x}) = \lambda \|\mathbf{x}\|_1$ minimiert. Weil die zweite Funktion nicht differenzierbar ist, kann diese Gleichung nicht einfach mit Gradientenabstieg gelöst werden. Dazu wird ein Vektor \mathbf{y} gesucht, der sowohl möglichst gering vom Schritt des Gradientenabstiegs zur Anpassung der Samples abweicht, als auch den Absolutwert der zweiten Funktion $h(\mathbf{y})$ klein lässt. Dieser Vektor ist das Ergebnis $\mathbf{x}_{[n+1]}$ der Iteration:

$$\mathbf{x}_{[n+1]} = \arg \min_{\mathbf{y}} \left[\frac{1}{2\mu} \|\mathbf{y} - \{\mathbf{x}_{[n]} - \mu \nabla g(\mathbf{x}_{[n]})\}\|_2^2 + h(\mathbf{y}) \right] \quad (24)$$

Dies lässt sich auch mit dem Proximal Operator, angewandt auf einen Schritt des Gradientenabstiegs zur Anpassung der Samples, schreiben

$$\mathbf{x}_{[n+1]} = prox_h(\mathbf{x}_{[n]} - \mu \nabla g(\mathbf{x}_{[n]})) \quad , \quad (25)$$

wobei der Proximal Operator

$$prox_h(\mathbf{x}_{[n]}) = \arg \min_{\mathbf{y}} \left[\frac{1}{2\mu} \|\mathbf{y} - \mathbf{x}_{[n]}\|_2^2 + h(\mathbf{y}) \right] \quad (26)$$

entspricht.

Im Falle von $h(\mathbf{x}) = \lambda \|\mathbf{x}\|_1$ lässt sich ein Minimum für den Proximal Operator über Nullsetzten des Subgradienten berechnen [14]. Hierbei entspricht der Proximal Operator dem Soft Thresholding Operator $S_\lambda(\mathbf{x})$:

$$prox_h(\mathbf{x}) = S_\lambda(\mathbf{x}) = \begin{cases} \mathbf{x}_i - \lambda & \text{für } \mathbf{x}_i > \lambda \\ 0 & \text{für } -\lambda_s \leq \mathbf{x}_i \leq \lambda \\ \mathbf{x}_i + \lambda & \text{für } \mathbf{x}_i < \lambda \end{cases} \quad (27)$$

Der Soft Thresholding Operator wird also bei jeder Iteration auf das Ergebnis eines Gradientenabstiegs angewandt. Er nähert alle Koeffizienten \mathbf{x} um λ der Null an, so dass kleine Koeffizienten ganz verschwinden. Zusammengefasst entspricht eine Iteration von Iterative Soft Thresholding folgender Anweisung:

$$\mathbf{x}_{[n+1]} = S_\lambda(\mathbf{x}_{[n]} - \mu \nabla g(\mathbf{x}_{[n]})) \quad (28)$$

5.5 Löser SPGL1 für Basis Pursuit Denoising

Der Algorithmus Spectral Projected Gradient zur Minimierung der ℓ_1 -Norm, kurz SPGL1 [15] löst das Basis Pursuit Denoising

$$\min \|\mathbf{x}\|_1 \quad \text{mit} \quad \|\mathbf{Ax} - \mathbf{y}\|_2 \leq \sigma \quad (29)$$

indem der Lasso Pursuit

$$\min \|\mathbf{Ax} - \mathbf{y}\|_2 \quad \|\mathbf{x}\|_1 \leq \tau \quad (30)$$

gelöst wird und Parameter τ des Lasso Pursuit so angepasst wird, dass dessen Lösung dem Ergebnis des Basis Pursuit Denoising mit festgelegtem σ entspricht.

Es wird die Funktion $\phi(\tau) = \|\mathbf{Ax}_\tau - \mathbf{b}\|_2$ mit optimaler Lösung des Lasso Pursuit $\tilde{\mathbf{x}}_\tau$ zu jedem Parameter τ des Lasso Pursuit definiert. Um damit den Basis Pursuit Denoising zu lösen, muss $\phi(\tau)$ dem festgelegtem Wert σ entsprechen. Um diesen Punkt zu finden, wird die Nullstelle $\phi(\tau) - \sigma$ mit dem Newtonverfahren gesucht:

$$\tau_{[n+1]} = \tau_{[n]} - \frac{\phi(\tau_{[n]}) - \sigma}{\frac{d}{d\tau} \phi(\tau)} \quad (31)$$

Dies ist möglich, weil $\phi(\tau)$ eine differenzierbare Funktion ist. Der Algorithmus zur Lösung des Lasso Pursuits ist das Iterative Soft Thresholding. Dabei wird λ des Soft Thresholding Operators geeignet gewählt, damit $\|\mathbf{x}\|_1$ knapp unterhalb $\tau_{[n]}$ liegt.

Insgesamt wird beginnend mit $\tau = 0$ nacheinander der Lasso Pursuit näherungsweise gelöst und daraus eine bessere Schätzung für $\tau_{[n]}$ berechnet. Dieses Prinzip wird wiederholt angewandt, so dass sich der Algorithmus der Lösung des Basis Pursuit Denoising nähert.

5.6 Löser FPC_AS für Lasso Pursuit

Beim Fix Point Continuation on Active Set Algorithmus [16] (FPC_AS) wird eine Variante des Lasso Pursuit gelöst:

$$\min \left(\|\mathbf{Ax} - \mathbf{y}\|_2^2 + \lambda \|\mathbf{x}\|_1 \right) \quad (32)$$

Zuerst kommt Iterative Soft Thresholding zum Einsatz, um sich der Lösung des Problems anzunähern. Sobald das Problem mit ausreichender Genauigkeit gelöst wurde, wird die zweite Phase eingeleitet. Es werden die Null-Koeffizienten auf Null festgesetzt und das Problem mit einem quasi-Newtonverfahren für die Koeffizienten ungleich Null gelöst. Das Verfahren heißt quasi-Newtonverfahren, weil die zweite Ableitung des Newtonverfahrens zur Optimierung nicht berechnet, sondern aus vorherigen Iterationen abgeschätzt wird. Der Vorteil eines solchen Verfahrens ist es, dass es meist schneller konvergiert.

Dieser Algorithmus beginnt wie SPGL1 mit einem geschätzten Parameter λ und nähert diesen, durch wiederholtes Lösen des oberen Schemas und Anpassen des Parameters λ , der tatsächlichen Lösung an. Dabei wird λ aus dem größten Element des Gradienten der vorherigen Lösung $\|\nabla sgr(\mathbf{x})\|_\infty$ abgeschätzt. Die Abschätzung von λ beruht darauf, dass der Betrag des Gradienten der exakten Lösung kleiner gleich λ ist. Um das zu zeigen, wird das Lasso Pursuit mit dem Subgradienten gelöst.

Dazu ein Vorwort: Eine Ableitung wird durch die Konvergenz des Differentialquotienten von zwei Seiten definiert. Ist eine Funktion nicht ableitbar (Ableitung nicht eindeutig), so ist das Subdifferential eine Verallgemeinerung der Ableitung, in dem das Subdifferential dort Werte auf einem Intervall zwischen den Differentialkoeffizienten annimmt. Hier als Beispiel das Subdifferential von $\|\mathbf{x}\|_1$:

$$\partial|\mathbf{x}| = \begin{cases} sign(\mathbf{x}_i) & \text{wenn } \mathbf{x}_i \neq 0 \\ [-1, 1] & \text{wenn } \mathbf{x}_i = 0 \end{cases} \quad (33)$$

Das Subdifferential kann statt der Ableitung für nicht ableitbare Funktionen genutzt werden, um an der Nullstelle des Subdifferentials ein mögliches Minimum der Funktion zu finden. Angewandt auf den Lasso Pursuit bekommen wir folgendes Ergebnis:

$$0 \in \nabla sgr(\mathbf{x}) + \lambda \partial|\mathbf{x}| \quad (34)$$

Daraus folgt, dass der Betrag des Gradienten $|\nabla sgr(\mathbf{x})|$ für mindestens eine Lösung des Problems kleiner oder gleich λ ist, woraus λ abgeschätzt werden kann.

5.7 Löser Nesta für Basis Pursuit Denoising

NESTA [17] löst das Basis Pursuit Denoising:

$$\min \|\mathbf{x}\|_1 \quad \text{mit} \quad \|\mathbf{Ax} - \mathbf{y}\|_2 \leq \sigma \quad (35)$$

Der Algorithmus beruht auf Nesterovs Methode eine nicht differenzierbare Funktion durch eine glatte Funktion anzunähern. In diesem Fall wird die ℓ_1 -Norm von \mathbf{x} durch eine glatte Funktion angenähert:

$$\|\mathbf{x}\|_1 = \sum_i |\mathbf{x}_i| \quad (36)$$

$$= \max_{\mathbf{u}_i \in Q_d} \mathbf{u} \cdot \mathbf{x} \quad (37)$$

$$\approx f_\gamma(\mathbf{x}) := \max_{\mathbf{u}_i \in Q_d} \left(\mathbf{u} \cdot \mathbf{x} - \frac{\gamma}{2} \|\mathbf{u}\|_2 \right) \quad (38)$$

Dabei gilt $Q_d = [-1, 1]$, so dass \mathbf{u}_i Eins oder minus Eins ist. Für $\gamma \rightarrow 0$ entspricht die Näherung exakt der ℓ_1 -Norm von \mathbf{x} . Die Ableitung dieser Funktion ist Lipschitz-stetig mit $L = \frac{1}{\gamma}$. Sie beträgt:

$$\frac{\partial}{\partial \mathbf{x}_i} f_\gamma(\mathbf{x}) = \begin{cases} \frac{\mathbf{x}_i}{\gamma} & \text{für } \mathbf{x}_i < \gamma \\ sign(\mathbf{x}_i) & \text{sonst} \end{cases} \quad (39)$$

Es wird die Minimierung der ℓ_1 -Norm durch die Minimierung der glatten Funktion $f_\gamma(\mathbf{x})$ auf dem Lösungsbereich des Basis Pursuit Denoising $Q_P = \{\mathbf{x} : \|\mathbf{Ax} - \mathbf{y}\|_2 \leq \epsilon\}$ ersetzt:

$$\min_{\mathbf{x} \in Q_P} f_\gamma(\mathbf{x}) \quad (40)$$

Dadurch wird die Unstetigkeit der ℓ_1 -Norm bei Null umgangen. Ein Iterationsschritt wird mit NESTA so berechnet:

für Iteration [k]: (41)

$$\mathbf{v}_{[k+1]} = \operatorname{argmin}_{\mathbf{x} \in Q_P} \left(\frac{L}{2} \|\mathbf{x} - \mathbf{x}_{[k]}\|_2^2 + \nabla f(\mathbf{x}_{[k]}) \cdot (\mathbf{x} - \mathbf{x}_{[k]}) \right) \quad (42)$$

$$\mathbf{z}_{[k+1]} = \operatorname{argmin}_{\mathbf{x} \in Q_P} \left(\frac{L}{2} \|\mathbf{x} - \mathbf{x}_{[0]}\|_2^2 + \sum_{[i]=0}^k \alpha_{[i]} \nabla f(\mathbf{x}_{[i]}) \cdot (\mathbf{x} - \mathbf{x}_{[i]}) \right) \quad (43)$$

$$\mathbf{x}_{[k+1]} = \tau_{[k]} \mathbf{z}_{[k]} + (1 - \tau_{[k]}) \mathbf{y}_{[k]} \quad (44)$$

$\mathbf{v}_{[k]}$ entspricht einem Schritt der Länge $\mathbf{x} - \mathbf{x}_{[k]}$ in Richtung des Gradienten der glatten Funktion $f_\gamma(\mathbf{x})$. Dieser Schritt wird durch den Term $\frac{L}{2} \|\mathbf{x} - \mathbf{x}_{[k]}\|_2^2$ klein gehalten.

Der Term $\mathbf{z}_{[k]}$ mindert abrupte Änderungen im Gradienten, denn er entspricht einer gewichteten Summe aller vorheriger Gradienten. Auch in diesem Term wird die Entfernung vom Startwert $\mathbf{x}_{[0]}$ durch $\|\mathbf{x} - \mathbf{x}_{[0]}\|_2^2$ begrenzt. Das Ergebnis der Iteration $\mathbf{x}_{[k]}$ ist ein gewichteter Wert aus $\mathbf{v}_{[k]}$ und $\mathbf{z}_{[k]}$.

Theoretische wurde eine gute Konvergenz mit den Parametern $\alpha_{[k]} = \frac{1}{2(k+1)}$ und $\tau_{[k]} = \frac{2}{k+3}$ nachgewiesen, so dass diese Parameter im Algorithmus implementiert sind. Um die Werte von $\mathbf{y}_{[k]}$ und $\mathbf{z}_{[k]}$ zu berechnen, muss man die beiden Schritte als sogenannte Lagrange-Funktionen mit dem Parameter η schreiben (hier am Beispiel von $\mathbf{v}_{[k]}$ gezeigt):

$$\mathcal{L}(\mathbf{x}, \eta) = \frac{L}{2} \|\mathbf{x} - \mathbf{x}_{[k]}\|_2^2 + \frac{\eta}{2} (\|\mathbf{Ax} - \mathbf{y}\|_2^2 - \epsilon^2) + \nabla f(\mathbf{x}_{[k]}) \cdot (\mathbf{x} - \mathbf{x}_{[k]}) \quad (45)$$

Wir suchen ein \mathbf{x} , welches im Minimum der Lagrangefunktion liegt. Dazu wird das Extrema gesucht, in dem die Ableitung dieser Funktion am Extrema Null entspricht. Das Ergebnis kann in der Publikation zu Nesta nachgelesen werden.

Die Konvergenz des Algorithmus ist von der Lipschitz-Stetigkeit mit Lipschitzkonstante L abhängig, so dass sie mit kleinerem γ langsamer ist. Allerdings nimmt die Genauigkeit der Lösung mit kleinerem γ zu, weil die glatte Funktion genauer der ℓ_1 -Norm entspricht. Daraus wird die Fortsetzung des Algorithmus definiert. Der Algorithmus wird mehrfach mit jeweils kleinerem γ bis zur Konvergenz ausgeführt, wobei der Startwert $\mathbf{x}_{[0]}$ wird immer wieder aus der Lösung der vorherigen Rechnung übernommen wird.

5.8 Löser l1eq pd und linprog für Basis Pursuit

Basis Pursuit

$$\min \|\mathbf{x}\|_1 \quad \text{mit} \quad \mathbf{Ax} = \mathbf{y} \quad (46)$$

wurde durch den Löser l1eq pd (von ℓ_1 -magic)[19] und den Löser linprog (aus der Matlab Optimization Toolbox) gelöst. In beiden Fällen wird das Problem in ein lineares Optimierungsproblem umformuliert. Dieses entspricht der Minimierung eines Skalarprodukts zweier Vektoren mit weiteren Nebenbedingungen. Ein Problem ist dabei, dass ein lineares Optimierungsproblem keine Möglichkeit hat eine ℓ_1 -Norm zu bilden. In beiden Fällen wird durch Nebenbedingungen das Erfüllen der Gleichung $\mathbf{Ax} = \mathbf{y}$ gefordert und ein weiterer Vektor \mathbf{t} durch Ungleichungen größer gleich dem Betrag von \mathbf{x} festgesetzt. Das Skalarprodukt dieses Vektors \mathbf{t} mit einem Einheitsvektor wird durch den Algorithmus minimiert, so dass dadurch $\|\mathbf{x}\|_1$ minimiert wird.

Das lineare Optimierungsproblem wird bei linprog mit einem Simplex-Verfahren und bei l1eq pd mit einem Innere-Punkte-Verfahren gelöst. Der Raum der Lösungen wird durch die Nebenbedingungen auf einen Bereich (Simplex) begrenzt. Das Simplex-Verfahren sucht das Minimum des Problems, in dem es eine triviale Lösung des Problems auf dem Rand der Lösungen findet und dann dem Rand folgt. Dazu wird das gesamte Problem in ein lineares Gleichungssystem umgeschrieben, wobei die Ungleichungen durch positive Hilfsvariablen einfließen (sogenanntes Simplex-Tableau). Dann wird Gaußumformung wiederholt auf das Gleichungssystem so angewandt, dass das Skalarprodukt der beiden Vektoren minimiert wird, bis ein Minimum erreicht ist.[18]

Ein Innere-Punkte-Verfahren geht von einer trivialen Lösung des linearen Optimierungsproblems aus und nähert sich dem Minimum durch Gradientenabstieg. Dabei werden die Nebenbedingungen und das Skalarprodukt in mehrere Gleichungen umgeschrieben, welche durch Gradientenabstieg gleichzeitig minimiert werden.

5.9 Löser lsqlin für Lasso Pursuit

Der Lasso Pursuit

$$\min \|\mathbf{Ax} - \mathbf{y}\|_2 \quad \text{mit} \quad \|\mathbf{x}\|_1 \leq \tau \quad (47)$$

kann als ein Problem kleinster Abweichungsquadrat mit linearen Bedingungen formuliert werden und mit quadratischer Optimierung lsqlin (aus der Matlab Optimization Toolbox) gelöst werden. Dabei wird in erster Linie das Abweichungsquadrat der Samples $\|\mathbf{Ax} - \mathbf{y}\|_2^2$ minimiert. Durch weitere Nebenbedingungen wird die ℓ_1 -Norm $\|\mathbf{x}\|$ kleiner τ gehalten. Dieses Problem wird durch den Löser lsqlin mit einem Innere-Punkte-Verfahren gelöst.

5.10 Löser l1qc logbarrier für Basis Pursuit Denoising

Das Basis Pursuit Denoising

$$\min \|\mathbf{x}\|_1 \quad \text{mit} \quad \|\mathbf{Ax} - \mathbf{y}\|_2 \leq \sigma \quad (48)$$

wird als sogenanntes Second Order Cone Programm formuliert und mit dem Löser l1qc logbarrier (von ℓ_1 -magic)[19] gelöst. Das ist eine Kombination der Minimierung des Skalarprodukts zweier Vektoren mit Nebenbedingungen, welche lineare und quadratische Terme enthalten können. Die ℓ_1 -Norm von \mathbf{x} wird ähnlich wie bei der linearen Optimierung umformuliert. Die wichtige Nebenbedingung dieses Problems ist, dass der Fehler $\|\mathbf{Ax} - \mathbf{y}\|_2^2$ kleiner σ^2 bleibt. Der Logarithmus dieser Bedingung $\log_{10}(\|\mathbf{Ax} - \mathbf{y}\|_2^2 - \sigma^2)$ wird als sogenannte Barrierfunktion bezeichnet. Nähert sich die Bedingung Null an, so geht der Logarithmus gegen unendlich. Somit wird das Erfüllen dieser Bedingung beim Lösen des Minimierungsproblems sichergestellt. Das resultierende Problem wird ähnlich zu l1eq pd mit einem Innere-Punkte-Verfahren gelöst. Dabei wird statt der Gleichheitsbedingung $\mathbf{Ax} = \mathbf{y}$ die Barrierfunktion minimiert. [19]

5.11 Orthogonal Matching Pursuit

Beim Orthogonal Matching Pursuit (OMP) werden die Samples \mathbf{y} durch einen gierigen Algorithmus iterativ angepasst. In jeder Iteration wird ein Koeffizient \mathbf{x}_i zur Lösung hinzugefügt, der die Darstellung der Samples $\mathbf{y} = \mathbf{Ax}$ am besten verbessert. Dazu wird die Korrelation $\text{cor}(\mathbf{y})$ zwischen den Spalten der Transfomationsmatrix \mathbf{A} mit den Samples \mathbf{y} berechnet. Es werden die Samples mit der adjungierten Gesamtmatrix \mathbf{A}^\dagger transformiert, so dass die Amplitude des Ergebnisses der Korrelation entspricht:

$$\text{cor}(\mathbf{y}) = \overline{\mathbf{A}} \cdot \mathbf{y} \quad (49)$$

Die Samples werden aus einer Linearkombination der Spalten der Darstellungsmatrix \mathbf{A} mit den Koeffizienten \mathbf{x} als Faktoren gebildet. Von dieser Idee ausgehend wird beim OMP in jedem Schritt des Algorithmus ein Koeffizient \mathbf{x}_i mit der höchsten Korrelation zu den Samples hinzugefügt und dann ein lineares Gleichungssystem für alle bisher ausgewählten Koeffizienten mit lsqlin gelöst, um die Amplitude der Koeffizienten in der Lösung genau zu bestimmen.

Zur Beschleunigung wird die Lösung aus dem letzten Iterationsschritt als Startwert verwendet und mit verringriger Genauigkeit gerechnet. Nach der ersten Iteration hat man eine Annäherung \mathbf{y}' an die vorgegebenen Samples erreicht, so dass jetzt die Korrelation mit dem Anpassungsfehler $\text{cor}(\mathbf{y} - \mathbf{y}')$ berechnet wird, um den nächsten Koeffizienten auszuwählen. Sobald die gewünschte Anpassungsgenauigkeit der Samples oder eine maximale Anzahl an Koeffizienten erreicht wurde, wird der Algorithmus beendet. Dabei werden die Koeffizienten noch einmal mit lsqlin mit erhöhter Genauigkeit berechnet. Dieser Algorithmus ist ein Beispiel für einen gierigen Algorithmus, weil bei jeder Iteration jeweils ein Koeffizient hinzugefügt wird und im Nachhinein nicht mehr entfernt werden kann.

5.12 Interpolation

Zur Rekonstruktion des Signals wurde neben CS auch Interpolation der Samples herangezogen. Dies ist nur möglich, weil in unserem Fall die Samples identisch mit ausgewählten Signalpunkten sind. Somit können unbekannte Signalpunkte aus den Samples in ihrer Umgebung angenähert werden. Bei der Interpolation eines Signalpunkts mit seinen **nächsten Nachbarn** wird das Sample mit kleinsten Abstand zum gesuchten Datenpunkt gefunden und dessen Wert übernommen. Bei der **linearen** und **quadratischen** Interpolation werden die nächsten 2 bzw. 3 Samples durch ein Polynom ersten bzw. zweiten Grades angepasst und daraus der Wert des gesuchten Datenpunkts aus dem Wert des Polynoms an dieser Stelle berechnet. Zuletzt wurde die Interpolation mit **natürlichen nächsten Nachbarn** verwendet. Dazu muss zuerst der Begriff des Voronoi-Diagramms definiert. Befinden sich mehrere Punkte im Raum, so ist ein Voronoi-Diagramm eine Unterteilung des Raumes, so dass jeder Ort des Raumes dem nahe-liegendsten Punkt zugeordnet wird. Bei der Interpolation mit natürlichen nächsten Nachbarn wird ein Voronoi-Diagramm für die Samples erzeugt. Zusätzlich wird auch ein Voronoi-Diagramm der Samples zusammen mit einem gesuchten Punkt erstellt, wobei nur die Voronoi-Zelle um den gesuchten Punkt von Bedeutung ist. Diese sind in Abbildung 8 abgebildet. Um den Wert des gesuchten Punktes zu berechnen, wird die Überschneidung der Voronoi-Zelle des gesuchten Punktes mit den Voronoi-Zellen der Samples berechnet. Diese werden als Gewichtung mit den Werten der Samples multipliziert und alles aufaddiert, um den Wert des gesuchten Punktes zu erhalten.

