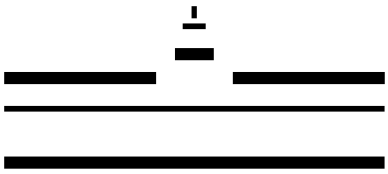

Entmischung mit Sparsity-Regularisierung und Anwendungen

Bachelorarbeit
zur Erlangung des akademischen Grades
Bachelor of Science

Westfälische Wilhelms-Universität Münster
Fachbereich Mathematik und Informatik
Institut für Numerische und Angewandte Mathematik

Betreuung:

Prof. Dr. Martin Burger

Eingereicht von:

Claudia Averkamp

Münster, Dezember 2011

Eidesstattliche Erklärung

Hiermit versichere ich, *Claudia Averkamp*, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe. Gedanklich, inhaltlich oder wörtlich übernommenes habe ich durch Angabe von Herkunft und Text oder Anmerkung belegt bzw. kenntlich gemacht. Dies gilt in gleicher Weise für Bilder, Tabellen, Zeichnungen und Skizzen, die nicht von mir selbst erstellt wurden.

Alle auf der CD beigefügten Programme sind von mir selbst programmiert worden.

Münster, 20. Dezember 2011

Claudia Averkamp

Inhaltsverzeichnis

1. Einleitung

In dieser Arbeit wollen wir uns mit dem Entmischen beschäftigen. Dabei nehmen wir an, dass eine Mischung aus verschiedenen Grundbestandteilen zusammengesetzt wurde, die in unterschiedlichen Anteilen miteinander gemischt wurden. Das Ziel des Entmischens ist es, festzustellen, welche Grundbestandteile in der Mischung vorkommen und in welchen Anteilen. Wir wollen hier davon ausgehen, dass die Grundbestandteile schon bekannt sind und uns mit dem Bestimmen der jeweiligen Anteile beschäftigen.

Eine erste Anwendungsmöglichkeit für das Entmischen ist die Sprachextraktion, bei der Audiosignale entmischt werden. Das Ziel dabei ist es, störende Hintergrundgeräusche, Musik oder andere Sprecher herauszufiltern. Am Ende behält man dann zum Beispiel nur noch einen Sprecher übrig, den man ohne störende andere Signale gut verstehen kann.

Ein anderes Anwendungsgebiet sind hyperspektrale Bilder. Dies sind Bilder mit einer hohen spektralen Auflösung, bei denen nicht nur der sichtbare Bereich des elektromagnetischen Spektrums von einer Kamera aufgenommen wurde, sondern auch die anderen, für einen Menschen nicht sichtbaren, Bereiche. Wenn man nun zum Beispiel mit einem Satelliten solche Bilder von der Erdoberfläche aufnimmt, so enthalten diese Aufnahmen sehr viele Informationen über das Terrain und man kann mit ihnen die vorkommenden Materialien und ihre Anteile in den einzelnen Bildpunkten eindeutig bestimmen. Dies findet unter anderem Anwendung in den Geowissenschaften und der Landwirtschaft.

Hier wollen wir uns mit dem Entmischen von Hyperspektralbildern und mit der Sprachextraktion beschäftigen und jeweils die Anteile bestimmen, in welchen die Grundbestandteile in den unterschiedlichen Mischungen vorkommen. In beiden Fällen sollen die gesuchten Lösungen eine gemeinsame Eigenschaft haben: Sparsity. Das heißt, dass der größte Teil der Einträge gleich Null ist, also in einer einzelnen Mischung, zum Beispiel in einem einzelnen Bildpunkt, kommen nur einige wenige der Grundbestandteile vor. Wir werden die Split-Bregman-Methode benutzen, um Lösungen mit dieser Eigenschaft

finden.

In Kapitel 2 dieser Arbeit wollen wir das Problem des Findens von Lösungen mit Sparsity erläutern und dazu die Split-Bregman-Methode vorstellen mit deren Hilfe man diese Lösungen bestimmen kann. Die Anwendung auf hyperspektrale Bilder folgt in Kapitel 3. Dort gibt es zuerst eine Einführung und dann wird ein gieriger Algorithmus zum Entmischen vorgestellt und dann die Split-Bregman-Methode angewandt auf das Entmischen von hyperspektralen Bildern. In Kapitel 4 folgt die Anwendung in der Sprachextraktion, bei der wieder die Split-Bregman-Methode verwendet wird. Die Implementierung dieser Methode für die Sprachextraktion folgt in Kapitel 5.

2. $l1$-Regularisierung mit der Split-Bregman-Methode

2.1. Sparsity und $l1$-Regularisierung

Wir wollen als erstes das Problem des Entmischens modellieren. Dazu sei f der Vektor, der die gemessenen Werte für die Mischung enthält, A die Matrix mit den Grundbestandteilen der Mischung und u der Vektor, der die Anteile der Grundbestandteile in der Mischung angibt. Dieses u suchen wir. Theoretisch würden wir nun annehmen, dass $f = Au$, doch Messwerte sind fast immer fehlerhaft. Also gilt $f \approx Au$. Wenn wir nun $||Au - f||_2^2$ betrachten, dann ist das der Unterschied zwischen Au und f. Diesen wollen wir minimieren, also ist es unser Ziel eine Lösung für

$$\min_u ||Au - f||_2^2 \tag{2.1}$$

zu finden.

Doch wir wollen nicht nur dieses Problem lösen, wir stellen auch noch eine weitere Bedingung an u: Sparsity. Sparstiy ist zum Beispiel bei Hyperspektralbildern eine sinnvolle Eigenschaft, denn diese Bilder haben eine relativ niedrige räumliche Auflösung und so enthält ein Bildpunkt eine relativ große Fläche. Nun ist die Annahme sinnvoll, dass in diesem einen Bildpunkt nicht alle Materialien des gesamten Bildes vorkommen, sondern nur einige wenige. Das heißt, dass der Lösungsvektor, der angibt, in welchen Anteilen welche Materialien in einem Pixel auftauchen, dünn besetzt sein sollte. Deswegen fodern wir Sparsity.

Als Maß für Sparsity kann man dabei die $l0$-“Norm“ betrachten

$$||u||_0 = \#\{i|u_i \neq 0\}$$

Diese ist eigentlich keine richtige Norm, denn es gilt nicht $||k \cdot u||_0 = |k| \cdot ||u||_0$ für einen Skalar k. Die anderen beiden Bedingungen für eine Norm sind erfüllt. Wir wollen sie im Folgenden trotzdem als $l0$-Norm bezeichnen. Wenn nun die $l0$-Norm eines Vektors klein ist, dann sind die meisten Einträge Null und der Vektor ist dünnbesetzt. Um also eine solche Lösung für unser Problem zu bekommen, liegt es nahe, die $l0$-Norm zu minimieren

$$\min_u ||u||_0 \quad \text{mit} \quad f = Au$$

Das Problem dabei ist, dass es sich nicht um ein konvexes Problem handelt und damit ist es schwer und aufwendig eine Lösung zu finden.
Wir wollen stattdessen die $l1$-Norm betrachten, d.h.

$$||u||_1 = \sum_i |u_i|$$

Das zugehörige Minimierungsproblem

$$\min_u ||u||_1 \quad \text{mit} \quad f = Au$$

ist ein konvexes Problem und damit viel einfacher zu lösen. Es hat sich gezeigt, dass die Minimierung der $l1$-Norm auch Lösungen liefert, die Sparsity besitzen (vgl. [1]).

Wie oben schon erläutert, ist es nicht sinnvoll die Nebenbedingung $f = Au$ zu fordern, sondern es sollte das Minimierungsproblem (2.1) gelöst werden. Damit kommen wir auf folgendes Problem, welches wir im weiteren Verlauf betrachten wollen.

$$\min_u(||Au - f||_2^2 + \mu||u||_1) \tag{2.2}$$

Dabei kontrolliert der Parameter μ die Stärke der Sparstiy.

2.2. Split-Bregman

Um das oben eingeführte Modell (2.2) zu minimieren wollen wir die Split-Bregman-Methode verwenden. Im Folgenden orientieren wir uns dazu an [5] und [3]. Dazu definieren wir als erstes den Bregmanabstand.

Definition 2.2.1. *Der Bregmanabstand zu einer konvexen Funktion J am Punkt v ist wie folgt definiert:*

$$D_J^p(u, v) = J(u) - J(v) - \langle p, u - v\rangle$$

Dabei ist p ein Subgradient von J bei v.

Definition 2.2.2. *Sei $J : \mathbb{R}^n \to \mathbb{R}$. Dann ist das Subdifferential von J im Punkt v gegeben durch*

$$\partial J(v) = \{p \in \mathbb{R}^n | J(u) - J(v) \geq \langle p, u - v\rangle \text{ für alle } u \in \mathbb{R}^n\}$$

$p \in \partial J(v)$ ist dann Subgradient.

