Wissenschaftliches Rechnen

wr@cg.tu-berlin.de

16. April 2024

Inhaltsverzeichnis

1. Zahlen

1.1. Motivation

Das wissenschaftliche Rechnen ist motiviert durch Systeme, deren Zustand sich durch Zahlen darstellen lässt. Den Vorgang, ein reales Phänomen durch Zahlen und Rechenvorschriften zu beschreiben nennen wir auch *Modellierung*. Bei der Modellierung findet oft eine teilweise *Diskretisierung* statt. Das heißt, dass einige von uns als kontinuierlich beobachtete Phänomene in endlich viele Teile zerlegt werden. Wir betrachten dazu einige Beispiele:

Bilder Wir betrachten einfache Grauwertbilder (man denke an eine Schwarz-Weiß Photographie). Ein solches Bild können wir beschreiben, in dem wir jedem Bildpunkt, also einem Vektor $\mathbf{x} \in \mathbb{R}^2$ den entsprechenden Grauwert, also eine reelle Zahl $g \in \mathbb{R}$ zuordnen. Ein Grauwertbild ist somit eine Funktion $g : \mathbb{R}^2 \mapsto \mathbb{R}$. Dabei sind die Bildpunkte auf ein gewisses Gebiet eingeschränkt, z.B. das Einheitsquadrat $\mathbf{x} \in [0, 1] \times [0, 1]$. Auch die Grauwerte können wir auf ein Intervall einschränken, denn dunkler als schwarz oder heller als weiß können die Bildpunkte nicht sein – also $g \in [0, 1]$. Damit ist die Beschreibung des Bildes eine Funktion $g : [0, 1]^2 \mapsto [0, 1]$.

Im Zuge der Modellierung wird nun typischerweise der Raum der Bildpunkte diskretisiert: Es werden nicht alle (unendlich vielen) Bildpunkte betrachtet, sondern ein regelmäßiges Gitter über das Bild gelegt. Wir können dies z.B. so schreiben: $\mathbf{x} \in [0, 1, \ldots, n-1] \times [0, 1, \ldots, m-1]$. Diesen Schritt motiviert man mit einer Kombination von zwei Argumenten: Auf der einen Seite gehen wir davon aus, dass das Bild letztendlich einem Menschen präsentiert wird, und dessen Wahrnehmung ist beschränkt. Die Anzahl der einzelnen Photorezeptoren auf der Netzhaut ist endlich. Ab einer bestimmten Anzahl von Bildpunkten würde eine weitere „Verfeinerung“ des Bildes nicht mehr wahrnehmbar sein. Mehr Bildpunkte führen also nicht zwangsläufig zu einer wahrnehmbaren Qualitätssteigerung des Bildes. Auf der anderen Seite haben Computer nur endliche Ressourcen. Insbesondere müssen diese Ressourcen oft möglichst geschickt auf verschiedene Dienste aufgeteilt werden. Man möchte also aus praktischen Gründen nicht unnötig viele Zahlen verwenden. Die Wahl der *Auflösung* (der Anzahl der Bildpunkte $n \cdot m$) ist ein Kompromiss zwischen den benötigten Ressourcen und der Qualität der Repräsentation.

Wir können also ein Bild als eine Reihe von reellen Zahlen im Intervall $[0, 1]$ darstellen. Wir schreiben die $n \cdot m$ Zahlen in Form eines Vektors $\mathbf{g} \in \mathbb{R}^{nm}$. Die Anordnung der Zahlen müssen wir dabei festlegen, aber es gibt hier offenbar verschiedene Möglichkeiten wie man von der zweidimensionalen Anordnung der Punkte in der Ebene zu einer Reihenfolge gelangt. Achtung: die Zahlen bilden keine Matrix, auch wenn sie Orten aus einem regelmäßigen Gitter zugeordnet werden können. Wichtig ist nur, dass sich ein Bild mit einer festen Anzahl von Werten beschreiben lässt. Alle Werte sind gleichberechtigt und können auf jede wohldefinierte Weise hintereinander in einen Vektor geschrieben werden.

Figuren Viele Objekte, die wir letztlich in der Form von Bildern präsentieren können, haben eine Repräsentation, die besser für die Verarbeitung geeignet ist, als ein Bild. Wir nehmen hier als Beispiel eine stark abstrahierte zweidimensionale Figur bestehend aus Liniensegmenten, die in etwa dem Skelett der Figur entsprechen. Die Liniensegmente sind hierarchisch organisiert –

in der Sprache der Informatik heißt das sie bilden einen Baum. Wir nehmen ferner an, dass sich aus physiologischen Gründen die Längen der Segmente nicht ändern. Dann geben wir die Orientierung und Position des Wurzelelementes explizit an. Alle anderen Segmente werden repräsentiert durch den Winkel zu dem Element des Elternknotens. Die Konstruktion der Figuren erfordert dann die Traversierung des Baums.

Eine Figur ist folglich beschrieben durch eine Position (zwei Koordinaten) und für jedes Segment einen Winkel. Wir können diese Zahlen wieder in einem Vektor zusammenfassen.

Audiosignale Töne und Geräusche sind Luftdruckschwankungen, die vom Trommelfell aufgenommen und anschließend weiterverarbeitet werden. Wir können diese Luftdruckschwankungen heute mit Mikrofonen in elektrische Signale umwandeln und diese dann durch Lautsprecher auch wieder als Luftdruckschwankungen ausgeben. Wir haben es also (ähnlich zu den Bildern) mit einer Funktion zu tun, die abhängig von einem Parameter (hier die Zeit) einen Wert annimmt. Wenn wir die Länge und maximale Amplitude des Signals beschränken, haben wir die Funktion $a : [0, 1] \mapsto [-1, 1]$. Nun haben wir die Erfahrung gemacht, dass in etwa 40000 einzelne Werte pro Sekunde genügen, um ein Audiosignal für Menschen zu repräsentieren. Die Verwendung von weiteren Werten ist für Menschen im Allgemeinen nicht wahrnehmbar. Es besteht also wieder die Möglichkeit, das Signal als Vektor $\mathbf{a} \in \mathbb{R}^n$ darzustellen, wobei man typischerweise $n = 40000t$ wählen würde, wenn t die Länge des Signals in Sekunden ist. Wir werden uns im Rahmen des wissenschaftlichen Rechnens auch damit auseinandersetzen, warum eine Zahl wie 40000 für den Vorgang des Hörens überhaupt existieren kann, und was es bedeutet, wenn man weniger Werte verwendet.

Eine andere Darstellung ist aber gerade im Bereich der Audiotechnik ebenfalls weit verbreitet: wir beschreiben die Funktion $a(t)$ als gewichtete Summe von Basisfunktionen $\{b_0(t), b_1(t), \ldots\}$, also

$$a(t) = x_0 \cdot b_0(t) + x_1 \cdot b_1(t) + \ldots + x_{n-1} \cdot b_{n-1}(t).$$

Da die Basisfunktionen fest sind, haben wir so das Audiosignal $a(t)$ tatsächlich als kontinuierliche Funktion beschrieben, sprich wir können die Funktion $a(t)$ an jeder beliebigen Stelle t auswerten. Das Signal ist beschrieben durch den Vektor von Koeffizienten $\mathbf{x} = (x_0, x_1, \ldots, x_{n-1}) \in \mathbb{R}^n$. Es mag zunächst so erscheinen, als ob eine sinnvolle Wahl von n für diese Art der Repräsentation nichts mit der Diskretisierung in der Zeit zu tun hätte. Dem ist aber nicht so – vielmehr wird sich die Erkenntnis, dass 40000 Werte pro Sekunde reichen, aus der Darstellung über geeignete Basisfunktionen ergeben.

Verbleibende Variablen Bei allen Beispielen haben wir ein kontinuierliches Phänomen durch Diskretisierung in einen endlichen Vektor $\mathbf{x} \in \mathbb{R}^n$ überführt. Es bleibt also die Frage, wie wir diese reellen Zahlen darstellen. Es ist aber klar, dass auch diese Werte unweigerlich diskretisiert werden, da der Computer zur Darstellung der Zahlen nur endliche Ressourcen hat. Es ist oft besser sich dieser Tatsache bewusst zu sein und direkt Überlegungen anzustellen, ob und wie solche Größen zu diskretisieren sind, anstatt sich darauf zu verlassen, dass die automatische Diskretisierung im Rechner die gewünschten Eigenschaften hat.

1.2. Natürliche Zahlen und Stellenwertsysteme

Wir beginnen mit den natürlichen Zahlen. Dabei führen wir einige Begriffe ein, die ganz allgemein wichtig sein werden. Danach fragen wir uns, wie man natürliche Zahlen auf dem Computer repräsentieren kann und gelangen zu Stellenwertsystemen.

1.2.1. Natürliche Zahlen

Die *natürlichen Zahlen* sind die Menge

$$\mathbb{N} = \{0, 1, 2, \ldots\}. \tag{1.1}$$

Wir betrachten also auch die Null als eine natürliche Zahl. Natürliche Zahlen kann man addieren und multiplizieren. Sowohl die Summe wie auch das Produkt natürlicher Zahlen ist wieder eine natürliche Zahl – wir sagen die natürlichen Zahlen sind gegenüber Addition und Multiplikation *abgeschlossen*:

$$a, b \in \mathbb{N}, \qquad a + b \in \mathbb{N}, \qquad a \cdot b \in \mathbb{N}. \tag{1.2}$$

Beide Operationen sind kommutativ, d.h. $a + b = b + a$ und $ab = ba$. Bezüglich beider Operationen gibt es ein *Neutralelement*. Für die Addition ist dies die Null ($a + 0 = a$) und für die Multiplikation die Eins ($a \cdot 1 = a$).

Wichtig für das Rechnen mit Zahlen (insbesondere zur Formulierung von Abbruchbedingungen) ist die Existenz einer Ordnung, also der Operation „kleiner“ bzw. „kleiner-gleich“. Wir können dies wie folgt definieren:

$$a \leq b \quad \Longleftrightarrow \quad \exists x \in \mathbb{N} : a + x = b \tag{1.3}$$

Wichtige Eigenschaften der Relation $\leq$ sind:

1. Reflexivität: $a \leq a$ (deswegen ist es wichtig, dass $x = 0$ ein Element der natürlichen Zahlen ist)
2. Antisymmetrie: $a \leq b \wedge b \leq a \quad \Longrightarrow \quad a = b$
3. Transitivität: $a \leq b \wedge b \leq c \quad \Longrightarrow \quad a \leq c$
4. Totalität: $a \leq b \vee b \leq a$

Die ersten drei Eigenschaften folgen unmittelbar aus der Definition. Ordnungen, die diese drei Eigenschaften erfüllen nennt man *partielle Ordnungen*. Zusammen mit Totalität ergeben sich *totale Ordnungen*. Totalität folgt übrigens nicht direkt aus der Definition; der Nachweis erfordert die Verwendung von *vollständiger Induktion*. Vollständige Induktion hängt eng mit den natürlichen Zahlen zusammen und kann auf alle Mengen angewendet werden, die *abzählbar* sind, also mit den natürlichen Zahlen identifiziert werden können.

1.2.2. Stellenwertsysteme

Zur Repräsentation von Zahlen auf dem Computer müssen wir uns auf einen endlichen Zeichenvorrat beschränken. Ein variabler Kompromiss zwischen der Zahl an verschiedenen Symbolen und der Anzahl der verwendeten Zeichen sind *Stellenwertsysteme*. Dabei verwendet man nur wenige verschiedene Symbole, z.B. in vielen Kulturen die zehn arabischen Ziffern $0, 1, 2, 3, 4, 5, 6, 7, 8, 9$. Zahlen, die größer als Neun sind können dargestellt werden, in dem man vereinbart, dass die (von rechts gesehen) zweite Ziffer mit der Anzahl der verschiedenen Ziffern zu multiplizieren ist – die Zahl 10 beschreibt also 1-mal die Größe des Zeichenvorrats, mit 200 meinen wir 2-mal die Größe des Zeichenvorrats mal die Größe des Zeichenvorrats.

Dieses System lässt sich für jede Zahl von Symbolen verwenden. Sei b die Anzahl der Zeichen, dann wird in einem Stellenwertsystem eine Zahl wie folgt dargestellt:

$$\underbrace{x_{n-1} \cdots x_1 x_0}_{\text{Darstellung mit } n \text{ Stellen}} = \sum_{i=0}^{n-1} x_i b^i \tag{1.4}$$

Die Anzahl der Stellen heißt wegen dieser Form auch *Basis*. Jede der Stellen x_i kann nur b verschiedene Werte annehmen, was wir wie folgt schreiben $x_i \in [0, 1, \ldots, b-1]$. Bei Darstellungen dieser Art ist es übrigens Konvention ein bestimmtes Stellenwertsystem zu verwenden, nämlich das *Dezimalsystem*, also das Stellenwertsystem mit $b = 10$.

Offensichtlich können mit Stellenwertsystemen, unabhängig von der Wahl der Basis, genau die natürlichen Zahlen dargestellt werden. Auf Computern (zumindest in Hardware) verwenden wir dabei die Binärbasis, also $b = 2$ und unterscheiden nur die Ziffern 0 und 1. Um tatsächlich mit den Zahlen rechnen zu können müssen wir die Operation $+$ und $\cdot$ durchführen und auch die Ordnungsrelation auswerten können.

Die Algorithmen zur Addition und Multiplikation entsprechen dem Verfahren, dass die meisten in der Schule als „schriftliches“ Addieren bzw. Multiplizieren gelernt haben. Im Binärsystem sind diese Algorithmen besonders einfach, wovon wir uns am Beispiel der Addition überzeugen.

Gegeben zwei Zahlen in binärer Stellenwertnotation $x_{n-1} \cdots x_0, y_{n-1} \cdots y_0$, sowie eine Zahl $z_n \cdots z_0$ zur Repräsentation der Summe. Die Summe initialisieren wir mit $z_i = 0$ und verwenden sie auch zur Speicherung der Überträge. Dann ergibt folgende Iteration das gewünschte Ergebnis:

1. Übertrag setzen, wenn mindestens zwei der drei Werte 1 sind: $z_{i+1} = x_i \cdot y_i + y_i \cdot z_i + z_i \cdot x_i$
2. Stellenwert ausrechnen - das entspricht der Parität der drei Werte: $z_i = x_i \oplus y_i \oplus z_i$,

Dabei bezeichnen $\oplus, +, \cdot$ die üblichen logischen Operationen XOR, OR, AND. Ausgehend von der Addition ist auch die Multiplikation einfach zu beschreiben.

Zum Vergleich zweier Zahlen beginnt man bei der Stelle $i = n - 1$ und dekrementiert i solange bis sich die Ziffern x_i und y_i unterscheiden.

Für eine praktische Implementierung dieser Konzepte bleibt noch die Frage, wie man mit der Anzahl der Stellen n umgeht. Die gängige Lösung besteht darin, n fest zu wählen. Die Wahl von n definiert dann einen *Typ*. Ein Beispiel ist die Wahl von $n = 8$ – dies erlaubt die Repräsentation der Zahlen $[0, \ldots, 255]$. Ein Vorteil dieser Vorgehensweise ist die besonders einfache Implementierung der Basisoperationen, da die Anzahl der Stellen konstant ist. Deswegen ist es auch plausibel zu behaupten, die asymptotische Komplexität der Basisoperationen $+, \cdot, \leq$

ist in $O(1)$. Andererseits können nicht alle Ergebnisse der arithmetischen Operationen mit den vorhandenen Stellen dargestellt werden. Dies sieht man auch direkt an dem Algorithmus zur Addition: während die Summanden n Stellen haben, mussten wir für das Ergebnis $n+1$ Stellen reservieren. Ist die Stelle z_n nach dem Abschluss der Addition nicht Null, so ist das Ergebnis zu groß, um mit den verfügbaren n Stellen repräsentiert zu werden (also größer oder gleich b^n). Man spricht dann von einem *Überlauf*. Das gesetzte Bit z_n ist ein Zeichen für den Überlauf. Das Ergebnis in den verbleibenden Stellen entspricht dem Rest, der beim Teilen durch b^n bleibt.

Eine Alternative zu einer festen Stellenanzahl ist die dynamische Anpassung. Dabei reserviert man Speicher für die notwendigen Stellen, die sich zwangsläufig aus den Operationen ergeben. Dieses Verfahren stößt erst an seine Grenzen, wenn der Speicher des Computers vollständig ausgeschöpft ist. Bei diesem Vorgehen kann man nicht mehr annehmen, die asymptotische Komplexität der Rechenoperationen sei konstant. Vielmehr hängt sie dann von der Größe der Zahl ab, mit der gerechnet wird.

Ein grundsätzliches Problem bleibt, dass die Inversen der Operationen $+$ und $\cdot$ nicht abgeschlossen sind gegenüber den natürlichen Zahlen. In einer praktischen Implementierung könnte man für die Subtraktion analog zum Überlauf umgehen: wenn das Ergebnis einer Operation kleiner als Null werden würde, wird das Ergebnis in das Intervall der verfügbaren Werte abgebildet und mit einem Bit signalisiert, dass es zu einem Fehler kam. Auch die Division kann man auf natürliche Zahlen einschränken („Division mit Rest“). Für viele Probleme ist dieses Vorgehen allerdings keine praktikable Lösung. Im nächsten Abschnitt werden wir daher den Zahlenraum – und damit auch die Repräsentation – geeignet erweitern.

1.3. Ganze Zahlen, Komplementdarstellung und Exzess-Code

Unabhängig von der Darstellung ist es für viele Anwendungen ungeschickt, dass die Gleichung $a+x=b$ für natürliche Zahlen $a, b \in \mathbb{N}$ nicht unbedingt eine Lösung x in den natürlichen Zahlen hat. Man kann die natürlichen Zahlen allerdings um weitere Zahlen „erweitern“ – das führt auf die *ganzen Zahlen* $\mathbb{Z}$. Die formale Erweiterung enthält auch eine mögliche Repräsentation, die allerdings nicht ökonomisch ist. Die üblichen Darstellungen der ganzen Zahlen diskutieren wir im Anschluss.

1.3.1. Ganze Zahlen

Wir betrachten also natürliche Zahlen $a, b \in \mathbb{N}$ und fordern die Lösbarkeit der Gleichung $a + x = b$. Der einfache „Trick“ die Zahl x zu beschreiben liegt darin, das Paar (a, b) zu verwenden. Wir sagen also einfach, x ist die Zahl, die die Gleichung $a + x = b$ erfüllt. Jetzt muss man allerdings alle wünschenswerten Eigenschaften nachweisen. Zunächst müssen wir beschreiben, unter welchen Umständen zwei Zahlen gleich sind – man sieht nämlich leicht, dass (a, b) nicht die einzige Repräsentation von x ist – z.B. beschreibt $(a + 1, b + 1)$ offenbar die gleiche Zahl. Dazu führen wir eine Äquivalenzrelation ein, die direkt aus der Gleichung $a + x = b$ folgt:

$$(a, b) \sim (c, d) \quad \Longleftrightarrow \quad a + d = b + c. \tag{1.5}$$

Dann beschreiben wir mit $[a, b]$ alle Elemente der Äquivalenzklasse von (a, b). Mit anderen Worten, das Paar (a, b) ist ein Repräsentant für eine (unendliche) Menge von Paaren, die alle die gleiche Zahl beschreiben. Die Menge der ganzen Zahlen ist die Menge der Äquivalenzklassen

$$\mathbb{Z} = \{[a, b], \quad a, b \in \mathbb{N}\}\,. \tag{1.6}$$

Die natürlichen Zahlen sind als Elemente der Form $[0, b]$ Teil der ganzen Zahlen und mit der Äquivalenzrelation verträglich. Es stellt sich die Frage, wie man mit dieser Repräsentation rechnet, also addiert, multipliziert usw.; ob die ganzen Zahlen gegenüber den Operationen abgeschlossen sind; ob die Elemente der ganzen Zahlen geordnet sind und wie man die Anordnung zweier Elemente bestimmt. Auf alle diese Fragen gibt es befriedigende Antworten.

Die Addition zweier ganzen Zahlen definiert man als

$$[a, b] + [c, d] := [a + c, b + d] \tag{1.7}$$

und die Multiplikation

$$[a, b] \cdot [c, d] := [ad + bc, ac + bd]. \tag{1.8}$$

Man kann leicht nachrechnen, dass dies den gewohnten Rechenregeln auf ganzen Zahlen entspricht. Eine Ordnung entsteht durch

$$[a, b] \leq [c, d] \quad \Longleftrightarrow \quad b + c \leq a + d.$$

Dabei hat man die Operationen auf die bereits definierten Operationen für die natürlichen Zahlen zurückgeführt: die Addition und Multiplikation sind gegenüber den natürlichen Zahlen abgeschlossen, weswegen alle vorkommenden Terme ($ad + bc$, etc.) natürliche Zahlen sind. Als nächstes kann man sich davon überzeugen, dass die Operationen mit der Äquivalenzrelation verträglich sind. Die Totalordnung der ganzen Zahlen lässt sich dann durch Verwendung der Totalordnung der natürlichen Zahlen zeigen (wir verzichten hier auf die einzelnen Beweise).

Zusätzlich können wir nun auch die additive Inverse einführen, die wir dann verwenden, um die Subtraktion zu definieren:

$$-[a, b] := [b, a], \qquad [a, b] - [c, d] := [a, b] + [d, c] \tag{1.9}$$

Das additiv inverse Elemente ebenfalls ganze Zahlen sind und die Subtraktion gegenüber den ganzen Zahlen abgeschlossen ist folgt unmittelbar aus den Definitionen.

1.3.2. Vorzeichendarstellung, 1-er Komplement

Jede ganze Zahl lässt sich also als Paar von natürlichen Zahlen beschreiben. Die üblichen Operationen könnten entsprechend der Definitionen auf die Operationen für die natürlichen Zahlen zurückgeführt werden. Wir wollen nach Möglichkeit allerdings nicht alle Paare einer Äquivalenzklasse verwenden, sondern ein bestimmtes auswählen, um Speicher zu sparen. Die verschiedenen gebräuchlichen Darstellungen der ganzen Zahlen ergeben sich aus einer bestimmten Wahl der Paare (a, b) für nicht-negative Zahlen (also $x \geq 0$) sowie nicht-positive Zahlen ($x \leq 0$).

In der folgenden Diskussion beschränken wir uns auf binäre Stellenwertrepräsentationen. Wie bereits bemerkt lassen sich die natürlichen Zahlen als $(0, x)$ beschreiben. Analog können wir

die nicht-positiven Zahlen $-y$ als $(y, 0)$ beschreiben. In der Umsetzung in eine Binärdarstellung entspricht dies der Idee, den Betrag der Zahl zu speichern und zusätzlich nach dem Vorzeichen zu unterscheiden. Dies wird in Binärdarstellung meist wie folgt gemacht:

$$\underbrace{x_{n-1}}_{\text{Vorzeichen}} \quad \underbrace{x_{n-2} \ \cdots \ x_1 \ x_0}_{\text{Betrag: } n-1 \text{ Stellen}} = (-1)^{x_{n-1}} \cdot \sum_{i=0}^{n-2} x_i 2^i, \qquad x_i \in [0, 1] \tag{1.10}$$

Möchte man zur Darstellung der Zahl bspw. 8 binäre Stellen verwenden, so würden 7 Stellen für den Betrag verbleiben. Man könnte also die Zahlen $[-127, +127]$ repräsentieren. Diese Art der Darstellung hat zwei Nachteile: 1. Die Addition von Zahlen benötigt eine Fallunterscheidung; 2. die Zahl Null ist doppelt repräsentiert, nämlich mit positivem und negativem Vorzeichen.

Warum funktioniert die Addition nicht mehr wie vorher? Addieren wir zwei positive Zahlen $(0, x) + (0, y)$ so ergibt sich $(0, x + y)$ und außer einem möglichen Überlauf gibt es keine Probleme. Gleiches gilt für die Addition zweier negativer Zahlen. Addiert man aber eine positive und eine negative Zahl $(0, x) + (y, 0)$ so erhält man (y, x). Diese Darstellung muss man auf entweder $(0, x - y)$ oder $(y - x, 0)$ bringen, je nachdem ob $x \geq y$ oder $x < y$ ist. Man hat also eine Fallunterscheidung.