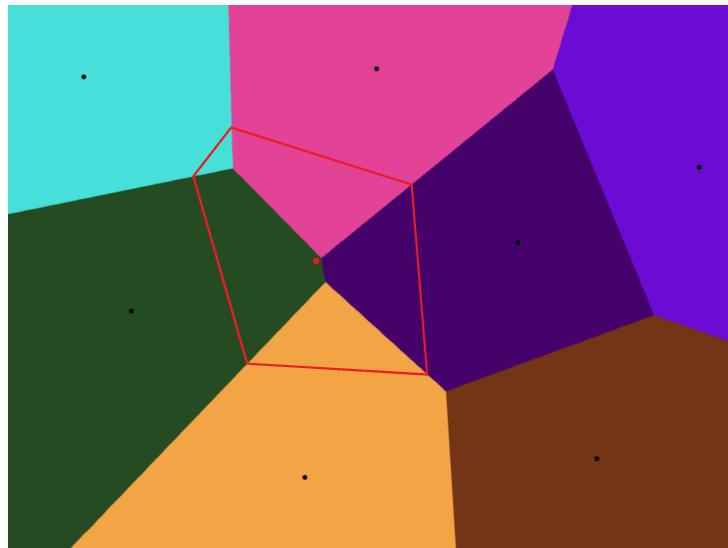


Abb. 8: Dargestellt sind Voronoi-Diagramme, welche für Interpolation mit natürlichen nächsten Nachbarn verwendet werden: Die schwarzen Punkte sind bekannte Samples. Diesen werden in einem Voronoi-Diagramm die Flächen unterschiedlicher Farbe zugeordnet. Zusätzlich wird ein gesuchter (roter) Punkt eingefügt und wieder ein Voronoi-Diagramm erstellt, wobei nur die Voronoi-Zelle um den roten Punkt als rotes Fünfeck dargestellt ist.

6 Vergleich unterschiedlicher diskreter Transformationen

In diesem Abschnitt sehen wir uns verschiedene Transformationen und ihre Vor- und Nachteile zur Darstellung von Signalen an. Die verwendete Transformationen sind die diskrete Kosinus-II-transformation (DCT), diskrete Fouriertransformation (DFT), eine reduzierte Fouriertransformation (CDFT), die Haar-Wavelets (HAAR) und die Daubechies-4-Wavelets (DB4). In Abbildung 9 sind die betrachteten Transformationen auf ein Beispielbild angewandt, wobei die Koeffizienten logarithmisch dargestellt sind. Die DCT und DFT stellen das Bild mit periodischen Schwingungen unterschiedlicher Frequenzen vollständig dar. Die HAAR und DB4 (Wavelets) stellen das Bild auf unterschiedlichen Größenskalen dar. Dabei entspricht das transformierte Bild in Richtung links oben zunehmend einem Tiefpassbild und nach rechts unten einem Hochpassbild des Originalbildes.

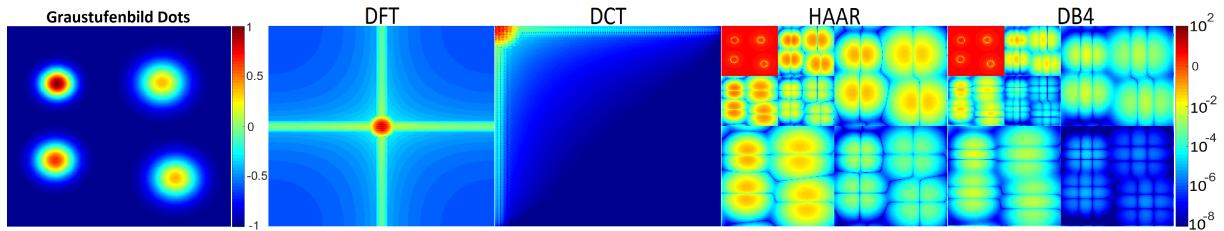


Abb. 9: Transformation eines Beispielbildes (ganz links) mit unterschiedlichen Transformationen logarithmisch dargestellt

6.1 Globale Transformationen

Fourier-, Sinus- und Kosinustransformation teilen ein Signal auf eine Reihe periodischer Schwingungen auf, mit denen dieses Signal repräsentiert wird. Die Koeffizienten werden durch Multiplikation des Signals $f(x)$ mit der Transformationsfunktion $\Psi(x, \omega)$ und anschließender Integration berechnet. Anschaulich berechnet man damit die Korrelation des Signals $f(x)$ mit der Transformationsfunktion für unterschiedliche Frequenzen ω .

$$\mathcal{F}\{f\}(\omega) = \int_{-\infty}^{\infty} f(x)\Psi(x, \omega)dx \quad (50)$$

Im diskreten Fall werden die Datenpunkte des Signals mit den diskreten Werten der Transformationsfunktion multipliziert und aufaddiert. Dabei sind die diskreten Werte der Transformationsfunktion in einer Matrix Ψ angeordnet.

$$\mathbf{x}_k = \sum_{t=0}^{N-1} \Psi_{kt} \mathbf{f}_t \quad (51)$$

Die zwei verwendeten Transformationen sind die diskrete Fouriertransformation (DFT):

$$\Psi_{kt} = \frac{1}{\sqrt{N}} e^{-\frac{i2\pi}{N} kt} \quad (52)$$

und die diskrete Kosinus-II-transformation (DCT):

$$\Psi_{kt} = \sqrt{\frac{2}{N}} \sqrt{\frac{1}{1 + \delta_{k,t}}} \cos\left(\frac{\pi}{N}(n + \frac{1}{2})k\right). \quad (53)$$

Diese sind beispielsweise für die Transformation eines Signals mit $N=5$ Datenpunkten in Abbildung 10 abgebildet. Die Zeilen der Transformationsmatrix sind orthogonal zueinander und die Matrix selbst unitär. Dadurch entspricht die Rücktransformationmatrix $\bar{\Psi}$ der adjungierten Darstellungsmatrix.

Diskrete Signale haben eine begrenzte Länge N , womit es nach der Transformation auch genau N Koeffizienten \mathbf{x}_k gibt. Unterscheidet sich ein Signal \mathbf{f} zu einer Zeile der Rücktransformationsmatrix Ψ nur um einen Faktor, so ist nur ein Koeffizient zur Darstellung dieses Signals erforderlich. Alle anderen Signale werden aus Linearkombination mehrerer Zeilen der Rücktransformationsmatrix gebildet. Die Werte einer Zeile der Transformationsmatrix entsprechen an äquidistanten Orten entnommenen Werten der zugrundeliegenden periodischen Transformationsfunktion. Besteht ein Signal aus solchen Werten einer Transformationsfunktion, welche zusätzlich verschoben wurde, so kann dieses Signal im Falle der DCT nicht mehr mit nur einem Koeffizienten dargestellt werden. Bei der DFT hingegen entspricht eine Verschiebung nur einer Multiplikation der Koeffizienten mit $e^{i\varphi}$ im Phasenraum, so dass die Anzahl an Koeffizienten dadurch nicht verändert wird und die Darstellung sparse bleibt.

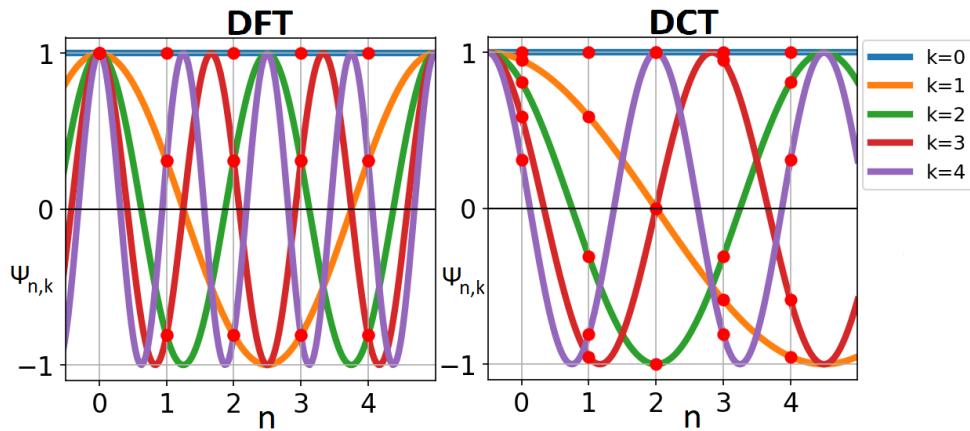


Abb. 10: Realteil der Schwingungen der DFT und DCT, so wie ihre diskreten Koeffizienten für die Transformation eines Signal mit $N=5$ Datenpunkten

Bei der DFT wird das Signal periodisch fortgesetzt, so dass der Randpunkt $\mathbf{x}_0 = \mathbf{x}_N$ Nachbar des Randpunktes \mathbf{x}_{N-1} ist. Das Signal muss an den Rändern von ähnlicher Amplitude (periodisch) sein, ansonsten kommt es dort zu Unstetigkeiten, so dass viele Koeffizienten zur Repräsentation des Signals nötig werden. Die Sinustransformationen und Kosinustransformationen setzen ein Signal gerade bzw. ungerade an den beiden Rändern fort. Es treten ähnliche Unstetigkeiten auf, weil das Signal mindestens an einem Rand ungerade fortgesetzt wird. Ausnahmen sind die Kosinustransformationen des Typs 1 und 2. Somit können unstetige Ränder mit diesen beiden Kosinustransformationen sparser dargestellt werden. Dabei ist die Kosinus-II-transformation die am häufigsten verwendete Transformation in der Bildverarbeitung. Mit ihr werden die Ränder des Signals weicher fortgesetzt und die Transformation läuft in Matlab doppelt so schnell wie die Kosinustransformationen 1. Aus diesem Grund wird in dieser Arbeit auch ausschließlich die Kosinustransformation vom Typ 2 verwendet. In Tabelle 1 sind die Vor- und Nachteile der DCT und DFT noch einmal kurz aufgezählt.

Tab. 1: Vor- und Nachteile globaler Transformationen

	Kosinus	Fourier
Koeffizienten	reell	komplex => Information doppelt vorhanden kann aber reduziert werden ! (CDFT)
Verschiebung	indirekt aus Linearkombination vieler Darstellungsmatrixspalten	berücksichtigt als Phasenfaktor $x \cdot e^{\frac{i2\pi}{N} k(t+\phi)} = x \cdot e^{\frac{i2\pi}{N} kt} \cdot e^{i\tilde{\phi}}$
periodischer Rand	nicht nötig (gerade Fortsetzung)	benötigt => sonst unstetiger Übergang

Neben der DCT und DFT wurde eine abgewandelte DFT, die CDFT in dieser Arbeit verwendet. Die DFT besitzt komplexe Koeffizienten. Werden reelle Signale transformiert, so sind die Hälfte der Koeffizienten komplex konjugierte der anderen Hälfte der Koeffizienten $\mathbf{x}_k = \mathbf{x}_{N-k}^*$. Die Ausnahme bilden der Koeffizient \mathbf{x}_0 und der Koeffizient $\mathbf{x}_{N/2}$ für Signale gerader Länge, die immer reell bleiben. Durch Nutzung der Symmetrie können die Koeffizienten in einem reellen Vektor angeordnet werden, so dass nur die Hälfte der Koeffizienten zum Darstellen des Signals notwendig werden. Dazu wurden Funktionen zur Reduzierung einer zweidimensionalen DFT in Matlab auf reelle Koeffizienten implementiert. Im weiteren Verlauf wird die DFT mit anschließender Reduzierung der Koeffizienten CDFT bezeichnet.

6.2 Lokale Transformationen (Wavelets)

Die zuvor genannten Transformationen haben gemeinsam, dass die Transformationsfunktion immer auf das gesamte Signal angewandt wird. Lokale Details erfordern somit sehr viele Linearkombinationen von Basisfunktionen, die sich größtenteils wieder aufheben müssen, so dass das Signal nicht sparse dargestellt wird. Bei den sogenannten Wavelet-Transformationen wird eine Transformationsfunktion $\Psi(x)$ auf Fenster unterschiedlicher Größe skaliert und verschoben. Dadurch wird das Signal auf verschiedenen Größenskalen und unterschiedliche Bereich unabhängig voneinander aufgelöst. Im kontinuierlichen Fall werden die Waveletfunktionen wie folgt konstruiert.

$$\Psi_{a,b}(x) = \frac{1}{\sqrt{a}} \Psi\left(\frac{x-a}{b}\right) \quad (54)$$

Auf diese Weise kann die Wavelettransformation analog zur Fouriertransformation berechnet werden:

$$T\{f(x)\}(a, b) = \int_{-\infty}^{\infty} f(x) \Psi_{a,b}(x) dx \quad (55)$$

Dabei ist a eine Verschiebung und b eine Skalierung der Ausgangsfunktion $\Psi(x)$. Im diskreten Fall wird oft eine Diskretisierung mit $a = 2^{-jk}$ und $b = 2^{-j}$ verwendet, so dass die Wavelets immer um einen Faktor 2 skaliert bzw. verschoben werden. Daraus ergibt sich, dass ein Signal mit Wavelets nur bis zu einer Ordnung k transformiert werden kann, wenn es k mal durch zwei teilbar ist. Zum Beispiel müssen ungerade Signale auf eine gerade Länge vergrößert werden, damit eine diskrete Wavelettransformation überhaupt angewendet werden kann.

In der Praxis wird die orthogonale, diskrete Wavelettransformation über Hochpassfilter h und Tiefpassfilter t definiert. Der Hochpassfilter wird dann auch als Waveletfilter und der Tiefpassfilter als Skalierungsfilter bezeichnet. Um die Koeffizienten zu berechnen, werden die Filter jeweils um zwei Signalpunkte verschoben und mit dem Signal multipliziert, um bei jeder Multiplikation jeweils einen Koeffizienten zu erhalten.

In dieser Arbeit wurden die einfachsten Wavelettransformationen aus der Gruppe der Daubechies-Wavelets verwendet. Die beiden implementierten Wavelettransformation waren die Haar-Wavelets (HAAR) und die Daubechies-4-Wavelets (DB4). Die zugehörigen Filter sind in Tabelle 2 und auf Abbildung 11 abgebildet. Diese Wavelets sind normiert und orthogonal zueinander. Sie können auch in einer unitären Transformationsmatrix geschrieben werden, so dass die inverse Matrix der transponierten Matrix entspricht. Eine HAAR-Transformation eines Signals aus 8 Elementen kann in erster Ordnung mit folgender Matrix durchgeführt werden:

$$\begin{pmatrix} t_0 & t_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & t_0 & t_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & t_0 & t_1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & t_0 & t_1 \\ h_0 & h_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & h_0 & h_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & h_0 & h_1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & h_0 & h_1 \end{pmatrix} \quad (56)$$

Dabei sind h_i und t_i Koeffizienten des zugehörigen Filters. Diese Transformation ergibt ein Hochpass- und Tiefpasssignal von jeweils halber Länge des ursprünglichen Signals. Dabei hat das Tiefpasssignal noch eine sehr große Ähnlichkeit zum ursprünglichen Signal. Aus diesem Grund wird das Tiefpasssignal noch einmal in einen Hochpass- und Tiefpassanteil transformiert. Daraus erhält man eine Auflösung des Signals auf einer weiteren Größenskala bzw. eine Transformation nächster Ordnung. Als Beispiel kann eine HAAR-Transformation dritter Ordnung auf folgende Weise konstruiert werden:

$$\mathbf{x} = \begin{pmatrix} t_0 & t_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ h_0 & h_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & & & & & & \\ \vdots & \vdots & & & & & & \\ 0 & 0 & & & & & & \end{pmatrix} \cdot \begin{pmatrix} t_0 & t_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & t_0 & t_1 & 0 & 0 & 0 & 0 \\ h_0 & h_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & h_0 & h_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & & & & \\ \vdots & \vdots & \vdots & \vdots & & & & \\ 0 & 0 & 0 & 0 & & & & \end{pmatrix} \cdot \begin{pmatrix} t_0 & t_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & t_0 & t_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & t_0 & t_1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & t_0 & t_1 \\ h_0 & h_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & h_0 & h_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & h_0 & h_1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & h_0 & h_1 \end{pmatrix} \cdot \mathbf{f}$$

$$(57)$$

Dabei sind $\mathbb{1}$ quadratische Einheitsmatrizen. Die Rücktransformation erfolgt durch Multiplikation der Koeffizienten mit den transponierten Matrizen in umgekehrter Reihenfolge. Für die DB4 kann die Transformation eines Signals aus 8 Elementen analog konstruiert werden:

$$\mathbf{x} = \begin{pmatrix} t_0 & t_1 & t_2 & t_3 & 0 & 0 & 0 & 0 \\ t_2 & t_3 & t_0 & t_1 & 0 & 0 & 0 & 0 \\ h_0 & h_1 & h_2 & h_3 & 0 & 0 & 0 & 0 \\ h_2 & h_3 & h_0 & h_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & & & & \\ \vdots & \vdots & \vdots & \vdots & & & & \\ 0 & 0 & 0 & 0 & & & & \end{pmatrix} \cdot \begin{pmatrix} t_0 & t_1 & t_2 & t_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & t_0 & t_1 & t_2 & t_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & t_0 & t_1 & t_2 & t_3 \\ t_2 & t_3 & 0 & 0 & 0 & 0 & t_0 & t_1 \\ h_0 & h_1 & h_2 & h_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & h_0 & h_1 & h_2 & h_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & h_0 & h_1 & h_2 & h_3 \\ h_2 & h_3 & 0 & 0 & 0 & 0 & h_0 & h_1 \end{pmatrix} \cdot \mathbf{f}$$

$$(58)$$

Diese Transformation ist nur von zweiter Ordnung, weil die DB4 vier statt zwei Filterkoeffizienten besitzen. Die Filter werden also paarweise in erster Ordnung auf das Signal angewendet und in allen weiteren Ordnungen auf das Ergebnis des Tiefpasses aus der vorherigen Ordnung. Somit erhält man durch den Hochpassfilter zuerst Koeffizienten für feinere, hochfrequente Signal-Details und mit jeder Ordnung immer gröbere, niederfrequente Details bis ein nicht mehr transformierbares Tiefpass-Signal übrig bleibt. Nur wenn das Signal eine Größe einer Potenz von zwei hat, bleibt nach der letzten Transformation mit dem Tiefpass nur ein Koeffizient übrig. Das Signal wird durch diese Schachtelung der Filter auf verschiedenen Skalen aufgelöst.

Tab. 2: Filterkoeffizienten für HAAR und DB4 Wavelets

Name	Waveletfilter h (Hochpass)	Skalierungsfilter (Tiefpass) t
HAAR	$\frac{1}{\sqrt{2}}[1, -1]$	$\frac{1}{\sqrt{2}}[1, 1]$
DB4	$\frac{1}{4\sqrt{2}}[1 - \sqrt{3}, -(3 - \sqrt{3}), 3 + \sqrt{3}, -(1 + \sqrt{3})]$	$\frac{1}{4\sqrt{2}}[1 + \sqrt{3}, 3 + \sqrt{3}, 3 - \sqrt{3}, 1 - \sqrt{3}]$

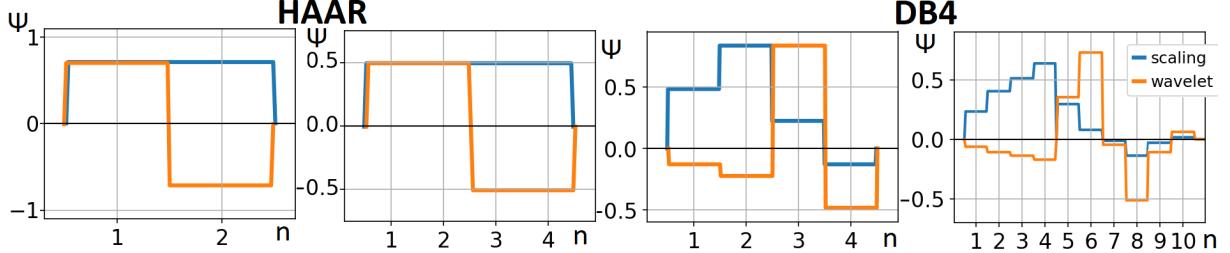


Abb. 11: Die Filterkoeffizienten für Haar- und DB4-Wavelets in erster und zweiter Ordnung.

Der Vorteil von Wavelets ist, dass das Signal auf unterschiedlichen Größenskalen durch die Transformation abgebildet wird. Lokale Details können durch weniger Koeffizienten dargestellt werden. Allerdings müssen die Abmessungen des Bildes im Optimalfall einer Potenz von zwei entsprechen. Außerdem ist der Rand auch hier unstetig, wenn das Bild nicht um die Anzahl an Filterkoeffizienten minus 2 gespiegelt wird. Tut man dies, so hat das Ergebnis nach jeder Transformation größere Abmessungen, als das Originalbild. Im Vergleich zur HAAR haben die DB4 einen weicheren Übergang zwischen den Koeffizienten der Filter, sie sind aber langsamer zu berechnen.

6.3 zweidimensionale Transformation für Löser

In dieser Arbeit geht es um CS von Bildern. Somit müssen die Transformationen für den zweidimensionalen Fall angepasst werden. Um eine Matrix bzw. ein Bild zu transformieren, werden die Transformationen nacheinander auf alle Zeilen und danach auf alle Spalten des Bildes angewandt. Die Reihenfolge ist hier nicht wichtig, da die üblichen Transformationen separierbar sind. Bei der CDFT werden die Koeffizienten dann nachträglich auf reelle Werte reduziert bzw. vor der Rücktransformation wieder in eine komplexe Matrix angeordnet. Bei den Wavelettransformationen hingegen ist die Transformation nur innerhalb einer Ordnung separierbar. Es lassen sich somit verschiedene Schemata für die 2D-Transformation definieren, von denen zwei implementiert wurden.

Alle aufgeführten Transformationen können entweder als Funktion oder als vierdimensionale Matrix für die Transformation von Bildern geschrieben werden. Gewöhnliche Algorithmen für CS unterstützen nur die Notation mit einer zweidimensionalen Gesamtmatrix A und eindimensionalen Vektoren y, x . Somit müssen die Samples y und Koeffizienten x eindimensional in einen Vektor angeordnet werden und die Transformationsmatrix A in eine zweidimensionale Matrix umgeschrieben werden. Wenn D Samples eines $N \times M$ großen Bildes bekannt sind, so ist die Matrix A von der Größe $D \times (N \cdot M)$. Somit können größere Matrizen schnell viel Arbeitsspeicher erfordern.

Bei einigen Lösen lässt sich die Matrix auch als Funktion übergeben. Dies hat den Vorteil, dass die Transformationsmatrix nicht explizit gespeichert werden muss und schneller berechnet werden kann, als eine Matrixmultiplikation.

6.4 Kohärenz zwischen Mess- und Darstellungsmatrix

Mit der Kohärenz von Matrizen kann eine Aussage über die Eindeutigkeit der Rekonstruktion mit der ℓ_1 -Norm beim CS getroffen werden (mehr dazu im Anhang). In diesem Abschnitt führen wir nur die Kohärenz von Matrizen ein und verwenden diese um qualitative Aussagen über unsere Darstellungsmatrizen

Ψ zu machen. Die Kohärenz

$$\mu(\mathbf{B}, \mathbf{C}) = \max_{i \neq j} \left(\frac{|(\mathbf{B}_i)^T \cdot \mathbf{C}_j|}{\|\mathbf{B}_i\|_2 \|\mathbf{C}_j\|_2} \right) \quad (59)$$

von Matrizen ist definiert als die maximale Korrelation der Spalten \mathbf{B}_i und \mathbf{C}_j der Matrizen \mathbf{B}, \mathbf{C} . Sie kann Werte im Intervall $[0, 1]$ einnehmen. Die Kohärenz gibt die größte Übereinstimmung der Spalten zweier unterschiedlicher Matrizen an. Ist die Kohärenz 0, so sind die Spalten orthogonal zueinander. Bei einer Kohärenz von 1 sind sie identisch.