Beim Bregmanabstand handelt es sich um keinen normalen Abstand, da die Bedingung der Symmetrie im Allgemeinen nicht erfüllt ist. Trotzdem misst er die Nähe zwischen den Punkten, denn es gilt $D_J^p(u, v) \geq 0$ und $D_J^p(u, v) \geq D_J^p(w, v)$ für alle w auf der verbindenden Linie zwischen u und v.
Es gilt zum einen $D_J^p(u, v) \geq 0$, denn nach Definition des Bregmanabstandes gilt:

$$D_J^p(u, v) = J(u) - J(v) - \langle p, u - v\rangle$$

Mit Definition 2.2.2 gilt:

$$J(u) - J(v) \geq \langle p, u - v\rangle$$

und es folgt

$$J(u) - J(v) - \langle p, u - v\rangle \geq 0$$

Außerdem ist noch zu zeigen, dass $D_J^p(u, v) \geq D_J^p(w, v)$ für alle w auf der verbindenden Linie zwischen u und v gilt. Dies ist äquivalent zur Ungleichung

$$D_J^p(u, v) - D_J^p(w, v) \geq 0$$

und lässt sich wie folgt zeigen:

$$\begin{aligned}
&J(u) - J(v) - \langle p, u - v\rangle - (J(w) - J(v) - \langle p, w - v\rangle)\\
=&J(u) - J(v) - \langle p, u\rangle + \langle p, v\rangle - J(w) + J(v) + \langle p, w\rangle - \langle p, v\rangle\\
=&J(u) - J(w) - \langle p, u\rangle + \langle p, w\rangle\\
=&J(u) - J(v) - \langle p, u - w\rangle\\
=&D_J^p(u, w) \geq 0
\end{aligned}$$

nach Definiton des Subdifferentials.

Wenn wir nun unser Problem (2.2) betrachten, dann definieren wir $||\Phi(u)||_1 = J(u)$, wobei hier $\Phi = Id$, und $H(u) = ||Au - f||_2^2$. Dann erhält man folgendes Minimierungsproblem ohne Nebenbedingungen

$$\min_u(J(u) + \lambda H(u)) \tag{2.3}$$

wobei J und H konvex.

Anstatt des Problems (2.3) kann man auch das folgende Problem lösen

$$u^{k+1} = \arg\min_u(D_J^p(u, u^k) + \lambda H(u)) = \arg\min_u(J(u) - \langle p^k, u - u^k\rangle + \lambda H(u))$$

Zur Vereinfachung nehmen wir an, dasss H differenzierbar ist, was für die Fälle ausreicht, die wir betrachten wollen. Es ergibt sich dann für die Iteration von p, dass gelten muss

$$0 = p^{k+1} - p^k + \nabla H(u^{k+1})$$

da wir annehmen, dass u^{k+1} optimal ist. Dabei ist p^{k+1} das Subdifferential von $J(u)$. Es ergibt sich also

$$p^{k+1} = p^k - \nabla H(u^{k+1})$$

Definition 2.2.3. *Die Bregmaniteration ist folgende iterative Regularisierungsmethode. Sei J linear, H linear und differenzierbar mit $H \geq 0$. Dann löst man folgende*

Iterationen:

$$u^{k+1} = \arg\min_u (D_J^p(u, u^k) + \lambda H(u)) \tag{2.4}$$

$$p^{k+1} = p^k - \nabla H(u^{k+1}) \tag{2.5}$$

Dazu wähle man als Startwerte $p^0 = 0$ *und* $u^0 = 0$.

Nun wollen wir die Bregmaniteration auf das allgemeine $l1$-Regularisierungsproblem anwenden.

$$\min_u (||\Phi u||_1 + \lambda H(u)) \tag{2.6}$$

Dabei gilt, dass $H(u)$ und $||\Phi u||_1$ konvexe Funktionale sind und außerdem, dass $\Phi(u)$ differenzierbar ist.

Die Idee ist nun, die $l1$- und $l2$-Anteile der Energie in (2.6) zu entkoppeln. Zuerst ändern wir das Problem zu

$$\min_{u,d} (||d||_1 + H(u)) \quad \text{mit} \quad d = \Phi(u)$$

welches äquivalent zu (2.6) ist. Um dieses Problem zu lösen, schreiben wir es in ein Problem ohne Nebenbedinungen um

$$\min_{u,d} (||d||_1 + H(u) + \frac{\lambda}{2}||d - \Phi(u)||_2^2) \tag{2.7}$$

Hierbei ersetzt der letzte Term die Nebenbedingung $d = \Phi(u)$, denn im Idealfall ist die $l2$-Norm gleich Null und dann gilt bereits $d = \Phi(u)$.

Nun wenden wir auf (2.7) die Bregmaniteration (2.4) und (2.5) an und setzen $J(u, d) = ||d||_1 + H(u)$. Dann erhalten wir folgende Iterationen:

$$
\begin{aligned}
(u^{k+1}, d^{k+1}) &= \arg\min_{u,d}(D_J^p(u, u^k, d, d^k) + \frac{\lambda}{2}||d - \Phi(u)||_2^2) \\
&= \arg\min_{u,d}(J(u,d) - \langle p_u^k, u - u^k\rangle - \langle p_d^k, d - d^k\rangle + \frac{\lambda}{2}||d - \Phi(u)||_2^2) \\
p_u^{k+1} &= p_u^k - \lambda(\nabla\Phi)^T(\Phi u^{k+1} - d^{k+1}) \\
p_d^{k+1} &= p_d^k - \lambda(d^{k+1} - \Phi u^{k+1})
\end{aligned}
$$

Nun führen wir eine neue Variable $b^k = \frac{p_d^k}{\lambda}$ ein. Dann erhält man $p_d^k = \lambda b^k$ und $p_u^k = -\lambda\Phi^T b^k$. In die oberen Iterationen eingesetzt, ergibt dies die neuen Iterationen.

$$
\begin{aligned}
(u^{k+1}, d^{k+1}) =& \arg\min_{u,d}(J(u,d) - \lambda\langle b^k, d - d^k\rangle - \\
& \lambda\langle b_d^k, \Phi(u - u^k)\rangle + \frac{\lambda}{2}||d - \Phi(u)||_2^2) \\
b^{k+1} =& b^k - d^{k+1} + \Phi u^{k+1}
\end{aligned}
$$

Dafür gelten die Startwerte $u^0 = 0$, $d^0 = 0$ und $b^0 = 0$.

Nun kann man die Iterationen für d^{k+1} und u^{k+1} nacheinander ausführen, sodass man einzelne Iterationen für diese beiden Werte erhält. Als erstes halten wir u^k fest um d^{k+1} zu erneuern und dann halten wir dieses neue d^{k+1} fest um u^{k+1} zu erhalten. Dadurch erhalten wir drei Iterationen, die man als die allgemeinen Split-Bregman-Iterationen bezeichnet.

Definition 2.2.4. *Für J linear, H linear und differenzierbar mit $H \geq 0$ und Φ differenzierbar sind folgende Iterationen die allgemeinen Split-Bregman-Iterationen:*

$$d^{k+1} = \arg\min_d(\frac{1}{\lambda}J(d) - \langle b^k, d - d^k\rangle + \frac{1}{2}||d - \Phi(u^k)||_2^2) \tag{2.8}$$

$$u^{k+1} = \arg\min_u(\frac{1}{\lambda}H(u) - \langle b^k, \Phi(u - u^k)\rangle + \frac{1}{2}||d^{k+1} - \Phi(u)||_2^2) \tag{2.9}$$

$$b^{k+1} = b^k - d^{k+1} + \Phi(u^{k+1}) \tag{2.10}$$

Als Startwerte wählt man $u^0 = 0$, $d^0 = 0$ und $b^0 = 0$.

Ist nun J die $l1$-Norm, dann hat das erste Unterproblem eine explizite Lösung. Das zweite Unterproblem ist einfach zu lösen, da wir H als differenzierbar vorausgesetzt

haben. In den folgenden Abschnitten wollen wir die Split-Bregman-Methode verwenden um Probleme beim Entmischen von Hyperspektralbildern und in der Sprachextraktion zu lösen.

3. Anwendung auf Hyperspektralbilder

3.1. Einführung in Entmischung für Hyperspektralbilder

Ein normales RGB-Bild wird durch seine Anteile in Rot, Grün und Blau in jedem Pixel beschrieben. Es lässt sich mathematisch als dreidimensionale Matrix $f \in \mathbb{R}^{m \times n \times 3}$ darstellen. Dabei steht $f_1 \in \mathbb{R}^{m \times n}$ für den Rotanteil, $f_2 \in \mathbb{R}^{m \times n}$ für den Grünanteil und $f_3 \in \mathbb{R}^{m \times n}$ für den Blauanteil in einem Bild.

Bei einem Hyperspektralbild nimmt man nun nicht nur die drei Anteile Rot, Grün und Blau auf, sondern Anteile, die über das ganze elektromagnetische Spektrum verteilt sind. Das heißt, man misst nicht nur die elektromagnetischen Wellen des sichtbaren Lichts im Bereich von 400 - 700 nm, sondern unter anderem auch den ultravioletten und infraroten Bereich, Röntgenstrahlung und Mikrowellen.
Diese kann man nicht mehr mit einer normalen Kamera aufnehmen oder mit dem menschlichen Auge sehen. Man benötigt hyperspektrale Sensoren. Ein Hyperspektralbild lässt sich also durch eine b-dimensionale Matrix $f \in \mathbb{R}^{m \times n \times b}$ darstellen, wobei b für die Anzahl der Bänder steht, die gemessen werden. Dabei ist jeder Pixel $f_i \in \mathbb{R}^b$ ein Vektor der Länge b.

Nun hat jedes Material durch seine physikalischen Eigenschaften ein einzigartiges elektromagnetisches Spektrum. Dieses entsteht, wenn Licht von einer Lichtquelle, in unserem Fall der Sonne, auf das Material fällt. Durch die physikalischen Eigenschaften des Materials wird nicht das komplette Sonnenlicht reflektiert, sondern nur ein bestimmter Anteil. Der Rest wird absorbiert. Dieser reflektierte Anteil wird dann von Sensoren gemessen. Allein mit Hilfe dieses Spektrums kann man ein Material eindeutig identifizieren. Diese Eigenschaften werden unter anderem in den Geowissenschaften, der Landwirtschaft und beim Militär eingesetzt.