Das Problem der Fallunterscheidung kann man weitgehend mit einer *Komplementdarstellung* umgehen. Wenn der Bereich der unterschiedlichen Beträge $0, b^{n-1} - 1$ ist, so werden die nicht-positiven Zahlen dargestellt als $(b^{n-1} - 1, y')$. Es wird also nicht der Betrag der nicht-positive Zahl y gespeichert, sondern der (notwendigerweise positive) Abstand zur kleinsten darstellbaren Zahl $-(b^{n-1} - 1)$, also

$$y < 0 \quad \Rightarrow \quad y' = b^{n-1} - 1 + y = b^{n-1} - 1 - |y|. \tag{1.11}$$

Diese Umwandlung ist im binären Stellenwertsystem sehr einfach durchzuführen: um y' zu bestimmen müssen für $y \leq 0$ einfach alle Stellen von $|y|$ invertiert werden, also eine Eins durch eine Null und eine Null durch eine Eins ersetzt werden. Diese Zahl nennt man das Komplement, oder genauer in diesem Kontext, das *1-er Komplement*.

Addiert man zwei Zahlen mit unterschiedlichem Vorzeichen ist die Behandlung der beiden möglichen Fälle nun einfacher als die bei der Verwendung von Beträgen: Wir haben

$$(0, x) + (b^{n-1} - 1, y') = (b^{n-1} - 1, x + y'). \tag{1.12}$$

Für den Fall, dass $x + y \leq 0$ ist, haben wir bereits eine korrekt Repräsentation. Ansonsten müssen wir von $x + y'$ die Zahl $b^{n-1} - 1$ subtrahieren. Die Subtraktion von b^{n-1} ist sehr einfach, denn das entsprechende Bit wird für die Darstellung von x und y' gar nicht verwendet. Man muss also gar nichts tun. Es bleibt also nur die Addition von 1. Ob dieser Fall eingetreten ist erkennt man praktischerweise am Übertrag der Addition $x + y'$. Offenbar ist die Bedingung für den Fall ein positives Ergebnis zu erhalten

$$\begin{aligned} x &> |y| \\ x - |y| &> 0 \\ x - (b^{n-1} - 1 - y') &> 0 \\ x + y' &> b^{n-1} - 1, \end{aligned} \tag{1.13}$$

und da wir als Wertebereich für x und y' die Zahlen $0, b^{n-1} - 1$ vereinbart haben entspricht dies genau dem Überlauf.

Der Fall zwei negative Zahlen zu addieren ist jetzt (anders als vorher bei der Vorzeichendarstellung) genau analog zum Fall ein negatives Ergebnis bei der Addition eine nicht-negativen und einer nicht-positiven Zahl zu erhalten. Hier müssen wir stets $b^{n-1}-1$ subtrahieren, also effektiv 1 addieren.

1.3.3. Exzess-Code, 2-er Komplement

Die Korrektur bei negativen Ergebnissen ist unbefriedigend. Außerdem bleibt das Problem der doppelten Repräsentation von Null. Beide Probleme lassen sich lösen, wenn man akzeptiert, dass der verfügbare Zahlenbereich nicht symmetrisch zur Null ist. Der verbreitete Ansatz ist die Verwendung des sog. *2-er Komplements*. Der Zahlenbereich wird auf $[-b^{n-1}, b^{n-1}-1]$ erweitert. Für 8 Bit mit Vorzeichen wäre dies der Bereich $[-128, 127]$. Negative Zahlen $y<0$ werden hier durch (b^{n-1}, y'), also durch den Abstand zur kleinsten Zahl dargestellt:

$$y<0 \quad \Rightarrow \quad y' = b^{n-1}+y = b^{n-1}-|y|. \tag{1.14}$$

Die Umwandlung einer positiven Zahl in eine negative Zahl ist im Vergleich zum 1-er Komplement etwas aufwändiger: Nach der Komplementbildung muss noch Eins addiert werden. Dafür werden die Operationen einfacher, was man an

$$(0,x)+(b^{n-1},y') = (b^{n-1}, x+y') \tag{1.15}$$

erkennen kann, den wie bei der Diskussion um das 1-er Komplement bedeutet die Subtraktion von b^{n-1} keine zusätzliche Operation. Sofort sieht man das auch an folgender Interpretation im Stellenwertsystem:

$$\underbrace{x_{n-1}}_{\text{Vorzeichen}} \quad \underbrace{x_{n-2} \ \cdots \ x_1 \ x_0}_{n-1 \text{ Stellen}} = -x_{n-1}2^{n-1} + \sum_{i=0}^{n-2} x_i 2^i, \qquad x_i \in [0,1]. \tag{1.16}$$

Die Addition erzeugt jetzt bei beliebigen positiven und negativen Zahlen immer das gewünschte Ergebnis, zumindest solange das Ergebnis im verfügbaren Zahlenbereich liegt. Ansonsten generiert die Addition ohne weiteres Zutun ein Ergebnis im *Restklassenring*: die Zahlen werden $\bmod b^n$ verstanden, d.h. zwei Zahlen x und y sind gleich, wenn ihre Differenz ein Vielfaches von b^n ist $x \sim y \Longleftarrow x = y + db^n, d \in \mathbb{Z}$. Für jede Klasse gibt es in $[-b^{n-1}, b^{n-1}-1]$ einen Repräsentanten und jede Operation bildet das Ergebnis automatisch auf diesen Repräsentanten ab.

Dem 2-er Komplement ähnlich ist die die Exzess-Darstellung. Sie entsteht, wenn man nicht nur die negativen sondern alle Zahlen verschiebt. Statt die Zahlen $[0, b^n-1]$ zu repräsentieren, verschiebt man die Zahlen um k, repräsentiert also alle Zahlen als (k, x'). Schreibt man im Stellenwertsystem x, so ist damit $x-k$ gemeint; d.h. eine Zahl x hat die Darstellung $x+k$. Eine übliche Wahl im binären Stellenwertsystem ist $k=b^{n-1}$, so dass man wie im 2-er Komplement den Wertebereich $[-b^{n-1}, b^{n-1}-1]$ und folgende Darstellung erhält:

$$\underbrace{x_{n-1} \ \cdots \ x_1 \ x_0}_{n \text{ Stellen}} = \sum_{i=0}^{n-1} x_i 2^i - k \underset{k=b^{n-1}}{=} (x_{n-1}-1)\ 2^{n-1} + \sum_{i=0}^{n-1} x_i 2^i, \quad x_i \in [0,1]. \tag{1.17}$$

Zur Addition muss man lediglich den *Exzess* entfernen. Zwei Zahlen x, y haben die Darstellung $(k,x), (k,y)$. Ihre Summe ist also $(2k, x+y)$. Um die gewünschte Darstellung $(k, x+y)$ zu erhalten, muss also k subtrahiert werden. Bei der Wahl $k=b^{n-1}$ geschieht dies einfach durch Invertieren der Stelle x_{n-1}.

1.4. Rationale Zahlen und Festkommadarstellung

Wenn wir Gleichungen lösen wollen, die neben Addition auch Multiplikation verwenden, dann brauchen wir auch inverse Elemente bzgl. der Multiplikation. Das bedeutet, wir wollen den Zahlenraum so erweitern, dass die Gleichung $ax = b$ in ganzzahligen Koeffizienten a, b (möglichst) immer gelöst werden kann. Eine Konstruktion analog zu den ganzen Zahlen führt auf die *rationalen Zahlen* $\mathbb{Q}$. Die ganze Zahl a entspricht dabei dem Nenner und die ganze Zahl b dem Zähler der rationalen Zahl. Repräsentiert werden rationale Zahlen in der Tat als Bruch, wobei sich aus der Wahl eines festen Nenners *Festkommadarstellungen* ergeben.

1.4.1. Rationale Zahlen

Wie bei der Konstruktion der ganzen Zahlen, erlaubt das Paar (a, b) durch die Gleichung $ax = b$ eine Beschreibung von x. Anders als bei den ganzen Zahlen, beschreibt nicht jedes Paar (a, b) eine Zahl. Für $a = b = 0$ ist die Gleichung $ax = b$ nach den Regeln der Multiplikation der ganzen Zahlen für jedes x erfüllt. Wegen der Regel $0 \cdot x = 0$ dürfen wir auch nicht erwarten, dass es Lösungen für $a = 0, b \neq 0$ gibt. Wir betrachten daher nur Paare (a, b) mit $a \in \mathbb{Z} \setminus \{0\}, b \in \mathbb{Z}$[1]

Auf den Paaren definiert man die Äquivalenzrelation

$$(a, b) \sim (c, d) \quad \Longleftrightarrow \quad ad = bc. \tag{1.18}$$

Die Menge der rationalen Zahlen ist die Menge der Äquivalenzklassen

$$\mathbb{Q} = \{[a, b], \quad a, b \in \mathbb{Z}\}\,. \tag{1.19}$$

Die Operationen und Relationen entstehen wieder aus gewohnten Rechenregeln. Für die Addition zweier rationaler Zahlen

$$[a, b] + [c, d] := [ac, ad + bc], \tag{1.20}$$

werden die Brüche auf „einen gemeinsamen Nenner“ (nämlich ac) gebracht. Neben Abgeschlossenheit und Verträglichkeit mit der Äquivalenzrelation muss man nun noch zeigen, dass keine Zahlen der Form $[0, b]$ entstehen. Dies sieht man für die Addition, da aus $a \neq 0, c \neq 0$ direkt $ac \neq 0$ folgt – um das streng nachzuweisen, verwendet man die Ordnungsrelation auf den ganzen Zahlen und unterscheidet die Fälle nach Vorzeichen von a und c. Auch bei der Einführung der Ordnung muss man auf das Vorzeichen achtgeben:

$$[a, b] \leq [c, d] \quad \Longleftrightarrow \quad (ac \leq 0 \wedge ad \leq bc) \vee (0 \leq ac \wedge bc \leq ad).$$

Bei der Definition der Inversen sehen wir, dass nicht jedes Paar eine Inverse hat. Das liegt daran, dass der Wertebereich von a und b unterschiedlich ist – nur für a hatten wir vereinbart $a \neq 0$. Da sich bei der Inversen von $[a, b]$ die Elemente vertauschen, ist die Inverse nur definiert, wenn $b \neq 0$ gilt. Die Zahl $[a, 0]$ ist in der Tat die Null, also auch das additive Neutralelement, wovon man sich leicht überzeugt. Als Inverse und Division ergeben sich

$$b \neq 0, \quad [a, b]^{-1} := [b, a], \qquad d \neq 0, \quad [a, b]/[c, d] := [a, b] \cdot [c, d]^{-1}. \tag{1.21}$$

[1] Dieses Vorgehen entspricht der klassischen Konstruktion der rationalen Zahlen. Es ist allerdings nicht wirklich notwendig, den Fall $a = 0$ auszuschließen. Lässt man $a = 0$ zu, so ergibt sich eine explizite Repräsentation des Werts $\pm\infty$. Ein Vorteil dieser Konstruktion ist, dass man dann bequem mit unendlichen Werten rechnen kann. Dies wird in Form von *homogenen Koordinaten* auch bei allen geometrischen Problemen gemacht.

1.4.2. Darstellung der rationalen Zahlen

Wie die ganzen Zahlen werden rationale Zahlen durch die Wahl bestimmter Paare (in diesem Fall von Zähler und Nenner) dargestellt. Eine mögliche und natürliche erscheinde Wahl wäre das Paar (a, b) immer vollständig zu kürzen. Dies wird nur dann gemacht, wenn man möglichst alle rationalen Zahlen (zumindest im Rahmen des verfügbaren Speichers) zulassen will.

Weitaus üblicher sind andere Darstellungen, bei denen man sowohl Zähler wie auch Nenner im Zahlenbereich einschränkt. Die *Festkomma*-Repräsentation ergibt sich aus der Idee einen festen Nenner zu wählen, also auf Paare der Form (k, x') zu beschränken. Das Prinzip ist ähnlich dem Exzess-Code: genau wie man beim Exzess-Code von der dargestellten Zahl noch einen festen Wert abziehen muss, wird bei der Festkommadarstellung die dargestellte Zahl durch den festen Nenner geteilt. Das ist besonders einfach, wenn der Nenner eine Potenz der Basis b ist, also $k = b^n$. Dann kann man die Division einfach als „Verschiebung des Kommas“ interpretieren:

$$x_{n-1} \cdots x_m, x_{m-1} \cdots x_1 x_0 = \frac{\sum_{i=0}^{n-1} x_i b^i}{b^m} = \sum_{i=0}^{n-1} x_i b^{i-m} \tag{1.22}$$

Zum Beispiel ergibt Division durch $100 = 10^2$ im Dezimalsystem für jede ganze Zahl 2 Stellen „hinter dem Komma“:

$$\frac{1234}{100} = 12{,}34 = 1 \cdot 10^1 + 2 \cdot 10^0 + 3 \cdot 10^{-1} + 4 \cdot 10^{-2} = \sum_{i=0}^{3} (4 - i) \cdot 10^{i-2}$$

Ein großer Vorteil dieser Darstellung liegt darin, dass die Addition und Subtraktion auf der Zahlendarstellung genau wie für ganze Zahlen durchgeführt werden können. Bei Multiplikation und Division muss man (ähnlich zur Addition und Subtraktion beim 2-er Komplement und der Exzesskodierung) das Ergebnis durch Multiplikation mit b^m, also Verschiebung der Stellen, korrigieren.

Der feste absolute Wertebereich, der sich aus der festen Wahl von b^m ergibt, ist für viele Anwendungen ungeeignet. Eine sehr nützliche und weit verbreitete Darstellung ergibt sich, durch Darstellung von Zähler und Exponenten m. Dabei beschreibt m die Position des Kommas, das durch Wahl von m „gleitet“. Den *Gleitkommazahlen* widmen wir im Folgenden einen eigenen Abschnitt.

1.5. Gleitkommazahlen

Die wesentliche Idee von Gleitkommazahlen besteht darin, den Nenner einer rationalen Zahl auf die Menge b^e einzuschränken und (neben dem Zähler) nur den *Exponenten* e zu speichern, also mit Paaren (b^e, x') zu rechnen. Statt vom Zähler x' spricht man bei Gleitkommazahlen von der *Mantisse* m; und statt durch b^e zu teilen wird mit b^e multipliziert (was letztlich nur das Vorzeichen des Exponenten anders interpretiert). Beides ist Konvention und hat keinen Einfluss auf die Eigenschaften der Zahldarstellung. Mit m und e wird also (bei fest vereinbarter Basis b) die Zahl $m \cdot b^e$ beschrieben.

Mantisse und Exponent sind ganze Zahlen und können mit den bereits eingeführten Systemen repräsentiert werden. Allerdings wird der Wertebereich der Mantisse dabei zugunsten einer eindeutigen Zahlendarstellung nicht vollständig ausgeschöpft. Wir sehen, dass durch führende

Nullen in der Mantisse mehrere Darstellungen einer Zahl möglich wären, z.B. $10{,}1 = 101 \cdot 10^{-1} = 1010 \cdot 10^{-2}$. Zu einer eindeutigen Darstellung gelangt man bei fester Anzahl von Stellen in der Mantisse dadurch, dass die führende Stelle von Null verschieden sein muss, also $m \geq 1$. Eine (weitere) Konvention schreibt vor, dass die erste (von Null) verschiedene Ziffer immer als die Stelle vor dem Komma zu verstehen ist, also $m < b$. Zusammen hat man damit $1 \leq m < b$, was den Exponenten festlegt. Nach diesen Konventionen *müssen* wir also für das oben gewählte Beispiel schreiben: $1{,}01 \cdot 10^1$.

Bevor wir nun Gleitkommazahlen in einem Stellenwertsystem konkret angeben können, müssen wir noch die Darstellung der Vorzeichen wählen. Dies wird bei Gleitkommazahlen typischerweise anders als bei ganzen Zahlen gemacht. Das Vorzeichen der Mantisse wird durch explizite Speicherung eines Vorzeichenbits repräsentiert. Für den Exponenten wird die Exzess-Codierung verwendet.

Es ergibt sich folgende Gleitkommadarstellung mit n_m Stellen in der Mantisse und n_e Stellen im Exponenten:

$$\underbrace{\pm}_{\text{Vorzeichen}} \underbrace{x_{n_m+n_e-1} \ \cdots \ x_{n_m}}_{n_e \text{ Stellen Exponent}} \underbrace{x_{n_m-1} \ \cdots \ x_0}_{n_m \text{ Stellen Mantisse}} \equiv \pm \cdot x_{n_m-1}, x_{n_m-2} \ \cdots \ x_0 \cdot b^e \equiv \pm m \cdot b^e$$

$$\begin{aligned} m &= \sum_{i=0}^{n_m-1} x_i b^{i-(n_m-1)}, \qquad 1 \leq m < b \\ e &= \sum_{i=0}^{n_e-1} x_{i+n_m} b^i - B, \qquad -B \leq e < b^{n_e} - B \\ x_i &\in [0, \ldots, b-1], \quad x_{n_m-1} \neq 0. \end{aligned} \tag{1.23}$$

Hierbei ist B der in diesem Kontext *Bias* genannte Exzess zur Darstellung des vorzeichenbehafteten Exponenten. In der Binärdarstellung wählt man wie erwähnt B typischerweise als höchstmögliche Zweierpotenz. Darüberhinaus ergibt sich eine weitere Einsparmöglichkeit: die erste Stelle der Mantisse x_{n_m-1} muss von Null verschieden sein. Im Binärsystem ergibt sich damit zwingend $x_{n_m-1} = 1$. Es gibt daher Umsetzungen des Gleitkommaformats, in denen diese Stelle eingespart wird, man also mit $n = n_m + n_e + 1$ Stellen eine Stelle mehr in der Mantisse erhält. Dies ist aber nicht bei allen Umsetzungen der Fall.

1.5.1. Wertebereich

Wir betrachten im Folgenden nur den positiven Teil der Gleitkommazahlen. Alle Aussagen gelten analog für die negativen Zahlen. Auf Basis der eingeführten Darstellung und der Forderung $x_{n_m-1} \neq 0$ ergibt sich direkt die kleinste (nicht-negative) Zahl, nämlich b^{-B}. Das bedeutet, die Null hat keine (direkte) Darstellung in den Gleitkommazahlen, was gemeinhin als unbefriedigend empfunden wird. Es ist üblich die Null dennoch zu codieren, und dafür auf andere eigentlich darstellbare Zahlen zu verzichten. Dafür wird meist der kleinste mögliche Exponent $e = -B$ verwendet, bei der die Darstellung des Exponenten nur aus Nullen besteht.

Durch die Verwendung des Exponenten zur Darstellung der Null, bleiben Vorzeichen und Mantisse für diesen Exponenten eigentlich ohne Verwendung. Man unterscheidet aber zumindest in den meisten Implementierungen zwischen $+0$ und -0. Dies ist sinnvoll, da Gleitkommazahlen immer Approximationen sind, also die Zahl 0 nicht unbedingt exakt Null bedeutet. Mit dem Vorzeichen kann man ausdrücken, ob der (sehr kleine) Wert eher positives oder negatives Vorzeichen hat.

Ferner verwenden einige Implementierungen die Mantisse zur Beschreibung weiterer Zahlen, die zwischen 0 und b^{-B} liegen. Dieser Zahlenbereich wird dann linear gefüllt. Man spricht hier von *sub-* oder *denormalisierten* Zahlen. Dieser Begriff kommt daher, dass man für diese Zahlen akzeptiert, dass die Zahl führende Nullen hat. Ohne subnormale Zahlen ist die kleinste Gleitkommazahl also wegen der Repräsentation der Null b^{-B+1}.

Auch der größtmögliche Exponent $e = b^{n_e} - 1 - B$ wird typischerweise nicht gemäß der Zahlendarstellung verwendet. Er repräsentiert meist unendlich, also ein Rechnungsergebnis, das über die größte darstellbare Zahl hinausgeht. Die größte darstellbare Zahl entsteht deshalb mit dem Exponenten $e = b^{n_e} - 2 - B$ und der größten Mantisse, hängt also auch von der Länge der Mantisse ab. Diese Zahl ist in jedem Fall kleiner als $b^{b^{n_e}-1-B}$, was uns für die Charaktisierung des Wertebereichs an dieser Stelle genügen soll.

Null und Unendlich können sehr einfach als Spezialfall in die Standardoperationen $+, -, \cdot, /$ integriert werden. Insbesondere erlauben die expliziten Darstellungen die Operationen $x + 0 = x, x + \infty = \infty, 0 \cdot x = 0, \infty \cdot x = \infty$, usw.

Wir halten fest, dass der Wertebereich der Standarddarstellung von der Basis b und der Repräsentation des Exponenten, also den Parametern n_e und B bestimmt ist. Um neben der kleinsten nicht mehr darstellbaren Zahl auch die größte darstellbare Zahl exakt anzugeben, braucht man auch noch die Länge der Mantisse n_m.

1.5.2. Approximationsfehler, Maschinengenauigkeit

Die Anzahl der Exponentenbits schränkt den Wertebereich der Gleitkommazahlen ein. Innerhalb dieses Wertebereichs führt die eingeschränkte Anzahl an Mantissenstellen dazu, dass nicht alle rationalen Zahlen dargestellt werden können. Wir beschäftigen uns nun mit dem Fehler, der bei der Darstellung von rationalen Zahlen als Gleitkommazahlen im Wertebereich entsteht.

Um bei den Rechnungen nicht stets auf die betragsmäßig kleinsten und größten darsttellbaren Zahlen Rücksicht nehmen zu müssen, führen wir die Menge $\mathbb{G} \subset \mathbb{Q}$ der (positiven) Gleitkommazahlen ein, die sich durch feste Wahl von b und n_m aber ohne Einschränkung des Exponenten ergibt:

$$\mathbb{G}(b, n_m) = \left\{ x \in \mathbb{R} : x = \sum_{i=0}^{n_m-1} x_i b^{i+e-n_m+1}, \quad x_i \in \{0, \ldots, b-1\}, e \in \mathbb{Z} \right\}. \tag{1.24}$$

Wenn n_m und b klar und nicht erheblich sind, schreiben wir auch kürzer $\mathbb{G}$. Ohne Festlegung der Darstellung des Exponenten (durch Wahl von n_e und B) ist diese Menge unendlich – anders als Gleitkommazahlen in der Praxis. Man beachte auch, dass die Menge $\mathbb{G}$ immer nur positive Elemente enthält, also $x \in \mathbb{G} \Rightarrow x > 0$. Insbesondere ist $0 \notin \mathbb{G}$. Wir werden im folgenden daher auch nur den Fehler positiver Zahlen betrachten; der Fehler im negativen Zahlenbereich ist identisch.

Zur Abbildung einer beliebigen rationalen Zahl $x \in \mathbb{Q}$ auf eine Gleitkommazahl $\hat{x} \in \mathbb{G}$ führen wir die Operation $G : \mathbb{Q}^+ \mapsto \mathbb{G}$, die jede positive rationale Zahl auf eine Gleitkommazahl aus $\mathbb{G}$ abbildet. Für eine solche Abbildung erwarten wir zumindest, dass rationale Zahlen x, für die es eine exakte Darstellung in $\mathbb{G}$ gibt, auf diese Darstellung abgebildet werden. Alle anderen rationalen Zahlen sollen zumindest entweder auf die nächst-kleinere oder nächst-größere

Gleitkommazahl abgebildet werden. Diese Forderung impliziert die erste Bedingung und wir schreiben

$$G(x) \in \{\max_{\hat{x}\in\mathbb{G}} \ \hat{x} \leq x, \min_{\hat{x}\in\mathbb{G}} \ \hat{x} \geq x\}. \tag{1.25}$$

Die Funktion wählt also entweder die größte Zahl $\hat{x}$, die nicht größer ist als x (Abrundung auf die nächst-kleinere Gleitkommazahl) oder die kleinste Zahl, die nicht kleiner als x ist (Aufrundung auf die nächst-größere Gleitkommazahl). In der Praxis ist es bequem immer abzurunden, weil man intern mit mehr als n_m Mantissenstellen rechnen kann, und dann nur die vorgesehenen n_m Stellen speichert (und die restlichen Stellen verwirft). Es gibt allerdings auch Implementierungen, die die nächste Zahl aus den Gleitkommazahlen bestimmen, also je nach Zahl auf- oder abrunden. Dieses Vorgehen führt zu kleineren Fehlern, ist aber aufwändiger. In der folgenden Diskussion gehen wir zunächst davon aus, dass abgerundet wird. Es ist nicht schwierig, die Argumente für kaufmännisches Runden oder Aufrunden anzupassen.