Für eine Messmatrix Φ , bestehend aus Zeilen einer Einheitsmatrix, ist die Kohärenz mit einer beliebigen Darstellungsmatrix Ψ gleich dem größten Eintrag in der Darstellungsmatrix. In Tabelle 3 ist die Kohärenz für die verwendeten Darstellungsmatrizen in dieser Arbeit aufgeführt. Es fällt auf, dass die Kohärenz von den in dieser Arbeit verwendeten Wavelets (es gibt auch andere Wavelets !) mit unserer Messmatrix sehr groß ist. Darüber hinaus werden alle Transformationen mit hoher lokaler Auflösung eine ähnlich hohe Kohärenz aufweisen.

Für CS sind zwei Matrizen von großer Bedeutung, die Messmatrix Φ und die Darstellungsmatrix Ψ . Bis jetzt wurde die Darstellung des Signals \mathbf{f} immer mit der Basis Ψ und den Koeffizienten \mathbf{x} gemacht. Im folgenden werden auch die Samples \mathbf{y} als eine (wenn auch unvollständige) Darstellung des Signals \mathbf{f} in der Basis Φ bezeichnet.

Ist ein Signal \mathbf{f} in einer Basis Φ stark ausgebreitet (\mathbf{y} nicht sparse), so darf die Darstellungsbasis Ψ in der das Signal (mit \mathbf{x}) sparse repräsentiert wird nicht kohärent (ähnlich) dazu sein.[6]

Jetzt gibt es zwei Möglichkeiten:

Wenn die Samples sparse besetzt sind, so enthalten sie nur sehr wenig Information über das Signal und eine Rekonstruktion mit CS wäre sehr ungenau. Weil die Wavelets sehr kohärent mit unserer Messmatrix sind, werden in den Wavelets sparse Signale meist auch nur sehr schlecht (sparse) mit unserer Messmatrix erfasst. In so einem Fall wird die Rekonstruktion in einer beliebigen Darstellungsbasis keine guten Ergebnisse liefern, es könnte sogar Interpolation möglicherweise bessere Ergebnisse ergeben.

Wenn das Signal in der Messmatrix stark ausgebreitet ist (besitzt viele große Koeffizienten), so muss die Darstellungsmatrix inkohärent zur Messmatrix sein, um das Signal sparse darstellen zu können. Die Kohärenz der in dieser Arbeit verwendeten Wavelets ist groß zu unserer Messmatrix, weswegen sie sich für sparse Darstellung (mit \mathbf{x}) eines solchen Signals nicht eignen. Also wären für so einen Fall die globalen Transformationen besser geeignet. Sobald das Signal in einer Darstellungsbasis ausreichend sparse ist, kann es hinreichend gut mit CS rekonstruiert werden.

Wird CS auf nicht sparse oder schlecht erfasste Signale (der Messmatrix geschuldet) angewandt, so unterscheidet sich das Ergebnis wenig von einfacher Interpolation. Dabei interpolieren globale Transformationen das Signal mit wenigen periodischen Schwingungen und erzeugen bei nicht-periodischen Signalen Rauschen im Ergebnis. Die verwendeten Wavelets lösen das Signal auf unterschiedlichen Größenskalen auf. Lokal stehen ihnen nur wenige Samples als Stützstellen zur Verfügung, so dass die Interpolation schlecht und uneindeutig ausfällt.

Die DFT, CDFT und DCT haben den Vorteil, dass diese Darstellungsbasen eine geringe Kohärenz zu unserer Messmatrix besitzen. Sie mögen im Spezialfall nicht die sparseste Wahl sein, jedoch sind sie für Signale, welche mit unserer Messmatrix gut erfasst werden meist ausreichend sparse. Außerdem sind sie schnell berechenbar. Besonders geeignet sind sie für periodische Signale mit weichen Übergängen, wie sie zum Beispiel in der Elektronenmikroskopie bei der Aufnahme atomarer Strukturen vorkommen.

Tab. 3: Kohärenz verwendeter diskreter Transformationen

Basis	CDFT	DFT	DCT	HAAR	DB4
Kohärenz	$\frac{1}{\sqrt{N}}$	$\frac{1}{\sqrt{N}}$	$\sqrt{\frac{2}{N}}$	$\frac{1}{\sqrt{2}}$	$\frac{3+\sqrt{3}}{4\sqrt{2}}$

7 Einführung in die Auswertung und Vergleich der Darstellungsbasen

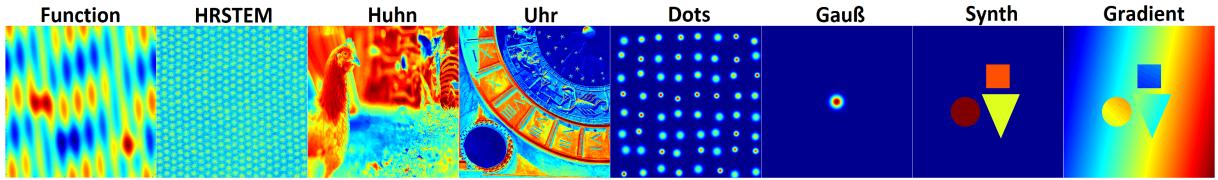


Abb. 12: Verwendete Graustufenbilder mit einer Größe von 1024x1024 Pixeln.

Um die Darstellungsbasis auszuwählen und die Algorithmen zu vergleichen, wurden die in Abbildung 12 abgebildeten Testbilder mit jeweils einer Größe von 1024x1024 Pixeln verwendet. Dabei wurden die Bilder so gewählt, dass sie unterschiedliche Merkmale von Bildern enthalten. Diese Bilder sind Graustufenbilder. Die Bilder Huhn und Uhr sind Beispiele für sehr detailreiche Bilder, Function und Dots sind Bilder mit vielen periodischen Strukturen. Gauß und Synth haben sehr lokal begrenzte Objekte, wobei Gauß weiche und Synth scharfe Kanten zwischen den Objekten und dem Hintergrund besitzen und Gradient ist ein Bild mit scharfen Kanten, weichen Übergängen und einem stark nicht-periodischen Rand. Die für CS verwendeten Transformationen sind die Fouriertransformation (DFT), eine reduzierte Fouriertransformation (CDFT), die Kosinustransformation (DCT), die Haar-Wavelets (HAAR) und die Daubechies-4-Wavelets (DB4).

Die Samples und Bilder werden vor jeder Rechnung auf einen Mittelwert von 0 und eine Varianz von 1 normiert:

$$\text{normiertes Bild} = \frac{\text{Bild} - \text{Mittelwert}}{\sqrt{\text{Varianz}}} \quad (60)$$

Dadurch werden die Bildpunkte gleichmäßig um den Nullpunkt verteilt und die Amplitude auf eine Standardabweichung von 1 normiert. Es werden auch die Samples vor der Anwendung der Algorithmen normiert und dadurch in eine einheitliche Form gebracht. Darüber hinaus bleibt der Koeffizient x_0 , der bei allen Transformationen außer der DB4 dem Mittelwert aller Bildpunkte entspricht, nahe Null, so dass die ℓ_1 -Norm der Koeffizienten durch diese Normierung verringert wird. Zur Erinnerung wird mit CS bei allen Algorithmen bis auf OMP und IHT die ℓ_1 -Norm der Koeffizienten minimiert. Außerdem lässt sich für ein so normiertes Bild die Ungenauigkeit durch Rauschen einfacher angeben. Der Bildfehler wird als Quadratische Abweichung der rekonstruierten Bildpunkte y^r von den tatsächlichen Werten y^b normiert mit der ℓ_2 -Norm des Bildes angegeben:

$$\text{Bildfehler} = \frac{\|y^b - y^r\|_2^2}{\|y^b\|_2^2} \quad (61)$$

Der Nenner entspricht durch die Normierung \sqrt{N} . Die oben genannte Gleichung für den Bildfehler ist aber grundlegender. Bei perfekter Übereinstimmung der Rekonstruktion mit dem Originalbild ist der Bildfehler Null und bei einem Signal-Rausch-Verhältnis von Eins ist der Bildfehler Eins.

7.1 Auswahl der Basis für Bilder

Zuerst wurde die Darstellung der Bilder in unterschiedlichen Basen untersucht. Dazu wurde der Bildfehler bei verschiedener Anzahl an großen Koeffizienten zur Darstellung eines Bildes aufgetragen. Am Beispiel des Bildes Huhn ist zu erkennen, dass der Bildfehler in Abbildung 13 (logarithmische Darstellung !) erst überexponentiell abnimmt, sich dann mit zunehmender Zahl an Koeffizienten weniger ändert und erst mit den letzten Koeffizienten ungleich Null abrupt bis zur Maschinengenauigkeit des Computers abfällt. Es fällt auf, dass selbst ein detailliertes Bild relativ wenige große Koeffizienten enthält. Das Bild Function kann bei kleiner bis mittlerer Genauigkeit mit weniger Koeffizienten in der DCT und DB4 dargestellt werden, denn die Kurve fällt am Anfang schneller ab. Das liegt daran, dass das Bild Function überwiegend periodisch ist.

Der Bildfehler in der CDFT und DFT nimmt Anfangs auch schnell ab, jedoch benötigen sie im weiteren Verlauf mehr Koeffizienten für einen äquivalenten Bildfehler, weil die Ränder von Function nicht periodisch sind. Trotzdem ist weder das Bild Huhn, noch Function in den verwendeten Basen sparse, was nicht optimal für CS ist. Zur Wahl der Darstellungsbasis ist besonders der Verlauf des Bildfehlers bei Darstellung mit wenigen Koeffizienten interessant, weil das Bild mit wenigen Koeffizienten rekonstruiert werden soll. Der Bildfehler ist bei Darstellung von Function mit wenigen Koeffizienten deutlich kleiner, so dass dieses Bild mit CS potenziell besser rekonstruiert werden könnte.

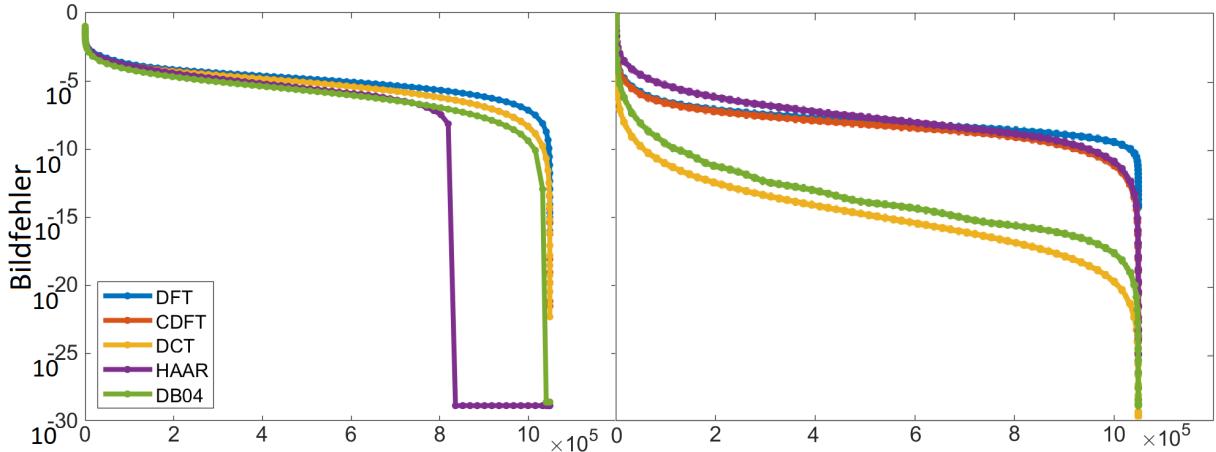


Abb. 13: Bildfehler gegen die Anzahl grösster Koeffizienten für das Bild Huhn (links) und Function (rechts) logarithmisch aufgetragen: Beide Bilder sind grob mit wenigen Koeffizienten darstellbar, zur exakten Darstellung sind allerdings alle nicht Null Koeffizienten notwendig. Beim Bild Function ist die Darstellungsgenauigkeit mit wenigen Koeffizienten etwas besser.

In Abbildung 14 ist die Anzahl der Koeffizienten für die Darstellung der Testbilder in den unterschiedlichen Basen mit einem Bildfehler von 1% ($1\% = 0,01$) abgebildet (beachte Skalierung). Ziel war es zu untersuchen, ob die Bilder mit leichten Fehlern sparse in den Basen darstellbar sind. Die Kompression erfolgte durch Vernachlässigen der Koeffizienten geringer Amplitude. Es ist deutlich zu erkennen, dass Bilder mit lokalen Details und sonst einer einheitlichen Intensität (Gauß) oder auch mit scharfen Kanten (Synth, Gradient) sehr gut mit den Wavelets darstellbar sind. Hingegen sind Bilder mit weichen Übergängen (Gauß), periodischen Strukturen (Function) und vielen periodisch angeordneten Objekten (Dots) besser in den globalen Transformationen darstellbar, wobei bei stark unstetigen Rändern (Gradient) die DCT der DFT vorzuziehen ist. Dabei enthalten Gauß und Gradient unterschiedliche Merkmale, welche jeweils in den lokalen und globalen Transformationen besser darstellbar sind, so dass hier unterschiedliche Transformationen ähnliche Resultate erzielen. Bilder mit vielen Details (Huhn, Uhr) lassen sich am besten mit den DB4 darstellen und Bilder mit starkem Rauschen (HRSTEM) sind in allen vorliegenden Basen sehr schlecht darstellbar. Es ist auffällig, dass die Bilder Gauß, Gradient, Function und Dots eine um mehrere Größenordnungen geringere Anzahl an Koeffizienten in den meisten Transformationen zur Darstellung mit einem Fehler von 1 % erfordern. Außerdem ist die CDFT immer sparser als die DFT, selbst wenn man außer acht lässt, dass die CDFT reelle statt komplexe Koeffizienten enthält. In der CDFT sind die Koeffizienten unabhängig voneinander, wohingegen bei der DFT immer ein Koeffizienten und sein komplex konjugiertes zusammen ein reelles Signal darstellen. Weil die Koeffizienten in der DFT komplex sind, enthalten zwei Koeffizienten in der DFT die gleiche Information, wie zwei Koeffizienten in der CDFT. Wenn allerdings nur der reelle oder komplexe Teil eines Koeffizienten in der DFT groß ist und der andere Teil wenig zur Darstellung beiträgt, so ist in der CDFT nur ein Koeffizient erforderlich, wohingegen die DFT immer noch zwei Koeffizienten erfordert. Dadurch ist die Darstellung in der CDFT immer sparser, als in der DFT.

Es gibt keine universale Basis für die Darstellung eines Bilds, sondern es muss für jedes Bild die passende Basis nach seinen Besonderheiten gewählt werden. Wichtig ist, dass Bilder mit weichen Übergängen und periodischen Strukturen (zum Beispiel bei Function und Dots), wie sie in der Elektronenmikroskopie bei Aufnahme atomarer Strukturen vorkommen, sich gut in der CDFT oder DCT darstellen lassen.

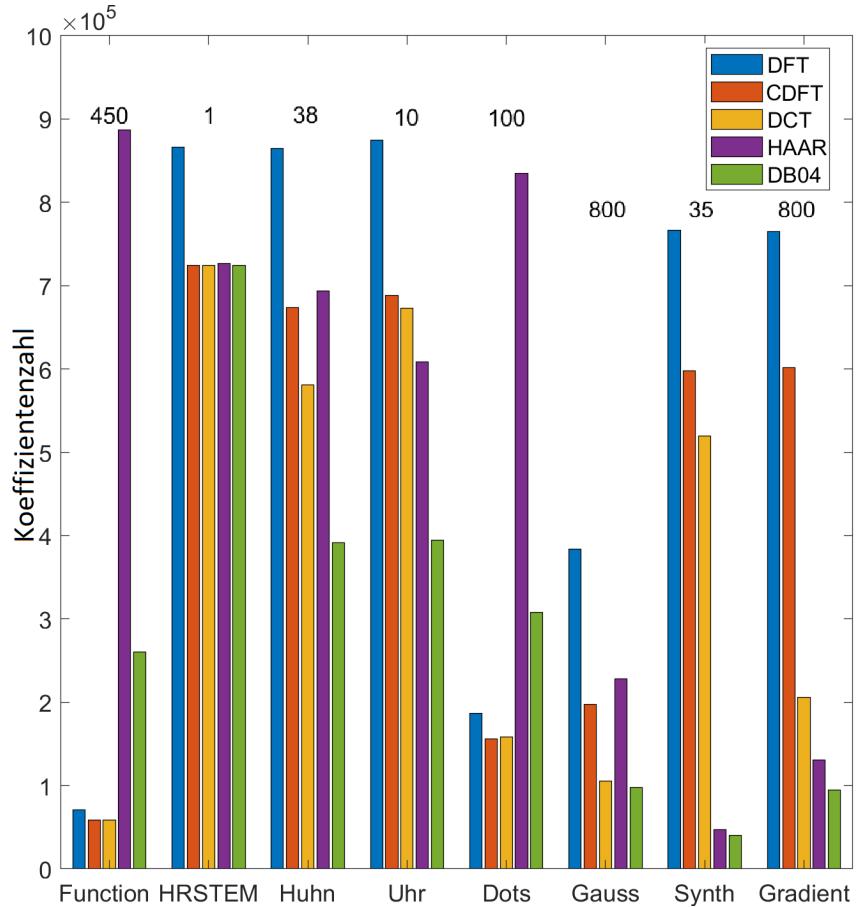


Abb. 14: Zahl der größten Koeffizienten der Bilder bei Kompression mit 1% Bildfehler in verschiedenen Basen (die Balken für jedes Bild sind um den Wert über den Balken zu teilen): Unterschiedliche Bilder sind in verschiedenen Basen besser darstellbar. Es eignen sich die DCT, DFT, CDFT für die Darstellung periodischer Strukturen und weicher Übergänge und die HAAR und DB4 für die Darstellung lokaler Details und scharfer Kanten.

7.2 Rekonstruktion in unterschiedlichen Basen

Als nächstes wurde eine Rekonstruktion der Bilder Huhn, Function und Gradient mit dem Algorithmus NESTA in allen Darstellungsbassen aus 10% Samples ausprobiert. Die Ergebnisse sind in Tabelle 4 abgebildet.

Tab. 4: Rekonstruktionsfehler einiger Bilder mit NESTA in verschiedenen Basen und Interpolation

Bild	CDFT	DFT	DCT	HAAR	DB4	Interpolation
Huhn	$3.0 \cdot 10^{-2}$	$2.5 \cdot 10^{-2}$	$2.7 \cdot 10^{-2}$	$5.9 \cdot 10^{-2}$	$5.5 \cdot 10^{-2}$	$1.6 \cdot 10^{-2}$
Function	$6.8 \cdot 10^{-5}$	$3.8 \cdot 10^{-5}$	$1.6 \cdot 10^{-7}$	$1.6 \cdot 10^{-2}$	$2.6 \cdot 10^{-3}$	$2.5 \cdot 10^{-3}$
Gradient	$2.3 \cdot 10^{-3}$	$1.8 \cdot 10^{-3}$	$2.0 \cdot 10^{-3}$	$5.8 \cdot 10^{-3}$	$2.0 \cdot 10^{-2}$	$3.0 \cdot 10^{-3}$

Zusätzlich zu CS wurde eine Interpolation der Samples mit vier Verfahren zum Vergleich herangezogen. Die Verfahren waren lineare und quadratische Interpolation, Interpolation mit nächstem Nachbarn und die Interpolation durch natürliche nächste Nachbarn. Das zuletzt aufgeführte Verfahren erzeugte marginal bessere Ergebnisse, so dass dessen Ergebnisse in Tabelle 4 aufgeführt sind. Die Ergebnisse dieser Untersuchung unterstützen die theoretischen Überlegungen aus Kapitel 6.4 zur „Kohärenz zwischen Mess- und Darstellungsmatrix“. Weil das Bild Huhn verhältnismäßig viele Koeffizienten in allen Basen benötigt, ist CS schlecht anwendbar. Die Rekonstruktionsgenauigkeit mit CS war sogar schlechter als Interpolation. Im Vergleich dazu ist das Bild Function mit wenigen Koeffizienten in der DCT (130 Koeffizienten mit einem Fehler von 1%) darstellbar. Die Rekonstruktion des Bildes mit CS in der DCT ergab deswegen ein deutlich besseres Ergebnis als Interpolation. Die Rekonstruktion in der DFT war, wegen nicht-periodischen Rändern, leicht schlechter und in den Wavelets, wegen einer geringeren Sparsity noch, deutlich schlechter. Der Einfluss der Sparsity des Bildes auf die Genauigkeit der Rekonstruktion lässt sich auch sehr gut an den beiden Ausschnitten des originalen Bildes und der Rekonstruktion für das Bild Huhn in Abbildung 15 und Bild Function in Abbildung 28 wiedererkennen.

Das Bild Gradient war in den DB4 (122 Koeffizienten bei 1% Kompressionsfehler) am sparsesten, die HAAR, DCT und DFT waren in dieser Reihenfolge weniger sparse. Dabei ist die Rekonstruktionsgenauigkeit in den Wavelets trotz ihrer Sparsity schlechter, als mit globalen Transformationen. Das Bild Gradient hat neben dem weichen Übergang im Hintergrund auch scharfe Kanten an den geometrischen Körpern in der Mitte des Bildes. Durch Samples, welche zufällig entnommenen Bildpunkten entsprechen, wird die Position der Kanten schlecht erfasst, so dass wenig Information für eine Rekonstruktion dieser Kanten vorliegt. Globale Transformationen passen die Samples mit weichen, periodischen Funktionen an. Dadurch wird zwar der Hintergrund gut rekonstruiert, die Kanten der Körper aber nicht. Die Wavelets hätten einen Vorteil zur Darstellung der scharfen Kanten, können diesen Vorteil aber nicht umsetzen, weil die Samples wenig Information über die genaue Lage dieser Kanten enthalten.

Somit können lokale Objekte, besonders solche mit scharfen Übergängen/Kanten, besser in den Wavelets dargestellt werden, aber unsere Messmatrix eignet sich nicht zum genau erfassen deren Position, so dass die Rekonstruktion mit CS keine guten Ergebnisse liefert. Wird in so einem Fall CS verwendet, do ist dass Ergebnis nur eine ungenaue Interpolation der Samples, weil die Sparsity der Koeffizienten nicht ausgenutzt werden kann. In so einem Fall kann das Bild entweder mit Interpolation, wie beim Bild Huhn, besser rekonstruiert werden, oder es enthält, wie beim Bild Gradient, einen periodischen Hintergrund, der einen großen Teil des Bildes ausmacht und mit CS in den globalen Transformationen besser dargestellt werden kann.

Ein Bild mit weichen, periodischen Übergängen kann gut mit unserer Messmatrix erfasst werden, ist aber wegen der hohen Kohärenz der Wavelets zur Messmatrix nicht sparse in den Wavelets darstellbar. Ein in den Wavelets sparses Bild kann aber sehr schlecht von der Messmatrix erfasst werden. Somit eignen sich die globalen Transformationen in jeder Hinsicht besser für CS mit unserer Messmatrix.

Aus diesem Grund und weil die DFT etwas schneller als die DCT zu berechnen ist, wurde für Rekonstruktion der Bilder die reelle Version der DFT, die CDFT bei der weiteren Analyse der Rekonstruktionsalgorithmen verwendet. Die Rekonstruktion von eindimensionalen Signalen wurde in der DCT gemacht, weil zu diesem Zeitpunkt noch keine CDFT vorlag. In der Anwendung kann die DCT für Bilder mit nicht-periodischem Rand durchaus leicht bessere Ergebnisse, als die CDFT, erzielen.