Ein Problem dabei ist allerdings, dass spektrale Sensoren im Allgemeinen eine schlechte räumliche Auflösung haben und so passiert es häufig, dass mehr als ein einzelnes Material in einem Pixel f_i vorkommt.

Um diese Situation nun mathematisch zu modellieren (vgl. [4]), wollen wir ein lineares Mischmodell betrachten. Dazu treffen wir die grundlegende Annahme, dass eine kleine Anzahl von Materialien die Oberfläche dominiert. Diese Materialen mit gut unterscheidbaren spektralen Signaturen nennen wir Endmember. Weiter nehmen wir an, dass diese Endmember räumlich getrennt auf der Oberfläche angeordnet sind. Dann können wir davon ausgehen, dass ein Bündel einfallender Strahlung immer nur mit einem Material interagiert.

Die Anteile, in denen das jeweilige Material in einem Pixel auftaucht, wollen wir Fractional Abundances nennen. Diese Anteile sind proportional zum Anteil des Materials auf der Oberfläche. Die von der Oberfläche reflektierte Strahlung übermittelt nun die Eigenschaften des Materials mit den gleichen Proportionen, wie es auf der Oberfläche vorkommt.
Damit herrscht eine lineare Beziehung zwischen den Fractional Abundances und dem Spektrum der reflektierten Strahlung. Mit diesen Informationen können wir ein lineares Mischmodell einführen.

Jeder Pixel f_i kann als Linearkombination der k Endmember $E_1, ..., E_k \in \mathbb{R}^b$ geschrieben werden:

$$f_i = a_1^i E_1 + ... + a_k^i E_k = \sum_{i=1}^{k} a^i E_i$$

Dabei ist $a^i = (a_1^i, ..., a_k^i)$ der Vektor der Fractional Abundances.

Für ein Bild mit n Pixeln ergibt sich

$$F = EA$$

wobei $F = (f_1, ..., f_n) \in \mathbb{R}^{b \times n}$, $E = (E_1, ..., E_k) \in \mathbb{R}^{b \times k}$ und $A = (a^1, ..., a^k) \in \mathbb{R}^{k \times n}$ sind. Dabei ist b die Anzahl der spektralen Bänder, n die Anzahl der Pixel und k die Anzahl der Endmember.

Für dieses Modell wollen wir eine Nichtnegativitätsbeschränkung $a_j^i \geq 0$ annehmen. Diese ist sinnvoll, da negative Anteile von Materialien auf der Oberfläche physikalisch keinen Sinn ergeben. Man kann zusätzlich die Summenbeschränkung $\sum_{j=1}^{k} a_j^i = 1$ fordern. Ob das sinnvoll ist, wollen wir zu einem späteren Zeitpunkt diskutieren.

Beim Entmischen handelt es sich nun um die Zerlegung eines gemischten Messwertes, in diesem Falle eines Pixels, in die Endmember, also eine Menge von einzelnen Spektren, und die zugehörigen Fractional Abundances. Dies ist notwendig, da in einem Pixel oft mehrere Materialien vorkommen. Einerseits kann das durch die schlechte räumliche Auflösung der Messgeräte passieren, andererseits durch Gegebenheiten des aufgenommenen Geländes, wie zum Beispiel einer homogenen Mischung mehrerer Materialien.

Beim Entmischen geht man in drei Schritten (vgl. [4]) vor. Zuerst kann man die Dimension der gemessenen Daten reduzieren, dann erfolgt die Endmemberdetermination und zuletzt die Inversion. Wir wollen kurz auf diese drei Schritte eingehen.

Die Reduzierung der Dimension ist ein Schritt, der nicht unbedingt notwendig ist. Man führt ihn durch, da die Daten oft eine sehr hohe Dimension haben und sich durch die Reduzierung Rechenzeit sparen lässt. Als Konsequenz muss man dann einen erhöhten Fehler in Kauf nehmen.

Der nächste Schritt ist die Endmemberdetermination. Hier geht es darum, die Spalten der Endmembermatrix E zu bestimmen. Dazu gibt es mehrere Möglichkeiten. Zum einen die empirische Variante, zum anderen die automatische. Dabei ist zu beachten, dass die einzelnen Endmember linear unabhängig sein müssen, damit die Lösung für das Lineare Mischmodell eindeutig ist.
Bei der empirischen Abschätzung der Endmember, werden diese durch Beobachtung und physikalische Intuition abgeschätzt. Für die automatische Determination gibt es unterschiedliche Algorithmen. Im weiteren Verlauf wollen wir davon ausgehen, dass die Endmember schon bestimmt wurden.

Der letzte Schritt beim Entmischen ist die Inversion, bei der man die Fractional Abundances a^i sucht. Dabei sind die geforderten Nebenbedingungen, vor allem $a^i \geq 0$, zu erfüllen.

Eine erste intuitive Möglichkeit die a^i zu bestimmen ist die Methode der kleinsten Quadrate. Man würde einfach

$$\min ||Ea - f||_2$$

berechnen. Dafür existiert die bekannte Lösung

$$a = (E^T E)^{-1} E^T f$$

Dabei stellt sich allerdings das Problem, dass die a^i häufig negativ sind. Außerdem wird die so erhaltende Lösung im Regelfall dicht sein. Wie oben schon erläutert, suchen wir aber dünnbesetzte Lösungen.

Im Folgenden wollen wir daher zwei unterschiedliche Algorithmen betrachten, die uns die geforderte Sparsity liefern. Der erste ist ein Ansatz von Greer [2], der die $l0$-Norm minimiert, was, wie im 1. Kapitel erläutert, die intuitive Möglichkeit ist um dünnbesetzte Lösungen zu erhalten.
Der zweite ist ein Algorithmus von Guo, Wittman und Osher [7], der mit $l1$-Regularisierung arbeitet und dafür das oben erläuterte Split-Bregman-Verfahren anwendet.

3.2. Entmischung mit einem gierigen Algorithmus

Für das Finden des Abundance-Vektors a, wollen wir folgendes minimieren (vgl. [2]):

$$a = \arg\min_{a'} \{||a'||_0 \mid ||Ea' - f||_2 < \epsilon ||f||_2\}$$

$$\text{mit} \quad a'_i \geq 0 \quad \text{und} \quad \sum_{i=1}^{k} a'_i = 1$$

Dabei zählt $||a'||_0$ die Anzahl der Nulleinträge in unserem gesuchten Vektor. Da wir eine dünnbesetzte Lösung suchen, ist es natürlich, wie oben schon erläutert, ein sinnvoller Ansatz die $l0$-Norm zu minimieren, um möglichst wenige Einträge ungleich Null zu haben. Der Parameter ϵ kontrolliert dabei, wie stark die Lösung vom gemessenen Bild abweichen darf. Wir fordern hier, dass die Summenbeschränkung gilt, da sich die Anteile der Materialien in einem Pixel zu Eins aufaddieren sollten.

Da es sich hier um ein nicht konvexes Problem handelt, ist es sehr schwer eine explizite Lösung anzugeben. Wir wollen uns also mit einem Algorithmus beschäftigen, der eine Näherung für das Problem angibt.

Um zu betrachten wie dieser Algorithmus arbeitet, wollen wir uns zuerst mit einer geometrischen Darstellung der gemessenen Pixel befassen. Dazu definieren wir den Begriff eines Simplex.

Definition 3.2.1. *Ein n-Simplex ist ein n-dimensionales Polytop, welches die konvexe Hülle seiner n+1 Ecken ist:*

$$\triangle^n = \{(t_0, ..., t_n) \in \mathbb{R}^{n+1} | \sum_{i=0}^{n} t_i = 1 \text{ und } t_i \geq 0 \ \forall i\}$$

Nun lassen sich die Pixel als Punkte in einem kartesischen Koordinatensystem schreiben. Dabei steht jede Koordinatenachse für ein spektrales Band. Also liegen die Punkte bei n Spektralbändern im $\mathbb{R}^n$ und die Koordinaten des Punktes sind durch die jeweiligen Werte im zugehörigen Spektralband gegeben.

Die jeweiligen Endmember bilden einen konvexen Kegel, in den die Pixel fallen, da sie positive Linearkombinationen der Endmember sind. Da wir in diesem Ansatz davon ausgehen, dass $\sum a_i' = 1$, fallen alle Punkte auf einen Simplex. Je nachdem wie groß der jeweilige Anteil der Endmembern in diesem Pixel ist, fallen die Punkte entweder auf die Ecken, Kanten oder ins Innere des Simplex.

Da wir davon ausgehen, dass die korrekte Lösung dünnbesetzt ist, sollten die meisten Punkte auf Ecken und Kanten des Simplex fallen und in die Mitte nur wenige. Da aber die Messungen, wie oben schon erläutert, oft fehlerhaft sind, verschieben sich die Punkte und fallen entweder ins Innere des Simplex, in den konvexen Kegel, den die Endmember aufspannen oder sie liegen ganz außerhalb.

Der Algorithmus von Greer versucht nun aus diesen fehlerhaften Messungen wieder eine dünnbesetzte Lösung zu erstellen. Dazu gehen wir davon aus, dass die Endmember schon bekannt sind.