Der Unterschied zwischen x und $G(x)$ hängt offenbar von den gewählten Parametern b und n_m in $\mathbb{G}(b, n_m)$ ab. Im Falle von Abrunden ist der Abstand groß, wenn x um eine kleine Zahl ϵ kleiner ist als eine der Gleitkommazahlen, also am oberen Ende des Intervalls zweier benachbarter Gleitkommazahlen liegt. Um den maximalen Abbildungsfehler zu verstehen, müssen wir also den Abstand zweier Gleitkommazahlen bestimmen. Für zwei beliebige, benachbarte Gleitkommazahlen ist der Abstand durch ihre Differenz gegeben. Wir betrachten die Mantissen zweier Zahlen, wobei die größere in der letzte Stelle x_0 um eins größer ist als die kleinere. Wir nehmen dabei an, dass kein Überlauf durch die Addition in der letzten Stelle auftritt.

$$\begin{array}{rccccccc} & x_{n_m-1} & , & x_{n_m-2} & \cdots & x_0+1 & \cdot b^e \\ - & x_{n_m-1} & , & x_{n_m-2} & \cdots & x_0 & \cdot b^e \\ \hline & 0 & , & 0 & \cdots & 1 & \cdot b^e \end{array} \tag{1.26}$$

Die Rechnung erfolgt hier wie bei der schriftlichen Subtraktion. Da die Zahlen den gleichen Exponenten e haben, können sie ohne vorherige Anpassung direkt subtrahiert werden. Das Ergebnis unter dem Strich entspricht noch nicht unseren Konventionen zur Darstellung in Gleitkommzahlen, da die führende Stelle nicht größer als Null ist. Die notwendige Verschiebung des Kommas führt zu

$$1 \quad , \quad 0 \quad \cdots \quad 0 \quad \cdot b^{e-(n_m-1)}. \tag{1.27}$$

Um eine konkrete Vorstellung über den Abstand benachbarter Gleitkommazahlen zu erhalten, wählen wir $b = 10$ und $n_m = 1$, d.h. ein in der Praxis wenig geeignetes aber intuitives Gleikommazahlen-Format mit einer Ziffer und Basis 10. Für den Abstand erhalten wir damit:[2]

e	Abstand
-2	0,01
-1	0,1
0	1,0
1	10,0
2	100,0

Vielleicht ein wenig überraschend hängt der Abstand also vom Exponenten ab. Zum Beispiel haben Gleitkommazahlen in $[1, 10)$ einen Abstand von 1, in $[10, 100)$ aber einen Abstand von 10. Wir sehen also, dass es nicht ***einen*** Abstand zwischen benachbarten Gleitkommazahlen gibt, sondern dass dieser von der Größe der Zahlen abhängt. Dies ist in Abbildung 1.1 graphisch

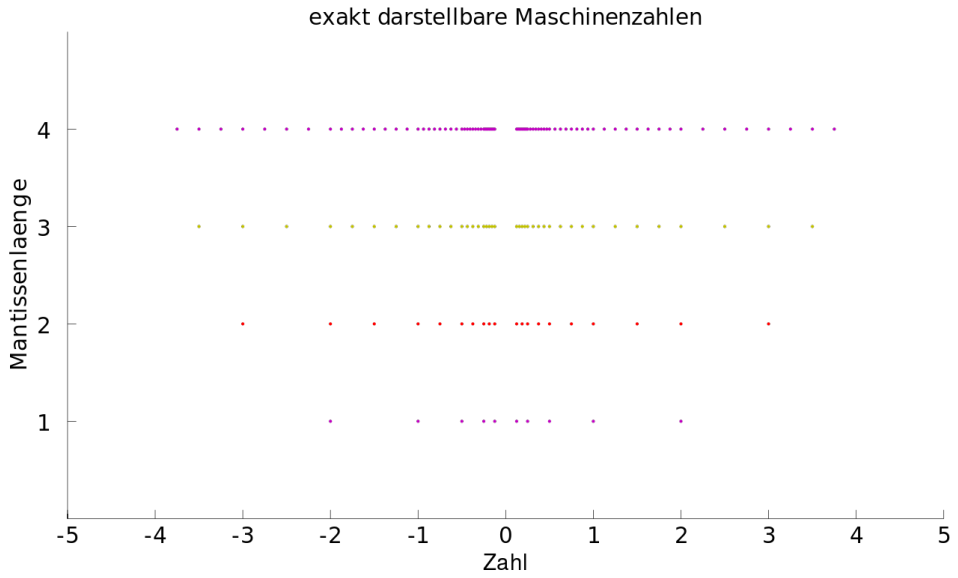


Abbildung 1.1.: *Abstände von Gleitkommazahlen mit Basis $b = 2$ (Quelle: `http://upload.wikimedia.org/wikipedia/de/0/0b/Gleitkommazahlen.svg`).*

dargestellt. Daraus folgt, dass auch die absolute Differenz zwischen einer rationalen Zahl $x > 0$ und ihrer Abbildung $G(x)$ in die Gleitkommazahlen von der Größe x abhängt. Zum Beispiel haben wir für die Abbildungsfehler:

$$\begin{aligned} G(3-\epsilon) &= 2 \qquad |3-\epsilon-G(3-\epsilon)| = 1-\epsilon, \qquad 1 > \epsilon > 0 \\ G(30-\epsilon) &= 20 \qquad |30-\epsilon-G(30-\epsilon)| = 10-\epsilon \qquad 10 > \epsilon > 0 \end{aligned}$$

Der größte Fehler ergibt sich für $\epsilon \to 0$ und entspricht dann genau der Intervallbreite.

Die Abhängigkeit der Abstände zwischen Gleitkommazahlen und damit der Abbildungsfehler von der Größe ist gewollt. Für viele Anwendungen ist nicht der absolute Fehler von Bedeutung sondern der relative. So erwartet man z.B. für die Bestimmung des Durchmessers der Erde einen anderen absoluten Fehler als für den Durchmesser eines Atoms. Der relative Fehler ergibt sich, wenn man den absoluten Fehler durch die Größe des gewünschten Werts teilt:

$$\delta = \frac{|x - \hat{x}|}{|x|}. \tag{1.28}$$

Hier ist x der korrekte Wert und $\hat{x}$ der fehlerbehaftete – in unserem Kontext ist $x \in \mathbb{Q}^+$ eine beliebige positive rationale Zahl und $\hat{x} \in \mathbb{G}$ eine im Allgemeinen fehlerbehaftete Approximation durch eine Gleitkommazahl.

Wie besprochen hängt der absolute Fehler $|x - G(x)|$, der durch die Abbildung G auf die Gleitkommazahlen entsteht von der Größe der Zahl x ab. Interessanterweise ist der relative Fehler $\frac{|x-G(x)|}{|x|}$ weitgehend unabhänangig von der Größe. Um dies zu sehen, schreiben wir

$$x = x' b^e, 1 \leq x' < b, \tag{1.29}$$

[2] Es sei daran erinnert, dass, auf Grund der Bedingung $x_{n_m-1} > 0$, der Exponent nicht frei wählbar ist, sondern eine inhärente Eigenschaft der Zahl ist.

stellen also $x \in \mathbb{Q}^+$ wie eine Gleitkommazahl in $\mathbb{G}$ dar, nur dass x' nicht auf eine feste Zahl von Stellen beschränkt ist und die Darstellung dadurch exakt wird. In dieser Darstellung beeinflusst die Abbildung auf die Gleitkommazahlen nur x', da Ab- oder Aufrunden nur den Wert von x' verändert. Es gilt also $G(x) = G(x'b^e) = G(x')b^e$. Betrachten wir jetzt den relativen Fehler

$$\frac{|x - G(x)|}{|x|} = \frac{|x'b^e - G(x')b^e|}{|x'b^e|} = \frac{|(x' - G(x'))|b^e}{|x'|b^e} = \frac{|x' - G(x')|}{|x'|}, \tag{1.30}$$

so sehen wir, dass Basis b und Exponent e keinen Effekt haben.

Da der Fehler offenbar unabhängig von der Größe von x beschränkt ist, liegt es nahe, nach dem größten relativen Fehler über alle $x \in Q^+$ bei der Abbildung auf $\mathbb{G}(b, n_m)$ zu fragen. Das heißt, wir suchen eine möglichst kleine positive Zahl ϵ für die gilt

$$\epsilon \geq \frac{|x - G(x)|}{|x|} = \frac{|x' - G(x')|}{|x'|}, \quad \forall x \in \mathbb{Q}^+. \tag{1.31}$$

Wir haben hier mit Absicht kein Minimierungsproblem für ϵ formuliert, denn die kleinstmögliche Wahl ist nicht nur von den Parametern b und n_m abhängig, sondern auch davon, in welchen Fällen G ab- und aufrundet. Eine nützliche kleine Schranke ϵ können wir dennoch mit folgender Überlegung finden: Der Bruch wird groß, wenn wir den Zähler so groß und den Nenner so klein wie möglich wählen.

Aus unserer Diskussion zu den minimalen Bedingungen für G wird klar, dass der Zähler nicht größer werden kann als der Abstand zweier Gleitkommazahlen. Wie oben erläutert ist der Abstand zweier benachbarter Gleitkommazahlen der Größenordnung b^e beschränkt durch b^{e-n_m+1}. Wegen $1 \leq x' < b$ ist hier $e = 0$. Aus dieser Einschränkung von x' ergibt sich auch der kleinste mögliche Nenner, nämlich $x' = 1$.

Aus diesen Überlegung erhalten wir die Schranke

$$\epsilon = b^{-n_m+1} \geq \frac{|x - G(x)|}{|x|}, \tag{1.32}$$

die für alle zulässigen Abbildungsfunktionen G auf $\mathbb{G}(b, n_m)$ gilt. Es ist unklar, ob diese Schranke für eine konkrete Abbildungsfunktion bestmöglich ist, also ob es im konkreten Fall nicht auch ein kleineres ϵ geben würde. Es ist allgemein üblich, keine kleineren Werte zu verwenden, da zumindest Funktionen G existieren, für die es keine kleinere Schranke gibt.

Den Wert ϵ nennt man auch *Maschinengenauigkeit*. Wie wir später sehen werden charakterisiert er nicht nur den maximalen relativen Fehler bei der Approximation einer rationalen Zahl durch eine Gleitkommazahl, sondern auch die von verschiedenen Operationen zu erwartende Genauigkeit.

Aus der Beobachtung, dass der größte Abbildungsfehler in der Nähe der Zahl 1 beobachtet wird und im wesentlichen vom Abstand zweier Gleitkommazahlen in diesem Intervall abhängt, kann man die Maschinengenauigkeit auch einfacher ausdrücken. Wenn G immer abrundet, dann suchen wir die kleinste Zahl $x \in \mathbb{Q}^+$ die wir zu 1 addieren müssen, so dass das Ergebnis nicht mehr 1 ist:

$$\epsilon = \arg\min_{x \in \mathbb{Q}^+} G(1 + x) > 1, \quad G \text{ rundet ab.} \tag{1.33}$$

1.5.3. Fehler bei Elementaroperationen, Auslöschung

Neben der notwendigen Approximation, um eine beliebige rationale Zahl im Gleitkommaformat darstellen zu können, treten weitere Fehler bei der Durchführung von Berechnungen mit Gleitkommazahlen auf. Selbst für die elementaren arithmetischen Operationen beobachten wir, dass das Ergebnis im Allgemeinen keine Gleitkommazahl ist, d.h. die Gleitkommazahlen $\mathbb{G}$ sind gegenüber den auf den rationalen Zahlen definierten Operationen Addition, Subtraktion, Multiplikation, Division nicht abgeschlossen. Davon überzeugen wir uns an einem Beispiel. Wir betrachten die rationalen Zahlen 6 und 8 im Format $\mathbb{G}(10, 1)$. Beide Zahlen lassen sich exakt darstellen. Für die Summe finden wir $G(6 + 8) = G(14) = 10$ und für das Produkt $G(6 \cdot 8) = G(48) = 40$. Diese Beobachtung fassen wir folgt zusammen

$$x * y \neq G(x * y) \neq G(x) * G(y) \neq G(G(x) * G(y)) \tag{1.34}$$

wobei "$*$" hier für eine beliebige elementare arithmetische Operation steht.

Da nach jeder Operation erneut auf die Gleitkommazahlen abgebildet werden muss und dabei immer ein Fehler entstehen kann gelten auch die sonst übliche Assoziativität für diese Operationen sowie die Distributivität zwischen Addition und Multiplikation nicht mehr.

Wir wissen, dass bei jeder Anwendung der Abbildung G der relative Fehler durch Maschinengenauigkeit ϵ beschränkt ist. Welcher Fehler entsteht bei der Durchführung einer elementaren Operation? Gegeben zwei Gleitkommazahlen $\hat{x}$ und $\hat{y}$ so könnten wir die Elementaroperation durch die Verwendung von zusätzlichem temporären Speicher exakt bestimmen (man braucht dafür nicht mehr als die doppelte Anzahl an Mantissenstellen) und anschließend auf die verfügbaren Gleitkommazahlen abbilden. Der relative Fehler, der bei Elementaroperationen auf Gleitkommazahlen ensteht entspricht dann genau dem Abbildungsfehler, ist also durch die Maschinengenauigkeit ϵ beschränkt. Diese Aussage gilt für alle Elementaroperationen.

Wenn wir zwei rationale Zahlen x und y betrachten und in gegebenen Gleitkommazahlen eine Elementaroperation approximieren, müssen wir sie zunächst in die Gleitkommazahlen abbilden, anschließend können wir die Operation ausführen und dann das Ergebnis wieder in die Gleitkommazahlen abbilden. Die Operation können wir wie bemerkt exakt durchführen, der Fehler der Abbildungen ist jeweils durch ϵ beschränkt. Ist deswegen auch der Fehler von $G(G(x) * G(y))$ relativ zu $x * y$ durch (ein kleines Vielfaches von) ϵ beschränkt? Nein! Der relative Fehler hängt von der Operation ab!

Wir betrachten den relativen Fehler $\delta = |G(x) * G(y) - x * y|/|x * y|$ der im Allgemeinen kleiner sein wird als $|G(G(x) * G(y)) - x * y|/|x * y|$. Dazu stellen wir x und y als Summe aus dem Teil, der in $\mathbb{G}$ liegt, und einem Rest dar. Für x bedeutet das

$$x = g_x + r_x, \quad g_x = G(x) = m_x b^{e_x} \in \mathbb{G}, \quad 0 \leq r_x < 1 \cdot b^{e_x} b^{-n_m+1} \tag{1.35}$$

und analog für y. Wir betrachten zunächst das Produkt:

$$xy = (g_x + r_x)(g_y + r_y) = g_x g_y + r_x g_y + r_y g_x + r_x r_y \tag{1.36}$$

$$G(x)G(y) = g_x g_y. \tag{1.37}$$

Damit ist der relative Fehler

$$\delta = \frac{r_x g_y + r_y g_x + r_x r_y}{g_x g_y + r_x g_y + r_y g_x + r_x r_y}. \tag{1.38}$$

Zur Abschätzung nach oben wählen wir den Zähler so groß wie möglich und den Nenner so klein wie möglich. Dazu nehmen wir für r im Zähler den Wert $b^e b^{-n_m+1}$ und im Nenner 0; für m im Zähler den Wert b und im Nenner 1.

$$\delta < \frac{(b \cdot b^{e_y} b^{e_x} b^{-n_m+1} + b \cdot b^{e_x} b^{e_y} b^{-n_m+1} + b^{e_y} b^{-n_m+1} b^{e_x} b^{-n_m+1}}{b^{e_x} b^{e_y}} \tag{1.39}$$

$$= b \cdot b^{-n_m+1} + b \cdot b^{-n_m+1} + b^{-n_m+1} b^{-n_m+1} = 2b\epsilon + \epsilon^2. \tag{1.40}$$

Der Fehler relativ zum korrekten Ergebnis xy ist also in der Tat in der Größenordnung der Maschinengenauigkeit. Ein analoges Ergebnis bekommt man für die Division.

Wir betrachten jetzt die Addition:

$$x + y = g_x + r_x + g_y + r_y \tag{1.41}$$

$$G(x) + G(y) = g_x + g_y. \tag{1.42}$$

Damit ergibt sich der relative Fehler

$$\delta = \frac{r_x + r_y}{g_x + g_y + r_x + r_y}. \tag{1.43}$$

Wieder verwenden wir die Definitionen von g und r und wählen im Zähler Abschätzungen nach oben und im Nenner Abschätzungen nach unten.

$$\delta < \frac{b^{e_y} b^{-n_m+1} + b^{e_x} b^{-n_m+1}}{b^{e_x} + b^{e_y}} \tag{1.44}$$

$$= b^{-n_m+1} = \epsilon \tag{1.45}$$

Der relative Fehler entspricht also genau der Maschinengenauigkeit.

Analog zur Summe erhalten wir für die Abschätzung des Fehlers für die Differenz von $x - y$ für $x \geq y$

$$\delta = \frac{|r_x - r_y|}{g_x - g_y + r_x - r_y} = \frac{|r_x - r_y|}{x - y} \tag{1.46}$$

Damit der Zähler möglichst klein wird wählt man x und y möglichst nah beisammen. Dabei soll der Zähler allerdings nicht Null werden. Dazu nimmt man eine beliebige Zahl $g \in \mathbb{G}$ und setzt $x = g$ und $y = g - \mu$. Dann ist $r_x = 0$ und für kleines μ wird r_y groß, also $r_y \approx b^{e_y} b^{-n_m+1}$. Zusammen erhält man

$$\delta < \frac{b^{e_y} b^{-n_m+1}}{\mu}. \tag{1.47}$$

Da sich μ beliebig klein wählen lässt ist der relative Fehler bei der Subtraktion also unbeschränkt.

Warum ist der Fehler bei Multiplikation, Division und Addition im Bereich der Maschinengenauigkeit, aber bei der Subtraktion beliebig groß? Intuitiv gesagt werden bei den ersten drei Operation jeweils alle Mantissenstellen für die Repräsentation des Ergebnisses verwendet. Bei der Subtraktion wird der Fehler groß, wenn die Zahlen nah beisammen sind, also die führenden Stellen der Mantisse alle Null werden. Man spricht hier von *Auslöschung* (der führenden Stellen) und verwendet nur einen (kleinen) Teil der Mantisse zur Darstellung des korrekten Ergebnis. Das erklärt warum der Fehler nicht mehr von der Anzahl der Mantissenstellen abhängt.

In der Praxis gilt es stets zu überlegen, ob es bei Subtraktion (oder Addition einer positiven und negativen Zahl) zu einem kleinen Ergebnis kommen kann. Solche Situationen sollte man soweit es geht vermeiden.

Abschließend betrachten wir noch ein Beispiel, in dem die relativen Fehler der Elementaroperation deutlich werden. Gegeben sei $\mathbb{G}(10,3)$ und drei Zahlen a, b, c, die wie folgt dargestellt werden:

	$\mathbb{R}$	$\mathbb{G}$	rel. Fehler
a	1,22	$1{,}22 \times 10^0$	0,0
b	3,34	$3{,}34 \times 10^0$	0,0
c	2,28	$2{,}28 \times 10^0$	0,0

In der 3-Ziffer Gleitkomma-Darstellung wollen wir $b^2 - 4\,a\,c$ berechnen, was zur Lösung einer quadratischen Gleichung bestimmt werden muss. Wir nehmen an, dass zunächst b^2 und $4\,a\,c$ berechnet werden, und dann die Differenz gebildet wird. Für die Zwischenergebnisse erhalten wir:

	$\mathbb{R}$	$\mathbb{G}$	rel. Fehler
b^2	11,1556	$1{,}12 \times 10^1$	0,00398
$4\,a\,c$	11,1264	$1{,}11 \times 10^1$	0,00237

Für die Gleitkomma-Darstellung des obigen Ergebnisses muss eine Rundung erfolgen, da nur drei Ziffern für die Mantisse zur Verfügung stehen. Der relative Fehler, welcher durch die Rundung entsteht, ist jedoch klein, so dass er für die meisten Anwendungen als vernachlässigbar gelten kann. Um das Ergebnis zu erhalten, muss noch eine Subtraktion ausgeführt werden. Hierfür erhalten wir:

	$\mathbb{R}$	$\mathbb{G}$	rel. Fehler
$b^2 - 4\,a\,c$	0,0292	$1{,}00 \times 10^{-1}$	2,42466

Trotz des sehr kleinen relativen Fehlers für b^2 und $4\,a\,c$ im Zwischenergebnis, erhalten wir durch die Subtraktion ein Ergebnis, welches für die meisten Anwendungen unbrauchbar ist.

1.6. Reelle Zahlen

Natürlich sind rationale Zahlen durch ihre endliche Darstellung beschränkt. Aber könnte man alle Zahlen als rationale Zahlen darstellen, wenn man nur genug Speicher zur Verfügung hätte und den Nenner als beliebige ganze Zahl beschreibt?

Betrachten wir ein Blatt Papier. Das Seitenverhältnis soll dem *goldenen Schnitt* ϕ entsprechen, eine geometrische Konfiguration, der man besondere Harmonie nachsagt, und die uns in vielen natürlichen Phänomenen begegnet. Ein Rechteck entspricht dem goldenen Schnitt, wenn man ein Quadrat mit der kürzeren Seitenlänge entfernt und der verbleibende Teil wieder ein Rechteck mit goldenem Schnitt ergibt. Offenbar muss ein solches Rechteck eine längere und eine kürzere Seite habe (ansonsten wäre nach dem Entfernen des Quadrates nichts mehr übrig). Bezeichne $a > 0$ die längere Seite und $b > 0$ die kürzere. Entfernt man ein Quadrat mit Seitenlänge b so

verbleibt ein Rechteck mit der längeren Seite b und der kürzeren Seite $a-b>0$. Die Forderung, dass die Verhältnisse der Seitenlängen gleich sein sollen führt auf

$$\frac{a}{b}=\frac{b}{a-b}. \tag{1.48}$$

Diese Gleichung enthält bereits den Nachweis, dass das Seitenverhältnis a/b keine rationale Zahl sein kann: wähle $a>0$ und $b>0$ aus den ganzen Zahlen. Nach unserer Konstruktion sind $0<b<a$ und $0<a-b<b$ zwei echt kleinere aber von Null verschiedene Zahlen, die das gleiche Verhältnis ausdrücken. Diesen Prozess können wir beliebig fortsetzen und erhalten immer kleinere ganze Zahlen, die stets das gleiche Verhältnis ausdrücken. Das kann aber nicht sein, denn a und b sind endlich und müssten irgendwann Null werden.