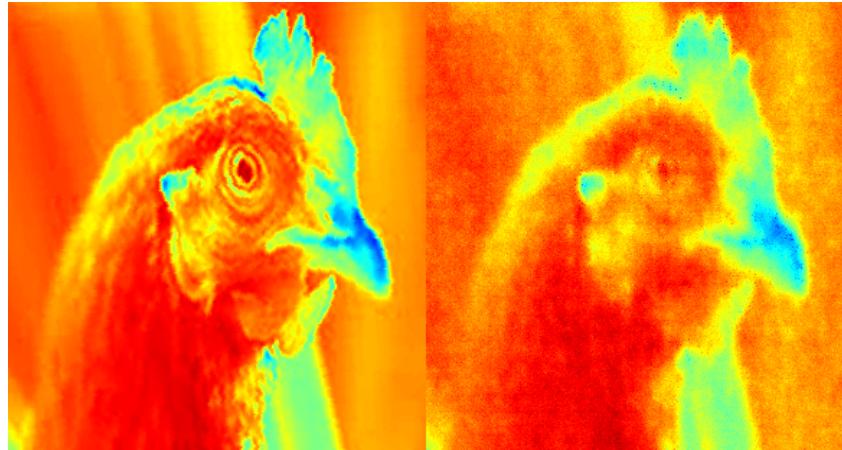


Abb. 15: Ausschnitt aus dem originalen (links) und mit NESTA in der CDFT rekonstruiertem Bild Huhn (rechts): Die Rekonstruktion in der CDFT liefert ein Bild mit weichen Übergängen, was in diesem Fall Rauschen erzeugt. Das Bild ist nicht sparse in der CDFT und wird deswegen mit einem größeren Fehler rekonstruiert.

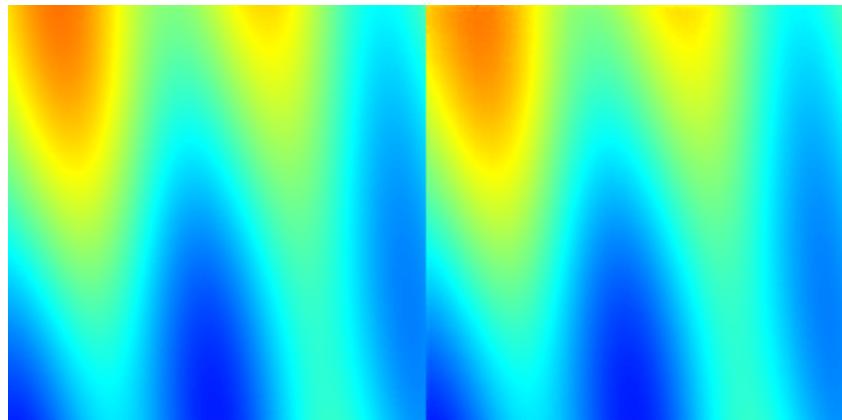


Abb. 16: Ausschnitt aus dem originalen (links) und mit NESTA in der CDFT rekonstruiertem Bild Function (rechts): Das Bild kann in der CDFT sehr gut rekonstruiert werden. Nur am oberen und linken Rand sind leichte Rekonstruktionsfehler, geschuldet einem nichtperiodischen Rand des Bildes, vorhanden.

8 Rekonstruktion

8.1 Rekonstruktion von Signalen (1D)

Zum Testen verschiedener Algorithmen für CS wurde anfangs die Rekonstruktion zweier eindimensionaler Signale in der DCT durchgeführt. Die DCT wurde verwendet, weil nicht alle Algorithmen komplexe Zahlen unterstützen und die CDFT zum Zeitpunkt der Tests noch nicht implementiert war. Die verwendeten zwei Testsignale sind in Bild 17 dargestellt.

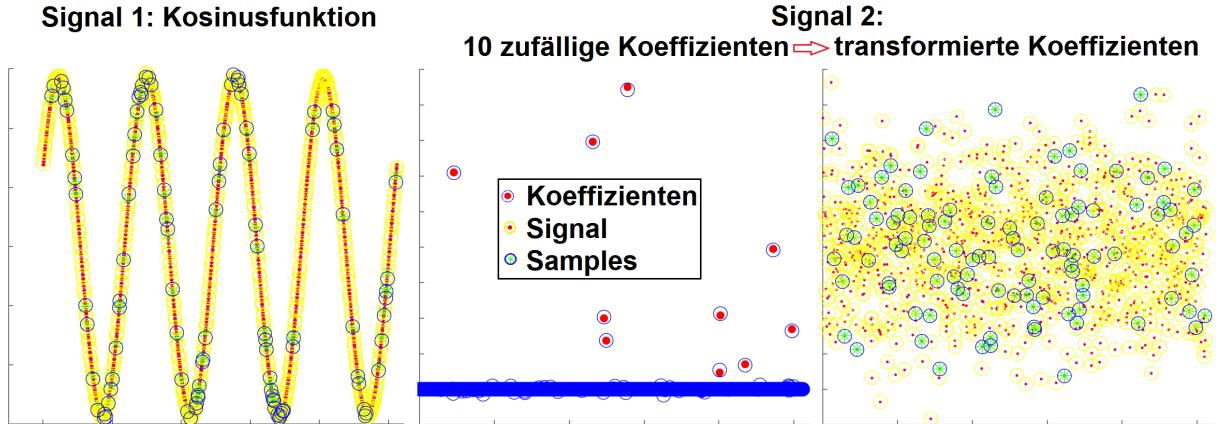


Abb. 17: 1D-Testsignale: links eine Kosinusfunktion (Signal 1) und mittig 10 zufällige Koeffizienten, welche mit DCT zurück transformiert das Signal (Signal 2) rechts ergeben.

Das erste Signal ist eine Kosinusfunktion. Diese ist nicht sparse in der DCT, weil die Funktion im Vergleich zu den DCT-Schwingungen verschoben ist. Das zweite Signal besteht aus 10 ausgewählten Koeffizienten welche mit der DCT in ein Signal zurück transformiert wurden. Somit ist das zweite Signal sparse in der DCT.

Für die Rekonstruktion verwendete Algorithmen sind Orthogonal Matching Pursuit (OMP), Iterative Hard Thresholding (IHT), die SPGL1 Löser für Basis Pursuit (spg bp), Basis Pursuit Denoising (spg bpdn) und Lasso Pursuit (spg lasso), die ℓ_1 -Magic Löser für Basis Pursuit (l1eq pd) und für Basis Pursuit Denoising (l1qc logbarrier), die Löser NESTA und FPC_AS und Matlab-Löser linprog und lsqlin zur Lösung linearer bzw. quadratischer Optimierungsprobleme. Der Rekonstruktionsfehler der Signale mit diesen Algorithmen ist in Abbildung 18 dargestellt.

Weil das zweite Signals sparser in der DCT ist, ist dessen Rekonstruktionsgenauigkeit genauer. Die beiden selbstgeschriebenen Löser OMP und IHT wurden so eingestellt, dass sie die Samples sehr genau anpassen. Deswegen wird das nicht sparse Signal mit mehr Koeffizienten angepasst, was in diesem Fall eine bessere Rekonstruktionsgenauigkeit ergibt. Dies kann allerdings für ein anderes Signal von Nachteil sein. Die anderen Löser, mit Ausnahme der Löser für Lasso Pursuit, haben etwas schlechtere, aber vergleichbare Ergebnisse.

Die meisten Löser konnten das zweite Signal sehr genau rekonstruieren. Die genauesten Löser waren OMP (Bildfehler: $7 \cdot 10^{-32}$) und linprog (Bildfehler: $3 \cdot 10^{-24}$) und bis auf FPC_AS (Bildfehler: $1 \cdot 10^{-4}$) und l1qc logbarrier ($1 \cdot 10^{-4}$) und die Löser für Lasso Pursuit hatten alle Löser einen geringeren Bildfehler als $6 \cdot 10^{-6}$. Die Löser für den Lasso Pursuit (spg lasso & lsqlin) schneiden für beide Signale besonders schlecht ab, weil der Parameter τ des Lasso Pursuit, welcher zwischen Anpassungsgenauigkeit der Samples und der Sparsity der Koeffizienten abwägt, schwierig abzuschätzen ist.

Die meisten Algorithmen laufen in unter einer Sekunde mit den Ausnahmen von IHT (3s), l1qc logbarrier (3s) und lsqlin (20s). Der wichtigste Unterschied zwischen den Algorithmen ist, dass linprog und lsqlin explizite Matrizen für die Transformation benötigen, wohingegen für die restlichen Algorithmen die Transformation mit Funktionen schneller und ohne explizite Speicherung der Transformationsmatrizen berechnet werden kann. Für ein Bild der Größe 1024x1024 Pixel wovon 6,25% des Bildes als Samples bekannt sind, besitzt die Matrix \mathbf{A} 2^{36} Elemente.

Bei einer Speicherung dieser Elemente mit einfacher Genauigkeit (4 Byte) bräuchte man allein für die Matrix $2^{38} \times 2^{38} \approx 256$ GB RAM. Für große Signale, wie es bei Bildern der Fall ist, ist es nicht sinnvoll die Matrix explizit zu speichern.

Aus diesen Gründen sind die Löser spg lasso, lsqlin und linprog nicht für die Rekonstruktion von Bildern geeignet und werden im weiteren nicht mehr betrachtet.

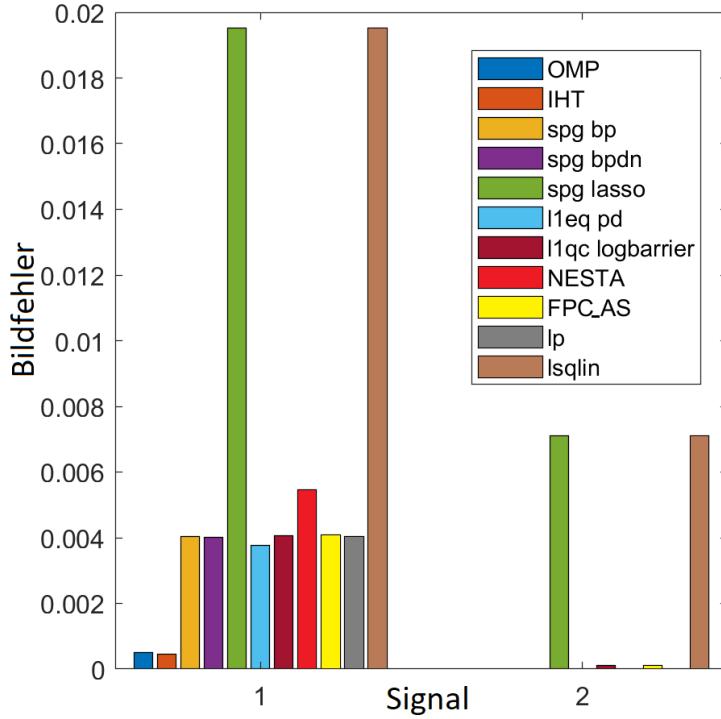


Abb. 18: Rekonstruktionsfehler mit CS für die beiden eindimensionalen Signale mit allen verwendeten Algorithmen: Löser für Lasso Pursuit schneiden besonders schlecht ab. Das erste Signal ist weniger sparsen, so dass es ungenauer rekonstruiert wird. Die beiden Löser OMP und IHT passen die Samples sehr genau an, was in diesem Fall für Signal eins eine bessere Rekonstruktion ergibt.

8.2 Rekonstruktion von Bildern (2D)

Als nächstes wurde der Satz von Testbildern aus 10% zufällig ausgewählten Samples in der CDFT rekonstruiert. Der Bildfehler der Rekonstruktion ist in Abbildung 19 abgebildet.

Die selbst implementierten Löser OMP und IHT beenden die Rechnung vorläufig nach einer maximalen Zahl an Iterationen, damit ihre Laufzeit in angemessenem Rahmen bleibt. Der Löser l1eq pd tut dies auch, weil er nach einer exakten Anpassung der Samples sucht und dabei nicht konvergiert. Deswegen werden die Samples mit diesen Algorithmen schlecht getroffen und der Rekonstruktionsfehler ist verglichen mit den anderen Algorithmen meist deutlich schlechter. Trotzdem brauchen diese drei Algorithmen, so wie auch l1qc logbarrier, ziemlich lange um ein Ergebnis zu berechnen (siehe Abbildung 20). Dabei ist für den implementierten IHT die Schätzung der Sparsity sehr schlecht, so dass die Ergebnisse nicht sparse und ungenau ausfallen. Für OMP ist es denkbar, gute Ergebnisse zu erzielen. Das Problem mit diesem Algorithmus ist, dass bei jeder Iteration ein lineares Gleichungssystem für alle gefundenen Koeffizienten gelöst werden muss. Dadurch nimmt seine Laufzeit für größere und nicht sparse Bilder sehr stark zu, womit sich dieser Algorithmus für größere Bilder nicht eignet.

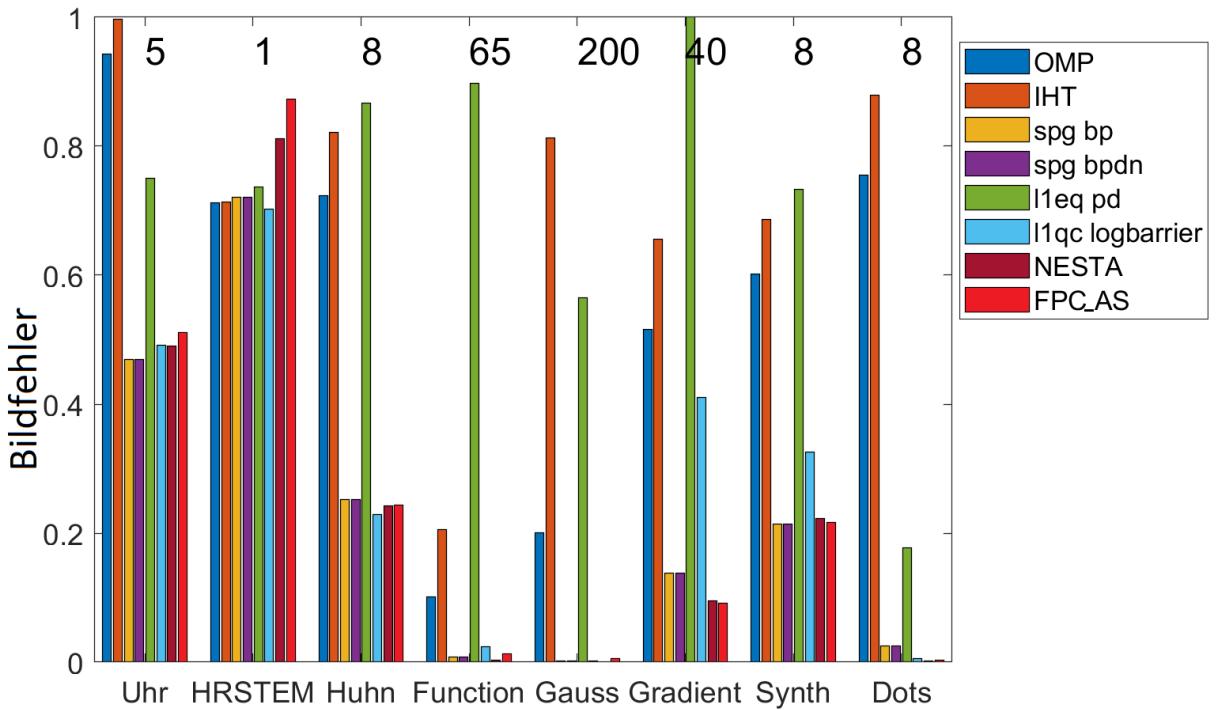


Abb. 19: Bildfehler für die Testbilder rekonstruiert mit unterschiedlichen Algorithmen (die Balken für jedes Bild sind um den Wert über den Balken zu teilen): IHT, OMP und l1ep pd schneiden besonders schlecht ab. Nur die Bilder Function, Gauss, Gradient und Dots wurden akzeptabel rekonstruiert, weil sie deutlich sparser in der CDFT sind.

Die Genauigkeit und Rechenzeit von Nesta, FPC_AS und den beiden Lösern von SPGL1 (spg bp und spg bpdn) fällt am besten aus. Bei den Bildern Function, Gauss, Gradient und Dots ist die Rekonstruktion genauer als 0.5%. Bei den anderen Bildern ist sie allerdings größer als 2%, weil diese Bilder weniger Sparse sind. Diese drei Algorithmen werden im weiteren Schritt genauer getestet. Dabei wird von den SPGL1-Lösern nur spg bpdn verwendet, weil dieser Löser das Basis Pursuit Denoising löst und demnach besser mit Rauschen umgehen sollte.

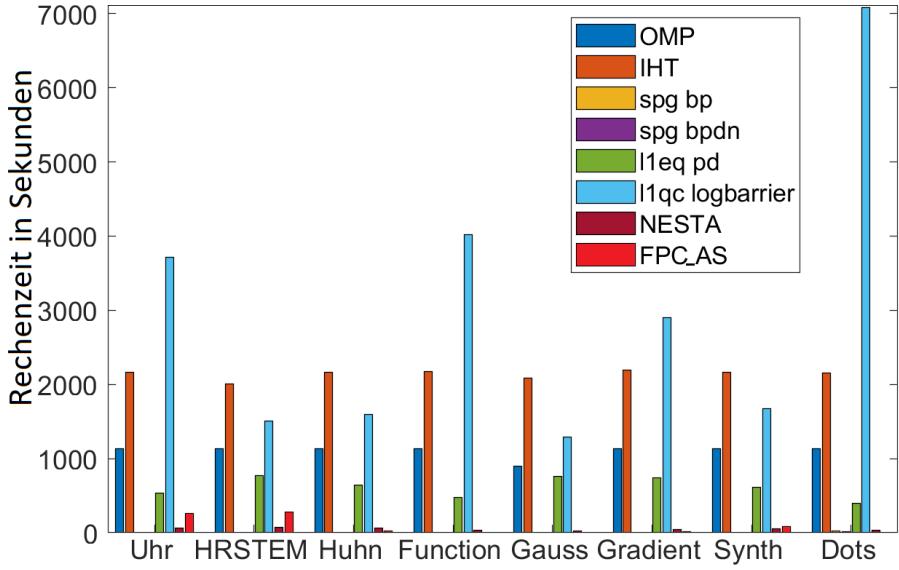


Abb. 20: Rechenzeit für die Rekonstruktion der Testbilder mit unterschiedlichen Algorithmen: OMP,IHT, l1eq pd und l1qc logbarrier brauchen besonders lange für die Rekonstruktion.

8.3 Rekonstruktion bei unterschiedlicher Anzahl an Samples

Um den Einfluss der Anzahl an Samples auf die Rekonstruktionsgenauigkeit zu testen, wurden die Bilder Function und Huhn mit den drei ausgewählten, schnellen Algorithmen rekonstruiert. Es wurde dabei die Anzahl an Samples variiert. Die Ergebnisse sind in Abbildung 21 und 22 dargestellt.

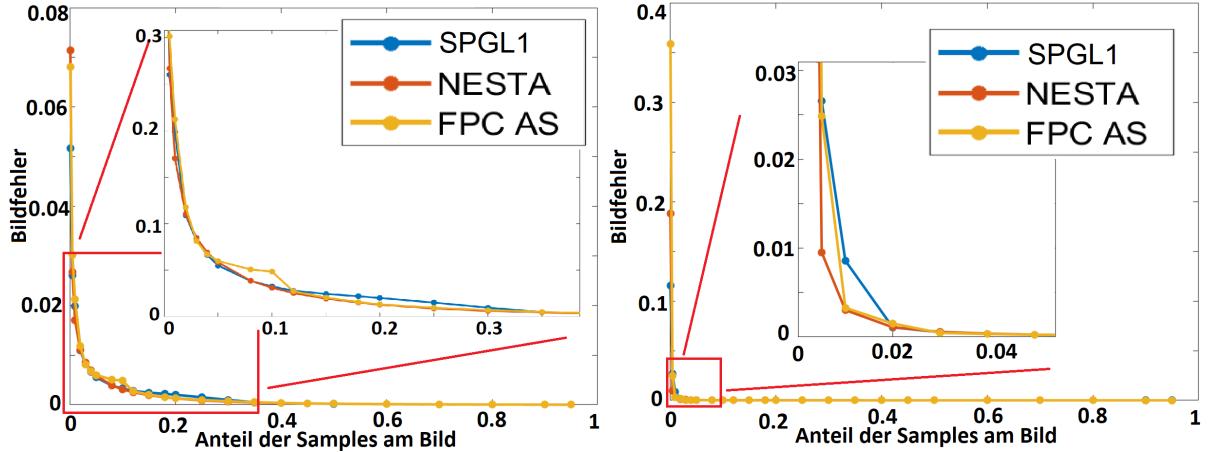


Abb. 21: Rekonstruktionsfehler bei verschiedener Anzahl an Samples für das Bild Huhn: der Bildfehler nimmt ungefähr exponentiell ab, FPC_AS hat fehler nimmt ungefähr exponentiell ab, SPGL1 hat für 5% bis 10% Samples einen Ausreißer nach oben. bei 1% Samples einen Ausreißer nach oben.
 Abb. 22: Rekonstruktionsfehler bei verschiedener Anzahl an Samples für das Bild Function: der Bildfehler nimmt ungefähr exponentiell ab, FPC_AS hat fehler nimmt ungefähr exponentiell ab, SPGL1 hat für 5% bis 10% Samples einen Ausreißer nach oben. bei 1% Samples einen Ausreißer nach oben.

Mit der Anzahl der Samples nimmt die Rekonstruktionsgenauigkeit bei wenigen Samples rapide zu. Bei einer höheren Anzahl an Samples bringen weitere Samples wenig Informationen dazu, so dass die Rekonstruktionsgenauigkeit kaum zunimmt. Bei tatsächlich sparsen Bildern wäre ein Optimum an Samples zu erwarten, ab dem die Rekonstruktionsgenauigkeit mit mehr Samples nicht mehr weiter zunimmt. In unserem Fall mit nicht sparsen Bildern gibt es kein ausgeprägtes Optimum. Trotzdem kann eine Grenze festgelegt werden, ab der die Genauigkeit der Rekonstruktion nicht mehr signifikant zunimmt.

Bei der Rekonstruktion ist der Algorithmus Nesta am zuverlässigsten. Vergleicht man die beiden Abbildungen, so nimmt der Bildfehler der Rekonstruktion für das Bild Function deutlich steiler ab und liegt auch signifikant unter dem Bildfehler für das Bild Huhn.

Das liegt daran, dass das Bild Function deutlich weniger Koeffizienten für eine gute Darstellung des Bildes in der CDFT benötigt. Fazit ist, dass die Rekonstruktionsgenauigkeit mit der Anzahl an Samples stark zunimmt, so dass eine Anzahl an Samples gewählt werden kann, um eine ausreichend gute Anpassung des tatsächlichen Bildes zu erreichen. Diese ist im Voraus allerdings nicht bekannt. Dabei benötigen sparse Bilder deutlich weniger Samples für eine gute Rekonstruktion. Von allen Algorithmen scheint Nesta etwas zuverlässiger zu sein.