Als erstes berechnet man die Lösung von $\min ||Ea - f||_2$ mit $a_i \geq 0$ und $\sum a_i = 1$ über die gesamte Menge der Endmember. Dies liefert aber eine dichte Lösung, denn die Punkte aus dem konvexen Kegel werden orthogonal auf das Simplex verschoben und nicht auf die Kante, wo sie wahrscheinlicher liegen sollten. Nun entfernt man den Endmember, der zur kleinsten Komponente von a gehört und berechnet noch einmal die Lösung von $\min ||Ea - f||_2$ mit $a_i \geq 0$ und $\sum a_i = 1$, aber diesmal über die kleinere Menge der Endmember. Dies führt man sukzessive fort, bis man den vorher festgelegten Genauigkeitsbereich verlassen hat. Dadurch werden die Lösungen innerhalb des Simplex auf die Ecken und Kanten verschoben, zu denen sie am Anfang nahe lagen, da wir davon ausgehen, dass die wirklichen Punkte dort liegen sollten und nur durch Messfehler verschoben wurden.

Am Ende dieses Algorithmus hat man eine dünnbesetze Lösung für die Fractional Abundances.

Der Algorithmus sieht folgendermaßen aus:

Input:

- $(b \times k)$-Matrix E mit k Endmembern im $\mathbb{R}^b$
- Signal f
- Parameter ϵ

Output:

- Abundance Vektor a

Algorithmus:

1. $E^1 = E$
2. Berechne die Lösung von a^1 durch Minimierung von $||E^1 a^1 - f||_2$ mit $\sum a_j^1 = 1$ und $a_j^1 \geq 0$
3. Setze $\delta = ||E^1 a^1 - f||_2$
4. $i = 1$
5. Solange $\delta < \epsilon$
    - Erstelle E^{i+1} durch Entfernen des Endmembers in E^i, der zur kleinsten Komponente vom Abundance-Vektor a^i gehört
    - Berechne die Lösung von a^{i+1} durch Minimierung von $||E^{i+1} a^{i+1} - f||_2$ mit $\sum a_j^{i+1} = 1$ und $a_j^{i+1} \geq 0$
    - Setze $\delta = ||E^{i+1} a^{i+1} - f||_2$

- $i = i + 1$

6. Gebe $a = a^{i-1}$ aus

3.3. $l1$-Regularisierung

Wie oben schon besprochen, ist die Minimierung der $l0$-Norm nicht einfach, da es sich dabei um ein nicht konvexes Problem handelt. Als alternative Herangehensweise wurde die $l1$-Regularisierung vorgestellt. Für den Fall des hyperspektralen Entmischens ergibt sich folgendes Modell von Guo, Wittman und Osher (vgl. [7]):

$$a^* = \arg\min_a ||a||_1 + \frac{\lambda}{2}||Ea - f||_2^2 \quad mit\ a_i \geq 1 \tag{3.1}$$

Dabei kontrolliert λ die Sparsity der Lösung und wie sehr die Endmember das ursprüngliche Signal f beschreiben.

Im Gegensatz zu dem oben vorgestellten gierigen Algorithmus von Greer wollen wir hier nicht die Summenbeschränkung erzwingen. Einerseits sorgt die $l1$-Norm dafür, dass die Summe aller Einträge der Fractional Abundances nahe 1 ist, denn wenn $a_i \geq 1$, dann gilt $||a||_1 = \sum a_i$ und wir minimieren ja die $l1$-Norm. Andererseits ist es nicht unbedingt sinnvoll, dass sich die Fractional Abundances zu 1 aufaddieren, da sie keine prozentualen Anteile von Materialien in einem Pixel darstellen.

Um diesen Term zu minimieren wollen wir das Split-Bregman-Verfahren aus Definition 2.2.4 verwenden. In diesem Fall gilt dann

$$J(a) = ||a||_1$$
$$\Phi = Id$$
$$H(a) = \frac{\lambda}{2}||Ea - f||_2^2 \quad \text{mit} \quad H \quad \text{ist differenzierbar}$$

Dazu wählen wir die Startwerte

$$d^0 = 0, \quad a^0 = 0, \quad b^0 = 0$$

Wir erhalten damit folgende Iterationen:

$$d^{k+1} = \arg\min_d \frac{1}{\lambda}||d||_1 - \langle b^k, d - d^k\rangle + \frac{1}{2}||d - a^k||_2^2 \tag{3.2}$$

$$a^{k+1} = \arg\min_a \frac{\mu}{2\lambda}||Ea - f||_2^2 + \langle b^k, a - a^k\rangle + \frac{1}{2}||d^{k+1} - a||_2^2 \tag{3.3}$$

$$b^{k+1} = b^k - (d^{k+1} - a^{k+1}) \tag{3.4}$$

Um mit diesen Iterationen zu rechnen, müssen wir sie noch explizit bestimmen. Für (3.4) ist das nicht nötig. Daher wollen wir uns als erstes mit (3.3) beschäftigen. Wir wissen, dass für ein Minimum gelten muss, dass die Ableitung an der entsprechenden Stelle Null ist. Daher wollen wir die Ableitung von (3.3) bestimmen und gleich Null setzen. Daraus erhalten wir eine explizite Formel für die Iteration. Wir bestimmen die Ableitungen der drei Summanden einzeln und setzen sie dann zusammen.

Betrachte also zuerst den Differenzenquotienten von $S_1(a) = \frac{\mu}{2\lambda}||Ea - f||_2^2$:

$$\begin{aligned}
\frac{\frac{\mu}{2\lambda}||E(a+\epsilon h) - f||^2 - \frac{\mu}{2\lambda}||Ea - f||^2}{\epsilon} &= \frac{\frac{\mu}{2\lambda}||Ea - f + \epsilon Eh||^2 - \frac{\mu}{2\lambda}||Ea - f||^2}{\epsilon} \\
&= \frac{\frac{\mu}{2\lambda}||Ea - f||^2 + \langle Ea - f, \epsilon Eh\rangle + ||\epsilon Eh||^2 - \frac{\mu}{2\lambda}||Ea - f||^2}{\epsilon} \\
&= \frac{\frac{\mu}{\lambda}\langle Ea - f, \epsilon Eh\rangle + \frac{\mu}{2\lambda}||\epsilon Eh||^2}{\epsilon} = \frac{\mu}{\lambda}\langle Ea - f, Eh\rangle + \frac{\epsilon\mu}{\lambda}||Eh||^2
\end{aligned}$$

Nun muss für $\epsilon \to 0$ für alle h gelten

$$\begin{aligned}
S_1'(a)h &= \frac{\mu}{\lambda}\langle Ea - f, Eh\rangle \\
&= \frac{\mu}{\lambda}\langle E^T(Ea - f), h\rangle
\end{aligned}$$

welches die Richtungsableitung ist. Damit ist unsere gesuchte Ableitung

$$S_1'(a) = \frac{\mu}{\lambda}E^T(Ea - f)$$

Nun betrachten wir den Differenzenquotienten von $S_2(a) = \langle b^k, a - a^k \rangle$:

$$\frac{b^k, a + \epsilon h - a^k\rangle - \langle b^k, a - a^k\rangle}{\epsilon} = \frac{\langle b^k, a - a^k\rangle + \langle b^k, \epsilon h\rangle - \langle b^k, a - a^k\rangle}{\epsilon}$$
$$= \frac{\langle b^k, \epsilon h\rangle}{\epsilon} = \langle b^k, h\rangle$$

Nun muss gelten

$$S_2'(a)h = \langle b^k, h\rangle$$

und unsere gesuchte Ableitung ist

$$S_2'(a) = b^k$$

Damit bleibt noch übrig, den Differenzenquotienten von $S_3(a) = \frac{1}{2}||d^{k+1} - a||^2$ zu betrachten:

$$\frac{\frac{1}{2}||d^{k+1} - (a + \epsilon h)||^2 - \frac{1}{2}||d^{k+1} - a||^2}{\epsilon}$$
$$= \frac{\frac{1}{2}||d^{k+1} - a||^2 - 2\langle d^{k+1} - a, \epsilon h\rangle + ||eh||^2 - \frac{1}{2}||d^{k+1} - a||^2}{\epsilon}$$
$$= \frac{\langle -(d^{k+1} - a), \epsilon h\rangle + \frac{1}{2}||eh||^2}{\epsilon} = \langle -(d^{k+1} - a), h\rangle + \frac{\epsilon}{2}||h||^2$$

Für $\epsilon \to 0$ gilt für alle h $S_3'(a)h = \langle -(d^{k+1} - a), h\rangle$ und unsere gesuchte Ableitung ist

$$S_3'(a) = -(d^{k+1} - a)$$

Dies ergibt zusammen folgende Optimalitätsbedingung

$$\frac{\mu}{\lambda}E^T(Ea - f) + b^k - d^{k+1} + a = 0$$
$$\Leftrightarrow E^TEa - E^Tf + \frac{\lambda}{\mu}b^k - \frac{\lambda}{\mu}d^k + 1 + \frac{\lambda}{\mu}a = 0$$
$$\Leftrightarrow a = (E^TE + \frac{\lambda}{\mu}Id)^{-1}(E^Tf + \frac{\lambda}{\mu}(d^{k+1} - b^k))$$

Nun müssen wir noch die Iteration für

$$d^{k+1} = \arg\min_d \frac{1}{\mu}||d||_1 - \langle b^k, d - d^k\rangle + \frac{1}{2}||d - a^k||_2^2$$

explizit berechnen. Wir nehmen dazu an, dass d die optimale Lösung des Problems ist. Nun machen wir eine Fallunterscheidung, um die jeweiligen Optimalitätsbedingungen zu bestimmen.