Auch wenn wir das Blatt Papier gemäß der DIN konstruieren, ist das Seitenverhältnis nicht rational. Der Grundgedanke des DIN-Papiers ist, dass das Seitenverhältnis beim Halbieren erhalten bleibt. Oder konkreter, wenn man zwei DIN A4 Papiere an der langen Seiten aneinander legt, entspricht das Ergebnis einem DIN A3 Papier. Beide Papiere haben das gleiche Seitenverhältnis. Bezeichnet wieder a die längere und b die kürzere Seite, so führt die Konstruktion auf

$$\frac{a}{b}=2ba \quad\Longleftrightarrow\quad a^2=2b^2. \tag{1.49}$$

Wir wählen wieder a und b aus den ganzen Zahlen so dass sie die Gleichung erfüllen. Offenbar können wir $2b^2$ ohne Rest durch zwei teilen, also auch a^2. Jede ganze Zahl a lässt sich schreiben als entweder $2c$ oder $2c+1$ für eine ganze Zahl c. Da $(2c+1)^2=4c^2+4c+1$ aber nicht durch zwei teilbar ist, muss a offenbar die Form $2c$ haben. Teilt man also $a^2=(2c)^2=4c^2=2b^2$ auf beiden Seiten durch 2 so gelangt man auf $2c^2=b^2$. Diese Gleichung entsprich der Ausgangssituation, nur hat b die Rolle von a übernommen und c die Rolle von b. Da wir beide Seiten der Ausgangsgleichung durch zwei geteilt haben gilt offenbar $b<a$ und $c<a$. Und wiederum ergibt sind der gewünschte Widerspruch daraus, dass a,b endlich sind und nur endlich oft kleiner werden können.

Es gibt also offenbar wohldefinierte Zahlen, die sich nicht als Bruch darstellen lassen - also keine rationalen und daher *irrationale* Zahlen sind. Die zwei Beispiele lassen sich allerdings als Nullstellen von (quadratischen) Polynomen darstellen. Wir nehmen diesen Gedanken im nächsten Abschnitt noch einmal auf. Selbst wenn wir die Beschreibung von Zahlen als Nullstellen von Polynomen zulassen, gibt es immer noch offenkundig wohldefinierte Zahlen, die sich nicht darstellen lassen. Ein bekanntes Beispiel ist das Verhältnis von Umfang und Durchmesser eines Kreises, die Kreiszahl π. Ein Beweis dafür, dass π nicht nur irrational, sondern vielmehr nicht Lösung eines Polynoms (mit endlichem Grad und ganzzahligen Koeffizienten) ist, führt für diesen Kurs zu weit. Alle Zahlen dieser Art bilden die *reellen* Zahlen $\mathbb{R}$.

Eines aber haben alle reellen Zahlen, also auch die irrationalen Zahlen ϕ und π (und alle anderen Zahlen dieser Art) gemein: sie lassen sich beliebig genau durch rationale Zahlen, also einen Bruch $p/q, p\in\mathbb{Z}, q\in\mathbb{N}$, approximieren. Wir beschränken uns zur Vermeidung von Fallunterscheidungen auf den Fall $x>0, x\in\mathbb{R}$. Wir behaupten also, dass sich p,q so wählen lassen, dass $|x-\frac{p}{q}|$ jede vorgegebene Schranke befriedigt. Dies sieht man sofort an

$$|x-\frac{p}{q}|\epsilon \quad\Longrightarrow\quad |qx-p|<\epsilon q, \tag{1.50}$$

da wir einfach $q>\epsilon^{-1}$ wählen können, so dass $\epsilon q>1$ wird und p als die nächste Ganzahl zu qx ausreicht. Der Nenner lässt aber auch beschränken. Konkret wollen wir zeigen, dass es stets

ein $q \leq n$ sowie entsprechendes p gibt, für das

$$|qx - p| < \frac{1}{n} \tag{1.51}$$

gilt. Das kann man durch eine Anwendung des Schubfachprinzips zeigen. Für jedes $k \in \{0, 1, \ldots, n\}$ können wir kx in einen Ganzzahlteil $\lfloor kx \rfloor$ und einen reellen Rest $0 \leq r_k < 1$ aufteilen. Es gibt offenbar nicht mehr als $n + 1$ solche Reste. Teilen wir das Intervall $[0, 1[$ in n gleiche Teile, so müssen also mindestens zwei der Reste in das gleiche (halboffene) Intervall der Breite $1/n$ fallen. Wir nennen die Indizes dieser zwei Reste i und j und nehmen an $i < j$. Es gilt also

$$\frac{1}{n} > |r_j - r_i| = |jx - \lfloor jx \rfloor - (ix - \lfloor ix \rfloor)| = |(j - i)x - (\lfloor jx \rfloor - \lfloor ix \rfloor)|. \tag{1.52}$$

Damit hat man $q = j - i$ und $p = \lfloor jx \rfloor - \lfloor ix \rfloor$ gefunden, die die gewünschten Eigenschaften besitzen. Teilt man die Ausgangsungleichung durch q so ergibt sich

$$\left| x - \frac{p}{q} \right| < \frac{1}{nq} \leq \frac{1}{q^2}. \tag{1.53}$$

Da $q \geq 1$ können wir also die rechte Seite durch entsprechende Wahl von n beliebig klein machen und finden jeweils einen Bruch mit Nenner nicht größer als n der die gewünschte Approximation ergibt.

Wir erinnern uns, dass wir auch von den rationalen Zahlen auf dem Rechner immer nur eine endliche Teilmenge darstellen konnten. Die Aussage oben zur Approximation von x gilt gleichermaßen für irrationale x wie für rationale, selbst wenn wir annehmen in der gegebenen Repräsentation gibt es keine p, q so dass $x = p/q$. Rationale und irrationale Zahlen lassen sich als in einem gewissen Sinn (den wir hier nicht weiter präzisieren wollen) gleich gut oder gleich schlecht durch rationale Zahlen repräsentieren. Diese Beobachtung rechtfertigt, für irrationale Zahlen keine weitere Repräsentation einzuführen. Auf Sonderfälle, in denen dies gerechtfertigt ist und gemacht wird, gehen wir hier nicht weiter ein.

2. Lineare Gleichungssysteme

2.1. Motivation: Computertomographie

Ziel der Computertomographie ist es, aus der Abschwächung einer Strahlung auf die Absorption und damit zumeist die Dichte innerhalb eines Objektes zu schließen. In vielen Fällen gibt die Dichte Aufschluss über die innere Zusammensetzung. In menschlichem Gewebe sind z.B. Knochen oder Zähne viel dichter als anderes Gewebe – sie können mit der Computertomographie dreidimensional rekonstruiert werden.

Die Strahlen müssen also das Material durchdringen können. Abhängig von der Dichte des Materials entlang des Strahls wird er mehr oder weniger abgeschwächt. Ein Strahl und ein mögliches Testobjekt sind in Abb. 2.1, links dargestellt. Offensichtlich braucht man eine große Anzahl von Strahlen, um eine Aussage über die Verteilung des Materials im Inneren des Objektes treffen zu können.

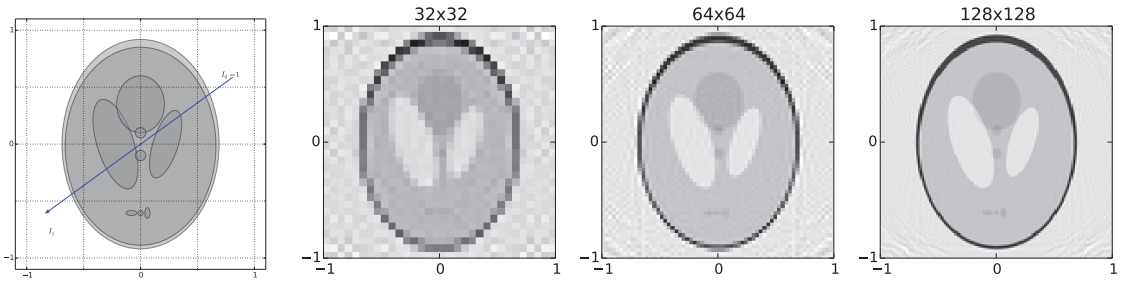


Abbildung 2.1.: *Ein Objekt und mögliche Rekonstruktionen durch Computertomographie. Links ist die Dichteverteilung des Objekts dargestellt. Ein Strahl (blau) der das Objekt durchdringt, wird abhängig von der Dichte entlang des Strahls abgeschwächt. Die Information über die Abschwächung vieler Strahlen erlaubt die diskrete Approximation der Verteilung, hier durch stückweise konstante Rekonstruktion auf einem regulären Gitter.*

Eine beliebige kontinuierliche Dichteverteilung ließe sich offenbar mit einer endlichen Zahl von Messwerten grundsätzlich nicht rekonstruieren. Wir müssen also die Dichteverteilung geeignet einschränken. Wir tun das, indem wir ein reguläres Gitter über das Gebiet legen, und annehmen, in jedem Gitterkästchen wäre die Dichte konstant. Dann lässt sich die Dichteverteilung durch einen Vektor von reellen Zahlen beschreiben, eine Zahl pro Gitterelement. Da uns jeder Strahl einen reellen Wert liefert, mag es möglich sein, mit n Strahlen die Dichtewerte von n Gitterelementen zu berechnen. Im Folgenden werden wir sehen, dass dies möglich sein kann, aber nicht notwendigerweise sein muss.

Dazu betrachten wir zunächst einen einzelnen Strahl den wir mit der Ausgangsintensität a erzeugen und der nach Austritt aus dem Objekt nur noch die gemessene Intensität g hat. Wir überlegen uns jetzt was passiert, wenn der Strahl ein einzelnes Gitterelement mit der Gitterlänge 1 entlang einer der Gitterachsen durchläuft. Wir beobachten, dass die Abschwächung mit einem konstanten Faktor c modelliert werden kann, also $g = c \cdot a$. Eine solche Beobachtung könnte man machen, indem man verschiedene Intensitäten a wählt und dann feststellt, dass die Abschwächung nicht von der Wahl von a abhängt.

Was passiert, wenn der Strahl zwei Elemente durchläuft? Offenbar wird der nach dem ersten Kästchen auf ca abgeschwächte Strahl ein weiteres mal um den Faktor c abgeschwächt, also $g = c^2a$. Wie stark ist die Abschwächung in einem „halben“ Kästchen? Diese muss offenbar $c^{1/2}$ sein, denn nur dann erhalten wir $c^{1/2} \cdot c^{1/2} = c$ wie erwartet. Wir sehen also, dass die Abschwächung von der Länge des Weges l im Material abhängt. Die Weglänge geht dabei als Exponent ein, also $g = c^l a$.

Für jeden erzeugten Strahl kennen wir die *Geometrie*. Wir wissen also, welche Gitterelemente er schneidet und welchen Weg er in den geschnittenen Gitterelementen zurücklegt. Wir indizieren Strahlen und Gitterelemente und nehmen dabei an, die Anzahl sowohl der Strahlen wie auch der Gitterelemente sei n. Dann bezeichnet $l_{i,j}$ die Länge des Weges, die der Strahl mit dem Index i im Gitterelement mit Index j zurücklegt. Schneidet der i-te Strahl das Gitterelement j gar nicht, so ist die Länge des Weges 0. Da die Strahlen immer nur einen kleinen Teilmenge aller Gitterelemente schneiden erwarten wir, dass die meisten Längen $l_{i,j}$ Null sind. Wichtig ist, dass wir die Längen aus der Anordnung der Strahlen (die wir selbst bestimmen) und der Wahl des Gitters (das wir selbst festlegen) unabhängig vom Objekt und der Messung von Strahlen bereits vorab ausrechnen können. Dafür müssen wir „nur“ die Strahlen gegen die Gitterlinien schneiden und dann die Abstände der Schnittpunkte bestimmen.

Die unbekannten Variablen hingegen sind die Dichten im Inneren des Objektes, die wir als konstant für jedes Gitterelement angenommen haben. Wir schreiben c_j für die konstante Dichte des Elementes mit dem Index j. Jetzt können wir die gesamte Abschwächung des Strahls mit dem Index i formulieren:

$$g_i = c_0^{l_{i,0}} \cdot c_1^{I_{i,1}} \cdots c_{n-1}^{l_{i,n-1}} \cdot a. \tag{2.1}$$

Diese Gleichungen lassen keine einfache Lösung der Unbekannten c_j zu. Wir machen das Problem zugänglicher, indem wir (aus dem Anwendungsszenario motiviert) annehmen, alle Größen seien positiv, so dass wir auf beiden Seiten die Logarithmusfunktion anwenden können.

$$\log \frac{g_i}{a} = l_{i,0} \log c_0 + l_{i,1} \log c_1 + \cdots + l_{i,n-1} \log c_{n-1}. \tag{2.2}$$

Nun verwenden wir die Variablenbezeichnungen $g'_i = \log g_i/a$ und $c'_j = \log c_j$. Dann ergeben sich n lineare Gleichungen in den n Unbekannten c'_j, die wir in Matrixnotation schreiben können:

$$\begin{pmatrix} l_{0,0} & l_{0,1} & \dots & l_{0,n-1} \\ l_{1,0} & l_{1,1} & \dots & l_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ l_{n-1,0} & l_{n-1,1} & \dots & l_{n-1,n-1} \end{pmatrix} \begin{pmatrix} c'_0 \\ c'_1 \\ \vdots \\ c'_{n-1} \end{pmatrix} = \begin{pmatrix} g'_0 \\ g'_1 \\ \vdots \\ g'_{n-1}. \end{pmatrix} \tag{2.3}$$

Diese *linearen* Gleichungen können wir auch kürzer als $\mathbf{L}\mathbf{c}' = \mathbf{g}'$ schreiben, wobei $\mathbf{L} \in \mathbb{R}^{n \times n}$ die Matrix der Längen der Strahlen in den Gitterlementen ist, der Vektor $\mathbf{g}' \in \mathbb{R}^n$ die Logarithmen der Verhältnisse von Ausgangsintensität zu gemessener Intensität der Strahlen enthält, und $\mathbf{c}'$ der Vektor der die Logarithmen der gesuchten Dichten in jedem Gitterelement beschreibt. Offenbar ergibt sich $\mathbf{L}$ aus dem Aufbau unserer Messung und $\mathbf{g}'$ enthält die Messergebnisse. Die gesuchten Dichten können wir einfach mit Hilfe der Exponentialfunktion ausrechnen, wenn wir einen Vektor $\mathbf{c}'$ finden, der die Matrix-Vektor-Gleichung erfüllt.

2.2. Definition

Unter einem *linearen Gleichungssystem* (LGS) versteht man eine Menge von m linearen Gleichungen in n gemeinsamen Variablen. Es ist üblich, die Variablen mit den dazugehörigen Koeffizienten auf die linke Seite und die Konstanten auf die rechte Seite der Gleichungen zu schreiben:

$$\begin{aligned} a_{0,0}\,x_0 + a_{0,1}\,x_1 + \ldots + a_{0,n-1}\,x_{n-1} &= b_0 \\ a_{1,0}\,x_0 + a_{1,1}\,x_1 + \ldots + a_{0,n-1}\,x_{n-1} &= b_1 \\ &\vdots \\ a_{m-1,0}\,x_0 + a_{m-1,1}\,x_1 + \ldots + a_{m-1,n-1}\,x_{n-1} &= b_{m-1}. \end{aligned} \tag{2.4}$$

Viele technische Probleme werden als LGS modelliert. Schon bei wenigen Variablen gibt es meistens keine einfach zu „sehende“ Lösung mehr und man muss numerische Methoden verwenden. Das Gebiet der *numerischen linearen Algebra* ist inzwischen sehr groß geworden und wir können hier nur in Stichpunkten die Grundlagen besprechen.

2.3. Matrix-Darstellung eines linearen Gleichungssystems

Lineare Gleichungssysteme lassen sich mit Matrizen und Vektoren darstellen. Dazu werden die Koeffizienten a_{ij} des Gleichungssystems zu einer Matrix

$$\mathbf{A} = \begin{pmatrix} a_{0,0} & a_{0,1} & \ldots & a_{0,n-1} \\ a_{1,0} & a_{1,1} & \ldots & a_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m-1,0} & a_{m-1,1} & \ldots & a_{m-1,n-1} \end{pmatrix}, \tag{2.5}$$

und die Unbekannten x_j und rechten Seiten b_i zu Vektoren

$$\mathbf{x} = (x_0, x_, \ldots, x_{n-1})^\mathsf{T} \quad \mathbf{b} = (b_0, b_1, \ldots, b_{m-1})^\mathsf{T} \tag{2.6}$$

zusammengefasst. Das System von Gleichungen (2.4) lässt sich dann als $\mathbf{Ax} = \mathbf{b}$ schreiben:

$$\begin{aligned} &\underbrace{\begin{pmatrix} a_{0,0} & a_{0,1} & \ldots & a_{0,n-1} \\ a_{1,0} & a_{1,1} & \ldots & a_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m-1,0} & a_{m-1,1} & \ldots & a_{m-1,n-1} \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_{n-1} \end{pmatrix}}_{} = \begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_{m-1} \end{pmatrix}. \\ &= \begin{pmatrix} a_{0,0}\,x_0 + a_{0,1}\,x_1 + \ldots + a_{0,n-1}\,x_{n-1} \\ a_{1,0}\,x_0 + a_{1,1}\,x_1 + \ldots + a_{1,n-1}\,x_{n-1} \\ \vdots \\ a_{m-1,0}\,x_0 + a_{m-1,1}\,x_1 + \ldots + a_{m-1,n-1}\,x_{n-1} \end{pmatrix} \end{aligned} \tag{2.7}$$

Durch Ausmultiplizieren der Matrix-Vektor Gleichung kann man sehen, dass diese zum ursprünglichen Gleichungssystem äquivalent ist.

Wir werden ein LGS im Folgenden auch wie folgt darstellen:

$$\left(\begin{array}{cccc|c} a_{0,0} & a_{0,1} & \ldots & a_{0,n-1} & b_0 \\ a_{1,0} & a_{1,1} & \ldots & a_{1,n-1} & b_1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{m-1,0} & a_{m-1,1} & \ldots & a_{m-1,n-1} & b_{m-1} \end{array}\right) \tag{2.8}$$

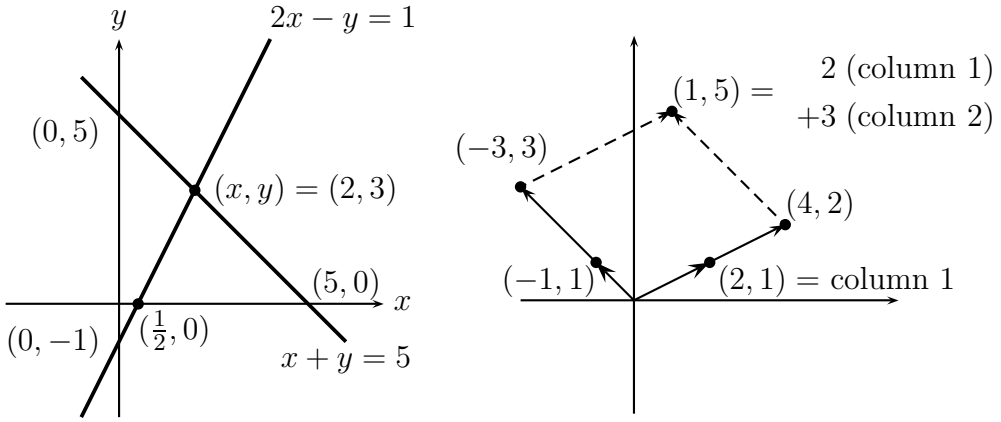


Abbildung 2.2.: *Zeilenweise (links) und spaltenweise (rechts) Interpretation des Gleichungssystem in Gl. 2.9.*

2.4. Geometrische Interpretation von Gleichungssystemen

Lineare Gleichungssysteme haben zwei geometrische Interpretationen: entlang der Zeilen und entlang der Spalten. Um diese zu erläutern, betrachten wir ein einfaches LGS mit 2 Unbekannten:[1]

$$2x - y = 1 \tag{2.9a}$$
$$x + y = 5. \tag{2.9b}$$

Zeilen: lineare Gleichungen als Hyperebenen Wenn wir ein LGS mit n Unbekannten zeilenweise interpretieren, dann entspricht jede Zeile einer $(n-1)$-dimensionalen Hyperebene in $\mathbb{R}^n$. Die Lösung der Gleichungen ist der Schnittpunkt der Ebenen. Für das System in Gl. 2.9 mit zwei Unbekannten entsprechen die Zeilen also 1-dimensionalen Hyperebenen im $\mathbb{R}^2$, d.h. Geraden. Durch Auflösen nach y erhalten wir explizite Ausdrücke für die Geraden:

$$y_0 = 2x - 1 \tag{2.10a}$$
$$y_1 = -x + 5. \tag{2.10b}$$

Diese Geraden sind graphisch in Abb. 2.2 (links) dargestellt.

Spalten: Basisvektoren Wenn wir ein LGS spaltenweise interpretieren, dann formen die Spalten der Matrix eine Basis, in der der Vektor auf der rechten Seite ausgedrückt werden soll. Dies geschieht durch die Wahl von (unbekannten) Gewichten, ein Gewicht für jeden der Spaltenvektoren. Gl. 2.9 können wir also wie folgt interpretieren:

$$x \begin{pmatrix} 2 \\ 1 \end{pmatrix} + y \begin{pmatrix} -1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 5 \end{pmatrix} \tag{2.11}$$

Dies ist graphisch in Abb. 2.2 (rechts) dargestellt.

[1] Das Beispiel ist aus G. Strang, *Linear Algebra and Its Applications*. Thomson, Brooks/Cole, 2006.

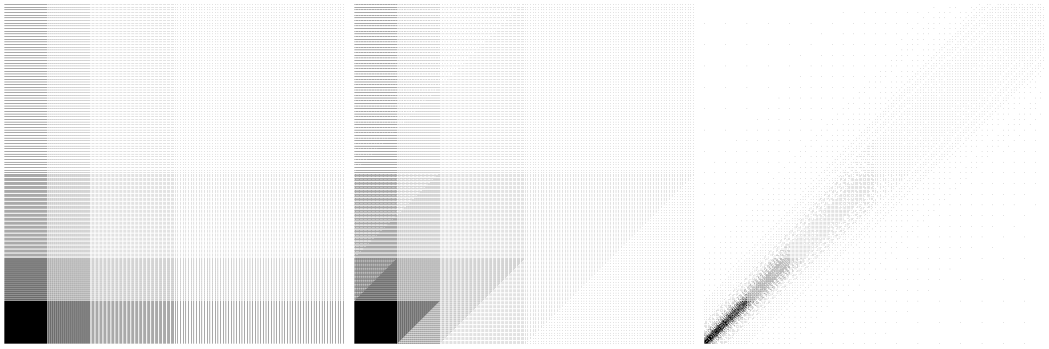


Abbildung 2.3.: *Elemente der Ebene, die sich in Gleitkommazahlen darstellen lassen (links). Produkte einer Matrix mit einem Vektor von Gleitkommazahlen, also die Elemente, die sich in der Basis der Spaltenvektoren in Gleitkommazahlen darstellen lassen, sind nur eine Teilmenge (mitte, rechts). Die Teilmenge hängt von der Basis ab, hier* $(1,0)^\mathsf{T}, (1,1)^\mathsf{T}$ *(mitte) und* $(1,15/16)^\mathsf{T}, (15/16,1)^\mathsf{T}$ *(rechts).*

Spaltenraum in Gleitkommazahlen Die Interpretation des LGS als Darstellung der rechten Seite im Raum, der durch die Spalten der Matrix aufgespannt wird, hilft auch die Probleme im Zusammenhang mit Gleitkommazahlen zu verstehen. Dazu betrachten wir wieder ein Beispiel im $\mathbb{R}^2$. Die Vektoren $\mathbf{b} \in \mathbb{G}^2$ stellen alle Zahlenpaare dar, die sich mit Gleitkommazahlen repräsentieren lassen. Sie sind für eine Wahl von $\mathbb{G}$ in Abb. 2.3 (links) gezeigt.