8.4 Untersuchung zur Skalierung der Laufzeit der Algorithmen

Zusätzlich wurden verschiedene Ausschnitte der beiden Bilder Huhn und Function rekonstruiert, um die Skalierung der Rechenzeit in Abhängigkeit von der Bildgröße zu bestimmen. Alle drei Algorithmen skalieren für beide Bilder ungefähr linear mit der Anzahl an Pixeln im Bild, wobei Nesta der langsamere Algorithmus davon ist. In Abbildung 23 sind die Ergebnisse für das Bild Huhn dargestellt. Es sollte laut Publikation möglich sein Nesta weiter zu beschleunigen, in dem die Transformationsmatrix \mathbf{A} zerlegt wird. Dies wurde allerdings nicht getestet.

Beim Untersuchen der Rechenzeit ist es wichtig, dass diese Algorithmen aus vielen Teilschritten bestehen. Die Rechenzeit hängt von der Genauigkeit der Anpassung, bei der diese Algorithmen ihre Rechnung beenden. Ein weiterer Faktor ist der Anzahl der Fortsetzungen (Wiederholungen der Algorithmen mit neuem Startwert und neuen Parametern). Der überwiegende Anteil der Rechenzeit beläuft sich bei diesen Algorithmen auf die Hin- und Rücktransformation mit der Matrix \mathbf{A} , so dass eine schnelle Transformation wichtig ist.

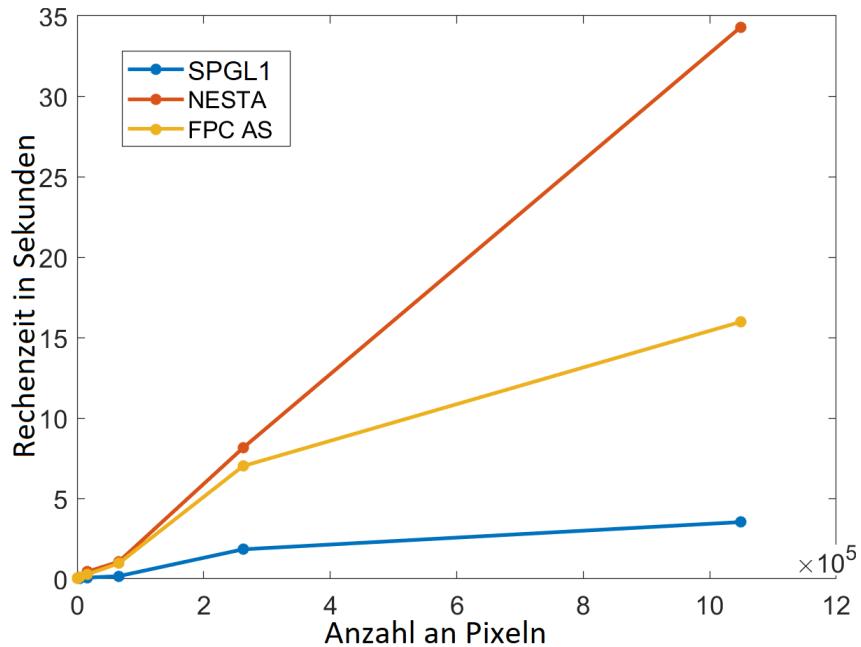


Abb. 23: Rechenzeit für die Rekonstruktion unterschiedlich großer Ausschnitte des Bildes Huhn: Die Rechenzeit skaliert ungefähr proportional zur Pixelanzahl. Dabei ist Nesta langsamer und SPGL1 schneller als FPC_AS.

9 Rekonstruktion von HAADF-Simulationen

9.1 Rekonstruktion von simulierten HAADF-Aufnahmen mit und ohne Rauschen

Zum Testen der Rekonstruktionsgenauigkeit von STEM-Bildern wurden Multislice-Simulationen (ohne Frozen-Lattice) einer Probe aus Silizium, mit Siliziumgermanium entlang zweier Kanäle, verwendet. Es gab zwei Version der Simulation, eine für die „Probe“ des Mikroskops Titan 30-800 und die andere für eine aberrationskorrigierte „Probe“. Ein horizontaler Streifen des simulierten Bildes ist in Abbildung 24 abgebildet. Das gesamte Bild hat eine Größe von 523x3016 Pixel.



Abb. 24: Horizontaler Ausschnitt der Multislice-Simulation einer Siliziumprobe mit Siliziumgermanium entlang zweier Kanäle, aufgenommen in $[1\ 1\ 0]$ -Richtung ohne Aberrationskorrektur der „Probe“. Es lassen sich deutlich die Kanäle mit Siliziumgermanium erkennen, wobei im linken Kanal die Germaniumkonzentration einen weichen Übergang hat und im rechten Kanal die Germaniumkonzentration abrupt abnimmt.

Für die beiden Bilder wurde mit Poisson-Rauschen unterschiedliche Elektronenintensität ähnlich einer realen Aufnahme simuliert. Bei der Angabe der Elektronenintensität in Elektronen pro Quadratnanometer (e^-/nm^2) handelt es sich um einen Relativwert, wie viele Elektronen bei einer Intensität von Eins am HAADF-Detektor ankommen. Je größer dieser Wert, um so geringer das Rauschen. Zur Veranschaulichung sind Ausschnitte der Simulation mit unterschiedlichem Rauschen in Abbildung 25 und 26 abgebildet.

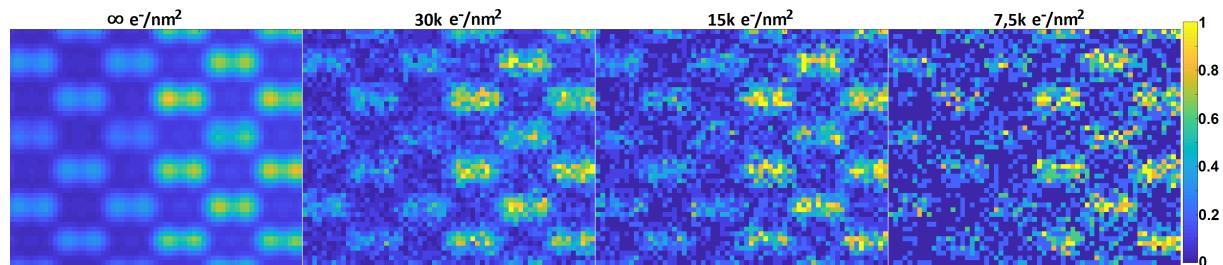


Abb. 25: Ausschnitt aus der Stemsim-Simulation mit nicht aberrationskorrigierter „Probe“. bei Aufnahme mit unterschiedlicher Elektronenintensität.

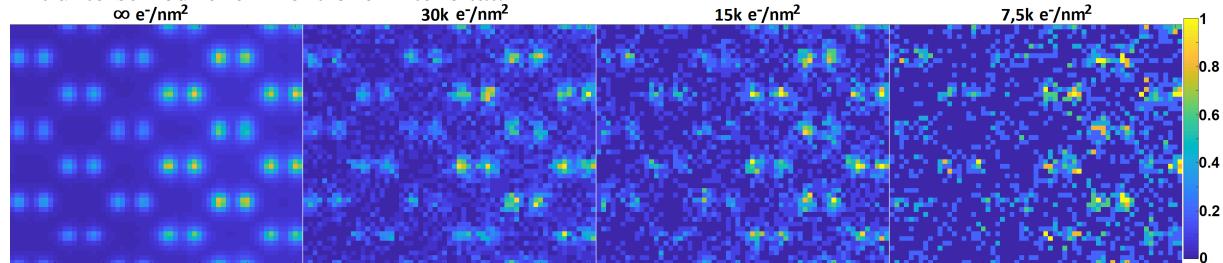


Abb. 26: Ausschnitt aus der Stemsim-Simulation mit aberrationskorrigierter „Probe“. bei Aufnahme mit unterschiedlicher Elektronenintensität.

Diese Bilder wurden mit den drei ausgewählten Rekonstruktionsalgorithmen und mit OMP aus 10%, 25% und 50% Samples rekonstruiert. Die Rekonstruktion mit Rauschen war deutlich ungenauer, als ohne Rauschen. In Abbildung 27 ist der Bildfehler für eine Rekonstruktion aus 25% Samples dargestellt und alle Rekonstruktionsfehler sind in Tabelle 5 wiedergegeben.

SPGL1 hatte große Probleme mit Rauschen, aber auch Nesta und FPC_AS erzielten mit Rauschen deutlich schlechtere Ergebnisse. Trotz der scheinbaren Periodizität des Bildes reichten 200 mit OMP bestimmte Koeffizienten nicht aus, um das Bild gut darzustellen, so dass das Gibbs-Phänomen an den Rändern der Siliziumgermanium-Kanäle auftrat. Mehr Koeffizienten mit OMP zu bestimmen ist nicht praktisch, weil bei jeder Iteration ein lineares Gleichungssystem für alle gefundenen Koeffizienten gelöst werden muss, was den Algorithmus extrem langsam macht.

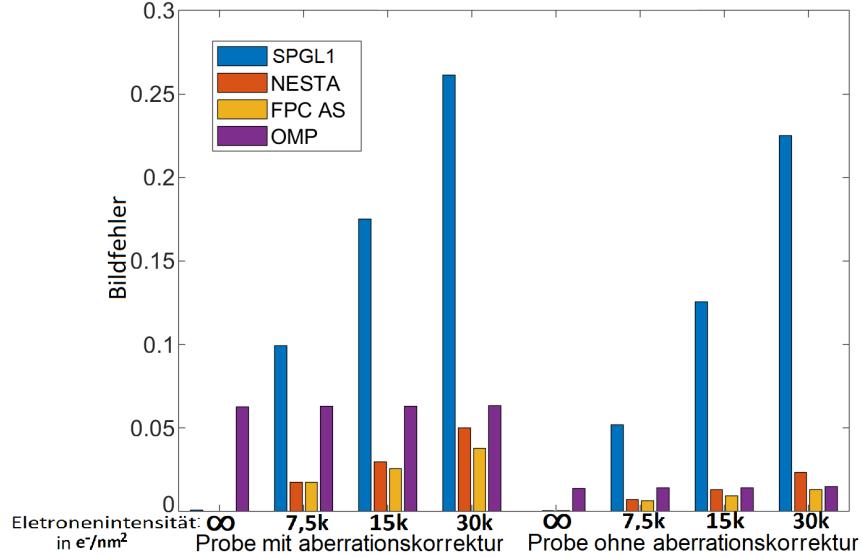


Abb. 27: Rekonstruktionsgenauigkeit aus 25% Samples für beide Multislice-Simulationen der Probe bei unterschiedlichem Rauschen: Die Rekonstruktionsgenauigkeit nimmt mit erhöhtem Rauschen deutlich ab. SPGL1 und OMP schneiden besonders schlecht ab.

Tab. 5: Bildfehler der Rekonstruktion der Multislice-Simulation bei unterschiedlich starkem Rauschen angegeben in Prozent

Algorithmus	„Probe“ mit Aberrationskorrektur				„Probe“ ohne Aberrationskorrektur			
	$\infty e^-/\text{nm}^2$	$30k e^-/\text{nm}^2$	$15k e^-/\text{nm}^2$	$7,5k e^-/\text{nm}^2$	$\infty e^-/\text{nm}^2$	$30k e^-/\text{nm}^2$	$15k e^-/\text{nm}^2$	$7,5k e^-/\text{nm}^2$
Rekonstruktion aus 10% Samples								
SPGL 1	0.21%	8.27%	13.91%	22.92%	$5.15 \cdot 10^{-2}\%$	4.01%	8.75%	16.49%
NESTA	0.13%	3.16%	4.73%	6.89%	$6.24 \cdot 10^{-5}\%$	1.14%	1.88%	3.14%
FPC_AS	0.12%	4.36%	6.10%	8.37%	$5.19 \cdot 10^{-5}\%$	1.52%	2.26%	3.33%
OMP	6.28%	6.33%	6.41%	6.48%	1.41%	1.44%	1.48%	1.71%
Rekonstruktion aus 25% Samples								
SPGL1	$7.86 \cdot 10^{-2}\%$	9.91%	17.52%	26.12%	$7.84 \cdot 10^{-5}\%$	5.21%	12.57%	22.49%
NESTA	$1.11 \cdot 10^{-5}\%$	1.77%	2.98%	5.02%	$7.09 \cdot 10^{-8}\%$	0.71%	1.31%	2.35%
FPC_AS	$1.43 \cdot 10^{-5}\%$	1.74%	2.56%	3.78%	$7.81 \cdot 10^{-8}\%$	0.66%	0.93%	1.32%
OMP	6.27%	6.29%	6.32%	6.35%	1.40%	1.43%	1.43%	1.48%
Rekonstruktion aus 50% Samples								
SPGL 1	$1.15 \cdot 10^{-2}\%$	10.92%	20.55%	42.14%	$1.32 \cdot 10^{-5}\%$	8.48%	14.53%	34.89%
NESTA	$1.06 \cdot 10^{-8}\%$	1.22%	2.28%	4.16%	$4.62 \cdot 10^{-9}\%$	0.51%	1.10%	2.24%
FPC_AS	$7.54 \cdot 10^{-9}\%$	1.23%	1.82%	2.81%	$1.11 \cdot 10^{-9}\%$	0.57%	0.86%	1.34%
OMP	6.26%	6.28%	6.28%	6.30%	1.40%	1.41%	1.41%	1.43%
nur verrauscht ohne Rekonstruktion								
vrrauscht	0%	21.85%	43.78%	87.46%	0%	15.53%	31.06%	62.02%

Das Rauschen wurde als Parameter σ , welcher dem Bildfehler entspricht, angegeben. Dieser Parameter multipliziert mit der Wurzel aus der Anzahl der Samples \sqrt{D} entspricht in guter Näherung dem Parameter σ des Basis Pursuit Denoising, weil die Samples vor jeder Rechnung ähnlich zum Bild normiert werden. Nach der Rekonstruktion wird die Normierung rückgängig gemacht. Dabei sind die beiden Algorithmen bei hohem Rauschen instabil. Nesta stürzt manchmal ab einem Rauschen von $\sigma = 0.6$ und SPGL1 ab einem Rauschen von $\sigma = 0.2$ ab. Für FPC_AS ist der Parameter fürs Rauschen nicht richtig dokumentiert. Durch Ausprobieren wurde festgestellt, dass σ unverändert übergeben, ordentliche Ergebnisse liefert, welche verglichen mit den anderen Lösern meist sogar etwas besser ausfallen. Der verwendete Parameter σ für die Rekonstruktion mit Rauschen wurde für FPC_AS in der Nähe des tatsächlichen Bildfehlers ($1 \cdot 10^{-5}; 0.54; 0.7; 0.85$) festgelegt und für NESTA auf ($10^{-5}; 0.5; 0.6; 0.7$) und für SPGL1 auf ($10^{-5}; 0.15; 0.2; 0.2$) reduziert, damit diese Algorithmen stabil laufen. OMP dagegen wurde solange ausgeführt, bis 200 Koeffizienten gefunden wurden, wobei diese für eine ordentliche Rekonstruktion nicht ausgereicht haben.

Anschließend wurde die mittlere Intensität der Atome parallel zu den Siliziumgermanium-Kanälen mit dem Programm Image-Eval ausgewertet. Als Beispiel ist das Intensitätsprofil für die Rekonstruktion mit NESTA in Abbildung 30 gezeigt. Dabei ist der Vergleich des vollständig aufgenommenen Bildes mit starkem Rauschen von $7,5k^-/\text{nm}^2$ mit den Rekonstruktionen der Datensätze mit abnehmendem Rauschen ($15k^-/\text{nm}^2, 30k^-/\text{nm}^2, \infty^-/\text{nm}^2$) bei Abnahme der Anzahl an Samples (50%, 25%, 10%) interessant. Durch diese Wahl an Samples und des Rauschens wird eine Aufnahme der Probe mit der gleichen Elektronenintensität simuliert, wobei ein Kompromiss zwischen Aufnahme weniger Datenpunkte mit geringem Rauschen oder mehr Datenpunkte mit erhöhten Rauschen getroffen wird. Als Referenz ist auch das Intensitätsprofil des originalen Bildes abgebildet. Auf den ersten Blick sehen diese Kurven ähnlich aus, nur die Standardabweichung der Intensität der stark verrauschten Aufnahme ist deutlich größer im Vergleich zu den originalen und rekonstruierten Bildern. In Abbildung 31 ist ein vergrößerter Ausschnitt des linken Übergangs des ersten bzw. zweiten Plateaus höherer Intensität dargestellt. An diesen Kurven ist zu erkennen, dass CS trotz nur 10% Samples beim unverrauschten Bild ein praktisch identisches Intensitätsprofil zum Originalbild aufweist. Die Rekonstruktion von verrauschten Bildern weicht sichtbar vom Originalbild ab und hat vor allem einen weicheren Übergang an den Grenzen der Plateaus.

Das war zu erwarten, weil bei Berücksichtigung von Rauschen die Samples ungenauer angepasst werden und somit weniger Koeffizienten zur Darstellung der Samples verwendet werden (tatsächlich kleinere ℓ_1 -Norm von \mathbf{x}). Im Gegensatz dazu können Bereiche mit starken Änderungen, wie etwa an den Grenzen der Plateaus, nur mit vielen Fourierkoeffizienten (bzw. einer höheren ℓ_1 -Norm von \mathbf{x}) dargestellt werden. Das stark verrauschte Bild weist verglichen dazu eine stärkere Variation der Intensität auf, folgt dem Intensitätsverlauf des Originalbildes jedoch besser als die verrauschten, rekonstruierten Bilder. Diese Fehler fallen bei der Simulation mit der „Probe“ ohne Aberrationskorrektur nicht so stark auf (siehe Anhang). Ein Ausschnitt der rekonstruierten Bilder mit NESTA am linken Übergang von reinem Silizium zu Siliziumgermanium kann in Abbildung 28 und 29 betrachtet werden.

Somit ist CS für unverrauschte Elektronenmikroskopieaufnahmen von stark periodischen Strukturen, wie diese bei atomarer Auflösung von Kristallstrukturen vorkommen, ein hervorragendes Mittel, um die Beleuchtungsintensität der Probe mit Elektronen zu verringern. In der Praxis wird allerdings durch Rauschen ein Qualitätsverlust Zustandekommen.

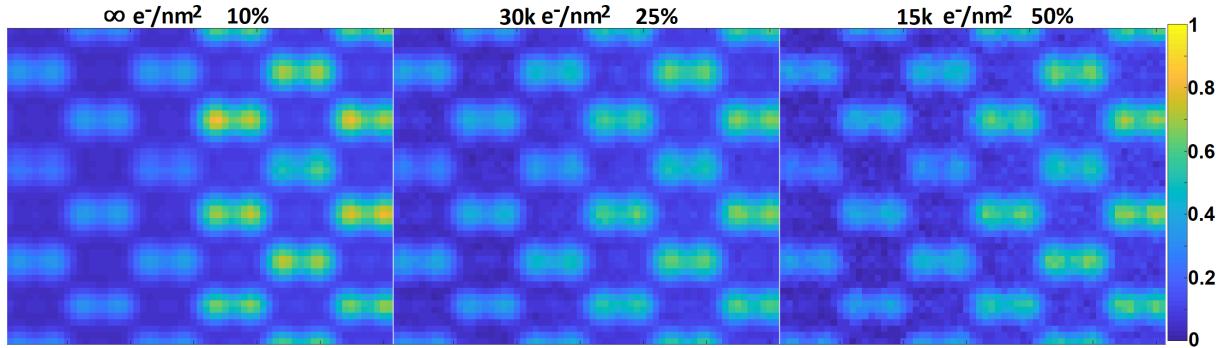


Abb. 28: Ausschnitt aus der mit NESTA rekonstruierten Aufnahme für die Stemsim-Simulation ohne Aberrationskorrektur der „Probe“ mit unterschiedlicher Elektronenintensität in e^-/nm^2 und Anzahl an Samples in % angegeben.

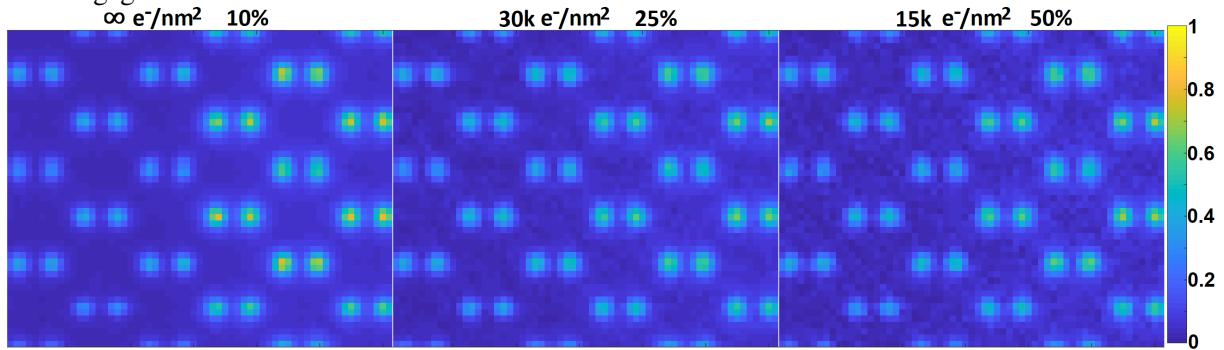


Abb. 29: Ausschnitt aus der mit NESTA rekonstruierten Aufnahme für die Stemsim-Simulation mit Aberrationskorrektur der „Probe“ mit unterschiedlicher Elektronenintensität in e^-/nm^2 und Anzahl an Samples in % angegeben.

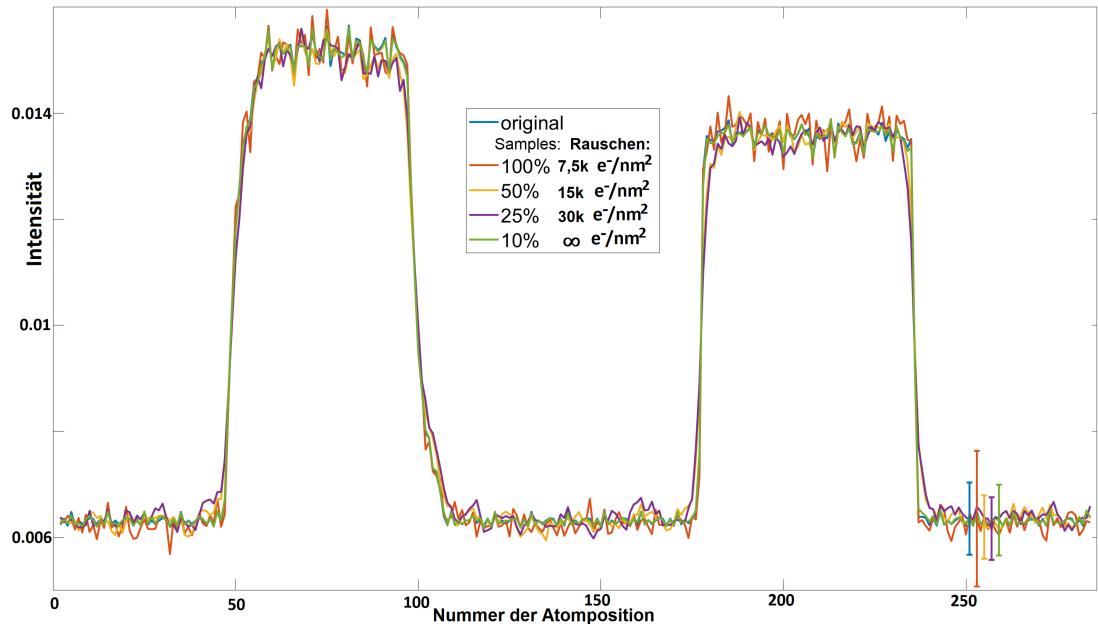


Abb. 30: Intensitätprofil der originalen und mit NESTA rekonstruierten Siliziumprobe mit aberrationskorrigierter „Probe“. Siliziumgermanium befindet sich an den Plateaus erhöhter Intensität. In der Legende ist die Anzahl der Samples in Prozent und die Elektronenintensität in e^-/nm^2 angegeben. Rechts unten ist die Standardabweichung der Intensität (gemittelt über das gesamte Intensitätsprofil) mit Fehlerbalken eingezeichnet. Die Profile sehen auf den ersten Blick ähnlich aus.