1. Fall: $d > 0$

Für das Subdifferential des Betrages $p \in \partial||d||_1$ gilt $p = 1$ für $d > 0$. Also ergibt sich zusammen mit den anderen Ableitungen als Optimalitätsbedingung

$$\begin{aligned} \frac{1}{\lambda} - b^k + d - a^k &= 0 \\ \Rightarrow d = a^k + b^k - \frac{1}{\lambda} &> 0 \\ \Rightarrow a^k + b^k &> \frac{1}{\lambda} \end{aligned}$$

2. Fall: $d < 0$

Hier gilt für das Subdifferential des Betrages $p = -1$ und es ergibt sich als Optimalitätsbedingung

$$\begin{aligned} -\frac{1}{\lambda} - b^k + d - a^k &= 0 \\ \Rightarrow d = a^k + b^k + \frac{1}{\lambda} &< 0 \\ \Rightarrow a^k + b^k &< -\frac{1}{\lambda} \end{aligned}$$

3. Fall: $d = 0$

Das Subdifferential ist gegeben durch $\{|p| \leq 1\}$ und es ergibt sich als Optimalitätsbedingung

$$|a^k + b^k| \leq \frac{1}{\lambda}$$

Also ergibt sich zusammen

$$d^{k+1} = sgn(a^k + b^k)\max(|a^k + b^k| - \frac{1}{\lambda}, 0) = shrink(a^k + b^k, \frac{1}{\lambda})$$

Damit ergibt sich zusammen folgendes Iterationsschema:

Initialisiere: $a^0 = 0$, $d^0 = 0$, $b^0 = 0$

While $||a^{k+1} - a^k||_2 // ||a^{k+1}||_2 > \epsilon$

1. $d^{k+1} = shrink(a^k + b^k, \frac{1}{\lambda})$
2. $a^{k+1} = (E^T E + \frac{\lambda}{\mu} Id)^{-1}(E^T f + \frac{\lambda}{\mu}(d^{k+1} - b^k))$
3. $b^{k+1} = b^k - d^{k+1} - a^{k+1}$

end While

Nachdem wir nun zwei Möglichkeiten zur Bestimmung der Fractional Abundances beim Entmischen von Hyperspektralbildern betrachtet haben, wollen wir im nächsten Abschnitt zur zweiten Anwendung, der Sprachextraktion, übergehen.

4. Anwendung in der Sprachextraktion

4.1. Einführung in die Sprachextraktion und Modell

Wir wollen nun zum zweiten Anwendungsgebiet, der Sprachextraktion, übergehen. Genau wie bei elektromagnetischen Wellen handelt es sich auch bei akustischen Signalen um Wellen. Sie überlagern sich also genau wie diese und man kann ähnlich mit ihnen arbeiten.

Bei den Hyperspektralbildern haben wir unseren Schwerpunkt auf die Inversion des Problems gelegt. Es ging darum, die Fractional Abundances zu finden, also die Anteile von reinen Materialien bzw. Endmembern in einem Pixel. Dabei waren wir davon ausgegangen, dass die Endmember uns schon bekannt sind. Dieses Problem haben wir mit l1-Regularisierung gelöst.

Bei der Sprachextraktion ist es unser Ziel, einen bestimmten Sprecher aus mehreren Aufnahmen der gleichen Situation herauszufiltern. Hier kann man einige Parallelen zum Entmischen bei hyperspektralen Bildern ziehen.

Ein gemessener Pixel $f_i \in \mathbb{R}^b$ entspricht der Aufnahme $x_i(t)$ mit einem Mikrofon. Dies sind unsere Daten, die wir entmischen wollen. Die Endmember, also die reinen Materialien, entsprechen den einzelnen Sprach- bzw. Geräuschquellen. Wir wollen hier eine davon aus den Mischungen rekonstruieren. Die Fractional Abundances tauchen auch hier auf und geben an, wie sehr sich das Signal der ursprünglichen Quelle durch die Eigenschaften des Raumes verändert, bis es beim Mikrofon angekommen ist.

Der Schwerpunkt bei der Sprachextraktion liegt noch viel stärker auf Blind Source Separation (BSS), also dem Entmischen, wenn die Endmember nicht bekannt sind, als

bei den hyperspektralen Bildern. Bei diesen war es auch möglich, die Endmember von Hand zu finden oder in Datenbanken nachzuschlagen und die Fractional Abundances waren von großem Interesse. Bei der Sprachextraktion möchte man aber eines oder mehrere der ursprünglichen Sprachsignale, also sozusagen Endmember, finden. Dafür ist es aber trotzdem entscheidend, die Faktoren zu finden, die angeben, wie sich die Signale auf dem Weg zum Mikrofon verändert haben. Wir wollen uns auf diesen Aspekt konzentrieren und ihn wieder mit $l1$-Regularisierung lösen.

Bei dem Mischmodell, welches wir gleich aufstellen wollen, ergibt sich noch ein weiterer Unterschied zu dem Modell für das Entmischen von hyperspektralen Bildern. Die Mischung wird in diesem Falle gefaltet, da die Signale nicht nur von der aktuellen Zeit abhängen, sondern auch von der Vergangenheit. Dies kommt durch die entstehende Zeitverzögerung bei der Schallausbreitung im Raum und den Reflektionen des Schalls von Objekten, besonders in geschlossenen Umgebungen, zustande.

Wir wollen nun das schnelle Sprachextraktionsmodell betrachten. Dazu nehmen wir an, dass zwei Sensoren und zwei Soundquellen gegeben sind, dabei sollte mindestens eine Quelle ein Sprachsignal sein. Bezeichne nun s_T das gesuchte Zielsprachsignal und s_B die Hintergrundstörung. Dann lässt sich folgendes Mischmodell von Osher (vgl.[3]) aufstellen:

$$x_i(t) = h_{i1} * s_B(t) + h_{i2} * s_T(t) \tag{4.1}$$

für $i = 1, 2$ und t sei der Parameter für die Zeit.

Nun wollen wir das Sprachsignal s_T erhalten. Dazu werden wir versuchen die Störung s_B zu elimieren um nur noch das gewünschte Sprachsignal zu erhalten. Dabei wollen wir die Annahme machen, dass das Zielsprachsignal Pausen enthält. Dies ist für gesprochene Sprache eine sinnvolle Annahme, denn zwischen den einzelnen Wörten und am Ende von Sätzen macht jeder Mensch kurze Pausen beim Sprechen. Ansonsten wäre Sprache nicht zu verstehen. Während dieser kurzen Pausen wollen wir die Struktur der Hintergrundstörung analysieren und diese Information benutzen, um sie dann zu eliminieren.

Wir definieren dazu D als Vereinigung von disjunkten Zeitintervallen mit $s_T \approx 0$ und Störung s_B aktiv. Dann haben wir mit (4.1) zwei Gleichungen für die beiden

Mischungen, die wir erhalten:

$$x_1(t) = h_{11} * s_B(t) + h_{12} * s_T(t)$$
$$x_2(t) = h_{21} * s_B(t) + h_{22} * s_T(t)$$

Nun wählen wir $t \in D$, woraus folgt $s_T(t) = 0$. Damit vereinfachen sich die Gleichungen zu

$$x_1(t) = h_{11} * s_B(t)$$
$$x_2(t) = h_{21} * s_B(t)$$

Wenn wir nun die erste Gleichung auf beiden Seiten mit h_{21} und die zweite mit h_{11} falten, dann erhalten wir

$$h_{21} * x_1(t) = h_{21} * h_{11} * s_B(t)$$
$$h_{11} * x_2(t) = h_{21} * h_{11} * s_B(t)$$

Daraus folgt dann

$$h_{21} * x_1(t) - h_{11} * x_2(t) \approx 0$$

da $s_T(t)$ nur ungefähr Null ist für $t \in D$.

Wir suchen nun ein Paar Sparsity-Filter u_i, die die Energie von $u_2 * x_1 - u_1 * x_2$ in D minimieren. Dabei gilt idealerweise $u_1 \approx h_{11}$ und $u_2 \approx h_{21}$. Die Lösungen sollen also dünnbesetzte akustische Raumimpulsantworten sein. Um die Sparsity dieser Filter zu erreichen, wenden wir wieder die $l1$-Regularisierung an, was uns auf folgendes konvexes Optimierungsproblem führt

$$(u_1^*, u_2^*) = \arg\min_{u_1,u_2}(\frac{1}{2}||u_2 * x_1 - u_1 * x_2||_2^2 + \frac{\eta^2}{2}(\sum_{i=1}^{2} u_1(1) - 1)^2 + \mu(||u_1||_1 + ||u_2||_1)) \quad (4.2)$$