Betrachten wir nun das Ergebnis des Produktes $\mathbf{Ax}$ für eine feste Matrix $\mathbf{A} \in \mathbb{G}^{2\times 2}$ und alle Vektoren $\mathbf{x} \in \mathbb{G}^2$ dann erhalten wir nicht alle möglichen Paare von Gleitkommazahlen. Warum nicht? Alle Rechenoperationen müssen jeweils wieder auf eine Gleitkommazahl abgebildet werden. Dabei fallen die Produkte $\mathbf{Ax}$ und $\mathbf{Ax}'$ für verschiedene Vektoren $\mathbf{x} \neq \mathbf{x}'$ durch Rundung auf den nächsten verfügbaren Vektor in $\mathbb{G}^2$ möglicherweise zusammen, also $\mathbf{Ax} = \mathbf{Ax}'$. Da die Menge der Gleitkommazahlen aber endlich ist, gehen mit dem Zusammenfallen von Werten unweigerlich andere mögliche Ergebnisse $\mathbf{b} \in \mathbb{G}^2$ verloren. Das heißt, es gibt Vektoren $\mathbf{b} \in \mathbb{G}^2$ für die es kein $\mathbf{x} \in \mathbb{G}^2$ gibt, so dass $\mathbf{Ax} = \mathbf{b}$.

Der Verlust ist dabei abhängig von $\mathbf{A}$ bzw. den Spaltenvektoren. Bilden die Spaltenvektoren rechte Winkel (oder fast rechte Winkel) und haben ähnliche Längen so ist der Effekt wenig ausgeprägt (siehe Abb. 2.3 (mitte)). Je kleiner der Winkel zwischen zwei Vektoren oder je unterschiedlicher die Längen, desto mehr Linearkombinationen der Basisvektoren fallen auf den gleichen Punkt in $\mathbb{G}^2$ und desto mehr Punkte der Menge $\mathbb{G}^2$ fehlen. (Abb. 2.3 (rechts)).

Es ist wichtig einzusehen, dass der Verlust nicht mehr „rückgängig“ gemacht werden kann. Jeder Wechsel in eine andere Basis (also jedes Produkt mit einer Matrix) bedeutet potentiellen zusätzlichen Verlust von verfügbaren Gleitkommazahlen. Insbesondere würde die Multiplikation mit der Inversen $\mathbf{A}^{-1}$ nichts „retten“, sondern vielmehr die Situation ein weiteres Mal verschlechtern.

2.5. Kondition

Der zuletzt besprochene Effekt, nämlich der Verlust von eigentlich verfügbaren Punkten im Vektorraum der Gleitkommazahlen, gilt selbstverständlich auch in höheren Dimensionen. Wie können wir den Effekt ganz allgemein messen? Da wir für ein gegebenes LGS keine Lösung erwarten können betrachten wir das *Residuum* $\mathbf{r} = \mathbf{Ax} - \mathbf{b}$. Um den Effekt zu studieren, der durch die Einschränkung auf Gleitkommazahlen entsteht, nehmen wir an, dass das LGS für

gegebene $\mathbf{A} \in \mathbb{G}^{n\times n}, \mathbf{b} \in \mathbb{G}^n$ eine eindeutige Lösung $\mathbf{x}^* \in \mathbb{R}^n$ im Vektorraum über den reellen Zahlen hat (sprich, wir nehmen an die Matrix $\mathbf{A}$ hat eine nicht verschwindende Determinante, wobei wir die Determinante in den reellen Zahlen ausrechnen dürfen). Es gilt also $\mathbf{A}\mathbf{x}^* = \mathbf{b}$.

In den Gleitkommazahlen wird man $\mathbf{x} \in \mathbb{G}^n$ stets so wählen wollen, dass die Norm des Residuums $\|\mathbf{r}\|$ möglichst klein ist. Wir nehmen an, dass $\mathbf{x}$ entsprechend gewählt worden ist und wollen den maximalen relativen Fehler des Residuums abschätzen. Der relative Fehler ist in diesem Fall also der absolute Fehler des Residuums geteilt durch den gewünschten Wert $\mathbf{b}$, also

$$\frac{\|\mathbf{r}\|}{\|\mathbf{b}\|} = \frac{\|\mathbf{A}\mathbf{x} - \mathbf{b}\|}{\|\mathbf{b}\|} = \frac{\|\mathbf{A}\mathbf{x} - \mathbf{A}\mathbf{x}^*\|}{\|\mathbf{A}\mathbf{x}^*\|} = \frac{\|\mathbf{A}(\mathbf{x} - \mathbf{x}^*)\|}{\|\mathbf{A}\mathbf{x}^*\|}. \tag{2.12}$$

Für welche Vektoren wird dieser Bruch besonders groß? Wir gelangen zu einer Abschätzung wenn wir das Problem getrennt für Zähler und Nenner betrachten. Damit der Bruch groß wird, sollte der Zähler möglichst groß und der Nenner möglichst klein gewählt werden.

Wir betrachten die Norm des Matrix-Vektor-Produktes genauer. Es gibt offenbar für jeden Vektor $\mathbf{y}$ eine Zahl $c_\mathbf{y}$, so dass $\|\mathbf{A}\mathbf{y}\| = c_\mathbf{y}\|\mathbf{y}\|$ gilt. Die Zahl $c_\mathbf{y}$ ist unabhängig von der Länge $\|\mathbf{y}\|$ des Vektors – sie ist also eine Eigenschaft der Matrix $\mathbf{A}$, die für alle Vektoren der Form $\lambda\mathbf{y}, \lambda \neq 0$ gilt. Um sie für die Vektoren der Form $\lambda\mathbf{y}$ zu bestimmen, genügt es einen Vektor zu verwenden. Es ist üblich hierfür einen Einheitsvektor zu nehmen: $\|\mathbf{y}\| = 1$ womit man direkt

$$c_\mathbf{y} = \|\mathbf{A}\mathbf{y}\|, \qquad \|\mathbf{y}\| = 1 \tag{2.13}$$

erhält.

Um nun eine obere Schranke für den Zähler zu finden, wählen wir das größtmögliche $c_\mathbf{y}$:

$$\|\mathbf{A}(\mathbf{x} - \mathbf{x}^*)\| \leq \max_{\|\mathbf{y}\|=1} \|\mathbf{A}\mathbf{y}\| \cdot \|\mathbf{x} - \mathbf{x}^*\|. \tag{2.14}$$

Für den kleinsten möglichen Nenner gilt analog:

$$\|\mathbf{A}\mathbf{x}^*\| \leq \min_{\|\mathbf{y}\|=1} \|\mathbf{A}\mathbf{y}\| \cdot \|\mathbf{x}^*\|. \tag{2.15}$$

Durch diese obere Schranke auf den Zähler und untere Schranke für den Nenner erhalten wir eine (pessimistische) obere Schranke für den relativen Fehler $\|\mathbf{r}\|/\|\mathbf{b}\|$, nämlich

$$\frac{\|\mathbf{A}(\mathbf{x} - \mathbf{x}^*)\|}{\|\mathbf{A}\mathbf{x}^*\|} \leq \frac{\max_{\|\mathbf{y}\|=1} \|\mathbf{A}\mathbf{y}\|}{\min_{\|\mathbf{y}\|=1} \|\mathbf{A}\mathbf{y}\|} \cdot \frac{\|\mathbf{x} - \mathbf{x}^*\|}{\|\mathbf{x}^*\|} \approx \frac{\max_{\|\mathbf{y}\|=1} \|\mathbf{A}\mathbf{y}\|}{\min_{\|\mathbf{y}\|=1} \|\mathbf{A}\mathbf{y}\|} \cdot \epsilon. \tag{2.16}$$

In der letzten Abschätzung haben wir ausgenutzt, dass die Maschinengenauigkeit ϵ definiert war als der maximale relative Fehler, der bei der Abbildung einer reellen Zahl auf eine Gleitkommazahl entstehen kann – hier besteht $\mathbf{x}^*$ aus reellen Einträgen und $\mathbf{x}$ aus Gleitkommazahlen.

Da man den Verlust durch Maschinengenauigkeit ohnehin nicht verhindern kann, repräsentiert der Bruch

$$\kappa(\mathbf{A}) = \frac{\max_{\|\mathbf{y}\|=1} \|\mathbf{A}\mathbf{y}\|}{\min_{\|\mathbf{y}\|=1} \|\mathbf{A}\mathbf{y}\|} \tag{2.17}$$

genau den zusätzlichen Verlust durch Multiplikation mit der Matrix $\mathbf{A}$. Die Zahl $\kappa(\mathbf{A})$ wird *Kondition* der Matrix $\mathbf{A}$ genannt. Da die Vektoren $\|\mathbf{y}\| = 1$ eine Kugel bilden, kann man die Kondition geometrisch als die Verzerrung einer Kugel interpretieren, oder intuitiv als das Verhältnis von stärkster Verlängerung zu stäkster Verkürzung eines Vektors beschreiben.

Aus der Definition der Kondition folgt direkt, dass die Inverse einer Matrix $\mathbf{A}^{-1}$ die gleiche Konditionszahl wie die Matrix $\mathbf{A}$ hat, also

$$\kappa\left(\mathbf{A}^{-1}\right) = \kappa(\mathbf{A}). \tag{2.18}$$

Es ist wichtig sich zu erinnern, dass diese Effekte in Gleitkommazahlen auftreten können, aber nicht müssen. Für geeignete Zahlen lassen sich auch schlecht konditionierte LGS exakt in Gleitkommazahlen lösen. Die Kondition beschreibt vielmehr, was bei einem Wechsel in eine andere Basis (also dem Produkt mit der entsprechenden Matrix) passieren *könnte*. Auch die Effekte bei mehreren Basiswechseln können sich unter günstigen Umständen auslöschen, aber grundsätzlich muss man davon ausgehen, dass jeder zusätzliche Basiswechsel die Situation verschlechtert. Zum Beispiel ist die Multiplikation mit $\mathbf{A}$ und dann $\mathbf{A}^{-1}$ im Allgemeinen mit erheblichem Verlust an verfügbarer Gleitkommagenauigkeit verbunden, wenn die Kondition von $\mathbf{A}$ schlecht ist. Die Verluste heben sich hierbei normalerweise nicht auf.

Bei der Konstruktion von Algorithmen versucht man deswegen, wann immer man eine Wahl hat, nach Möglichkeit Matrizen mit einer kleinen Konditionszahl zu verwenden; denn dann ist sichergestellt, dass nur die ohnehin unvermeidlichen Fehler durch die Maschinengenauigkeit auftreten. Die kleinstmögliche Kondition ist 1 und wird von allen Matrizen erreicht, die alle Längen um den gleichen Faktor skalieren. Dies sind die orthogonalen Matrizen multipliziert mit einem beliebigen Skalar.

2.6. Lineare Gleichungssysteme in Diagonalform

Lineare Gleichungssysteme mit quadratischer Diagonalmatrix

$$\begin{pmatrix} a_{0,0} & 0 & \dots & 0 \\ 0 & a_{1,1} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & a_{n-1,n-1} \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_{n-1} \end{pmatrix} = \begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_{n-1} \end{pmatrix} \tag{2.19}$$

lassen sich direkt lösen. Die n Gleichungen haben keine gemeinsamen Variablen, sprich die Variable x_i hängt nur von der i-ten Zeile des LGS, also $a_{i,i}$, ab:

$$x_i = b_i / a_{i,i}. \tag{2.20}$$

Im Falle $a_{i,i} = 0$ haben wir

$$0\, x_i = b_i. \tag{2.21}$$

Wenn $b_i = 0$ dann kann die Unbekannte x_i beliebige Werte annehmen, da $0\, x_i = 0$ für jedes $x_i \in \mathbb{R}$ gilt. Im Falle $b_i \neq 0$ hat das LGS keine Lösung.

2.7. Lineare Gleichungssysteme in Dreiecksform

Ein (quadratisches) LGS ist in *unterer Dreicksform*, wenn es folgende Form hat:

$$\begin{pmatrix} a_{0,0} & 0 & \dots & 0 \\ a_{1,0} & a_{1,1} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n-1,0} & a_{n-1,1} & \dots & a_{n-1,n-1} \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_{n-1} \end{pmatrix} = \begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_{n-1} \end{pmatrix} \tag{2.22}$$

Diese Systeme sind von besonderer Bedeutung, da eine Lösung besonders effizient berechnet werden kann. Betrachten wir die erste Zeile so ergibt sich die Lösung für x_0 direkt:

$$x_0 = \frac{b_0}{a_{0,0}}. \tag{2.23}$$

Nach der Berechnung von x_0 ist in der zweiten Zeile nur noch x_1 unbekannt und wir können wieder direkt für die Variable lösen:

$$x_1 = \frac{b_1 - a_{1,0}x_0}{a_{1,1}} \tag{2.24}$$

Dies kann Zeile für Zeile fortgesetzt werden bis man in der letzten Zeile angekommen ist:

$$x_{n-1} = \frac{b_{n-1} - a_{n-1,0}x_0 - \ldots - a_{n-1,n-2}x_{n-2}}{a_{n-1,n-1}}. \tag{2.25}$$

Die Lösung wird also von oben nach unten schrittweise berechnet und man spricht daher von *Vorwärtseinsetzen*. Die allgemeine Rechenvorschrift für das Verfahren lautet:

$$x_i = \frac{b_i - \sum_{j=0}^{i-1} a_{i,j}x_j}{a_{i,i}} \qquad (i = 0, \ldots, n-1) \tag{2.26}$$

Im Falle einer *oberen* Dreiecksmatrix

$$\begin{pmatrix} a_{0,0} & a_{0,1} & \ldots & a_{0,n-1} \\ 0 & a_{1,1} & \ldots & a_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & a_{n-1,n-1} \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_{n-1} \end{pmatrix} = \begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_{n-1} \end{pmatrix} \tag{2.27}$$

geht man analog von unten nach oben vor. Man spricht daher von *Rückwärtseinsetzen*:

$$\begin{aligned} x_{n-1} &= \frac{b_{n-1}}{a_{n-1,n-1}} \\ x_{n-2} &= \frac{b_{n-2} - a_{n-2,n-1}x_{n-1}}{a_{n-2,n-2}} \\ &\vdots \\ x_0 &= \frac{b_0 - a_{0,1}x_1 - \ldots - a_{0,n-1}x_{n-1}}{a_{0,0}}, \end{aligned} \tag{2.28}$$

oder allgemein:

$$x_i = \frac{b_i - \sum_{j=i+1}^{n-1} a_{i,j}x_j}{a_{i,i}} \qquad (i = n-1, \ldots, 0). \tag{2.29}$$

2.8. Beispiel: Schnitt von Gerade und Ebene

Wir wollen den Schnittpunkt einer Ebene und einer Gerade im Raum bestimmen. Dazu nehmen wir an, dass die Ebene E und die Gerade G in Parameterform gegeben sind:

$$E : x = \begin{pmatrix} -3 \\ 1 \\ 1 \end{pmatrix} + \alpha \begin{pmatrix} 1 \\ -1 \\ -1 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ -1 \\ 2 \end{pmatrix}, \quad \text{mit } \alpha, \beta \in \mathbb{R} \tag{2.30}$$

$$G : x = \begin{pmatrix} 2 \\ -3 \\ 2 \end{pmatrix} + \mu \begin{pmatrix} 1 \\ -2 \\ 3 \end{pmatrix}, \quad \text{mit } \mu \in \mathbb{R} \tag{2.31}$$

Ein Schnittpunkt von Gerade und Ebene ist ein Punkt, welcher sowohl die Ebenengleichung als auch die Geradengleichung erfüllt. Durch Gleichsetzen der beiden Gleichungen erhalten wir ein Gleichungssystem in den Unbekannten α, β und μ. Dieses ist linear, da die Unbekannten nur als gewichtete Summanden auftreten:

$$-3 + \alpha = 2 + \mu \tag{2.32a}$$

$$1 - \alpha - \beta = -3 - 2\mu \tag{2.32b}$$

$$1 - \alpha + 2\beta = 2 + 3\mu \tag{2.32c}$$

Durch das Addieren von Vielfachen dieser Gleichungen bzw. das Austauschen von Gleichungen ändert sich die Lösungsmenge des Gleichungssystems nicht. Eine Lösungsstrategie besteht nun darin, durch diese Operation drei neue Gleichungen zu erhalten, die eine Dreiecksstruktur haben: Die dritte Gleichung hängt nur von einer Variable (z.B. μ) ab, die zweite Gleichung von zwei Variable (z.B. μ und α) usw. Die Formalisierung dieses Verfahrens heißt *Gaußsche Eliminationsmethode*. Wir haben bereits in Abschnitt (2.7) gesehen, dass sich ein System mit oberer Dreiecksstruktur effizient durch Rückwärtseinsetzen lösen lässt. Die Gaußsche Eliminationsmethode zusammen mit Rückwärtseinsetzen ergibt also einen Algorithmus zur Lösung linearer Gleichungssysteme.

Die Darstellung von Gleichungssystemen in Matrixform ermöglicht eine übersichtliche Durchführung der Gauß-Elimination, welche wir in Abschnitt 2.9 formell einführen werden. Addieren von Gleichungen entspricht dann dem Addieren von Zeilen der Matrix und von Elementen des Vektors auf der rechten Seite. Darüber hinaus, und für unsere Zwecke natürlich noch viel wichtiger, ist die Matrixdarstellung die Voraussetzung für eine effiziente Umsetzung des Verfahrens auf dem Computer. Gl. 2.32 hat dann die Form:

$$\begin{pmatrix} 1 & 0 & -1 \\ -1 & -1 & 2 \\ -1 & 2 & -3 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \\ \mu \end{pmatrix} = \begin{pmatrix} 5 \\ -4 \\ 1 \end{pmatrix} \tag{2.33}$$

und noch kompakter lässt sich das System wie folgt darstellen:

$$\left(\begin{array}{ccc|c} 1 & 0 & -1 & 5 \\ -1 & -1 & 2 & -4 \\ -1 & 2 & -3 & 1 \end{array}\right) \tag{2.34}$$

Um unser Beispiel in Dreiecksform zu überführen, müssen wir alle Elemente in der erste Spalte, bis auf das erste, eliminieren. Wenn wir also die erste mit der zweiten Zeile addieren und die erste mit der dritten, so erhalten wir das folgende, äquivalente Gleichungssystem:

$$\left(\begin{array}{ccc|c} 1 & 0 & -1 & 5 \\ 0 & -1 & 1 & 1 \\ 0 & 2 & -4 & 6 \end{array}\right) \tag{2.35}$$

wobei wir auch die rechte Seite transformiert haben. Das Ziel der Dreiecksform ist schon fast erreicht. Das Element A_{32} der Matrix kann Null gesetzt werden, in dem wir das doppelte der zweiten Zeile mit der dritten addieren. Dann erhalten wir:

$$\left(\begin{array}{ccc|c} 1 & 0 & -1 & 5 \\ 0 & -1 & 1 & 1 \\ 0 & 0 & -2 & 8 \end{array}\right) \tag{2.36}$$

Das umgeformte System lässt sich jetzt einfach durch Rückwärtseinsetzen lösen. Aus der letzten Zeile entnehmen wir das $-2\mu = 8$ und damit $\mu = -4$. Einsetzen dieses Wertes in der zweiten Zeile liefert $-\beta - 4 = 1$, so dass $\beta = -5$. Analog erhalten wir $\alpha = 1$. Den Schnittpunkt erhält man nun indem man entweder α und β in die Ebenengleichung oder μ in die Geradengleichung einsetzt. Explizit erhalten wir also für den Schnittpunkt mithilfe der Ebenengleichung:

$$x = \begin{pmatrix} -3 \\ 1 \\ 1 \end{pmatrix} + \alpha \begin{pmatrix} 1 \\ -1 \\ -1 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ -1 \\ 2 \end{pmatrix} = \begin{pmatrix} -3 \\ 1 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ -1 \\ -1 \end{pmatrix} - 5 \begin{pmatrix} 0 \\ -1 \\ 2 \end{pmatrix} = \begin{pmatrix} -2 \\ 5 \\ -10 \end{pmatrix} \tag{2.37a}$$

und mit der Geradengleichung erhalten wir:

$$x = \begin{pmatrix} 2 \\ -3 \\ 2 \end{pmatrix} + \mu \begin{pmatrix} 1 \\ -2 \\ 3 \end{pmatrix} = \begin{pmatrix} 2 \\ -3 \\ 2 \end{pmatrix} - 4 \begin{pmatrix} 1 \\ -2 \\ 3 \end{pmatrix} = \begin{pmatrix} -2 \\ 5 \\ -10 \end{pmatrix}. \tag{2.37b}$$

In jedem Schritt des Algorithmus haben wir zunächst ein Diagonalelement, das sogenannte Pivotelement, gewählt und mit ihm die Spalte darunter „eliminiert“. Dies kann allerdings nicht funktionieren, wenn das Pivotelement Null ist. In diesem Fall tauscht man einfach die Zeile des Pivotelements mit einer Zeile weiter unten in der Matrix, sodass auf der Diagonale ein Element ungleich Null steht und fährt wie gewohnt fort.

Es kann passieren, dass am Ende des Verfahrens ein System der Form

$$\left(\begin{array}{ccc|c} x_{0,0} & x_{0,1} & x_{0,2} & a \\ 0 & x_{1,1} & x_{1,2} & b \\ 0 & 0 & 0 & c \end{array}\right) \quad , \quad c \neq 0 \tag{2.38a}$$

oder der Form

$$\left(\begin{array}{ccc|c} x_{0,0} & x_{0,1} & x_{0,2} & a \\ 0 & x_{1,1} & x_{1,2} & b \\ 0 & 0 & 0 & 0 \end{array}\right) \tag{2.38b}$$

entsteht. Dies entspricht den degenerierten Konfigurationen von Ebene und Gerade: die Gerade ist parallel zur Ebene in einer Distanz ungleich Null, so dass es keinen Schnittpunkt gibt, oder in einer Distanz Null, so dass jeder Punkt auf der Gerade eine Lösung ist. Gleichung 2.38a

entspricht dem ersten Fall, dass heißt es gibt keine Lösung. Dies folgt algebraisch, da die Gleichung einen Widerspruch enthält, was auch für allgemeine Gleichungssysteme anzeigt, dass es keine Lösung gibt. Gleichung 2.38b entspricht unendlich vielen Lösungen. Dies hat zur Folge, dass man für die letzte Variable einen beliebigen Wert angeben kann. Für jeden dieser Werte erhält man einen Lösungsvektor. Enthält die Matrix k dieser Null-Zeilen, so erhält man einen $k-$dimensionalen Lösungsraum.

2.9. Gaußsche Eliminationsmethode und Pivoting

2.9.1. Gaußsche Eliminationsmethode

Sei $\mathbf{Ax} = \mathbf{b}$ ein lineares Gleichungssystem. Bei der Gaußschen Eliminationsmethode wird die Matrix $\mathbf{A}$ schrittweise auf eine obere Dreiecksmatrix $\mathbf{R}$ reduziert. Anschließend erhält man die Lösung des Gleichungssystems mit Rückwärtseinsetzen. Bezeichnet $\mathbf{A}^{(0)} = A$ und $\mathbf{b}^{(0)} = b$, so erhalten wir in jedem Schritt, $0 < k \leq n-1$, eine Matrix

$$\mathbf{A}^{(k)} = \begin{pmatrix} a_{0,0}^{(0)} & a_{0,1}^{(0)} & \cdots & \cdots & \cdots & a_{0,n-1}^{(0)} \\ & a_{1,1}^{(1)} & \cdots & \cdots & \cdots & a_{1,n-1}^{(1)} \\ & & \ddots & & & \vdots \\ & & & a_{k,k}^{(k)} & \cdots & a_{k,n-1}^{(k)} \\ & & & \vdots & & \vdots \\ & & & a_{n-1,k}^{(k)} & \cdots & a_{n-1,n-1}^{(k)} \end{pmatrix}, \tag{2.39}$$

die in den Spalten bis zum Index k unterhalb der Diagonalen nur Nullen enthält. Auch die rechten Seiten werden dabei transformiert:

$$\mathbf{b}^{(k)} = (b_0^{(k)}, \ldots, b_{n-1}^{(k)})^T. \tag{2.40}$$

Entscheidend ist offenbar der Schritt von k zu $k+1$, der schrittweise in der Spalte mit Index k die gewünschten Nullen unterhalb der Diagonale erzeugt. Dies geschieht wie im vorangegangenen Kapitel illustriert. Jede Zeile $i > k$ wird durch eine Linearkombination der Zeilen k und i ersetzt, so dass der Eintrag (i, k) zu Null wird. Dabei lassen wir die die Zeile i unverändert (d.h. setzt den Faktor für diese Zeile auf 1) und addieren ein Vielfaches der Zeile k. Der notwendige Faktor für die k-te Zeile ist

$$m_{ik} = -a_{i,k}^{(k)} / a_{k,k}^{(k)}, \tag{2.41}$$

denn

$$m_{ik} \cdot a_{k,k}^{(k)} + 1 \cdot a_{i,k}^{(k)} = \frac{-a_{i,k}^{(k)}}{a_{k,k}^{(k)}} a_{k,k}^{(k)} + a_{i,k}^{(k)} = -a_{i,k}^{(k)} + a_{i,k}^{(k)} = 0, \tag{2.42}$$

wie gewünscht.