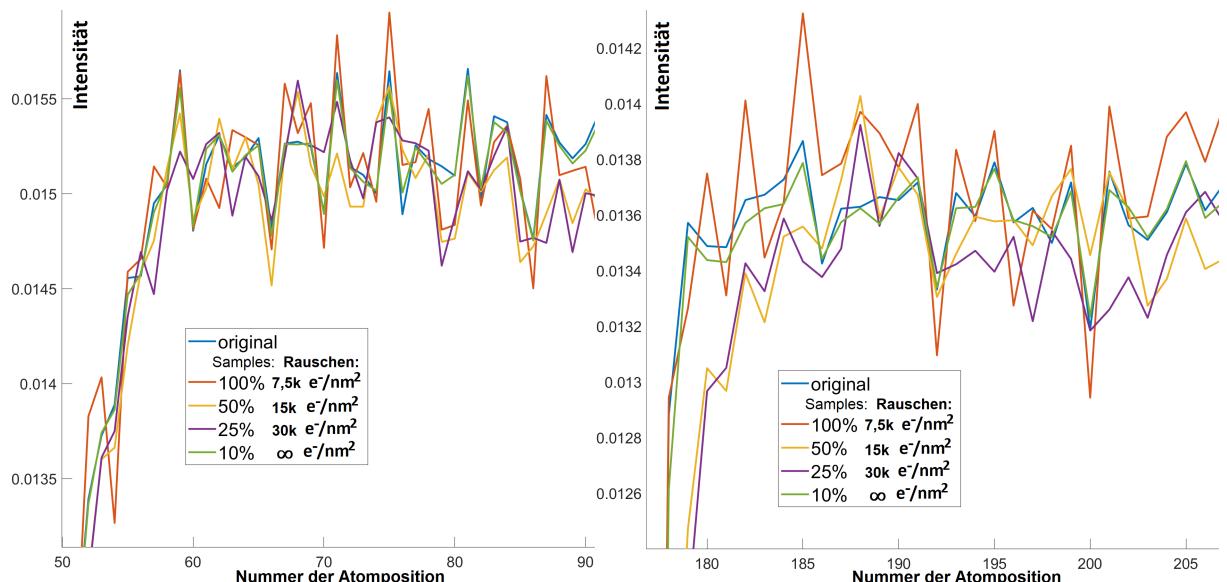


Abb. 31: Vergrößerter Ausschnitt des Intensitätsprofils der originalen, verrauschten und rekonstruierten Bilder einer Siliziumprobe mit aberrationskorrigierter „Probe“: Der Intensitätsverlauf ist für das rekonstruierte Bild fast identisch zum Original. Für das vollständig aufgenommene, verrauschte Bild weicht dieser etwas vom original ab. Die Rekonstruktion aus den verrauschten Samples liefert ein noch etwas schlechteres Ergebnis.

9.2 Rekonstruktion einer Multislice-Simulation und Vergleich der mittleren Intensitäten für unterschiedliche Probendicken

Um aus einer STEM-Aufnahme die örtliche Dicke der Probe zu bestimmen wird eine Multislice-Simulation (mit Frozen-Lattice) der Intensität am HAADF-Detektor für unterschiedliche Probendicken erstellt. Die vorliegende Simulation besteht aus 628 Slices mit jeweils 20x19 Pixeln, dies entspricht einer Probendicke zwischen 0 nm und 200 nm. Als Beispiel sind hier die Intensitäten nach Durchgang des 200.ten (Probendicke: 63,5 nm) und des 628.ten (Probendicke: 200 nm) Slices in Bild 32 dargestellt.

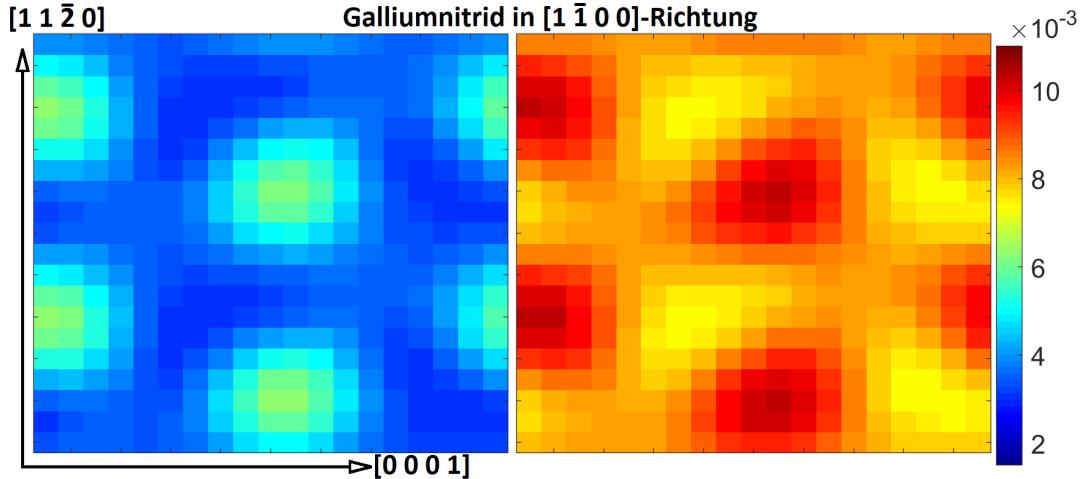


Abb. 32: Zwei Slices der Simulation einer Galliumnitrid-Probe aufgenommen in $[1\ 1\ \bar{2}\ 0]$ Richtung. Das linke Slice (Nr. 200) entspricht einer Probendicke von 63,5 nm und das rechte Slice (Nr. 628) einer Probendicke von 200 nm.

Aus diesen Slices wurden an identischen Positionen Samples entnommen, aus denen die Slices in der CDFT wieder rekonstruiert wurden. Die CDFT eignet sich hier am besten, weil die Bilder periodische Strukturen enthalten, welche auch am Rand periodisch sind. Es wurden Rekonstruktionen mit 10% bis 25% Samples für 100 verschiedene, zufällig ausgewählte Positionen der Samples durchgeführt und der mittlere so wie der maximale Fehler der Rekonstruktion untersucht. Die Rekonstruktionsgenauigkeit ist von der Auswahl der Samples abhängig. Die Position der Samples wird in einer Maske angegeben.

Bei der Rekonstruktion schneidet der Algorithmus OMP, gefolgt von NESTA und FPC_AS am besten ab. Mit 10% Samples ist die Rekonstruktion nicht sehr zuverlässig. Für die Rekonstruktion aus 25% Samples ist der Rekonstruktionsfehler deutlich geringer. Dies kann anhand des mittleren und maximalen Bildfehlers für 100 unterschiedlichen Masken in Abbildung 33 nachvollzogen werden. In allen Fällen gibt es Ausreißer für einige Masken. Bei Verwendung nicht zufälliger Masken waren diese Fehler noch deutlich größer. Selbst bei Rekonstruktion aus 25% Samples bleibt der maximale Fehler der Rekonstruktion für nicht zufälligen Masken mit 20% bis 40% ziemlich hoch. Versucht man die Slices aus 25% Samples, welche die Pixel größter Intensität enthalten, wiederherzustellen, so ist der maximale Bildfehler sogar 64%. Das Problem bei nicht zufälligen Masken ist, dass über einige Teile des Bildes kaum Informationen vorliegen, so dass das Bild aus den Samples schlecht rekonstruiert werden kann. Eine zufällige Auswahl an Samples hat eine geringe Wahrscheinlichkeit, dass ähnliche Bildpunkte systematisch ausgewählt werden, so dass bei einer solchen Wahl der Samples die Rekonstruktion besser funktioniert. Deswegen wurden in den vorherigen Tests die Samples auch immer zufällig ausgewählt.

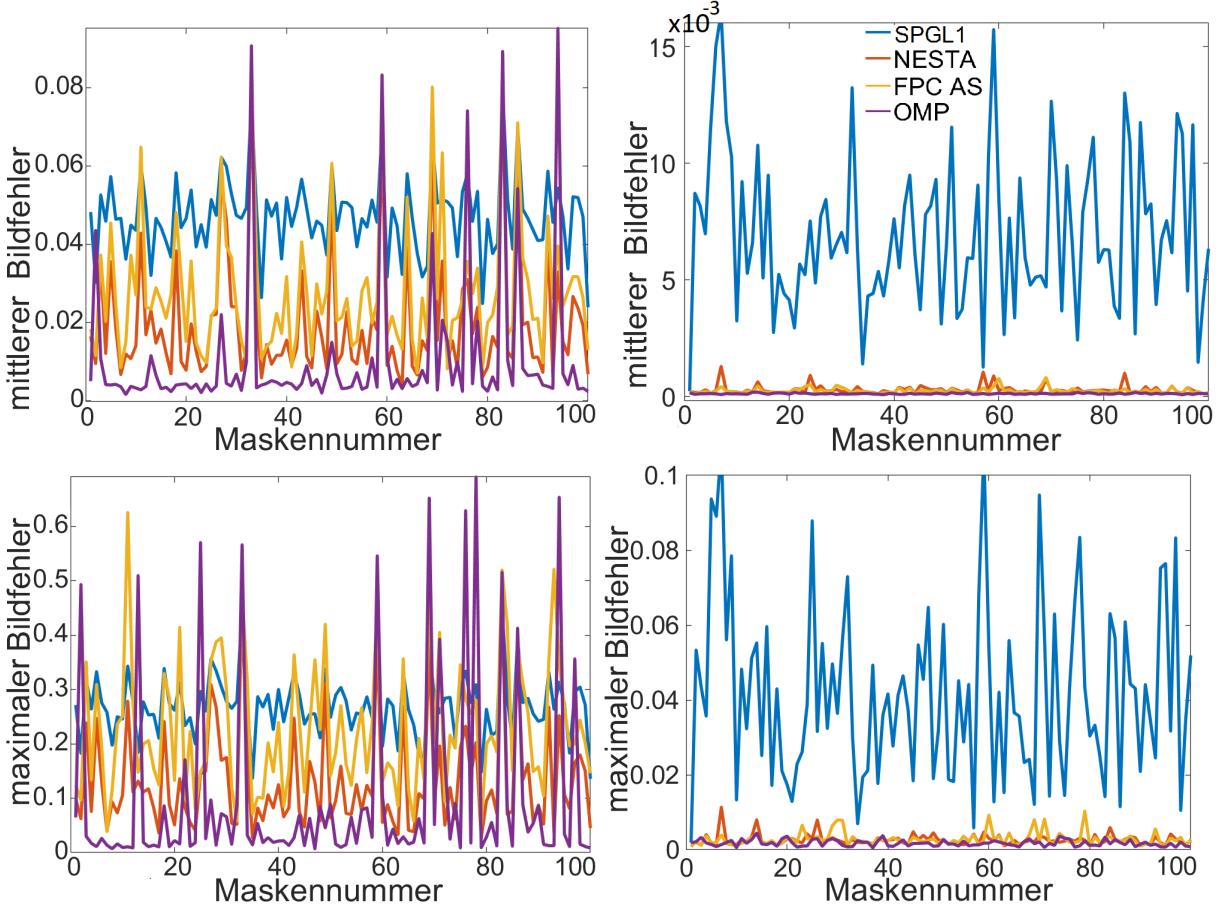


Abb. 33: Der mittlere (oben) und maximale (unten) Bildfehler bei der Rekonstruktion der Slices mit verschiedenen Algorithmen aus 10% Samples (links) bzw. 25% Samples (rechts) für 100 verschiedene Masken: OMP liefert die besten Ergebnisse. Bei kleiner Anzahl an Samples gibt es größere Ausreißer für einige Masken.

In Abbildung 34 ist der Rekonstruktionsfehler einer besonders schlechten Rekonstruktion der Slices für eine Maske mit 10% Samples dargestellt. Trotzdem repräsentiert diese Abbildung den typischen Verlauf des Rekonstruktionsfehlers. OMP hat dabei die genaueste Rekonstruktionsgenauigkeit, abgesehen von wenigen Ausreißern. Diese Ausreißer traten bei den meisten Masken nicht auf. Wie zu erkennen ist, tritt maximale Fehler der Rekonstruktion meist bei den ersten Slices auf. Das liegt wahrscheinlich daran, dass durch die Frozen-Lattice-Näherung bei der Multislice-Simulation, durch zufälliges Auslenken der Atome im Kristallgitter, ein periodisches Ergebnis mit leichtem Rauschen erzeugt wird. Weil in den ersten Slices das Bild eine geringe Intensität besitzt, ist das Verhältnis des Signals zum Rauschen hier größer, was die Rekonstruktionsgenauigkeit für die ersten Slices stark senkt.

Weil die mittlere Intensität des Bildes zur Bestimmung der Dicke der Probe verwendet wird, wurde diese für die Samples, das Originalbild und die verschiedenen Rekonstruktionen berechnet. In Abbildung 35 ist der Mittelwert für eine Rekonstruktion mit OMP aus 15% Samples zusammen mit den Mittelwerten der Samples und der originalen Slices aufgetragen. Diese Werte sind dabei für alle 100 Masken zusammen abgebildet, so dass die Breite einer Kurven das Vertrauensintervall des Mittelwertes mit der jeweiligen Methode wiedergibt. Es ist deutlich zu erkennen, dass die Rekonstruktion eine bessere Abschätzung der mittleren Intensität eines Slices zulässt, als dies rein aus dem Mittelwert der Samples (einer unvollständigen Simulation) möglich wäre. Bei höherer Anzahl an Samples war die Rekonstruktion noch genauer, so dass sich die Mittelwerte der Rekonstruktion kaum mehr von denen der originalen Simulation unterscheidet.

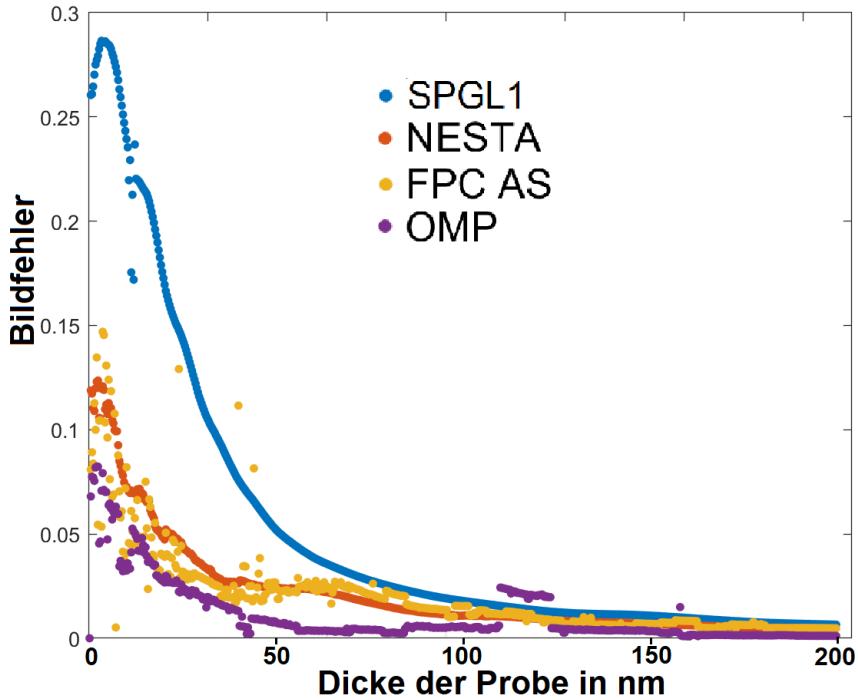


Abb. 34: Rekonstruktionsfehler einer besonders schlechten Rekonstruktion einer Multislice-Simulation (mit Frozen-Lattice) aus 10% Samples. Der Rekonstruktionsfehler nimmt einen typischen Verlauf, wie bei den meisten anderen Masken beobachtet wurde. Er nimmt mit der Nummer der Slices ab und hat bei einigen Slices Ausreißer. Überwiegend hat OMP den kleinsten Rekonstruktionsfehler, wobei in diesem Beispiel ein ungewöhnlich hoher Rekonstruktionsfehler bei einer Probendicke zwischen 110 nm und 125 nm für die Rekonstruktion mit OMP auftritt.

Es wurde auch der mittlere und maximale prozentuale Fehler des Mittelwerts berechnet und für eine Rekonstruktion aus 10% und 25% Samples in Abbildung 36 dargestellt. Diese Werte sind für verschiedene Anzahl an Samples auch in Tabelle 6 abgebildet.

Diese Werte sind ein Ansatzpunkt, um die Zuverlässigkeit des Mittelwertes einer Rekonstruktion bei einer gegebenen Anzahl an Samples abzuschätzen. Auch hier schneidet der Löser OMP am besten ab. OMP unterscheidet sich auch in seiner Funktionsweise deutlich von den anderen Lösern, denn es wird die ℓ_0 -Norm der Koeffizienten klein gehalten statt die ℓ_1 -Norm zu minimieren. Dadurch berechnet OMP eine tatsächlich sparse Rekonstruktion des Signals.

Die einzelnen Slices sind zwar nicht sparse, sie können aber durch eine sparse Darstellung gut angenähert werden. OMP erzielt wahrscheinlich aus diesem Grund bessere Ergebnisse. Dieser Algorithmus eignet sich gut, um eine unvollständige Simulation zu vervollständigen und den Mittelwert daraus für die Bestimmung der Probendicke zu verwenden.

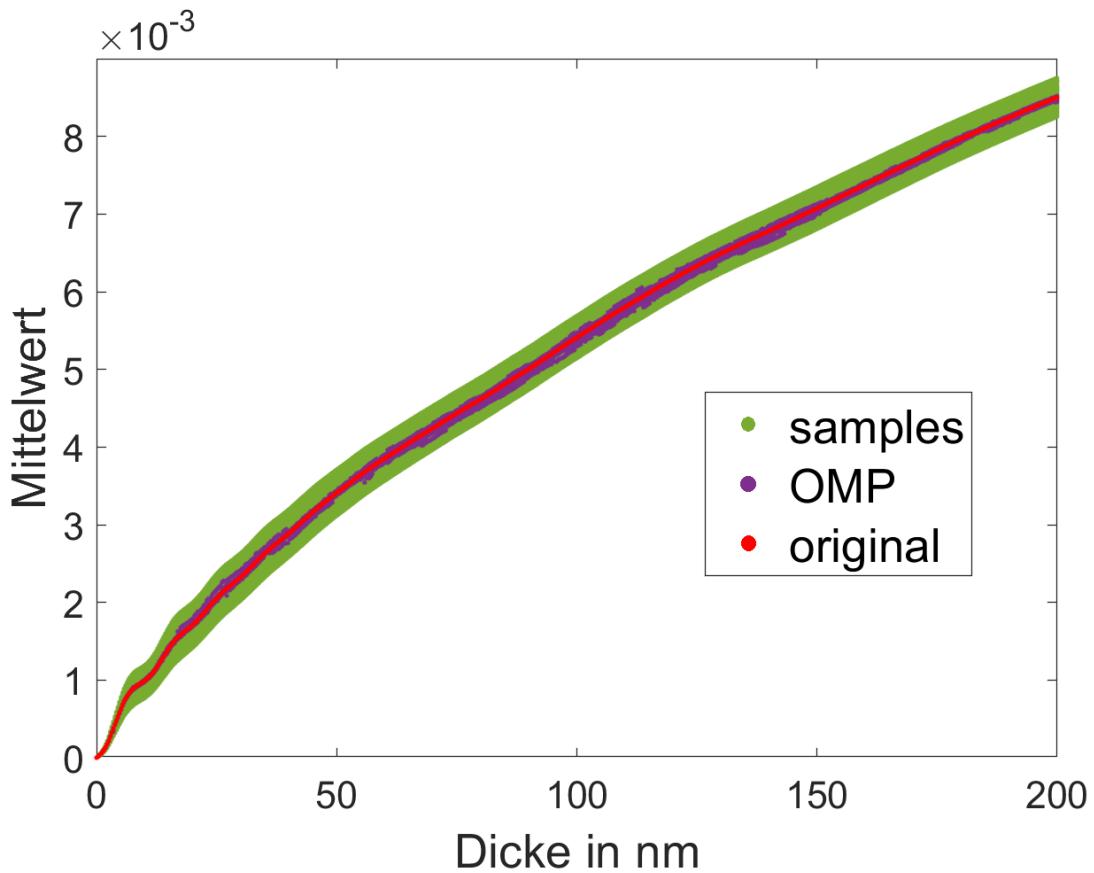


Abb. 35: Die mittlere Intensität des Originalbildes, der Samples und einer Rekonstruktion mit OMP für 100 unterschiedliche Masken aus 15% Samples: Die Rekonstruktion des Bildes ergibt einen deutlich genaueren Mittelwert, als dies allein aus den Samples möglich wäre.

Tab. 6: Der mittlere und maximale prozentuale Mittelwertfehler für Rekonstruktion mit verschiedenen Lösern und für unterschiedliche Anzahl an Samples angegeben in Prozent.