Hierbei verhindert der zweite Term Nulllösungen, der dritte reguliert die Sparsity der Lösung. Dieses Problem lässt sich wie folgt in Matrixform umschreiben:

$$u^* = \arg\min_{u}(\frac{1}{2}||Au - f||_2^2 + \mu||u||_1) \quad (4.3)$$

Sei nun L_D die Länge von D und L die Länge von u_i. Dann hat die $(L_D+1)\times 2L$-Matrix A die folgende Form:

$$A = \begin{pmatrix} x_1(1) & x_1(2) & \dots & \dots & x_1(L_D-1) & x_1(L_D) & \eta \\ & x_1(1) & \dots & \dots & x_1(L_D-2) & x_1(L_D-1) & 0 \\ & & \ddots & & & \vdots & \vdots \\ & & & x_1(1) & \dots & x_1(L_D-L+1) & 0 \\ -x_2(1) & -x_2(2) & \dots & \dots & -x_2(L_D-1) & -x_2(L_D) & \eta \\ & -x_2(1) & \dots & \dots & -x_2(L_D-2) & -x_2(L_D-1) & 0 \\ & & \ddots & & & \vdots & \vdots \\ & & & -x_2(1) & \dots & -x_2(L_D-L+1) & 0 \end{pmatrix}^T$$

u^* ist definiert durch $u^* = (u_1, u_2)^T$ und der Vektor f mit Länge $L_D + 1$ ist definiert durch

$$f = (0, 0, \dots, \eta)^T$$

Da für $t \in D$ die Relation $h_{21} * x_1(t) - h_{11} * x_2(t) \approx 0$ gilt, folgt, dass für $t \notin D$ gilt, dass $h_{21} * x_1(t) - h_{11} * x_2(t) \approx s_T(t)$. Dies lässt sich mit Hilfer der Sparsity-Filter auch schreiben als $\hat{s}_T(t) = u_2 * x_1 - u_1 * x_2 \approx h_{21} * x_1(t) - h_{11} * x_2(t)$. Nun lässt sich folgende Umformung machen:

$$\begin{aligned} & h_{21} * x_1(t) - h_{11} * x_2(t) \\ =& h_{21} * (h_{11} * s_B(t) + h_{12} * s_T(t)) - h_{11} * (h_{21} * s_B(t) + h_{22} * s_T(t)) \\ =& h_{21} * h_{11} * s_B(t) + h_{21} * h_{12} * s_T(t) - h_{11} * (h_{21} * s_B(t) - h_{11} * h_{22} * s_T(t) \\ =& h_{21} * h_{12} * s_T(t) - h_{11} * h_{22} * s_T(t) \\ =& (h_{21} * h_{12} - h_{11} * h_{22}) * s_T(t) \end{aligned}$$

Damit ergibt sich als Formel für unser gesuchtes Sprachsignal

$$\hat{s}_T(t) \approx (h_{21} * h_{12} - h_{11} * h_{22}) * s_T(t). \tag{4.4}$$

Hierbei klingt $\hat{s}_T(t)$ für das menschliche Ohr fast genauso wie $s_T(t)$, wie sich später bei der Anwendung zeigen wird. Außerdem konnte die Hintergrundstörung $s_B(t)$ herausgefiltert werden. Hierbei sind wir von der Annahme ausgegangen, dass die Hintergrundgeräusche sich nicht großartig ändern und die gefundenen Schätzer für h_{11} und h_{21} auch gelten, wenn $t \notin D$.

4.2. $l1$-Regularisierung zur Lösung des Problems

Wir wollen nun wieder mit dem Split-Bregman-Verfahren aus Definition 2.2.4 die Sparsity-Filter u_1 und u_2 finden. Dabei setzen wir

$$J(u) = \mu||u||_1$$
$$\Phi = Id$$
$$H(u) = \frac{1}{2}||Au - f||_2^2$$

Dadurch erhalten wir folgende Iterationen

$$d^{k+1} = \arg\min_d(\frac{\mu}{\lambda}||d||_1 - \langle b^k, d - d^k\rangle + \frac{1}{2}||d - \alpha^k||_2^2) \tag{4.5}$$

$$u^{k+1} = \arg\min_u(\frac{1}{2\lambda}||Au - f||_2^2 + \langle b^k, u - u^k\rangle + \frac{1}{2}||d^{k+1} - u||_2^2) \tag{4.6}$$

$$b^{k+1} = b^k - (d^{k+1} - u^{k+1}) \tag{4.7}$$

mit den Startwerten $d^0 = 0$, $u^0 = 0$ und $b^0 = 0$.

Rechnet man dies, wie oben schon gezeigt, explizit nach, dann kommt man auf folgendes Iterationsschema

Initialisiere $u^0 = 0$, $d^0 = 0$, $b^0 = 0$

While $||u^{k+1} - u^k||_2/||u^{k+1}||_2 > \epsilon$

- $d^{k+1} = shrink(u^k + b^k, \frac{\lambda}{\mu})$
- $u^{k+1} = (\lambda Id + A^TA)^{-1}(A^Tf + \lambda(d^{k+1} - b^k))$
- $b^{k+1} = b^k - (d^{k+1} - u^{k+1})$

end While

Um den Algorithmus komplett zu machen, würde jetzt noch das Festestellen der Zeit fehlen, in der die Quelle $s_T \approx 0$ ist. Ansonsten kann man die Matrix A nicht aufstellen. In [3] wird dazu die Methode DUET vorgestellt. Da das aber nicht mehr mit der Inversion des Problems mit Hilfe von $l1$-Regularisierung zusammenhängt und sehr aufwendig ist, wollen wir uns hier nicht damit beschäftigen.

5. Implementierung

5.1. Beschreibung des Programms

Wir wollen nun ein Programm zum Entmischen von zwei Soundquellen aus zwei Mischungen mit Split-Bregman betrachten. Damit das Ergebnis kontrollierbar ist, werden die zwei Mischungen im Programm erzeugt und nicht aufgenommen.
Als erstes liest das Programm zwei Audiodateien ein, wovon eine als gesuchtes Signal s_T und die andere als Hintergrundstörung s_B definiert wird. Um zu gewährleisten, dass die benötigte Zeit D, in der $s_T \approx 0$ ist, bekannt ist, wird s_T zeitlich verschoben, sodass, wenn die beiden gemischt werden, zuerst nur s_B zu hören ist. Als nächstes werden die beiden Quellen gemischt. Dazu werden als erstes die Vorfaktoren h_{ij} erstellt, indem an einer vorher festgegeben Anzahl zufälliger Stellen die Nulleinträge eines Vektors durch normalverteilte Zufallszahlen ersetzt werden. Mit diesen Vorfaktoren werden dann die beiden Mischungen anhand des Modells (4.1) erstellt.
Zum Entmischen werden nun erst die Matrix A und der Vektor f bestimmt und dann wird Split-Bregman angewandt, um den Sparsity-Filter u zu finden. Zum Schluss wird die entmischte Quelle $\hat{s_T}$ nach der Formel $\hat{s}_T(t) = u_2 * x_1 - u_1 * x_2$ erstellt. Als Ausgaben erhält man den Plot der Ursprungsquelle gegen die entmischte Quelle, den Vorfaktor h_{11} gegen den errechneten Filter u_1 und die Soundausgaben der beiden Mischungen x_1 und x_2, der Ursprungsquelle s_T und der entmischten Quelle $\hat{s_T}$.

5.2. Parameter und Ergebnisse

Die für die Ausführung des Programmes gewählten Parameter werden im Folgenden vorgestellt. Dabei hat sich ein Unterschied ergeben, ob die Quelle, auf die man das Entmischen anwendet, eine Sprachquelle ist, wie eigentlich vorgesehen, oder ob es sich um Musik handelt. Der Parameter μ, der die Sparsity der Lösung kontrolliert, ergibt für 10^{-1} bei Musik die besten Klangergebnisse. Wenn man ihn größer wählt, dann

klingt das Ergebnis sehr blechern, was bei Musik störend ist. Will man dagegen eine Sprachquelle erhalten, dann sollte man den Paramter μ größer wählen. 10^{-3} hat Ergebnisse von guter Qualität geliefert. Die Sprache klingt zwar blechern, ist aber sehr gut verständlich. Wählt man den gleichen Parameter wie bei Musik, dann ist das gesprochene Wort nicht annähernd so gut zu verstehen.

Als weitere Parameter wurde $L = 200$ für die Länge des Nachhalls gewählt. Dieser Parameter sollte nicht zu groß sein, denn ansonsten wird der Rechenaufwand zu groß. Für Mischungen mit großem Nachhall wurde in [3] ein anderer Algorithmus vorgestellt.

Desweiteren wurde der Parameter η auf 1 gesetzt. Er garantiert, dass der Eintrag $u_1(1)$ wie im Modell gefordert nahe bei 1 liegt. Der Parameter λ wurde als 2μ gewählt. Um die Annahme zu simulieren, dass die Vorfaktoren h_{ij} dünn besetzt sind, wurden hier zehn Einträge ungleich Null gewählt.

Im Folgenden wurden drei unterschiedliche Versuche gemacht. Beim ersten Mal wurden zwei Sprachaufnahmen (siehe Abb. 5.1) gemischt. In der Abbildung ist als erstes das Urpsrungssignal s_T geplottet, welches wir am Ende nach dem Entmischen wieder erhalten wollen. Das nächste ist die Hintergrundstörung s_B. In diesem Falle handelt es sich um zwei unterschiedliche Sprachaufnahmen, nur das die eine zeitlich verschoben wurde. Danach wurden die beiden Mischungen x_1 und x_2 geplottet. Der vorletzte Plot in der Abbildung ist das entmischte Signal $\hat{s_T}$ und der letzte Plot ist das Urprungssignal s_T mit Vorfaktor $h_{21} * h_{12} - h_{11} * h_{22}$. Beim zweiten Mal wurden zwei Musiksignale gemischt (siehe Abb. 5.2). Der erste Plot ist das Ursprungssignal s_T, der zweite das entmischte Signal $\hat{s_T}$ und der letzte das Urprungssignal s_T mit Vorfaktor $h_{21} * h_{12} - h_{11} * h_{22}$. Schlussendlich wurden ein Musik- und ein Sprachsignal gemischt (siehe Abb. 5.3). Dabei hat sich im Vergleich von Originalquelle s_T und rekonstruiertem Signal $\hat{s_T}$ jeweils Folgendes ergeben.

Beide Signale haben eine sehr ähnliche Struktur, auch klingen sie ähnlich. Die Qualität leidet allerdings durch den Entmischungsprozess. Außerdem wird das Signal nach dem Entmischen schwächer.