Als Algorithmus können wir den Schritt von $k \to k+1$ wie folgt beschreiben:

$$m_{ik} = -a_{i,k}^{(k)} \Big/ a_{k,k}^{(k)} \qquad i = k+1, \ldots, n-1 \tag{2.43}$$

$$a_{i,j}^{(k+1)} = a_{i,j}^{(k)} + m_{ik}\, a_{k,j}^{(k)} \qquad i,j = k+1, \ldots, n-1 \tag{2.44}$$

$$b_{i}^{(k+1)} = b_{i}^{(k)} + m_{ik}\, b_{k}^{(k)} \qquad i = k+1, \ldots, n-1. \tag{2.45}$$

Es ist für die weitere Diskussion nützlich zu sehen, dass diese Operationen (wie jede lineare Operation) durch eine Matrixmultiplikation ausgedrückt werden können. Es ergibt sich die folgende Matrix

$$\mathbf{L}^{(k)} = \begin{pmatrix} 1 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & m_{k+1,k} & 1 & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & m_{n-1,k} & 0 & \cdots & 1 \end{pmatrix}. \tag{2.46}$$

mit der sowohl die Matrix $\mathbf{A}^{(k+1)} = \mathbf{L}^{(k)}\mathbf{A}^{(k)}$ wie auch die rechte Seite $\mathbf{b}^{(k+1)} = \mathbf{L}^{(k)}\mathbf{b}^{(k)}$ transformiert werden.

2.9.2. Notwendigkeit von Pivoting

Offensichtlich ist $m_{ik} = -a_{i,k}^{(k)}/a_{k,k}^{(k)}$ nicht definiert, wenn $a_{k,k}^{(k)} = 0$ ist. In diesem Fall muss man die Zeile k mit einer anderen Zeile $i > k$ mit $a_{i,k}^{(k)} \neq 0$ tauschen. Sollte es keine entsprechende Zeile geben, so ist das LGS unterbestimmt und der Wert von x_k kann frei gewählt werden. Wenn man immer eine entsprechende Zeile finden kann, so findet man mit dem Gaußschen Eliminationsverfahren im Prinzip immer ein LGS in Dreiecksform und kann durch Rückwärtseinsetzen die Lösung bestimmen.

Warum nur „im Prinzip“? Wie zuvor erwähnt findet bei jedem Basiswechsel ein potentieller Verlust an verfügbaren Gleitkommazahlen statt. Jeder Schritt des Gaußschen Eliminationsverfahrens ist ein Basiswechsel, der für den k-ten Schritt mit der in Gl. 2.46 eingeführten Matrix $\mathbf{L}^{(k)}$ repräsentiert wird. Zur Beurteilung der potentiellen Verluste durch den Wechsel in das durch $\mathbf{L}^{(k)}$ beschriebene System suchen wir nach der größten und kleinsten Länge eines abgebildeten Einheitsvektors, also $\min / \max_{\|\mathbf{y}\|=1} \|\mathbf{L^{(k)}y}\|$. Wir machen es uns einfach und betrachten als mögliche Einheitsvektoren nur die kanonischen Basisvektoren $\mathbf{e}_j$. Dadurch unterschätzen wir die Kondition eventuell, aber als Strategie eine nicht zu schlecht konditionierte Matrix $\mathbf{L}^{(k)}$ zu finden, dient es uns trotzdem.

Außer dem Basisvektor $\mathbf{e}_k$ bildet $\mathbf{L}^{(k)}$ alle anderen kanonischen Einheitsvektoren $\mathbf{e}_j, j \neq k$ auf sich selbst ab: $\mathbf{L}^{(k)}\mathbf{e_j} = \mathbf{e}_j, j \neq k$. Wir sehen also, dass es Vektoren gibt, die nicht länger werden, und damit ist sicher, dass

$$\min_{\|\mathbf{y}\|=1} \|\mathbf{L^{(k)}y}\| \leq \|\mathbf{L}^{(k)}\mathbf{e}_{j\neq k}\| = \|\mathbf{e}_{j\neq k}\| = 1. \tag{2.47}$$

Damit haben wir, dass jede untere Schranke auf die maximale Länge gleichzeitig eine untere Schranke für die Kondition ist, denn die Kondition beschreibt das Verhältnis der maximalen und minimalen Länge. Für den Basisvektor $\mathbf{e}_k$ erhalten wir

$$\mathbf{L}^{(k)}\mathbf{e_k} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ m_{k+1,k} \\ \vdots \\ m_{n-1,k} \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ -a^{(k)}_{k+1,k}/a^{(k)}_{k,k} \\ \vdots \\ -a^{(k)}_{n-1,k}/a^{(k)}_{k,k} \end{pmatrix}. \tag{2.48}$$

Diesen Vektor können wir als untere Schranke für die maximale Länge eines abgebildeten Einheitsvektors nehmen. Damit haben wir dann die gewünschte untere Schranke für die Kondition, nämlich

$$\begin{aligned} \kappa\left(\mathbf{L}^{(k)}\right) &= \frac{\max_{\|\mathbf{y}\|=1} \|\mathbf{L^{(k)}y}\|}{\min_{\|\mathbf{y}\|=1} \|\mathbf{L^{(k)}y}\|} \\ &\geq \max_{\|\mathbf{y}\|=1} \|\mathbf{L^{(k)}y}\| \\ &\geq \|\mathbf{L}^{(k)}\mathbf{e}_k\| \\ &= \left(1 + \left(\frac{a^{(k)}_{k+1,k}}{a^{(k)}_{k,k}}\right)^2 + \cdots + \left(\frac{a^{(k)}_{n-1,k}}{a^{(k)}_{k,k}}\right)^2\right)^{1/2} \\ &= \frac{\left(\sum_{i=k}^{n-1}\left(a^{(k)}_{i,k}\right)^2\right)^{1/2}}{|a^{(k)}_{k,k}|}. \end{aligned} \tag{2.49}$$

Die Kondition der Matrix kann also nur dann klein sein, wenn $|a^{(k)}_{k,k}|$ groß im Verhältnis zu den Elementen $|a^{(k)}_{i,k}|$ ist. Ist hingegen, $|a^{(k)}_{k,k}|$ sehr klein (im Verhältnis zu den anderen Elementen der gleichen Spalte) so werden die Brüche $|m_{ik}|$ groß und die Kondition der Transformationsmatrix $\mathbf{L}^{(k)}$ schlecht. In diesem Fall droht der Verlust von Genauigkeit in den Gleitkommazahlen. Das dieses Szenario in der Praxis eintreten kann zeigen wir weiter unten an einem einfachen Beispiel.

Da der Tausch von Zeilen keine Verluste bedeutet, wird man also nicht nur für den Fall $a^{(k)}_{k,k} = 0$ Zeilen tauschen, sondern dies grundsätzlich tun, mit dem Versuch eine schlechte Kondition der Matrix $\mathbf{L}^{(k)}$ zu verhindern. Das Element $a^{(k)}_{i,k}$, dass durch Tausch in die Zeile k gebracht wird nennt man *Pivotelement*. Eine einfache Strategie für die Auswahl ist als Pivotelement immer das betragsmäßig größte Element der Spalte k zu wählen. Damit hat man sichergestellt, dass alle Faktoren $|m_{ik}| \leq 1$ sind. Dieses Vorgehen nennt man auch *Spaltenpivotisierung*. Es ist wegen der unbekannten Konsequenzen auf die nachfolgenden Schritte des Verfahrens nur eine Heuristik – allerdings eine die sich in der Praxis sehr bewährt hat.

Beispiel Wir wollen an einem einfachen Beispiel zeigen, dass die Wahl des falschen Pivotelements katastrophale Folgen haben kann, und dass ein Tausch auf Basis der einfachen

Spaltenpivotisierung zu einem vernünftigen Ergebnis führt. Wir nehmen dabei an, dass die Berechnungen im Gleitkommazahl-Format $\mathbb{G}(10,3)$ mit drei Ziffern in der Mantisse erfolgen, d.h.

$$\mathbb{G} \ni x = \pm x_2\,,\, x_1\, x_0 \;\times\; 10^e \tag{2.50}$$

wobei x_0, x_1, x_2 für die drei Ziffern stehen; zum Beispiel

$$x = 0{,}312 \;\rightarrow\; \hat{x} = +3{,}12 \times 10^{-1} \tag{2.51}$$

und die Abbildung $G : \mathbb{R} \to \mathbb{G}$ von den reellen Zahlen $\mathbb{R}$ in die Gleitkommazahlen $\mathbb{G} \equiv \mathbb{G}(10,3)$ erfolgt entsprechend den normalen Rundungsregeln.

Betrachtet werden soll das folgende Gleichungssystem:

$$\begin{array}{rcrcl} 0{,}0001\, x & + & 1{,}00\, y & = & 1{,}00 \\ 1{,}00\, x & + & 1{,}00\, y & = & 2{,}00 \end{array} \tag{2.52}$$

welches als exakte Lösung

$$x = \frac{10000}{9999} \approx 1{,}0001, \quad y = \frac{9998}{9999} \approx 0{,}9999 \tag{2.53}$$

besitzt. In $\mathbb{G}(10,3)$ erhalten wir also durch die notwendige Rundung für das Ergebnis:

$$x = 1{,}0001 = 1{,}0001 \times 10^0 \;\rightarrow\; \hat{x} = 1{,}00 \times 10^0 \tag{2.54a}$$

$$y = 0{,}9999 = 9{,}9999 \times 10^{-1} \;\rightarrow\; \hat{y} = 1{,}00 \times 10^0. \tag{2.54b}$$

In Matrixform sieht das Gleichungssystem wie folgt aus:

$$Ax = \begin{pmatrix} 0{,}0001 & 1{,}00 \\ 1{,}00 & 1{,}00 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 1{,}00 \\ 2{,}00 \end{pmatrix} = b \tag{2.55}$$

und da alle Werte exakt in unserer Gleitkomma-Darstellung abgebildet werden können haben wir $\hat{A} = G(A) = A$ und es ist keine Rundung bei den Eingangsdaten notwendig und diese können exakt in $\mathbb{G}(10,3)$ dargestellt werden. Führen wir nun Gauß-Elimination durch, so können wir das Matrixelement $a_{1,0}$ Null setzen, wenn die erste und zweite Zeile–geeignet skaliert–addiert werden. Der Skalierungsfaktor ergibt sich aus dem Pivotelement $a_{0,0}$:

$$m_{10} = \frac{a_{1,0}}{a_{0,0}} = \frac{1{,}0}{0{,}0001} = 10000.00 \;\rightarrow\; \hat{m}_{10} = 1{,}00 \times 10^4. \tag{2.56}$$

Die Elemente, welche durch die Addition modifiziert werden müssen, sind (hier mit einen Strich angegeben):

$$a'_{1,1} = \hat{a}_{1,1} - \hat{m}_{10} \cdot \hat{a}_{0,1} \tag{2.57a}$$

$$= 1{,}00 - 10000{,}00 \cdot 1{,}00 \tag{2.57b}$$

$$= -9{,}999 \times 10^3 \tag{2.57c}$$

und die Repräsentation im Gleitkommazahl-Format ist demzufolge

$$\rightarrow \hat{a}'_{1,1} = -1{,}00 \times 10^4. \tag{2.57d}$$

Analog erhalten wir für die rechte Seite des Gleichungssystems

$$b'_1 = \hat{b}_1 - \hat{m}_{10} \cdot \hat{b}_0 \tag{2.57e}$$
$$= 2{,}00 - 10000{,}00 \cdot 1{,}00 \tag{2.57f}$$
$$= -9{,}998 \times 10^3 \tag{2.57g}$$
$$\rightarrow \hat{b}'_1 = -1{,}00 \times 10^4 \tag{2.57h}$$

Das Gleichungssystem ist in oberer Dreiecksform in $\mathbb{G}(10, 3)$ also durch

$$\hat{A}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 1{,}0 \times 10^{-4} & 1{,}0 \times 10^0 \\ 0{,}0 \times 10^0 & -1{,}0 \times 10^4 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 1{,}0 \times 10^0 \\ -1{,}0 \times 10^4 \end{pmatrix} \tag{2.58}$$

gegeben. Die Lösung erhalten wir durch Rückwärtseinsetzen:

$$\hat{y} = 1{,}0 \times 10^0 = 1{,}00, \tag{2.59}$$
$$\hat{x} = 0{,}0 \times 10^0 = 0{,}00. \tag{2.60}$$

Der relative Fehler für x bezüglich der bestmöglichen Lösung in $\mathbb{G}(10, 3)$ ist damit:

$$E_{\mathrm{r}}(\hat{x}) = \frac{|1{,}00 - 0{,}00|}{|1{,}00|} = 1{,}00 \tag{2.61}$$

und damit sehr groß.

Wir wollen nun betrachten was passiert, wenn wir vor der Durchführung der Gauß-Elimination die Zeilen vertauschen. Das heißt, wir betrachten das System:

$$\begin{pmatrix} 1{,}00 & 1{,}00 \\ 0{,}0001 & 1{,}00 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 2{,}00 \\ 1{,}00 \end{pmatrix} \tag{2.62}$$

was weiterhin die gleiche Lösung wie Gl. 2.52 besitzt. Gauß-Elimination ergibt dann:

$$m_{10} = \frac{a_{1,0}}{a_{0,0}} = \frac{1{,}00 \times 10^{-4}}{1{,}00} \;\rightarrow\; \hat{m}_{10} = 1{,}00 \times 10^{-4} \tag{2.63a}$$

$$a'_{1,1} = a_{1,1} - m_{10} \cdot a_{0,1} \tag{2.63b}$$
$$= 1{,}00 - 0{,}0001 \cdot 1{,}00 \tag{2.63c}$$
$$= 0{,}9999 \tag{2.63d}$$
$$\rightarrow \hat{a}'_{1,1} = 1{,}00 \times 10^0 \tag{2.63e}$$

$$b'_1 = b_1 - m_{10} \cdot b_0 \tag{2.63f}$$
$$= 1{,}00 - 0{,}0001 \cdot 2{,}00 \tag{2.63g}$$
$$= 0{,}9998 \tag{2.63h}$$
$$\rightarrow \hat{b}'_1 = 1{,}00 \times 10^0 \tag{2.63i}$$

In oberer Dreiecksform in $\mathbb{G}(10, 3)$ erhalten wir also

$$\hat{A}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 1{,}00 & 1{,}00 \\ 0{,}00 & 1{,}00 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 2{,}00 \\ 1{,}00 \end{pmatrix}. \tag{2.64}$$

Durch Rückwärtseinsetzen erhalten wir also $\hat{x} = 1{,}00$ und $\hat{y} = 1{,}00$ und das Ergebnis ist gleich dem exakten Ergebnis in unserem Gleitkommazahl-Format aus Gl. 2.54.

In Pseudo-Code kann die Gauss-Elimination mit Pivoting wie folgt beschrieben werden:

1. Wähle im Eliminationsschritt $A^{(k)} \to A^{(k+1)}$ ein $p \in \{k, \ldots, n\}$, so dass

$$\left|a_{p,k}^{(k)}\right| \geq \left|a_{j,k}^{(k)}\right| \text{ für alle } j \in \{k, \ldots, n\}$$

2. Vertausche die Zeilen k und p und führe den Eliminationsschritt aus.

Neben dem hier beschriebenen Spalten-Pivoting existiert auch Zeilen- und totales Pivoting. Wir werden diese nicht näher betrachten.

3. Ausgleichsrechnung

3.1. Motivation

Wir hatten gesehen, dass sich die Rekonstruktion der diskreten Dichteverteilung aus der Abschwächung einzelner Strahlen als System von linearen Gleichungen modellieren lässt. Um n Dichtewerte zu ermitteln, brauchen wir dabei genau n Strahlen. Die Geometrie der Strahlen müssen wir dabei so wählen, dass die resultierende Systemmatrix $\mathbf{L}$ nicht singulär wird. Die Verwendung von mehr als n Strahlen führt auf ein überbestimmtes LGS $\mathbf{L}\mathbf{c}' = \mathbf{g}'$. Da wir bereits wissen, dass es eine Lösung $\mathbf{c}'$ geben muss, könnte man hoffen, dass sich das überbestimmte Gleichungssystem lösen lassen wird. Dies ist jedoch wegen der unweigerlich auftretenden Messungenauigkeiten und -fehler in der Praxis nicht zu erwarten.

Was machen wir also wenn ein überbestimmtes LGS der Form

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \qquad \mathbf{A} \in \mathbb{R}^{n\times m}, \mathbf{x} \in \mathbb{R}^m, \mathbf{b} \in \mathbb{R}^n, \quad n > m \tag{3.1}$$

keine Lösung hat? Es liegt Nahe, das *Residuum*

$$\mathbf{r} = \mathbf{A}\mathbf{x} - \mathbf{b} \tag{3.2}$$

zu betrachten und $\mathbf{x}$ so zu wählen, dass das Residuum bzgl. einer geeigneten Norm $\|\cdot\|$ möglichst klein wird. Die Verwendung der Länge des Residuums

$$\|\mathbf{r}\|_2 = \left(r_0^2 + \ldots + r_{n-1}^2\right)^{1/2} \tag{3.3}$$

führt auf die *lineare Ausgleichsrechnung*. In der Folge verzichten wir auf die genauere Bezeichnung der Norm und meinen mit $\|\cdot\|$ immer die *euklidische* oder *L2*-Norm $\|\cdot\|_2$.

3.2. Lineare Ausgleichsrechnung und Normalengleichung

Um einen allgemeinen Ansatz zu entwickeln, schauen wir uns zunächst den einfachsten Fall an, nämlich $n = 2, m = 1$. In diesem Fall ist die „Matrix“ $\mathbf{A}$ ein Spaltenvektor $\mathbf{a} \in \mathbb{R}^2$, genau wie die rechte Seite $\mathbf{b} \in \mathbb{R}^2$. Die gesuchte Lösung ist ein Skalar $x \in \mathbb{R}$. Im Allgemeinen wird sich $\mathbf{a} \cdot x = \mathbf{b}$ nicht erfüllen lassen, weswegen wir x so wählen, dass $\|\mathbf{r}\| = \|x\mathbf{a} - \mathbf{b}\|$ möglicht klein ist. Die Situation ist in Abb. 3.1 dargestellt.

Es ist geometrisch „intuitiv klar“, dass das Residuum senkrecht auf der Geraden $\mathbf{a} \cdot x$ stehen muss, um dessen Länge zu minimieren. Wir können dies aber auch mathematisch plausibel machen. Dazu betrachten wir eine beliebige Wahl von x und bezeichnen mit $\hat{x}$ den Fußpunkt des Lotes von $\mathbf{b}$ auf die durch $\mathbf{a}$ definierte Ursprungsgerade (siehe Abb. 3.1 (b)). Dann ist $\mathbf{r}$ die Hypotenuse im rechtwinkligen Dreieck mit den Eckpunkten $\mathbf{a} \cdot x, \mathbf{a} \cdot \hat{x}, \mathbf{b}$. Nach Pythagoras gilt für die Quadrate der Längen

$$\|\mathbf{r}\|^2 = \|\mathbf{b} - \mathbf{a} \cdot \hat{x}\|^2 + \|\mathbf{a} \cdot x - \mathbf{a} \cdot \hat{x}\|^2 = \|\mathbf{b} - \mathbf{a} \cdot \hat{x}\|^2 + \|\mathbf{a} \cdot (x - \hat{x})\|^2 \tag{3.4}$$

Da die Längen positiv sind, ist die Minimierung von $\|\mathbf{r}\|$ gleichbedeutend mit der Minimierung von $\|\mathbf{r}\|^2$. Der Ausdruck $\|\mathbf{b} - \mathbf{a} \cdot \hat{x}\|^2$ ist unabhängig von der Wahl von x. Das heißt es bleibt

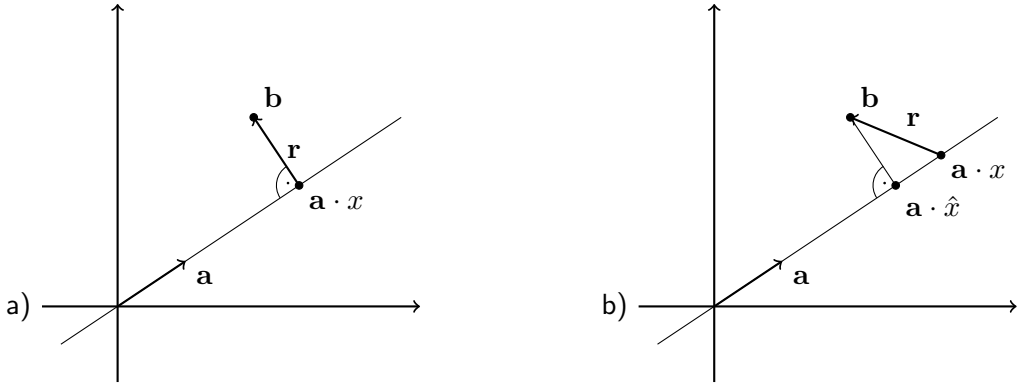


Abbildung 3.1.: *Die „beste" Lösung für das überbestimmte LGS* $\mathbf{a} \cdot x = \mathbf{b}$ *ergibt sich, wenn das Residuum* $\mathbf{r}$ *senkrecht auf der Geraden* $x\mathbf{a}$ *steht (a). Dies kann man einsehen, wenn man das Residuum anders wählt, und dann das rechtwinklige Dreieck* $\mathbf{a} \cdot x, \mathbf{a} \cdot \hat{x}, \mathbf{b}$ *betrachtet.*

nur den Term $\|\mathbf{a} \cdot (x - \hat{x})\|^2$ zu minimieren. Dieser Term ist nicht-negativ und nimmt mit der Wahl $x = \hat{x}$ den (kleinstmöglichen) Wert Null an. Wie behauptet ist es also optimal, $\mathbf{r}$ so zu wählen, dass es auf der Geraden $\mathbf{a} \cdot x$ senkrecht steht.

Wir können nun das gesuchte x ausrechnen, indem wir fordern, dass das Skalarprodukt zwischen $\mathbf{a}$ und $\mathbf{r}$ Null wird:

$$0 = \mathbf{a}^\mathsf{T}\mathbf{r} = \mathbf{a}^\mathsf{T}(\mathbf{a} \cdot x - \mathbf{b}) \quad \Longleftrightarrow \quad \mathbf{a}^\mathsf{T}\mathbf{a} \cdot x = \mathbf{a}^\mathsf{T}\mathbf{b}. \tag{3.5}$$

Das heißt, für jeden nicht verschwindenden Vektor $\mathbf{a}$ finden wir die Orthogonalprojektion von $\mathbf{b}$ auf die Ursprungsgerade $\mathbf{a} \cdot x$ als $x = \mathbf{a}^\mathsf{T}\mathbf{b}/\mathbf{a}^\mathsf{T}\mathbf{a}$. Diese Orthogonalprojektion minimiert die Länge des Vektors zwischen $\mathbf{a} \cdot x$ und $\mathbf{b}$. Diese Beobachtungen gelten nicht nur für $n = 2$, sondern für jedes n.