Anzahl Samples		Samples	SPGL1	NESTA	FPC_AS	OMP
10%	mittlerer rel. Mittelwertfehler	18,77%	18,64%	17,43%	17,95%	4,41%
10%	maximaler rel. Mittelwertfehler	40,03%	39,90%	39,96%	39,96%	38,02%
15%	mittlerer rel. Mittelwertfehler	14,61%	13,20%	9,44%	8,99%	0,79%
15%	maximaler rel. Mittelwertfehler	25,57%	22,76%	21,95%	26,18%	9,47%
20%	mittlerer rel. Mittelwertfehler	11,45%	8,86%	3,58%	2,70%	0,29%
20%	maximaler rel. Mittelwertfehler	22,25%	17,19%	12,75%	13,75%	2,26%
25%	mittlerer rel. Mittelwertfehler	10,41%	6,06%	1,55%	1,20%	0,28%
25%	maximaler rel. Mittelwertfehler	19,14%	10,46%	3,46%	3,75%	1,73%

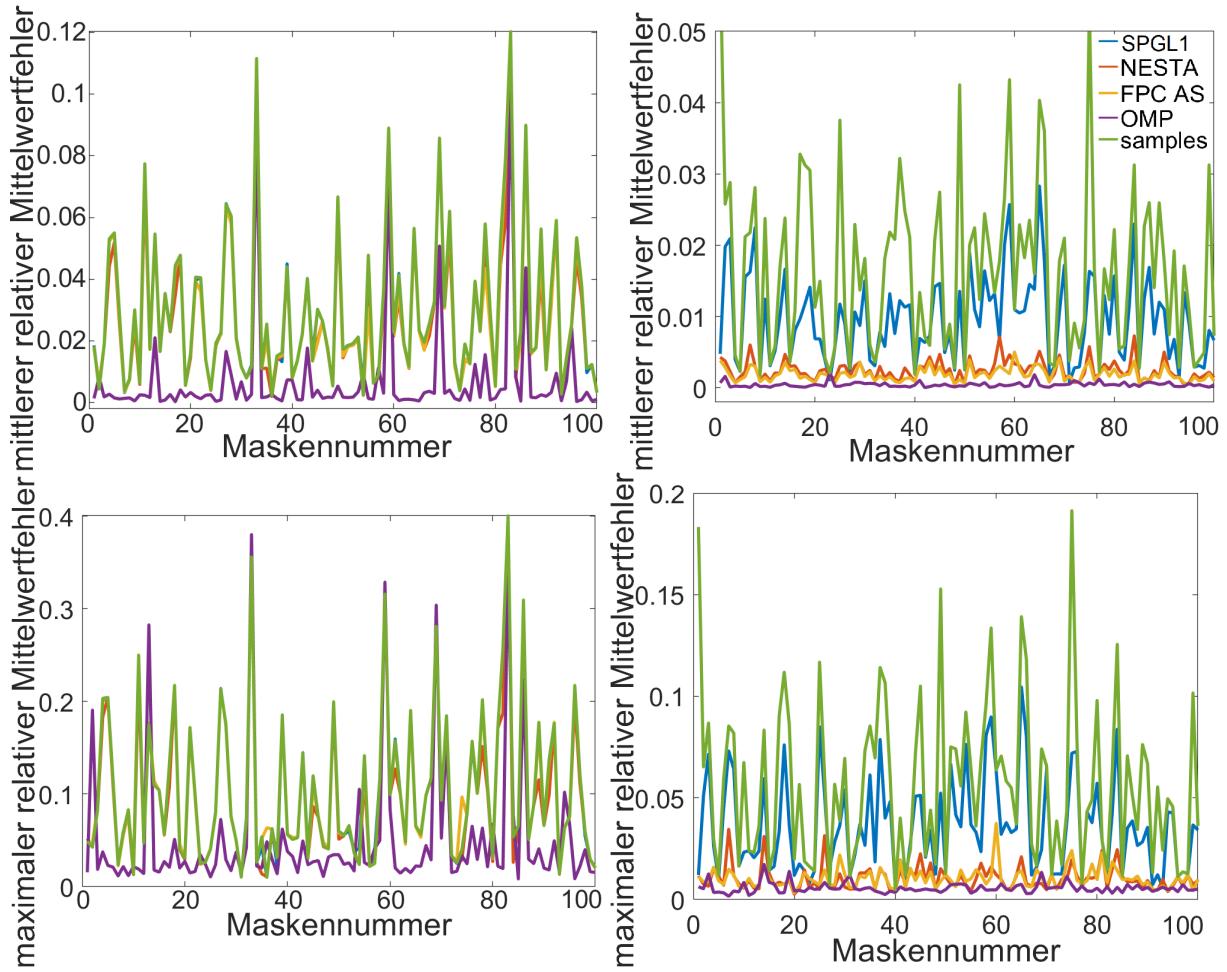


Abb. 36: Die mittlere (oben) und maximale (unten) prozentuale Abweichung des Mittelwertes der Samples und der Rekonstruktionen der Slices mit verschiedenen Algorithmen aus 10% Samples (links) bzw. 25% Samples (rechts) für 100 verschiedene Masken: Wieder liefert OMP die besten Ergebnisse und es gibt größere Ausreißer bei kleiner Anzahl an Samples.

10 Fazit

In dieser Arbeit wurde eine Einleitung in die Elektronenmikroskopie und das Compressed Sensing gegeben. Darauf aufbauend wurden in Kapitel 4 aktuelle Anwendungen von Compressed Sensing in der Elektronenmikroskopie vorgestellt. Als nächstes wurden in Kapitel 5 die verwendeten Algorithmen für Compressed Sensing eingeführt. In Kapitel 6 wurden fünf verschiedene Transformationen für die Darstellungsmatrix vorgestellt. Globale Transformationen können Signale mit weichen Übergängen und periodischen Strukturen sparse auflösen. Dagegen lösen Daubechies-4-Wavelets und Haar-Wavelets ein Signal auf unterschiedlichen Größenskalen auf, so dass lokale Objekte und scharfe Ränder sparse dargestellt werden können. Aus der Kohärenz der Messmatrix mit den unterschiedlichen Darstellungsmatrizen wurde festgestellt, dass die verwendeten Wavelets nur solche Bilder sparse darstellen, welche mit der Messmatrix schlecht erfasst werden können. In Kapitel 7 wurde dies an einigen Beispielen bestätigt. Es wurde nachgewiesen, dass eine Rekonstruktion mit Compressed Sensing nur für solche Bilder sinnvoll ist, welche sich in einer Darstellungsbasis zumindest näherungsweise sparse darstellen lassen. Somit ist eine sparse Darstellung eines Bildes notwendig aber nicht ausreichend für eine gute Rekonstruktion mit Compressed Sensing, wobei die Wavelets wegen der hohen Kohärenz zur Messmatrix schlechtere Ergebnisse liefern. Die globalen Transformationen haben eine geringere Kohärenz mit der Messmatrix und eignen sich zugleich besser für die Darstellung von periodischen Strukturen, welche bei elektromikroskopischer Aufnahme atomarer Strukturen vorkommen. Deswegen wurde die reduzierte diskrete Fouriertransformation für die weitere Rekonstruktion von Bildern verwendet. Die diskrete Kosinustransformation könnte auch verwendet werden, vor allem bei Bildern mit nicht-periodischen Rändern. In Kapitel 8 wurden die Algorithmen für die Rekonstruktion von Signalen und Bildern getestet. Beim Testen der Algorithmen wurden viele ausgeschlossen, weil ihre Parameter (für den Lasso Pursuit) im Voraus nicht abzuschätzen sind, sie viel Speicher zum speichern einer expliziten Matrix benötigen oder diese langsam für große Bilder konvergieren. Nach ausführlichem Testen stellte sich der Algorithmus NESTA gefolgt von FPC_AS und SPGL1 als besonders schnell und zuverlässig heraus. Die Rechenzeit dieser Algorithmen hängt hauptsächlich von der Ausführungszeit der Transformation ab und wächst grob linear mit der Anzahl an Pixeln im Ergebnis. Dabei hatte auch OMP gute Ergebnisse erzielt, jedoch nimmt bei diesem Algorithmus die Rechenzeit für große und nicht sparse Bilder stark zu. In Kapitel 9 wurden die Algorithmen dann auf simulierte HAADF-STEM-Aufnahmen angewandt. Die Rekonstruktion von nicht verrauschten HAADF-STEM-Aufnahmen mit wenigen Samples (10%) lieferte bessere Ergebnisse als eine stark verrauschte, vollständige Aufnahme. Waren die Samples verrauscht, so waren die Ergebnisse der Rekonstruktion deutlich schlechter, wobei FPC_AS leicht besser mit Rauschen zurecht kam. Somit lässt sich Compressed Sensing prinzipiell für die Reduzierung der Elektronenintensität verwenden, in der Praxis wird das Verfahren durch Rauschen schlechtere Ergebnisse erzielen.

Als nächstes wurde eine kleine Multislice-Simulation (mit Frozen-Lattice) rekonstruiert, um die Anwendung von Compressed Sensing zur Verringerung der Simulationszeit zu untersuchen. Der Mittelwert der Simulation für unterschiedliche Dicken der Probe ist wichtig, um die Dicke der Probe aus einer tatsächlichen HAADF-STEM-Aufnahme abzuschätzen. Bei der Rekonstruktion stellte sich OMP als die beste Wahl heraus. Dabei brachte die Rekonstruktion der Samples messbar bessere Übereinstimmung mit dem Mittelwert einer vollständigen Simulation, als ein einfacher Mittelwert über die Samples. Eine zufällige Auswahl an Samples hat eine geringe Wahrscheinlichkeit, dass ähnliche Bildpunkte systematisch ausgewählt werden, so dass bei einer solchen Wahl der Samples die Rekonstruktion besser funktioniert. Insgesamt eignet sich OMP gut für die Rekonstruktion einer Multislice-Simulation, wobei die Genauigkeit der Rekonstruktion auf Kosten der Berechnung von mehr Samples immer erhöht werden kann.

Verzeichnisse

Tabellenverzeichnis

1	Vor- und Nachteile globaler Transformationen	18
2	Filterkoeffizienten für HAAR und DB4 Wavelets	21
3	Kohärenz verwendeter diskreter Transformationen	22
4	Rekonstruktionsfehler einiger Bilder mit NESTA in verschiedenen Basen und Interpolation	26
5	Bildfehler der Rekonstruktion der Multislice-Simulation bei unterschiedlich starkem Rauschen angegeben in Prozent	34
6	Der mittlere und maximale prozentuale Mittelwertfehler für Rekonstruktion mit verschiedenen Lösern und für unterschiedliche Anzahl an Samples angegeben in Prozent.	41

Abbildungsverzeichnis

1	Prinzipieller Aufbau eines Elektronenmikroskops.	3
2	Signal und Betrag der Fourierkoeffizienten eines Signals: einige Signale können mit wenigen Koeffizienten dargestellt werden. In diesem Beispiel sind es exakt drei Koeffizienten.	6
3	Betrag der Fouriertransformation vom Bild Huhn: viele Koeffizienten sind sehr klein.	6
4	Beispielbild Huhn: ein Graustufenbild mit vielen Details.	6
5	Beispielsweise 10% Samples aus dem Bild Huhn entnommen.	6
6	Mit CS aus den Samples rekonstruiertes Bild: Huhn erkennbar, aber viel Rauschen vorhanden.	6
7	Beispiel für ein Gleichungssystem $\mathbf{Ax} = \mathbf{y}$ mit zwei Koeffizienten \mathbf{x}_1 und \mathbf{x}_2 und einer zufälligen linearen Gleichung. Mögliche Lösungen dieses Gleichungssystems für die Koeffizienten werden mit der roten Geraden dargestellt. Die Lösungen werden durch eine minimale ℓ_0 -Norm (gelb), ℓ_1 -Norm (grün) und ℓ_2 -Norm (violett) eingegrenzt, so dass eindeutige Lösungen (blaue Kreise) erhalten werden. In diesem Fall ist die Lösung mit minimaler ℓ_0 -Norm und ℓ_1 -Norm identisch und die Lösung mit minimaler ℓ_2 -Norm unterscheidet sich davon.	8
8	Dargestellt sind Voronoi-Diagramme, welche für Interpolation mit natürlichen nächsten Nachbarn verwendet werden: Die schwarzen Punkte sind bekannte Samples. Diesen werden in einem Voronoi-Diagramm die Flächen unterschiedlicher Farbe zugeordnet. Zusätzlich wird ein gesuchter (roter) Punkt eingefügt und wieder ein Voronoi-Diagramm erstellt, wobei nur die Voronoi-Zelle um den roten Punkt als rotes Fünfeck dargestellt ist.	16
9	Transformation eines Beispielbildes (ganz links) mit unterschiedlichen Transformation logarithmisch dargestellt	17
10	Realteil der Schwingungen der DFT und DCT, so wie ihre diskreten Koeffizienten für die Transformation eines Signal mit N=5 Datenpunkten	18
11	Die Filterkoeffizienten für Haar- und DB4-Wavelets in erster und zweiter Ordnung.	21
12	Verwendete Graustufenbilder mit einer Größe von 1024x1024 Pixeln.	23
13	Bildfehler gegen die Anzahl größter Koeffizienten für das Bild Huhn (links) und Function (rechts) logarithmisch aufgetragen: Beide Bilder sind grob mit wenigen Koeffizienten darstellbar, zur exakten Darstellung sind allerdings alle nicht Null Koeffizienten notwendig. Beim Bild Function ist die Darstellungsgenauigkeit mit wenigen Koeffizienten etwas besser.	24
14	Zahl der größten Koeffizienten der Bilder bei Kompression mit 1% Bildfehler in verschiedenen Basen (die Balken für jedes Bild sind um den Wert über den Balken zu teilen): Unterschiedliche Bilder sind in verschiedenen Basen besser darstellbar. Es eignen sich die DCT, DFT, CDFT für die Darstellung periodischer Strukturen und weicher Übergänge und die HAAR und DB4 für die Darstellung lokaler Details und scharfer Kanten.	25

15	Ausschnitt aus dem originalen (links) und mit NESTA in der CDFT rekonstruiertem Bild Huhn (rechts): Die Rekonstruktion in der CDFT liefert ein Bild mit weichen Übergängen, was in diesem Fall Rauschen erzeugt. Das Bild ist nicht sparse in der CDFT und wird deswegen mit einem größeren Fehler rekonstruiert.	27
16	Ausschnitt aus dem originalen (links) und mit NESTA in der CDFT rekonstruiertem Bild Function (rechts): Das Bild kann in der CDFT sehr gut rekonstruiert werden. Nur am oberen und linken Rand sind leichte Rekonstruktionsfehler, geschuldet einem nichtperiodischen Rand des Bildes, vorhanden.	27
17	1D-Testsignale: links eine Kosinusfunktion (Signal 1) und mittig 10 zufällige Koeffizienten, welche mit DCT zurück transformiert das Signal (Signal 2) rechts ergeben.	28
18	Rekonstruktionsfehler mit CS für die beiden eindimensionalen Signale mit allen verwendeten Algorithmen: Löser für Lasso Pursuit schneiden besonders schlecht ab. Das erste Signal ist weniger sparse, so dass es ungenauer rekonstruiert wird. Die beiden Löser OMP und IHT passen die Samples sehr genau an, was in diesem Fall für Signal eins eine bessere Rekonstruktion ergibt.	29
19	Bildfehler für die Testbilder rekonstruiert mit unterschiedlichen Algorithmen (die Balken für jedes Bild sind um den Wert über den Balken zu teilen): IHT, OMP und l1ep pd schneiden besonders schlecht ab. Nur die Bilder Function, Gauss, Gradient und Dots wurden akzeptabel rekonstruiert, weil sie deutlich sparser in der CDFT sind.	30
20	Rechenzeit für die Rekonstruktion der Testbilder mit unterschiedlichen Algorithmen: OMP,IHT, l1eq pd und l1qc logbarrier brauchen besonders lange für die Rekonstruktion.	31
21	Rekonstruktionsfehler bei verschiedener Anzahl an Samples für das Bild Huhn: der Bildfehler nimmt ungefähr exponentiell ab, FPC_AS hat für 5% bis 10% Samples einen Ausreißer nach oben.	31
22	Rekonstruktionsfehler bei verschiedener Anzahl an Samples für das Bild Function: der Bildfehler nimmt ungefähr exponentiell ab, SPGL1 hat bei 1% Samples einen Ausreißer nach oben.	31
23	Rechenzeit für die Rekonstruktion unterschiedlich großer Ausschnitte des Bildes Huhn: Die Rechenzeit skaliert ungefähr proportional zur Pixelanzahl. Dabei ist Nesta langsamer und SPGL1 schneller als FPC_AS.	32
24	Horizontaler Ausschnitt der Multislice-Simulation einer Siliziumprobe mit Siliziumgermanium entlang zweier Kanäle, aufgenommen in [1 1 0]-Richtung ohne Aberrationskorrektur der „Probe“. Es lassen sich deutlich die Kanäle mit Siliziumgermanium erkennen, wobei im linken Kanal die Germaniumkonzentration einen weichen Übergang hat und im rechten Kanal die Germaniumkonzentration abrupt abnimmt.	33
25	Ausschnitt aus der Stemsim-Simulation mit nicht aberrationskorrigierter „Probe“. bei Aufnahme mit unterschiedlicher Elektronenintensität.	33
26	Ausschnitt aus der Stemsim-Simulation mit aberrationskorrigierter „Probe“. bei Aufnahme mit unterschiedlicher Elektronenintensität.	33
27	Rekonstruktionsgenauigkeit aus 25% Samples für beide Multislice-Simulationen der Probe bei unterschiedlichem Rauschen: Die Rekonstruktionsgenauigkeit nimmt mit erhöhtem Rauschen deutlich ab. SPGL1 und OMP schneiden besonders schlecht ab.	34
28	Ausschnitt aus der mit NESTA rekonstruierten Aufnahme für die Stemsim-Simulation ohne Aberrationskorrektur der „Probe“ mit unterschiedlicher Elektronenintensität in e^-/nm^2 und Anzahl an Samples in % angegeben.	36
29	Ausschnitt aus der mit NESTA rekonstruierten Aufnahme für die Stemsim-Simulation mit Aberrationskorrektur der „Probe“ mit unterschiedlicher Elektronenintensität in e^-/nm^2 und Anzahl an Samples in % angegeben.	36

30	Intensitätprofil der originalen und mit NESTA rekonstruierten Siliziumprobe mit aberrationskorrigierter „Probe“. Siliziumgermanium befindet sich an den Plateaus erhöhter Intensität. In der Legende ist die Anzahl der Samples in Prozent und die Elektronenintensität in e^-/nm^2 angegeben. Rechts unten ist die Standardabweichung der Intensität (gemittelt über das gesamte Intensitätsprofil) mit Fehlerbalken eingezeichnet. Die Profile sehen auf den ersten Blick ähnlich aus.	36
31	Vergrößerter Ausschnitt des Intensitätsprofils der originalen, verrauschten und rekonstruierten Bilder einer Siliziumprobe mit aberrationskorrigierter „Probe“: Der Intensitätsverlauf ist für das rekonstruierte Bild fast identisch zum Original. Für das vollständig aufgenommene, verrauschte Bild weicht dieser etwas vom original ab. Die Rekonstruktion aus den verrauschten Samples liefert ein noch etwas schlechteres Ergebnis.	37
32	Zwei Slices der Simulation einer Galliumnitrid-Probe aufgenommen in $[1\ 1\ \bar{2}\ 0]$ Richtung. Das linke Slice (Nr. 200) entspricht einer Probendicke von 63,5 nm und das rechte Slice (Nr. 628) einer Probendicke von 200 nm.	38
33	Der mittlere (oben) und maximale (unten) Bildfehler bei der Rekonstruktion der Slices mit verschiedenen Algorithmen aus 10% Samples (links) bzw. 25% Samples (rechts) für 100 verschiedene Masken: OMP liefert die besten Ergebnisse. Bei kleiner Anzahl an Samples gibt es größere Ausreißer für einige Masken.	39
34	Rekonstruktionsfehler einer besonders schlechten Rekonstruktion einer Multislice-Simulation (mit Frozen-Lattice) aus 10% Samples. Der Rekonstruktionsfehler nimmt einen typischen Verlauf, wie bei den meisten anderen Masken beobachtet wurde. Er nimmt mit der Nummer der Slices ab und hat bei einigen Slices Ausreißer. Überwiegend hat OMP den kleinsten Rekonstruktionsfehler, wobei in diesem Beispiel ein ungewöhnlich hoher Rekonstruktionsfehler bei einer Probendicke zwischen 110 nm und 125 nm für die Rekonstruktion mit OMP auftritt.	40
35	Die mittlere Intensität des Originalbildes, der Samples und einer Rekonstruktion mit OMP für 100 unterschiedliche Masken aus 15% Samples: Die Rekonstruktion des Bildes ergibt einen deutlich genaueren Mittelwert, als dies allein aus den Samples möglich wäre.	41
36	Die mittlere (oben) und maximale (unten) prozentuale Abweichung des Mittelwertes der Samples und der Rekonstruktionen der Slices mit verschiedenen Algorithmen aus 10% Samples (links) bzw. 25% Samples (rechts) für 100 verschiedene Masken: Wieder liefert OMP die besten Ergebnisse und es gibt größere Ausreißer bei kleiner Anzahl an Samples.	42
37	Intensitätprofil der originalen und mit NESTA rekonstruierten Siliziumprobe mit nicht aberrationskorrigierter „Probe“. Siliziumgermanium befindet sich an den Plateaus erhöhter Intensität. In der Legende ist die Anzahl der Samples in Prozent und das Rauschen in e^-/nm^2 angegeben. Rechts unten ist die Standardabweichung der Intensität (gemittelt über das gesamte Intensitätsprofil) mit Fehlerbalken eingezeichnet. Die Profile sehen auf den ersten Blick ähnlich aus.	42
38	Vergrößerter Ausschnitt des Intensitätsprofils der originalen, verrauschten und rekonstruierten Bilder einer Siliziumprobe mit nicht aberrationskorrigierter „Probe“: Der Intensitätsverlauf ist für das rekonstruierte Bild fast identisch zum Original. Für das vollständig aufgenommene, verrauschte Bild weicht dieser stärker vom Original ab. Die Rekonstruktionen aus den verrauschten Bildern liefern allerdings noch einmal deutlich schlechtere Ergebnisse.	42

Notation

Parameter	a
Vektor	\mathbf{a}
n-tes Element eines Vektors	\mathbf{a}_t
Matrix	\mathbf{A}
i,j-tes Element einer Matrix (i-te Zeile / j-te Spalte)	$\mathbf{A}_{i,j}$
n-te Iteration eines Element	$x_{[n]}$

Abkürzungsverzeichnis

Compressed Sensing	CS
Elektronenmikroskopie	EM
Rastertransmissionselektronenmikroskopie	STEM
Transmissionselektronenmikroskopie	TEM
Orthogonal Matching Pursuit	OMP
Iterative Hard Thresholding	IHT
Diskrete Fouriertransformation	DFT
komprimierte Diskrete Fouriertransformation	CDFT
Diskrete Kosinustransformation	DCT
Daubechies-4-Wavelets	DB4
Haar-Wavelets	HAAR
beziehungsweise	bzw.
aberrationskorrigiert	aberrkor.

Danksagung

Ich möchte mich an dieser Stelle bei allen bedanken, die mich beim Schreiben dieser Bachelorarbeit unterstützt haben.

An erster Stelle gebührt mein Dank an Prof. Dr. Andreas Rosenauer und seine Arbeitsgruppe der Elektronenmikroskopie, die dieses Thema für meine Bachelorarbeit angeboten haben, alle nötigen Simulationen bereitgestellt haben und mich während der gesamten Arbeit begleitet haben. Die gemeinsamen Mittagspausen und die Radtour werden mir in Erinnerung bleiben.

Insbesondere bin ich dankbar an Dr. Florian Krause für die Betreuung und die vielen hilfreichen Hinweise beim Schreiben dieser Arbeit. Mein Dank gilt außerdem an Dr. Christoph Maahr, der mir schnell bei Fragen weitergeholfen hat.

Vielen Dank an meine Prüfer Prof. Dr. Andreas Rosenauer und Prof. Dr. Thomas Schmidt, die sich die Zeit nehmen sich diese Arbeit durchzusehen.

Außerdem geht der Dank an meine Eltern, weil sie durch ihre Unterstützung mein Studium ermöglicht haben.

Nicht zuletzt möchte ich mich bei allen Kommilitonen bedanken, die sich Erzählung über den Fortschritt meiner Arbeit anhören mussten.

Anhang

Implementierung

komprimierte Fouriertransformation

Die Fouriertransformation erzeugt komplexe Koeffizienten, welche für reelle Bilder doppelt besetzt sind. Es wurde die Funktion `compress_coeff(x)` implementiert, welche die Symmetrie der Koeffizienten nutzt um doppelte Koeffizienten zu vernachlässigen und die Koeffizienten in eine reelle Matrix anzuordnen. Die Funktion `decompress_coeff(x)` macht diese Reduktion der Koeffizienten rückgängig.