Vergleicht man die vorher gewählten Vorfaktoren h_{ij} mit den Sparsity-Filtern, die man durch das Split-Bregman erhält, dann sieht man, dass sich beide sehr ähneln (siehe Abb. 5.4). Nun ist u_1 (geplottet in (a)) der errechnete Filter zum Faktor h_{11} (geplottet

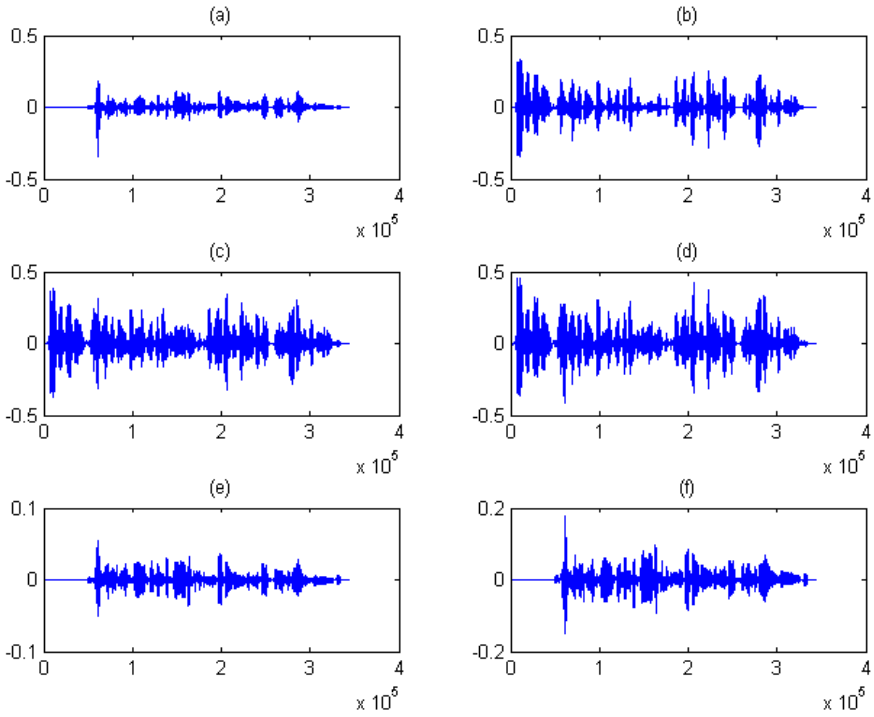

Abbildung 5.1.: Zweimal Sprache. (a) Ursprungssignal s_T (b) Ursprungssignal s_B (c) erste Mischung x_1 (d) zweite Mischung x_2 (e) entmischtes Signal $\hat{s_T}$ (f) Urprungssignal s_T mit Vorfaktor $h_{21} * h_{12} - h_{11} * h_{22}$

in (c)) und u_2 (geplottet in (b)) der Filter zu h_{21} (geplottet in (d)). Die Zacken im Graph nach oben und nach unten sind an den gleichen Stellen, aber sie sind nicht so stark ausgeprägt. Dies passt zu dem Höreindruck und dem Vergleich der Signale, dass das Entmischen ein gutes Ergebnis liefert. Bei den reinen Musikaufnahmen (siehe Abb. 5.5) und den reinen Sprachaufnahmen (siehe Abb. 5.4) zeigt sich eine größere Übereinstimmung und die Filter haben wirklich Sparsity. Bei der gemischten Aufnahme aus Musik und Sprache (siehe Abb. 5.6) zeigen sich viele kleinere Auschläge um die größeren herum. Dies lässt sich auch in der Aufnahme heraushören. Die Qualität ist insgesamt schlechter und leise im Hintergrund ist noch Musik zu hören. Trotzdem ist die Sprache eindeutig zu verstehen.

Plottet man die Faltungen $h_{21} * h_{12} - h_{11} * h_{22}$ (siehe Abb. 5.7 und Abb. 5.8), ergeben sich auf den ersten Blick sehr unruhige Graphen, doch wenn man beachtet, dass die Skala nur bis 200 bzw. 400 auf der x-Achse geht und diese sich bei den geplotteten Signalen bis zu $8 \cdot 10^4$ erstreckt, dann ist das ganze sinnvoller. Dazu kommt noch, dass die Faktoren durch die Faltung gemittelt werden und sich die großen Auschläge also am Anfang zentrieren.

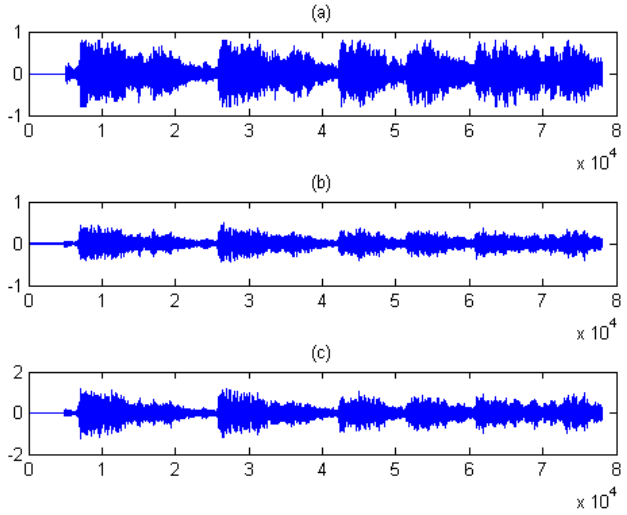

Abbildung 5.2.: Zweimal Musik. (a) Ursprungssignal s_T (b) Entmischtes Signal $\hat{s_T}$ (c) Urprungssignal s_T mit Vorfaktor $h_{21} * h_{12} - h_{11} * h_{22}$

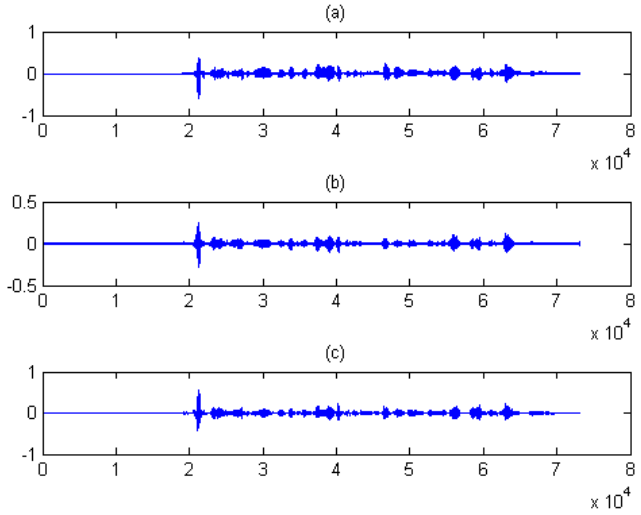

Abbildung 5.3.: Einmal Musik, einmal Sprache. (a) Ursprungssignal s_T (b) Entmischtes Signal $\hat{s_T}$ (c) Urprungssignal mit Vorfaktor $h_{21} * h_{12} - h_{11} * h_{22}$

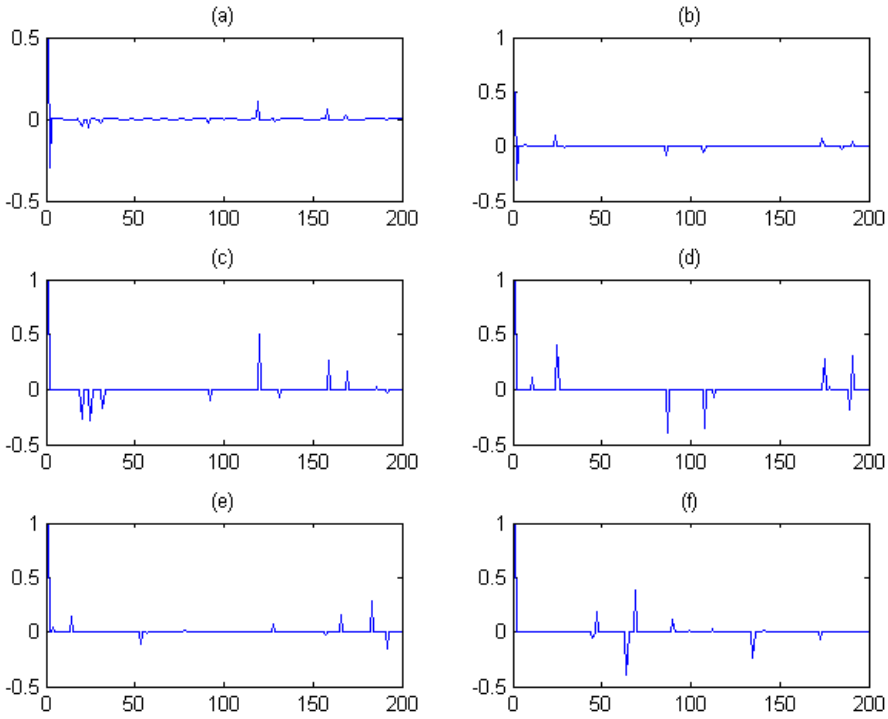

Abbildung 5.4.: Zweimal Sprache. (a) Sparsityfilter u_1 (b) Sparsityfilter u_2 (c) Vorfaktor h_{11} (d) Vorfaktor h_{21} (e) Vorfaktor h_{12} (f) Vorfaktor h_{22}