Wie hilft uns diese Erkenntnis weiter, um den allgemeinen Fall $\mathbf{A}\mathbf{x} \approx \mathbf{b}$ zu lösen? Dazu betrachten wir die Spalten von $\mathbf{A} = (\mathbf{a}_0, \mathbf{a}_1, \ldots, \mathbf{a}_{m-1})$. Wie zuvor argumentieren wir, dass das Residuum senkrecht stehen sollte, und zwar auf jedem der Spaltenvektoren. Nehmen wir nämlich an, $\mathbf{r}$ stünde nicht orthogonal auf $\mathbf{a}_i$, so könnte man sich (analog zum Fall oben) entlang von $\mathbf{a}_i$ bewegen, um die Länge von $\mathbf{r}$ zu minimieren. Es muss also für alle $\mathbf{a}_i$ gelten: $\mathbf{a}_i^\mathsf{T}\mathbf{r} = 0$. Betrachten wir diese Gleichungen gemeinsam, so sehen wir

$$\left.\begin{array}{rcccl} 0 & = & \mathbf{a}_0^\mathsf{T}\mathbf{r} & = & \mathbf{a}_0^\mathsf{T}(\mathbf{A}\mathbf{x} - \mathbf{b}) \\ 0 & = & \mathbf{a}_1^\mathsf{T}\mathbf{r} & = & \mathbf{a}_1^\mathsf{T}(\mathbf{A}\mathbf{x} - \mathbf{b}) \\ \vdots & & \vdots & & \vdots \\ 0 & = & \mathbf{a}_{m-1}^\mathsf{T}\mathbf{r} & = & \mathbf{a}_{m-1}^\mathsf{T}(\mathbf{A}\mathbf{x} - \mathbf{b}) \end{array}\right\} \quad \mathbf{0} = \mathbf{A}^\mathsf{T}(\mathbf{A}\mathbf{x} - \mathbf{b}). \tag{3.6}$$

Damit ergibt sich also ein LGS zur Bestimmung von $\mathbf{x}$, die *Normalengleichung*:

$$\mathbf{A}^\mathsf{T}\mathbf{A}\mathbf{x} = \mathbf{A}^\mathsf{T}\mathbf{b}. \tag{3.7}$$

Genau wie im Falle einer Unbekannten, reduziert man die unlösbare Gleichung durch Multiplikation mit den transponierten Basisvektoren. Als Lösung erhält man (für invertierbare Matrizen $\mathbf{A}^\mathsf{T}\mathbf{A}$)

$$\mathbf{x} = \left(\mathbf{A}^\mathsf{T}\mathbf{A}\right)^{-1}\mathbf{A}^\mathsf{T}\mathbf{b} \tag{3.8}$$

und damit als Ergebnis der Orthogonalprojektion auf den durch die Spalten von $\mathbf{A}$ aufgespannten linearen Unterraum

$$\mathbf{A}\mathbf{x} = \underbrace{\mathbf{A}\left(\mathbf{A}^\mathsf{T}\mathbf{A}\right)^{-1}\mathbf{A}^\mathsf{T}}_{\text{Projektor}}\ \mathbf{b}. \tag{3.9}$$

Die Matrix $\mathbf{A}\left(\mathbf{A}^\mathsf{T}\mathbf{A}\right)^{-1}\mathbf{A}^\mathsf{T} \in \mathbb{R}^{n\times n}$ ist der *Projektor*: diese Matrix projiziert jeden Vektor $\mathbf{b} \in \mathbb{R}^n$ auf den durch die Spalten von $\mathbf{A}$ aufgespannten Unterraum. Eine Projektionsmatrix $\mathbf{P}$ hat die Eigenschaft, dass die Projektion der Projektion keine Veränderung mehr herbeiführt, also $\mathbf{P}^2 = \mathbf{P}$. Davon kann man sich im Falle von $\mathbf{P} = \mathbf{A}\left(\mathbf{A}^\mathsf{T}\mathbf{A}\right)^{-1}\mathbf{A}^\mathsf{T}$ leicht überzeugen.

3.3. Lineare Regression

Die Normalengleichung dient auch zur Lösung eines wichtigen praktischen Problems, nämlich der *linearen Regression*. Hierbei erheben wir n Messdaten $y_0, \ldots, y_{n-1}$ an verschiedenen Parameterstellen $\mathbf{x}_0, \ldots, \mathbf{x}_{n-1}, \mathbf{x}_i \in \mathbb{R}^m$. Wir sind interessiert an der Funktion f, die aus den Parametern die Funktionsdaten erzeugt: $f : \mathbb{R}^m \mapsto \mathbb{R}$. Ohne weitere Annahmen, lässt sich die Funktion f nicht aus den Messdaten bestimmen. Wenn wir eine Annahme bzgl. der Form von f machen, so können wir f bei ausreichender Anzahl an Messpunkten näher bestimmen. Wie bei der Computertomographie, ist die Anzahl der Variablen zur Definition der Funktion f dabei meist kleiner als die Anzahl der Messungen; und für die Messungen muss man vermuten, dass sie mit Rauschen behaftet sind, und/oder wegen der vereinfachenden Modellannahmen nicht alle eine gemeinsame Funktion efüllen werden.

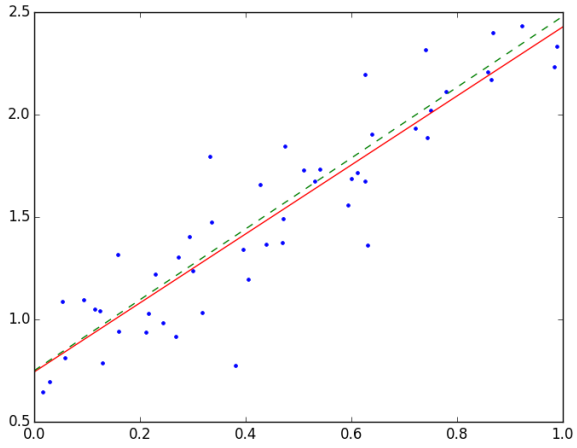


Abbildung 3.2.: *Einzelne Messergebnisse (blaue Punkte) wurden aus einer linearen Gleichung (grün gestrichelt) mit normalverteiltem Rauschen behaftet. Mit linearer Regression wurde der rot dargestellte lineare Zusammenhang geschätzt.*

Ein gängiger Ansatz ist die Verwendung von linearen Funktionen f, also

$$f(\mathbf{x}) = c_0x_0 + c_1x_1 + \ldots c_{m-1}x_{m-1} + b = \mathbf{x}^\mathsf{T}\mathbf{c} + b = (\mathbf{x}^\mathsf{T}, 1)\begin{pmatrix}\mathbf{c}\\ b\end{pmatrix}. \tag{3.10}$$

Für $m = 1$, also ein-dimensionale Parameterwerte ist die Situation in Abb. 3.2 illustriert. Man möchte $\mathbf{c} = (c_0, \ldots, c_{m-1})^\mathsf{T}, b$ so wählen, dass

$$y_i \approx f(\mathbf{x}_i) = \mathbf{x}_i^\mathsf{T}\mathbf{c} + b = (\mathbf{x}^\mathsf{T}, 1)\begin{pmatrix}\mathbf{c}\\ b\end{pmatrix} \tag{3.11}$$

bzw. so dass der Fehler $y_i - \mathbf{c}^\mathsf{T}\mathbf{x}_i + b$ in allen Messwerten klein wird. Wenn wir dabei als Maß für den Gesamtfehler die Summe der Quadrate

$$\Big(y_0 - \mathbf{x}_0^\mathsf{T}\mathbf{c} + b\Big)^2 + \ldots + \Big(y_{n-1} - \mathbf{x}_{n-1}^\mathsf{T}\mathbf{c} + b\Big)^2 \tag{3.12}$$

so ist die Situation ähnlich zur linearen Ausgleichsrechnung, denn wir betrachten (das Quadrat) der Länge des Vektors

$$\mathbf{r} = \mathbf{y} - \begin{pmatrix}\mathbf{x}_0^\mathsf{T}, 1\\ \mathbf{x}_1^\mathsf{T}, 1\\ \vdots\\ \mathbf{x}_{n-1}^\mathsf{T}, 1\end{pmatrix}\begin{pmatrix}\mathbf{c}\\ b\end{pmatrix} \tag{3.13}$$

Hier ist also die Systemmatrix gegeben durch die Messparameter $\mathbf{x}_i$ und die Zahl 1, die in den Zeilen auftauchen. Die Messungen $\mathbf{y}$ sind die rechte Seite des Systems. Und die Unbekannten sind die Parameter $\mathbf{c}, b$, die die gesuchte lineare Funktion beschreiben. Wie zuvor, finden wir das kürzeste Residuum durch Lösung der Normalengleichung, die hier die Form

$$\begin{pmatrix}\mathbf{x}_0 & \mathbf{x}_1 & \ldots & \mathbf{x}_{n-1}\\ 1 & 1 & \ldots & 1\end{pmatrix}\begin{pmatrix}\mathbf{x}_0^\mathsf{T}, 1\\ \mathbf{x}_1^\mathsf{T}, 1\\ \vdots\\ \mathbf{x}_{n-1}^\mathsf{T}, 1\end{pmatrix}\begin{pmatrix}\mathbf{c}\\ b\end{pmatrix} = \begin{pmatrix}\mathbf{x}_0 & \mathbf{x}_1 & \ldots & \mathbf{x}_{n-1}\\ 1 & 1 & \ldots & 1\end{pmatrix}\mathbf{y} \tag{3.14}$$

annimmt. Da wir in Bezug auf unsere Messungen in jedem Datenpunkt das Quadrat des Fehlers minimieren, spricht man bei dieser Lösung von der *Methode der kleinsten Fehlerquadrate*.

Eine wichtige Eigenschaft der Ausgleichsrechnung ist, dass der Schätzfehler für die Funktionsparameter im Regelfall mit der Anzahl der Datenpunkte abnimmt. Für die lineare Regression ist dies in Abb. 3.3 und der folgenden Tabelle dargestellt (gemittelt über mehrere Instanzen des Rauschens):

n	c	b
korrekt	1,73	0,75
50	0,075	0,043
100	0,056	0,032
150	0,048	0,028
200	0,040	0,022
250	0,035	0,019
300	0,032	0,019
350	0,029	0,017
400	0,028	0,016
450	0,025	0,015
500	0,025	0,015

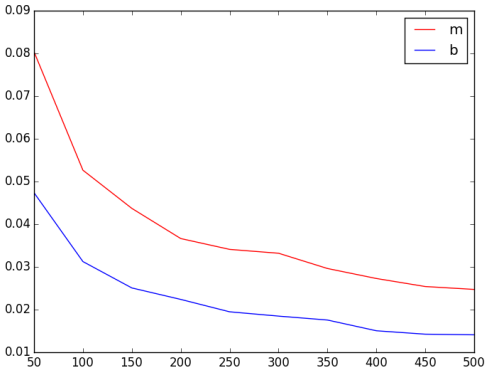


Abbildung 3.3.: *Fehler in c und b in Abhängigkeit von der Anzahl der Datenpunkte.*

3.4. Lösen der Normalengleichung

Jedes überbestimmte LGS $\mathbf{Ax} = \mathbf{b}$ können wir also durch Multiplikation beider Seiten in die Normalform $\mathbf{A}^\mathsf{T}\mathbf{Ax} = \mathbf{A}^\mathsf{T}\mathbf{b}$ bringen. Die neue Systemmatrix $\mathbf{A}^\mathsf{T}\mathbf{A}$ ist quadratisch – das LGS also lösbar sofern $\mathbf{A}^\mathsf{T}\mathbf{A}$ vollen Rang hat. Natürlich können wir das Gaußsche Eliminationsverfahren zur Lösung verwenden. Aber die Matrix $\mathbf{A}^\mathsf{T}\mathbf{A}$ hat offenbar eine besondere Struktur, und es stellt sich die Frage ob es nicht Verfahren gibt, die die besondere Konstruktion ausnutzen.

3.4.1. Eigenschaften von $\mathbf{A}^\mathsf{T}\mathbf{A}$

Zunächst erinnern wir uns, dass das Produkt einer Matrix mit ihrer Transponierten immer *symmetrisch* ist:

$$\left(\mathbf{A}^\mathsf{T}\mathbf{A}\right)^\mathsf{T} = \mathbf{A}^\mathsf{T}\mathbf{A}^{\mathsf{T}^\mathsf{T}} = \mathbf{A}^\mathsf{T}\mathbf{A} \tag{3.15}$$

In der Matrix $\mathbf{B} = \mathbf{A}^\mathsf{T}\mathbf{A}$ sind also die Elemente $b_{ij} = b_{ji}$, was Anlass zur Hoffnung geben kann, einen effizienteren Algorithmus zur Lösung von LGS mit $\mathbf{B}$ zu finden.

Darüber hinaus ist $\mathbf{A}^\mathsf{T}\mathbf{A}$ *positiv semidefinit* (PSD). Damit meint man, dass die quadratische Funktion $\mathbf{x}^\mathsf{T}\mathbf{A}^\mathsf{T}\mathbf{Ax}$ für jede Wahl $\mathbf{x}$ nur nicht-negative Werte annehmen kann:

$$\mathbf{x}^\mathsf{T}\mathbf{A}^\mathsf{T}\mathbf{Ax} = (\mathbf{Ax})^\mathsf{T}(\mathbf{Ax}) = \mathbf{y}^\mathsf{T}\mathbf{y} = \|\mathbf{y}\|^2 \geq 0. \tag{3.16}$$

Hat $\mathbf{A}$ vollen Spaltenrang, so ist $\mathbf{A}^\mathsf{T}\mathbf{A}$ regulär (also invertierbar). Wir sehen das daran, dass jedes Element $\mathbf{x}$ im Kern von $\mathbf{A}^\mathsf{T}\mathbf{A}$, also $\mathbf{A}^\mathsf{T}\mathbf{Ax} = 0$, auch im Kern von $\mathbf{A}$ liegt: multipliziert man $\mathbf{A}^\mathsf{T}\mathbf{Ax} = 0$ von links mit $\mathbf{x}^\mathsf{T}$ so ergibt sich direkt $\|\mathbf{Ax}\|^2 = 0$ und damit wie behauptet $\mathbf{Ax} = 0$. Hätte also $\mathbf{A}^\mathsf{T}\mathbf{A}$ einen nicht-leeren Kern (wäre also nicht invertierbar), so würde dies auch für $\mathbf{A}$ gelten. Sind die Spalten von $\mathbf{A}$ linear unabhängig so ist also $\mathbf{A}^\mathsf{T}\mathbf{A}$ invertierbar. Wir können die lineare Unabhängigkeit der Spalten von $\mathbf{A}$ in der Praxis oft dadurch erreichen, dass wir weitere Messungen (also Zeilen) hinzufügen. Wir können also in der Regel davon ausgehen (oder dies durch geeignete Messungen erreichen), dass $\mathbf{A}^\mathsf{T}\mathbf{A}$ invertierbar ist.

Hat also $\mathbf{A}^\mathsf{T}\mathbf{A}$ keinen Kern, so ist $\mathbf{A}^\mathsf{T}\mathbf{A}\mathbf{x} \neq 0$ für alle $\mathbf{x} \neq \mathbf{0}$. In diesem Fall ist $\mathbf{A}^\mathsf{T}\mathbf{A}\mathbf{x} = 0$ nicht nur positiv semidefinit sondern *positiv definit* (PD):

$$\mathbf{x} \neq \mathbf{0} \quad \Longleftrightarrow \quad \mathbf{x}^\mathsf{T}\mathbf{A}^\mathsf{T}\mathbf{A}\mathbf{x} > 0 \tag{3.17}$$

Für praktische Probleme gehen wir also davon aus, dass $\mathbf{A}^\mathsf{T}\mathbf{A}$ eine symmetrische PD Matrix ist.

3.4.2. Eigenschaften von P(S)D-Matrizen

Die positiv (semi-)definiten Matrizen haben einige besondere Eigenschaften, die manchmal bei Berechnungen nützlich sein können. Im Folgenden gehen wir davon aus, dass $\mathbf{B}$ eine PD-Matrix ist, also $\mathbf{x}^\mathsf{T}\mathbf{B}\mathbf{x} > 0$ für alle $\mathbf{x} \neq \mathbf{0}$ gilt – die diskutierten Eigenschaften gelten in analoger Weise für PSD-Matrizen (sprich, man ersetzt das Zeichen „$>$“ durch „$\geq$“ und den Begriff „positiv“ durch „nicht-negativ“).

Unsere erste Beobachtung ergibt sich, wenn wir den Eintrag x_i des Vektors $\mathbf{x}$ in $\mathbf{x}^\mathsf{T}\mathbf{B}\mathbf{x}$ Null setzen. Dann spielen die Zeile und Spalte mit dem Index i in $\mathbf{B}$ keine Rolle. Das heißt, auch Matrizen, die durch Löschen einer Zeile und Spalte mit dem gleichen Index entstehen sind wieder PD-Matrizen. Auch aus dieser Matrix kann man wieder korrespondierende Zeilen und Spalten entfernen und die resultierenden Matrizen sind wiederum PD-Matrizen. Diese Beobachtung gilt in gleicher Weise für PSD-Matrizen. Mit anderen Worten: ist $\mathbf{B}$ eine PD Matrix, so gilt diese Eigenschaft auch für alle Matrizen, die aus $\mathbf{B}$ durch Löschen von ein oder mehrerer korrespondierender Zeilen und Spalten entstehen.

Löscht man aus $\mathbf{B}$ alle Zeilen und Spalten bis auf den Index i, so bleibt nur das Diagonalelement $b_{i,i}$ stehen. Da gelten muss $xb_{i,i}x = x^2 b_{i,i} > 0$ für alle $x \neq 0$ sind also insbesondere die Diagonalemente einer PD-Matrix positiv.

Ist $\mathbf{B}$ eine PD-Matrix, so auch $s\mathbf{B}$ für beliebiges $s > 0$. Dies ergibt sich sofort durch Multiplikation der Ungleichung $\mathbf{x}^\mathsf{T}\mathbf{B}\mathbf{x} > 0$ auf beiden Seiten mit s. Ist zusätzlich $\mathbf{A}$ eine PD-Matrix, so ist die Matrix $\mathbf{A} + \mathbf{B}$ eine PD-Matrix:

$$\mathbf{x}^\mathsf{T}(\mathbf{A} + \mathbf{B})\,\mathbf{x} = \underbrace{\mathbf{x}^\mathsf{T}\mathbf{A}\mathbf{x}}_{>0} + \underbrace{\mathbf{x}^\mathsf{T}\mathbf{B}\mathbf{x}}_{>0} > 0 \tag{3.18}$$

Nimmt man diese beiden Eigenschaften zusammen sind also für zwei PD-Matrizen $\mathbf{A}, \mathbf{B}$ auch alle Matrizen $s\mathbf{A} + t\mathbf{B}$ für $s, t > 0$ wiederum PD-Matrizen.

Alle PD-Matrizen haben positive Determinante! Um das zu sehen, betrachten wir zunächst die Identität $\mathbf{I}$, die eine PD-Matrix ist und positive Determinante hat. Alle Matrizen der Form $(1-t)\mathbf{I} + t\mathbf{B}, t \in [0,1]$ sind nach unserer Beobachtung oben PD-Matrizen, insbesondere sind alle regulär. Die Funktion $\det((1-t)\mathbf{I} + t\mathbf{B})$ hat also keine Nullstellen im Intervall $t \in [0,1]$. Ferner ist sie (als lineare Funktion) stetig und für $t = 0$ nimmt sie den Wert 1 an. Als stetige Funktion ohne Nulldurchgang nimmt sie also für alle $t \in [0,1]$ positive Wert an, insbesondere auch für $t = 1$ und damit die (beliebig gewählte) PD-Matrix $\mathbf{B}$.

Oft ist es einfacher mit einer Matrix zu rechnen, wenn man sicher sein kann, dass sie regulär ist, also PD und nicht nur PSD ist. Dies kann man nach unseren Überlegungen immer wie folgt erreichen. Sei $\mathbf{A}$ eine PSD-Matrix, und $\mathbf{B}$ wie zuvor eine PD-Matrix. Dann ist $s\mathbf{A} + t\mathbf{B}$ für $s, t > 0$ eine PD-Matrix. Eine häufige Wahl ist $\mathbf{A} + \epsilon\mathbf{I}$ oder $(1-\epsilon)\mathbf{A} + \epsilon\mathbf{I}$: durch Addition eines kleinen positiven Vielfachen der (positiv definiten) Identitätsmatrix wird aus der PSD-Matrix $\mathbf{A}$ eine PD-Matrix. Dieser Vorgang wird manchmal *Regularisierung* genannt.

3.4.3. Cholesky-Zerlegung

Wir haben gesehen, dass sich LGS gut lösen lassen, wenn die beteiligten Matrizen Dreiecksform haben. Interessanterweise lässt sich jede symmetrische PD Matrix als Produkt einer Dreiecksmatrix mit ihrer eigenen Transponierten schreiben. Diese Eigenschaft werden wir konstruktiv nachweisen, sprich wir geben direkt einen Algorithmus an, der die Elemente der Dreiecksmatrix konstruiert.

Wir wollen also die symmetrische PD Matrix $\mathbf{B}$ schreiben als $\mathbf{L}\mathbf{L}^\mathsf{T}$, wobei $\mathbf{L}$ eine untere Dreiecksmatrix ist (in diesem Fall ist also $\mathbf{L}^\mathsf{T}$ eine obere Dreiecksmatrix). Ausrechnen der einzelnen Einträge der Matrizen ergibt

$$\begin{pmatrix} b_{0,0} & b_{0,1} & b_{0,2} & \dots \\ b_{1,0} & b_{1,1} & b_{1,2} & \dots \\ b_{2,0} & b_{2,1} & b_{2,2} & \dots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix} = \begin{pmatrix} l_{0,0} & 0 & 0 & \dots \\ l_{1,0} & l_{1,1} & 0 & \dots \\ l_{2,0} & l_{2,1} & l_{2,2} & \dots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix} \begin{pmatrix} l_{0,0} & l_{1,0} & l_{2,0} & \dots \\ 0 & l_{1,1} & l_{2,1} & \dots \\ 0 & 0 & l_{2,2} & \dots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix} = \begin{pmatrix} l_{0,0}^2 & l_{1,0}l_{0,0} & l_{2,0}l_{0,0} & \dots \\ l_{1,0}l_{0,0} & l_{1,0}^2 + l_{1,1}^2 & l_{1,0}l_{2,0} + l_{1,1}l_{2,1} & \dots \\ l_{2,0}l_{0,0} & l_{2,0}l_{1,0} + l_{2,1}l_{1,1} & l_{2,0}^2 + l_{2,1}^2 + l_{2,2}^2 & \dots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix} \tag{3.19}$$

Die entscheidende Beobachtung liegt darin, dass sich die Unbekannten $l_{i,j}$ zeilenweise von links nach rechts direkt bestimmen lassen: für jedes Element kommt genau ein unbekannter Eintrag $l_{i,j}$ hinzu. Wir beginnen mit

$$b_{0,0} = l_{0,0}^2 \quad \Longrightarrow \quad l_{0,0} = \sqrt{b_{0,0}}, \tag{3.20}$$

wobei wir wissen, dass $b_{0,0}$ als Diagonalelement einer symmetrischen PD Matrix immer positiv ist und damit $l_{0,0}$ existiert und auch positiv ist. Das ist nützlich zum Ausrechnen der weiteren Elemente der Zeile, wie $b_{0,1} = l_{1,0}l_{0,0}, \quad b_{0,2} = l_{2,0}l_{0,0}$ usw., also allgemein

$$b_{0,i} = l_{i,0}l_{0,0} \quad \Longrightarrow \quad l_{i,0} = \frac{b_{0,i}}{l_{0,0}} \qquad i > 0. \tag{3.21}$$

Wir wissen bereits, dass der Nenner $l_{0,0}$ positiv ist. In der nächsten Zeile können wir das erste Element überspringen, denn $\mathbf{B}$ ist symmetrisch. Für das Diagonalelement erhalten wir

$$b_{1,1} = l_{1,0}^2 + l_{1,1}^2 \quad \Longrightarrow \quad l_{1,1} = \sqrt{b_{1,1} - l_{1,0}^2}. \tag{3.22}$$

Hier nutzen wir aus, dass $l_{1,0}$ bereits bei der Bearbeitung der ersten Zeile ausgerechnet worden ist. Aber warum ist der Ausdruck unter Wurzel positiv? Setzen wir die bekannten Werte ein, so erhalten wir die Bedingung

$$b_{1,1} - l_{1,0}^2 = b_{1,1} - \left(\frac{b_{0,1}}{l_{0,0}}\right)^2 = b_{1,1} - \frac{b_{0,1}^2}{b_{0,0}} > 0 \tag{3.23}$$

Multipliziert man die Ungleichung rechts auf beiden Seiten mit (der positiven Zahl) $b_{0,0}$ so ergibt sich $b_{0,0}b_{1,1} - b_{0,1}^2 > 0$. Dieser Ausdruck ist die Determinante der oberen linken 2×2 Matrix von $\mathbf{B}$. Nach unseren Vorüberlegungen sind alle Matrizen, die durch Löschen von korrespondierenden Zeilen und Spalten entstehen wieder PD-Matrizen, also auch die obere

2×2-Matrix von $\mathbf{B}$. Und alle PD-Matrizen haben positive Determinante. Folglich ist der Ausdruck unter der Wurzel positiv und damit auch $l_{1,1}$.