Wavelet-Transformationen

Es wurden die HAAR und DB4 implementiert. Der Grund dafür war ein Verständnis für die Transformationen zu entwickeln. Außerdem ändern die in Matlab vorhandenen Transformationen die Bildgröße, so dass die vorhandenen Transformationen nicht für CS geeignet waren. Für die Transformation muss ein Signal (**f**) als Zeilenvektor, die Transformation (trafo) als Zeichenkette „haar“ oder „db04“ und die Ordnung der Transformation (iter) als Zahl übergeben werden. Die Transformation eines eindimensionalen Signals in die Koeffizienten erfolgt mit `x=wave_1d_multi(f,trafo,iter)` und die Rücktransformation mit `f=iwave_1d_multi(x,trafo,iter)`. Für die zweidimensionale Transformation wurden zwei verschiedene Schemata umgesetzt:

- Transformation um eine Ordnung zur Zeit, so dass die Ordnung zeilen- und spaltenweise identisch ist (Funktion: `wave_2d_standard(X,trafo,iter)`)
- vollständige zeilenweise Transformation und anschließend spaltenweise Transformation. Ordnungen können unterschiedlich sein (Funktion: `wave_2d_nonstandard(X,trafo,[iter1,iter2])`)

Dabei muss der Funktion eine Matrix **X** statt des Vektors übergeben werden. Für das zweite Schema wird die Transformationsordnung als Vektor mit zwei Zahlen angegeben. Die erste Zahl gibt die Ordnung für die zeilenweise Transformation und die zweite Zahl für die spaltenweise Transformation. Die anderen Argumente sind identisch. Für alle Transformationen liegen Beispiel-Skripte (test_1d.m und test_2d.m) im Ordner Wavelets vor.

Vereinfachte Funktion zur Rekonstruktion mit CS

Für eine einfache Rekonstruktion von Samples wurde eine Funktion erstellt. Die Funktion

```
[result,fit_error,comp_time] = reconstruct(samples,indices,dim,trafo,solver,sigma)
```

nimmt die Samples (samples) mit ihren Positionen (indices) und der Größe des Bildes als Vektor mit der Anzahl an Elementen in einer Zeile und Spalte (dim) entgegen. Des weiteren muss eine Transformation (trafo: cdft, dft2, dct2, haar, db04) und ein Löser (solver: spgl1, nesta, fpcas) als Zeichenkette und das vorhandene Rauschen (sigma) als Zahl angegeben werden. Die Funktion gibt das Ergebnis der Rekonstruktion (result) zusammen mit dem Fehler der Rekonstruktion (fit_error) und der Rechenzeit (comp_time) aus. Dazu gibt es ein Beipiel-Skript (final.m). Sollten die implementierten Algorithmen OMP oder IHT verwendet werden, so muss auf die Funktionen direkt zugegriffen werden, weil diese Funktionen mehr Argumente benötigen. Genaueres zu den Parametern ist im Abschnitt zu dem jeweiligen Algorithmen beschrieben. Zu den beiden Algorithmen sind auch Beispiel-Skripte (finalOMP.m und finalIHT.m) vorhanden.

Bei all diesen Lösern wird die Matrix als Funktion übergeben. Dies hat den Vorteil, dass die Transformationsmatrix nicht explizit gespeichert werden muss und schneller berechnet werden kann, als eine Matrixmultiplikation.

Der Funktion für die Rücktransformation $\mathbf{y} = AT(\mathbf{x}) = \mathbf{Ax}$ werden $N \times M$ Koeffizienten \mathbf{x} als Vektor übergeben, welche innerhalb der Funktion in eine Matrix umgeformt, transformiert und zurück in einen Vektor angeordnet werden. Bei diesen Lösern musste auch eine Multiplikation mit der adjungierten Matrix $\mathbf{x} = AA(\mathbf{y}_n) = \bar{\mathbf{A}} \cdot \mathbf{y}$ für einen Vektor \mathbf{y} formuliert werden.

OMP

```
x = OMP(samples,dim,indices,sparsity,start_lsqlin_OptimalityTolerance,final_lsqlin_OptimalityTolerance  
accuracy,AA,AT)
```

Der Algorithmus OMP nimmt die Samples (samples), Positionen der Samples (indices) und Größe des Bildes (dim) entgegen. Der implementierte Algorithmus wird beendet, sobald die maximale Sparsity (sparsity) oder die geforderte Genauigkeit (accuracy) erreicht wurde. Anschließend werden die endgültigen Koeffizienten mit einer höheren Genauigkeit (final_lsqlin_OptimalityTolerance) berechnet. Zusätzlich müssen die Hintransformation und dessen adjungierte (AA und AT) angegeben werden.

IHT

```
x = IHT(samples,dim,indices,accuracy,min_change,maxiter,min_step,step_search,  
start_sparsity,max_sparsity,sparsity_step,AA,AT)
```

Der Algorithmus IHT nimmt die Samples (samples), Positionen der Samples (indices) und Größe des Bildes (dim) entgegen. In dieser Arbeit implementierte IHT startet mit einer Sparsity von *start_sparsity* und erhöht diese um *sparsity_step* bis maximal zur *max_sparsity*. Zusätzlich müssen die Hin- und Rücktransformationen (AA und AT) angegeben werden. Der IHT wird abgebrochen, wenn die Fitgenauigkeit $accuracy = \|\mathbf{y} - \mathbf{y}^r\|_2^2 / \|\mathbf{y}\|_2^2$ der rekonstruierten Samples \mathbf{y}^r überschritten wird. Wenn sich die Fitgenauigkeit um weniger als um *min_change* ändert, wird die sparsity erhöht und bei erreichen der *max_sparsity* wird der Algorithmus abgebrochen.

Die anfängliche Schrittweite des IHT wurde auf 10 gesetzt. Es wird der Gradient berechnet und ein Schritt in die Richtung größter Abnahme gemacht und der Hard Thresholding Operator auf das Ergebnis angewandt und dann die Fitgenauigkeit berechnet. Die Schrittweite wird um den Faktor *step_search* bis maximal *min_step* immer wieder verkleinert solange die Fitgenauigkeit um mindestens *min_change* zunimmt. Wird *min_step* unterschritten, so wird die Sparsity erhöht und die Schrittweite erneut gesucht. Ändert sich die Fitgenauigkeit um weniger als *min_change* oder nimmt sogar ab, so wird der Schritt mit der größten Fitgenauigkeit gemacht.

Nyquist Shannon Theorem

Das Nyquist Shannon Theorem besagt, dass ein Signal bestehend aus Frequenzen (der Fourierbasis) bis zu einer Maximalfrequenz $|f| < \frac{f_s}{2}$ mit einer Mindestfrequenz von f_s aufgenommen werden muss, um es vollständig zu erfassen. Für den Beweis wird ein Signal im Frequenzraum durch eine Rechteckfunktion auf einen Frequenzbereich $|f| < \frac{f_s}{2}$ begrenzt.

$$X_s(f) = X(f) * rect(f/f_s) \quad (62)$$

$$rect(x) = \begin{cases} 1 & 0 \leq x \leq 1 \\ 0 & sonst \end{cases} \quad (63)$$

Transformiert man $X(f)$ in den Realraum unter Verwendung der Poissonschen Summenformel $\sum_n x(n) = \sum_k X(k)$ ergibt sich eine Sinus-Cardinalis-Interpolation der Funktion $x(t)$

$$x(t) = \sum_{n=-\infty}^{\infty} x(nT) * sinc\left(\frac{(t-nT)}{T}\right) \quad (64)$$

Das bedeutet, dass eine unendliche Anzahl an Datenpunkten gemessen mit einer Frequenz von mindestens f_s nötig sind um das Signal exakt aufzunehmen. Mit dieser Abtastung wären alle Frequenzen kleiner $\frac{f_s}{2}$ erfasst. Bei der Aufnahme eines Signals mit endlich vielen Datenpunkten ist dieses Theorem nur von theoretischer Bedeutung.

Kontinuierliche, sparse Signale können mit Compressed Sensing rekonstruiert werden, auch wenn die Samples mit einer kleineren Frequenz, als der Niquist-Frequenz f_s abgetastet werden.

Äquivalenz der Rekonstruktion mit ℓ_0 und ℓ_1 -Norm

Für eine Rekonstruktion des Signals mit der ℓ_1 -Norm muss die Matrix A entweder die Restricted Isometry Property (RIP) oder die Null Space Property (NSP) erfüllen.

Kohärenz als Rekonstruktionsgarantie

Eine Eindeutigkeit der Rekonstruktion mit CS ist gegeben, wenn die Koeffizienten $\mathbf{Ax} = \mathbf{y}$ erfüllen, s-sparse sind und folgende Ungleichung erfüllen [21] :

$$(2s - 1)\mu(\mathbf{A}, \mathbf{A}) < 1 \quad (65)$$

Kern einer Matrix (Null Space)

Der Kern einer Matrix A ist die Menge aller Vektoren η , welche transformiert mit dieser Matrix den Nullvektor ergeben.

$$\eta \in Ker(\mathbf{A}) \setminus \{\mathbf{0}\} \quad \Leftrightarrow \quad \mathbf{A}\eta = \mathbf{0} \quad (66)$$

Null Space Property (s-NSP)

Liegt ein Problem vor, dessen Lösung N Koeffizienten besitzt. Zum zeigen der Nullspace Property nimmt man eine Menge $S \subset \{1, 2, N\}$, welche kleiner gleich s Elemente enthält. Die dazu komplementäre Menge heißt S_C . Mit den Elementen aus der Menge S (Elemente als Positionsangabe) wird ein maximal s-sparsier Vektor η_S und der dazu komplementäre Vektor η_{S_C} aus allen Vektoren des Kerns der Matrix η gebildet. Die Nullspace Property gilt, wenn folgende Ungleichung für alle möglichen Mengen S und Vektoren η_S erfüllt ist.[22]

$$\|\eta_S\|_1 \leq \|\eta_{S_C}\|_1 \quad (67)$$

Die Nullspace Property stellt sicher, dass die Lösung des Basis Pursuit identisch zum Ausgangsproblem ist. Um das zu zeigen nutzen wir den Zusammenhang :

$$\eta = \eta_S + \eta_{S^C} \quad (68)$$

Wir nehmen an, dass $\tilde{\mathbf{x}}$ die richtige, am s-sparse besetzte Lösung des linearen Gleichungssystems $\mathbf{Ax} = \mathbf{y}$ sei. So sind $\tilde{\mathbf{x}} + \eta$ die anderen möglichen Lösungen dieses Gleichungssystems. Ist die Sparsity s identisch zur ℓ_0 -Norm von $\tilde{\mathbf{x}}$ gewählt, so ist unter allen Vektoren η_S auch die richtige Lösung $\tilde{\mathbf{x}}$ enthalten und die anderen möglichen Lösungen (mit einer höheren Sparsity) sind in η_{S^C} enthalten. Wenn also die Nullspace Property gilt, ist die ℓ_1 -norm der sparsesten Lösung kleiner als die ℓ_1 -Norm der anderen Lösungen. Die Null Space Property ist sehr aufwendig zu prüfen, aktuelle Forschung beschränkt sich darauf die kleinste bzw. größte Schranke s für eine gewählte Matrix z.B. mit Monte Carlo abzuschätzen.

Restricted Isometry Property (RIP)

Die Restricted Isometry Property besagt, dass die Matrix A sich näherungsweise wie ein Orthonormalsystem für einen s-sparsen Vektor x verhält (Erhaltung der Norm). Die δ_s -RIP gilt, wenn folgende Gleichung mit kleinem δ_s erfüllt ist. Dabei ist $\delta_s \in (0, 1)$. Die RIP wird für alle möglichen s-sparsen Vektoren x geprüft und dann die Matrix A als δ_s -RIP mit der maximale Konstante δ_s bezeichnet. [23]

$$1 + \delta_s \leq \frac{\|Ax\|_2^2}{\|x\|_2^2} \leq 1 - \delta_s \quad (69)$$

Eine δ_{2s} -RIP von kleiner als 0.6248 garantiert, das Basis Pursuit dem Ausgangsproblem entspricht. Außerdem ist eine stabile Rekonstruktion bei Rauschen sichergestellt. [21]

Die RIP ist sehr aufwendig zu prüfen. Es gibt allerdings einen Beweis, das zufällige Matrizen (mit Gauß- oder BernoulliVerteilung) mit Wahrscheinlichkeit $1 - e^{-C_2 m}$ die δ_s -RIP mit $m \geq C_1 s \ln(eN/s)$ für gewisse Konstanten C_1, C_2 erfüllen. [24] Diese Matrizen brauchen für größere Probleme viel Speicherplatz und die Matrixmultiplikation ist langsamer als gut bekannte Transformationen.

Intensitätsprofil der rekonstruierten Multislice-Simulation ohne aberrationskorrigierter „Probe“

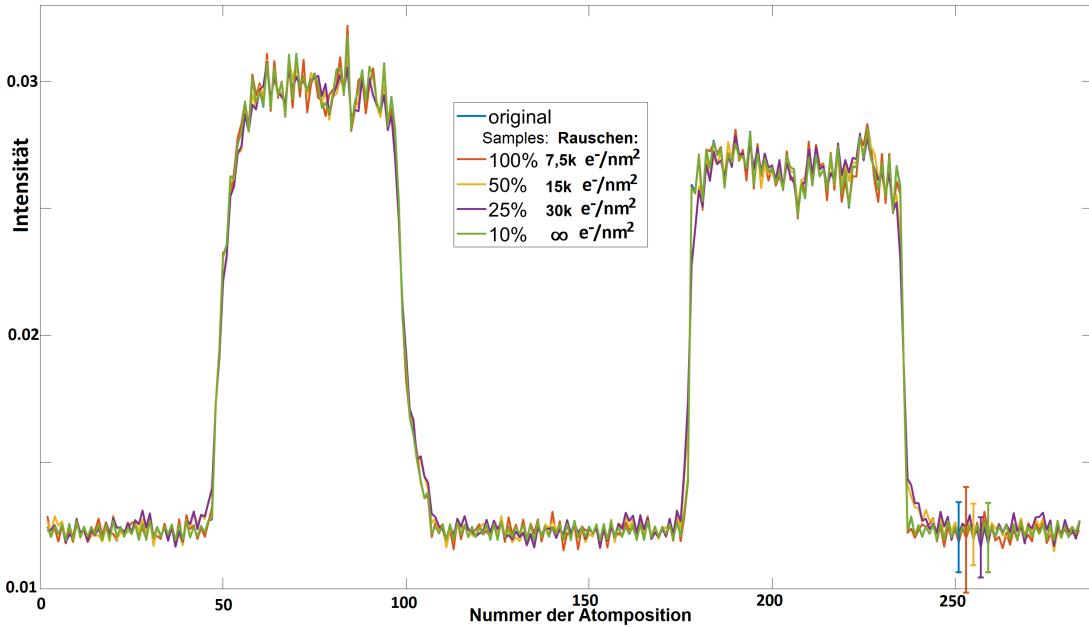


Abb. 37: Intensitätprofil der originalen und mit NESTA rekonstruierten Siliziumprobe mit nicht aberrationskorrigierter „Probe“. Siliziumgermanium befindet sich an den Plateaus erhöhter Intensität. In der Legende ist die Anzahl der Samples in Prozent und das Rauschen in e^-/nm^2 angegeben. Rechts unten ist die Standardabweichung der Intensität (gemittelt über das gesamte Intensitätsprofil) mit Fehlerbalken eingezeichnet. Die Profile sehen auf den ersten Blick ähnlich aus.

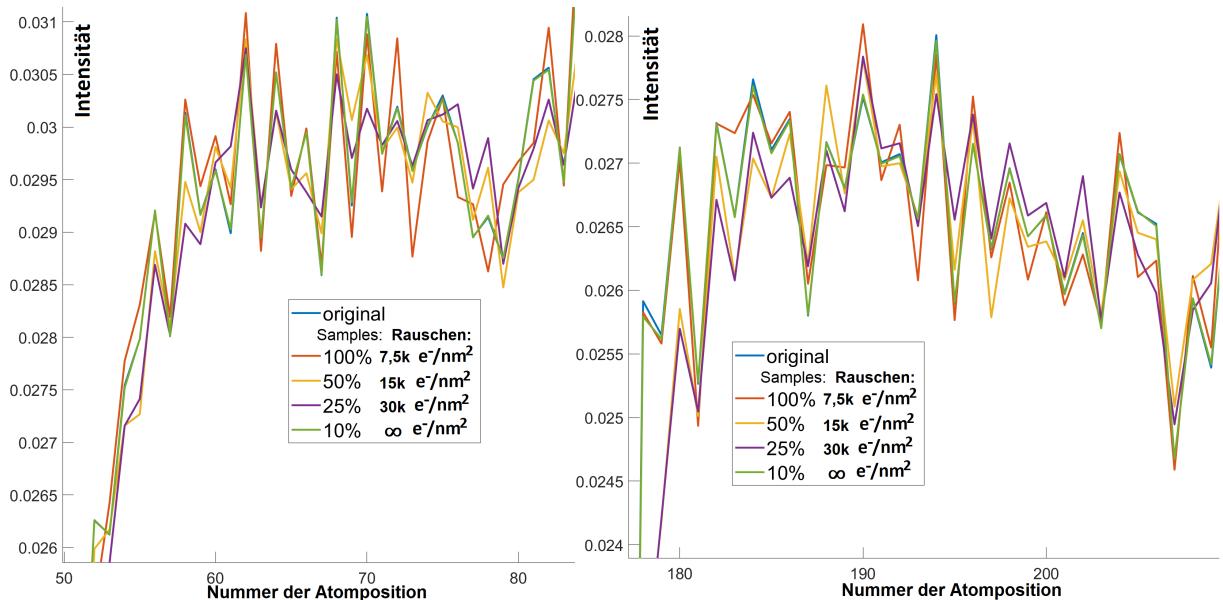


Abb. 38: Vergrößerter Ausschnitt des Intensitätsprofils der originalen, verrauschten und rekonstruierten Bilder einer Siliziumprobe mit nicht aberrationskorrigierter „Probe“: Der Intensitätsverlauf ist für das rekonstruierte Bild fast identisch zum Original. Für das vollständig aufgenommene, verrauschte Bild weicht dieser stärker vom Original ab. Die Rekonstruktionen aus den verrauschten Bildern liefern allerdings noch einmal deutlich schlechtere Ergebnisse.

Literatur

- [1] Knut Müller-Caspary, Oliver Oppermann, Tim Grieb, Florian F Krause, Andreas Rosenauer, Marco Schowalter, Thorsten Mehrtens, Andreas Beyer, Kerstin Volz, and Pavel Potapov. Materials characterisation by angle-resolved scanning transmission electron microscopy. *Scientific reports*, 6:37146, 2016.
- [2] AJV Griffiths and T Walther. Quantification of carbon contamination under electron beam irradiation in a scanning transmission electron microscope and its suppression by plasma cleaning. In *Journal of Physics: Conference Series*, volume 241, page 012017. IOP Publishing, 2010.
- [3] Armand Béché, Bart Goris, Bert Freitag, and Jo Verbeeck. Development of a fast electromagnetic beam blanker for compressed sensing in scanning transmission electron microscopy. *Applied Physics Letters*, 108(9):093103, 2016.
- [4] Andreas Rosenauer and Marco Schowalter. Stemsima new software tool for simulation of stem haadf z-contrast imaging. In *Microscopy of Semiconducting Materials 2007*, pages 170–172. Springer, 2008.
- [5] A Weickenmeier and H Kohl. Computation of absorptive form factors for high-energy electron diffraction. *Acta Crystallographica Section A: Foundations of Crystallography*, 47(5):590–597, 1991.
- [6] Emmanuel J Candès and Michael B Wakin. An introduction to compressive sampling [a sensing/sampling paradigm that goes against the common knowledge in data acquisition]. *IEEE signal processing magazine*, 25(2):21–30, 2008.
- [7] Dorothea Muecke-Herzberg, Patricia Abellan, Michael Sarahan, Iain Godfrey, Zineb Saghi, Rowan Leary, Andrew Stevens, Jackie Ma, Gitta Kutyniok, Feridoon Azough, et al. A compressive sensing based acquisition design for quantitative ultra-low dose high-resolution imaging and spectroscopy in the stem. In *European Microscopy Congress 2016: Proceedings*, pages 324–325. Wiley Online Library, 2016.
- [8] Andrew Stevens, Hao Yang, Weituo Hao, Lewys Jones, Colin Ophus, Peter D Nellist, and Nigel D Browning. Subsampled stem-ptychography. *Applied Physics Letters*, 113(3):033104, 2018.
- [9] Andrew Stevens, Libor Kovarik, Patricia Abellan, Xin Yuan, Lawrence Carin, and Nigel D Browning. Applying compressive sensing to tem video: a substantial frame rate increase on any camera. *Advanced Structural and Chemical Imaging*, 1(1):10, 2015.
- [10] Rowan Leary, Zineb Saghi, Paul A Midgley, and Daniel J Holland. Compressed sensing electron tomography. *Ultramicroscopy*, 131:70–91, 2013.
- [11] Laurène Donati, Masih Nilchian, Sylvain Trépout, Cédric Messaoudi, Sergio Marco, and Michael Unser. Compressed sensing for stem tomography. *Ultramicroscopy*, 179:47–56, 2017.
- [12] Giulio Guzzinati, Thomas Altantzis, Maria Batuk, Annick De Backer, Gunnar Lumbeeck, Vahid Samaee, Dmitry Batuk, Hosni Idrissi, Joke Hadermann, Sandra Van Aert, et al. Recent advances in transmission electron microscopy for materials science at the emat lab of the university of antwerp. *Materials*, 11(8):1304, 2018.
- [13] Thomas Blumensath and Mike E Davies. Iterative hard thresholding for compressed sensing. *Applied and computational harmonic analysis*, 27(3):265–274, 2009.
- [14] Ryan Tibshiran. Vorlesungsskript 7 & 8 zu convex optimization. <http://www.stat.cmu.edu/~ryantibs/convexopt-S15/scribes/>, 2015. [Online; aufgerufen am 20.10.2019].

- [15] Ewout Van Den Berg and Michael P Friedlander. Probing the pareto frontier for basis pursuit solutions. *SIAM Journal on Scientific Computing*, 31(2):890–912, 2008.
- [16] Zaiwen Wen, Wotao Yin, Donald Goldfarb, and Yin Zhang. A fast algorithm for sparse reconstruction based on shrinkage, subspace optimization, and continuation. *SIAM Journal on Scientific Computing*, 32(4):1832–1857, 2010.
- [17] Stephen Becker, Jérôme Bobin, and Emmanuel J Candès. Nesta: A fast and accurate first-order method for sparse recovery. *SIAM Journal on Imaging Sciences*, 4(1):1–39, 2011.
- [18] Tomá Bouda. Artikel aus der online publikationsplattform medium. 100 days of algorithms, day 99: Simplex. <https://medium.com/100-days-of-algorithms/day-99-simplex-1588dd2ebb05>, 2017. [Online; aufgerufen am 20.10.2019].
- [19] Emmanuel Candes and Justin Romberg. 11-magic: Recovery of sparse signals via convex programming. URL: www.acm.caltech.edu/11magic/downloads/11magic.pdf, 4:14, 2005.
- [20] Michael Elad and Alfred M Bruckstein. A generalized uncertainty principle and sparse representation in pairs of bases. *IEEE Transactions on Information Theory*, 48(9):2558–2567, 2002.
- [21] Simon Foucart and Holger Rauhut. A mathematical introduction to compressive sensing. *Bull. Am. Math.*, 54:25–26, 2017.
- [22] Yi Gao, Jigen Peng, Shigang Yue, and Yuan Zhao. On the null space property of ℓ_q -minimization for $0 < q \leq 1$ in compressed sensing. *Journal of Function Spaces*, 2015:1–10, 2015.
- [23] Richard Baraniuk, Mark Davenport, Ronald DeVore, and Michael Wakin. A simple proof of the restricted isometry property for random matrices. *Constructive Approximation*, 28(3):253–263, 2008.
- [24] Simon Foucart, Alain Pajor, Holger Rauhut, and Tino Ullrich. The gelfand widths of p -balls for $0 < p \leq 1$. *Journal of Complexity*, 26(6):629–640, 2010.