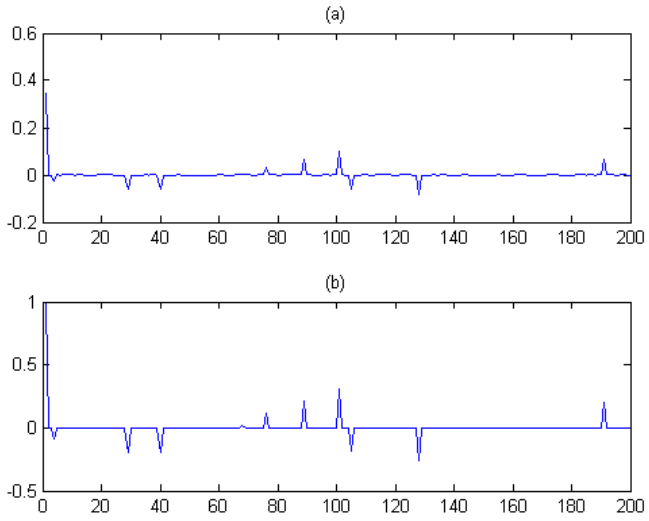

Abbildung 5.5.: Zweimal Musik. (a) Sparsityfilter u_1 (b) Vorfaktor h_{11}

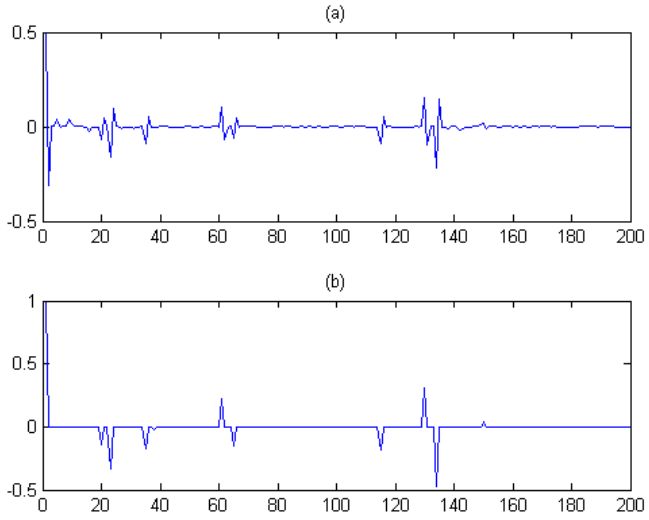

Abbildung 5.6.: Einmal Musik, einmal Sprache. (a) Sparsityfilter u_1 (b) Vorfaktor h_{11}

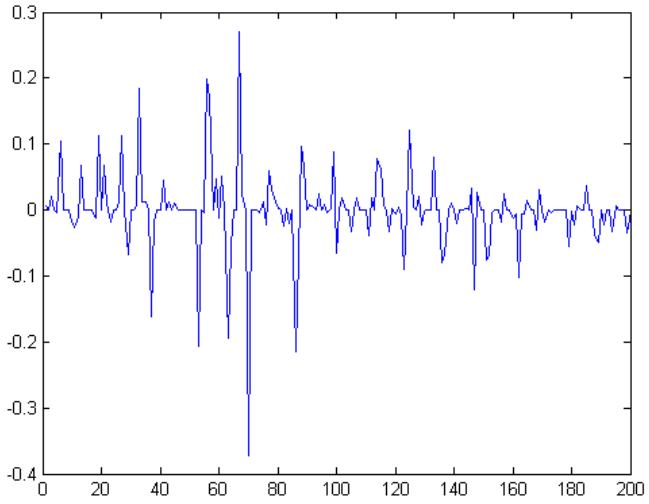

Abbildung 5.7.: $h_{21} * h_{12} - h_{11} * h_{22}$ für $L = 200$

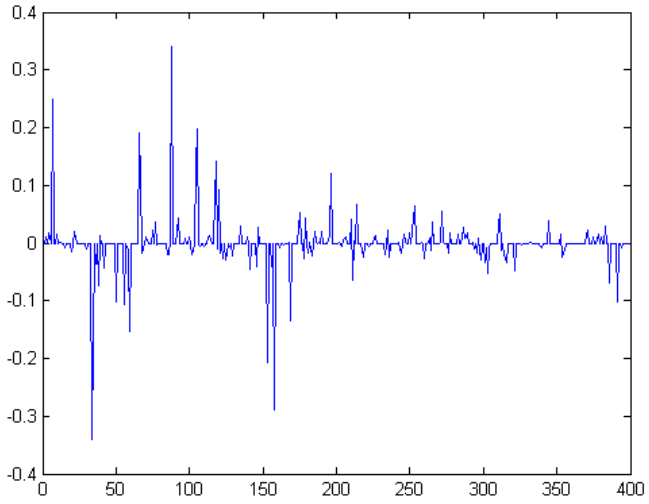

Abbildung 5.8.: $h_{21} * h_{12} - h_{11} * h_{22}$ für $L = 400$

6. Fazit und Ausblick

In der vorliegenden Arbeit haben wir uns mit der Entmischung mit Hilfe von Sparsity-Regularisierung beschäftigt. Dazu haben wir uns als erstes überlegt, warum es sinnvoll ist die $l1$-Norm zu minimieren um dünnbesetzte Lösungen zu erhalten und haben dann die Split-Bregman-Methode betrachtet um das Problem numerisch zu lösen.

Die Split-Bregman-Methode haben wir auf Hyperspektralbilder und auf Sprachextraktion angewandt. Dabei sind wir jedesmal davon ausgegangen, dass die Endmember bzw. die ursprünglichen Sprachsignale, also die Zeit, in der das gesuchte Signal ungefähr Null ist, schon bekannt sind. Dies ist in Wirklichkeit oft nicht so. Man müsste nun noch das Problem des Entmischens mit unbekannten Endmembern betrachten. Dies bezeichnet man als Blind Source Seperation (BSS).

Für die Sprachextraktion wird in [3] eine Möglichkeit vorgestellt, wie man die Zeitspanne, in der das zu entmischende Sprachsignal ungefähr Null ist, finden kann, wenn die ursprünglichen Signale nicht bekannt sind. Davon sollte man normalerweise ausgehen. Auch für das Entmischen von Hyperspektralbildern gibt es einige Arbeiten, die Verfahren zum BSS vorstellen, unter anderem auch [2] und [6].

Die Implementierung der Split-Bregman-Methode für die Sprachextraktion lieferte gute Ergebnisse. Die entmischten Signale sind deutlich zu verstehen, auch wenn sie schlechtere Qualität als die ursprünglichen Signale haben. Je nach Art der zu entmischenden Quelle ist es sinnvoll, den Sparsity-Parameter entsprechend anzupassen um das bestmögliche Ergebnis zu erlangen. Man kann diesen Algorithmus noch weiter ausbauen, denn hier wurde nur mit künstlich kreierten Mischungen von Sprache und Musik gearbeitet. Der nächste Schritt wäre es, dieses Verfahren so zu implementieren, dass die Mischungen direkt mit dem Mikrofon aufgenommen werden können.

A. Inhalt der CD

Die beigelegte CD enthält folgende Dateien:

- **Abbildungen:** Alle Abbildungen der Arbeit
- **Erstellung:** Programm zum Erstellen der Matrix A und dem Vektor f
- **Inhaltsverzeichnis:** Verzeichnis über alle Dateien auf der CD
- **Mischen:** Programm zum Mischen der eingelesenen Sprachsignale
- **Sprachsignal1:** Sprachsignal, erster Sprecher, Samplingrate: 44100, (Auschnitt aus „Der Zauberlehrling“ von Goethe)
- **Sprachsignal2:** Sprachsignal, zweiter Sprecher, Samplingrate: 44100, (Auschnitt aus „Der Zauberlehrling“ von Goethe)
- **Sprachsignal3:** Sprachsignal, erster Sprecher, Samplingrate: 8192 (Auschnitt aus „Der Zauberlehrling“ von Goethe)
- **StimmeMusik:** Programm zum Entmischen von Sprachsignal3 und Musiksignal aus Matlab („Hallelujah“ von Händel)
- **Unmixing:** Split-Bregman-Algorithmus
- **ZweiMusik:** Programm zum Entmischen von zwei Musiksignalen aus Matlab („Hallelujah“ von Händel)
- **ZweiStimmen:** Programm zum Entmischen von Sprachsignal1 und Sprachsignal2

Abbildungsverzeichnis

Literaturverzeichnis

[1] M. Elad A.Bruckstein, D. Donoho. From sparse solutions of systems of equations to sparse modeling of signals and images. *SIAM Review*, 51(1):34–81, 2009. 4

[2] J. Greer. Sparse demixing. *SPIE proceedings on Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery XVI*, 7695:765910–76591012, 2010. 13, 33

[3] Jack Sin Meng Yu, Wenye Ma and Stanley Osher. A convex speech extraction model and fast computation by the split bregman method. pages 1–8, 2010. 5, 22, 25, 27, 33

[4] J. Mustard N. Keshava. Spectral unmixing. *IEEE Signal Processing Mag.*, 19(1):44–57, 2002. 11, 12

[5] S. Osher T. Goldstein. The split bregman method for l1 regularized problems. *SIAM Journal on Imaging Sciences*, 2:323–343, 2009. 5

[6] M.E. Winter. N-findr: an algorithm for fast autonomous spectral end-member determination in hyperspectral data. *Proceedings of SPIE*, 3753:266–277, 1999. 33

[7] T. Wittman Z. Guo and S. Osher. L1 unmixing and its application to hyperspectral image enhancement. *Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery*, 7334:73341M–73341M9, 2009. 13, 16