Für alle weiteren Elemente in der Zeile ergibt sich

$$b_{1,i} = l_{1,0}l_{i,0} + l_{1,1}l_{i,1} \quad \Longrightarrow \quad l_{i,1} = \frac{b_{1,i} - l_{1,0}l_{i,0}}{l_{1,1}} \qquad i > 1. \tag{3.24}$$

Wiederum haben wir bereits alle notwendigen Elemente von $\mathbf{L}$ entweder in der ersten Zeile oder im ersten Schritt der zweiten Zeile ausgerechnet und können so $l_{i,1}$ direkt bestimmen.

Geht man in gleicher Art und Weise weiter, so erhält man allgemein für die Bestimmung des ersten (Diagonal-)Elements der Zeile i

$$l_{i,i} = \sqrt{b_{i,i} - \sum_{k=0}^{i-1} l_{i,k}^2} \tag{3.25}$$

und man kann für alle Audrücke zeigen, dass der Ausdruck unter der Wurzel positiv ist. Für die weiteren Elemente der Zeile ergibt sich

$$l_{i,j} = \frac{1}{l_{j,j}} \left(b_{i,j} - \sum_{k=0}^{j-1} l_{i,k}l_{j,k} \right), \qquad i > j. \tag{3.26}$$

Diese beiden Formeln beschreiben vollständig die *Cholesky*-Zerlegung einer symmetrischen PD Matrix. Sie stellen tatsächlich einen vernünftigen Algorithmus dar. Für gut konditionierte Matrizen wird keine Pivotisierung benötigt.

3.4.4. Vorwärts- / Rückwärtseinsetzen

Hat man das LGS $\mathbf{Bx} = \mathbf{c}$ in die Form $\mathbf{LL}^\mathsf{T}\mathbf{x} = \mathbf{c}$ gebracht, so lässt es sich durch zweimaliges Einsetzen lösen. Dazu setzt man $\mathbf{L}^\mathsf{T}\mathbf{x} = \mathbf{y}$ und löst zunächst durch Vorwärtseinsetzen das System $\mathbf{Ly} = \mathbf{c}$ nach $\mathbf{y}$ auf. Dann kann man $\mathbf{L}^\mathsf{T}\mathbf{x} = \mathbf{y}$ durch Rückwärtseinsetzen nach dem gesuchten $\mathbf{x}$ lösen.

Der wesentliche Vorteil der Cholesky-Zerlegung mit anschließendem Einsetzen liegt in der Geschwindigkeit: das gesamte Verfahren benötigt nur etwa halb so viele Operation wie das Gaußsche Eliminationsverfahren. Dass man auf Pivotisierung verzichten kann, erlaubt entweder eine einfache Implementierung oder den möglichen Zeilentausch für andere Optimierungen zu nutzen.

4. Eigenwerte und Eigenvektoren

4.1. Eigenwerte und Eigenvektoren

Sei $A \in \mathbb{R}^{n\times n}$ eine Matrix. Ein $\lambda \in \mathbb{R}$ heißt *Eigenwert* von A, wenn es einen von Null verschiedenen Vektor $v \in \mathbb{R}^n$ mit der der Eigenschaft $Av = \lambda v$ gibt. Den Vektor v bezeichnet man in diesem Fall als *Eigenvektor* zum Eigenwert λ. Die Menge

$$\mathrm{Eig}(A, \lambda) = \left\{ v \in \mathbb{R}^n : Av = \lambda v \right\} \tag{4.1}$$

der Eigenvektoren zu einem Eigenwert λ zusammen mit dem Nullvektor, heißt der *Eigenraum* zum Eigenwert λ. Die Menge der Eigenwerte $\sigma(A)$ wird als *Spektrum* der Matrix bezeichnet. Der betragsmäßig größte Eigenwert einer Matrix heißt *spektraler Radius*.

Eigenvektoren sind also besondere Vektoren für welche das Abbildungsverhalten der Matrix besonders einfach ist, d.h., durch eine Streckung bzw. Stauchung beschrieben ist.

Man beachte, dass ein Eigenvektor per Definition stets von Null verschieden ist. Ansonsten wäre nämlich jedes $\lambda \in \mathbb{R}$ ein Eigenwert, da immer $A0 = \lambda 0$ gilt. Im Gegensatz dazu kann ein Eigenwert durchaus Null sein.

- $\mathrm{Eig}(A, \lambda) = \ker(A - \lambda I)$: $Av = \lambda v \Leftrightarrow Av = \lambda I v \Leftrightarrow Av - \lambda I v = 0 \Leftrightarrow (A - \lambda I)v = 0$.
- $\mathrm{Eig}(A, \lambda)$ ist ein linearer Unterraum von $\mathbb{R}^n$.
- λ ist genau dann ein Eigenwert von A, wenn $\mathrm{Eig}(A, \lambda) \neq \{0\}$
- Eigenvektoren zu verschiedenen Eigenwerten sind linear unabhängig.

Satz 1. *$\lambda \in \mathbb{R}$ ist genau dann Eigenwert von A, wenn $\det(A - \lambda I) = 0$ ist.*

Beweis. Es gilt:

$$\begin{aligned} \lambda \text{ ist Eigenwert von } A &\Leftrightarrow \exists v \neq 0 : v \in \ker(A - \lambda I) \\ &\Leftrightarrow A - \lambda I \text{ ist nicht injektiv} \\ &\Leftrightarrow A - \lambda I \text{ ist nicht bijektiv} \Leftrightarrow \det(A - \lambda I) = 0 \end{aligned}$$ □

Die Funktion $\chi_A(t) = \det(A - tI)$ ist ein Polynom vom Grad n und heißt *charakteristisches Polynom* von A. Die Vielfachheit der Nullstelle heißt die *algebraische Vielfachheit* des Eigenwertes. Die Dimension des Eigenraumes $\mathrm{Eig}(A, \lambda)$ zu einem Eigenwert λ heißt *geometrische Vielfachheit* des Eigenwertes.

Satz 2. *Sei A eine quadratischen Matrix. Dann hat die transponierte Matrix A^T die gleichen Eigenwerte wie A.*

Beweis. Sei λ ein Eigenwert von A. Dann gilt

$$\det(A^T - \lambda I) = \det((A - \lambda I)^T) = \det(A - \lambda I)$$

Eine Matrix und ihre Transponierte haben also das gleiche charakteristische Polynom und daher auch die gleichen Eigenwerte. □

- Das Produkt aller Eigenwerte (gezählt entsprechend der algebraischen Vielfachheiten) ist gleich der Determinanten.
- Die Summe aller Eigenwerte (gezählt entsprechend der algebraischen Vielfachheiten) ist gleich der Spur der Matrix..

Über die Existenz von Eigenwerten lassen sich folgende Aussagen machen:

- Jede Matrix aus $\mathbb{C}^{n\times n}$ hat mindestens einen Eigenwert.
- Jede Matrix aus $\mathbb{K}^{n\times n}$ ($\mathbb{K} = \mathbb{R}, \mathbb{C}$) hat höchstens n Eigenwerte.
- Für ungerades n hat jede Matrix aus $\mathbb{R}^{n\times n}$ mindestens einen Eigenwert.

Satz 3. *Sei eine $\mathbb{R}^{n\times n}$ Matrix mit n verschiedene Eigenwerte gegeben. Dann bilden die Eigenvektoren eine Basis.*

Beweis. Angenommen $v_1, \ldots, v_n$ sind nicht linear unabhängig. Sei r die größte Zahl, so dass $v_1, \ldots, v_r$ linear unabhängig sind. Dann ist $r < n$ und $v_1, \ldots, v_{r+1}$ linear abhängig. Das heißt es gibt α_i so dass $\sum_{i=1}^{r+1} \alpha_i v_i = 0$ mit nicht allen $\alpha_i = 0$. Multiplikation beider Seiten mit A ergibt:

$$\sum_{i=1}^{r+1} A\alpha_i v_i = \sum_{i=1}^{r+1} \alpha_i \lambda_i v_i = 0\,.$$

Subtrahieren von $\lambda_{r+1} \sum_{i=1}^{r+1} A\alpha_i v_i = 0$ ergibt

$$\sum_{i=1}^{r} \alpha_i(\lambda_i - \lambda_{r+1}) v_i = 0\,.$$

Daraus folgt $\alpha_i(\lambda_i - \lambda_{r+1}) = 0$, da die v_i für $i = 1, \ldots, r$ linear unabhängig sind. Da die Eigenwerte unterschiedlich sind folgt somit $\alpha_i = 0$. Das bedeutet aber $\alpha_{r+1} v_{r+1} = 0$. Woraus wiederrum $\alpha_{r+1} = 0$ folgt, weil v_{r+1} Eigenvektor und daher von Null verschieden ist. Dies ist ein Widerspruch zu der Annahme das nicht alle α gleich Null sind. □

4.2. Diagonalisierbarkeit

Zwei Matrizen A, B heißen *ähnlich* wenn es eine invertierbare Matrix S gibt mit $B = S^{-1}AS$. Eine Matrix heißt *diagonalisierbar*, wenn sie ähnlich zu einer Diagonalmatrix ist.

Satz 4. *Sind A und B ähnliche Matrizen, so haben sie das gleiche charakteristische Polynom und daher die gleichen Eigenwerte.*

Beweis. Sei $B = S^{-1}AS$ mit $S \in \mathrm{GL}(n)$. Dann ist

$$\begin{aligned}\det(B - \lambda I) &= \det(S^{-1}AS - \lambda I) = \det(S^{-1}AS - \lambda S^{-1}S) \\ &= \det(S^{-1}(A - \lambda I)S) = \det(S^{-1})\det(A - \lambda I)\det(S) = \det(A - \lambda I)\end{aligned}$$ □

Satz 5. *Seien A, B ähnliche Matrizen mit $B = S^{-1}AS$. Ist v ein Eigenvektor von A, dann ist $v' = S^{-1}v$ ein Eigenvektor von B.*

Beweis. Sei λ der Eigenwert zum Eigenvektor v von A. Dann gilt:

$$Bv' = S^{-1}AS(S^{-1}v) = S^{-1}Av = S^{-1}\lambda v = \lambda S^{-1}v = \lambda v'$$ □

Satz 6. *Sei $A \in \mathbb{R}^{n\times n}$ eine quadratische Matrix. Dann sind die foldenden Aussagen äquivalent:*

1. *Die Matrix A ist diagonalisierbar.*
2. *Das charakteristische Polynom $\chi_A(t)$ zerfällt vollständig in Linearfaktoren:*

   $$\pm\chi_A(t) = (t - \lambda)^{m_1} \cdot \ldots \cdot (t - \lambda)^{m_r} \qquad \lambda_i \neq \lambda_j,\ 1 \leq i < j \leq r$$

   und für jeden Eigenwert λ_i gilt $\dim \mathrm{Eig}(A, \lambda_i) = m_i$.
3. $\sum_{i=1}^{r} \dim \mathrm{Eig}(A, \lambda_i) = n$
4. $\oplus_{i=1}^{r} \mathrm{Eig}(A, \lambda_i) = \mathbb{R}^n$

Beweis. TBD □

4.3. Symmetrische Matrizen

Eine Matrix $A \in \mathbb{R}^{n\times n}$ heißt *symmetrisch*, wenn $A = A^T$ gilt.

Satz 7. *Sei $A \in \mathbb{R}^{n\times n}$ eine symmetrische Matrix. Dann sind alle Eigenwerte reell. Ist $z = x + iy \in \mathbb{C}^n$ ein komplexer Eigenvektor von A zum Eigenwert λ, dann sind x und y, sofern von Null verschieden, reelle Eigenvektoren von A mit Eigenwert λ.*

Beweis. Sei $z \in \mathbb{C}^n$ ein Eigenvektor zum Eigenwert λ. Da A symmetrisch und reell ist, gilt $A = A^*$. Somit folgt:

$$\lambda\langle v, v\rangle = \langle \lambda v, v\rangle = \langle Av, v\rangle = \langle v, A^*v\rangle = \langle v, Av\rangle = \langle v, \lambda v\rangle = \bar{\lambda}\langle v, v\rangle \tag{4.2}$$

Nach Voraussetzung ist $v \neq 0$. Es ist somit $\lambda = \bar{\lambda}$ und daher λ reell.

Sei nun $z = x + iy$ ein Eigenvektor von A zum Eigenwert λ. Dann gilt:

$$Az = \lambda z \Leftrightarrow A(x + iy) = \lambda(x + iy) \Leftrightarrow Ax + iAy = \lambda x + i\lambda y$$

Da A, x und y reell sind, folgt durch Koeffizientenvergleich $Ax = \lambda x$ und $Ay = \lambda y$. Man beachte, dass x und y nicht beide Null sein können, da sonst auch $z = 0$ wäre und ein Widerspruch zur Annahme das z ein Eigenvektor ist. □

Lemma 8. *Sei $A \in \mathbb{R}^{n\times n}$ eine symmetrische Matrix und sei v ein Eigenvektor von A. Ist $w \in \mathbb{R}^n$ orthogonal zu v, dann ist auch Aw orthogonal zu v. Ist W ein Untervektorraum von $\mathbb{R}^n$ mit $A(W) \subset W$, dann ist auch $A(W^\perp) \subset W^\perp$.*

Beweis. Aw ist orthogonal zu v, denn

$$\langle Aw, v\rangle = \langle w, Av\rangle = \langle w, \lambda v\rangle = \lambda\langle w, v\rangle = 0\,.$$

Sei nun $A(W) \subset W$ und $u \in W^\perp$ beliebig. Für alle $w \in W$ gilt dann $\langle Au, w\rangle = \langle u, Aw\rangle = 0$, da nach Voraussetzung $Aw \in W$. Somit folgt $Au \in W^\perp$. □

Satz 9. *Sei V ein n-dimensionaler euklidischer Vektorraum und $A\colon V \to V$ ein selbstadjungierter Endomorphismus. Dann existiert eine Orthonormalbasis von V aus Eigenvektoren von A.*

Beweis. Nach Satz 7 gibt es einen reellen Eigenvektor v_1 von A. Sei $W = \operatorname{span}(v)$ der von v_1 aufgespannte Untervektorraum. Dann gilt $A(W) \subset W$ und somit nach Lemma 8 auch $A(W^\perp) \subset W^\perp$. Es ist $\dim W^\perp = n-1$. Einschränken von A auf $W^\perp$ ergibt eine symmetrische lineare Abbildung von $W^\perp \to W^\perp$. Durch wiederholen des Vorganges erhalten wir mittel Induktion eine Basis $v_2, \ldots, v_n$ von $W^\perp$ bestehend aus Eigenvektoren. $v_1, \ldots, v_n$ ist dann eine orthogonale Basis von V bestehend aus Eigenvektoren. Durch normieren wird diese zu einer Orthonormalbasis. □

Satz 10. *Sei $A \in \mathbb{R}^{n\times n}$ eine symmetrische Matrix und v_i und v_j zwei Eigenvektoren zu unterschiedlichen Eigenwerten $\lambda_i \neq \lambda_j$. Dann sind v_i und v_j orthogonal.*

Beweis. Aufgrund der Symmetrie von A gilt:

$$\lambda_i\langle v_i, v_j\rangle = \langle \lambda_i v_i, v_j\rangle = \langle Av_i, v_j\rangle = \langle v_i, Av_j\rangle = \langle v_i, \lambda_j v_j\rangle = \lambda_j\langle v_i, v_j\rangle \tag{4.3}$$

Somit ist $(\lambda_i - \lambda_j)\langle v_i, v_j\rangle = 0$. Da $\lambda_i \neq \lambda_j$ vorausgesetzt war, folgt $\langle v_i, v_j\rangle = 0$. □

Satz 11. *Sei $A \in \mathbb{R}^{n\times n}$ eine symmetrische Matrix. Dann gibt es eine orthogonale Matrix $Q \in O(n)$, so dass*

$$Q^TAQ = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)\,. \tag{4.4}$$

Mit anderen Worten: Jede symmetrische Matrix ist diagonalisierbar und es gibt eine Orthonormalbasis aus Eigenvektoren.

Beweis. TBD □

Beweis. Sei $A = Q\Lambda Q^T$ mit $\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$. Für $x = Qy$, gilt dann:

$$x^TAx = (Q^Tx)^T\Lambda Q^Tx = y^T\Lambda y = \sum_{i=1}^{n} \lambda_i y_i^2 \tag{4.5}$$

Dieser Ausdruck ist genau dann für alle $x \neq 0$ positiv (bzw. negativ), falls alle λ_i positiv (bzw. negativ) sind. □

Satz 12. *Sei $A \in \mathbb{R}^{n\times n}$ eine symmetrische Matrix und seien $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$ die nach Größe geordneten Eigenwerte von A. Dann gilt:*

$$\lambda_1 \, x^T x \geq x^T A x \geq \lambda_n \, x^T x \qquad \text{für alle } x \in \mathbb{R}^n \tag{4.6}$$

$$\lambda_{\max} = \lambda_1 = \max_{x\neq 0} \frac{x^T A x}{x^T x} = \max_{\|x\|=1} x^T A x \tag{4.7}$$

$$\lambda_{\min} = \lambda_n = \min_{x\neq 0} \frac{x^T A x}{x^T x} = \min_{\|x\|=1} x^T A x \tag{4.8}$$

Beweis. Betrachten zunächst den Fall, dass A eine Diagonalmatrix Λ mit absteigend sortierten Diagonalelementen $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$ ist. Dann gilt

$$x^T \Lambda x = \begin{pmatrix} x_1 & \cdots & x_n \end{pmatrix} \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_n \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \lambda_1 x_1^2 + \ldots \lambda_n x_n^2 = \sum_{i=1}^n \lambda_i x_i^2$$

und wir können abschätzen:

$$x^T \Lambda x = \sum_{i=1}^n \lambda_i x_i^2 \leq \lambda_1 \sum_{i=1}^n x_i^2 = \lambda_1 \, x^T x$$

Daraus erhalten wir $\max_{\|x\|=1} x^T \Lambda x \leq \lambda_1 x^T x$. Da für den Vektor $x = (1, 0, \ldots, 0)^T$ Gleichheit angenommen wird, folgt:

$$\max_{\|x\|=1} x^T \Lambda x = \lambda_1$$

Sei nun A beliebig und $A = Q\Lambda Q^T$ eine Eigenzerlegung von A. Dann gilt:

$$x^T A x = x^T Q \Lambda Q^T x = \left(Q^T x\right)^T \Lambda Q^T x$$

Sei nun $y = Q^T x$. Da $\|y\| = \|Q^T x\| = \|x\|$ und Q vollen Rang hat, können wir anstatt über alle x zu maximieren auch über alle y maximieren:

$$\max_{\|x\|=1} x^T A x = \max_{\|x\|=1} \left(Q^T x\right)^T \Lambda Q^T x = \max_{\|y\|=1} y^T \Lambda y = \lambda_1 \qquad \square$$

Eine symmetrische Matrix heißt *positiv definit* falls $x^T A x > 0$ für alle $x \neq 0$ und $x \in \mathbb{R}^n$ ist.

Satz 13. *Eine symmetrische Matrix A ist genau dann positiv (negativ) definit, wenn alle Eigenwerte von A positiv (negativ) sind.*

4.4. Direkte Vektoriteration (von-Mises-Verfahren)

Die nach Richard von Mises benannte Vektoriteration ist ein Verfahren um den Eigenvektor zum größten Eigenwert zu bestimmen. Die Grundidee ist sehr einfach: Gegeben sei eine eine Matrix $A \in \mathbb{R}^{n\times n}$ und ein beliebiger Startwert $x^{(0)} \in \mathbb{R}^n$. Wir betrachten die Folge:

$$x^{(k+1)} := A x^{(k)} \quad \text{für } k = 1, 2, \ldots$$

Ist nun λ ein einfacher Eigenwert, der betragsmäßig echt größer ist als alle anderen Eigenwerte, so lässt sich vermuten, dass dieser sich bei der obigen Iteration durchsetzt und $x^{(k)}$ somit gegen den größten Eigenvektor konvergiert.

Satz 14. *Sei $A \in \mathbb{R}^n$ eine diagonalisierbare Matrix mit den Eigenwerten $\lambda_1, \ldots, \lambda_n$ und $|\lambda_1| > |\lambda_i|$ für $i > 1$. Sei außerdem $y^{(0)} \in \mathbb{R}^n$ ein Vektor, welcher nicht orthogonal auf dem Eigenraum von λ_1 steht. Dann konvergiert die Folge*

$$y^{(k+1)} = \frac{x^{(k+1)}}{\|x^{(k+1)}\|} \quad \textit{mit} \quad x^{(k+1)} = Ay^{(k)}$$

gegen einen normierten Eigenvektor von A zum Eigenwert λ_1.

Beweis. Sei $v_1, \ldots, v_n$ eine Orthonormalbasis aus Eigenvektoren von A mit $Av_i = \lambda v_i$. Dann ist $y_0 = \sum_{i=1}^n \alpha_i v_i$ mit $\alpha_i = \langle x_0, v_i \rangle$. Nach Voraussetzung ist $\alpha_1 \neq 0$. Mittels Induktion ergibt sich

$$y^{(k)} = \frac{A^{(k)} y^{(0)}}{\|A^{(k)} y^{(0)}\|}$$

und wir erhalten:

$$x^{(k)} = A^k y^{(0)} = \sum_{i=1}^n \alpha_i A^k v_i = \sum_{i=1}^n \alpha_i \lambda_i^k v_i = \lambda_1^k \Bigg(\alpha_1 v_1 + \underbrace{\sum_{i=2}^n \alpha_i \left(\frac{\lambda_i}{\lambda_1} \right)^k v_i}_{=:r(k)} \Bigg)$$

Da für $i > 1$ nach Voraussetzung $|\lambda_i/\lambda_1| < 1$ ist, folgt $\lim_{k\to\infty} r(k) = 0$. Somit erhalten wir:

$$\lim_{k\to\infty} \operatorname{sgn}(\lambda_1)\, y^{(k)} = \lim_{k\to\infty} \operatorname{sgn}(\lambda_1) \frac{x^{(k)}}{\|x^{(k)}\|} = \lim_{k\to\infty} \operatorname{sgn}(\lambda_1) \frac{\lambda_1^k \alpha_1 v_1}{|\lambda_1|^k |\alpha_1| \|v_1\|} = \operatorname{sgn}(\alpha_1) v_1$$

Man beachte: $y^{(k)} \to v_1$ gilt nur bis auf einen Vorzeichenfaktor! □

Üblicherweise wählt man für den Startwert $y^{(0)} = (1, \ldots, 1)^T$. Sollte der unwahrscheinliche Fall $\alpha_1 = 0$ auftreten, so konvergiert das Verfahren zunächst gegen einen Eigenvektor zum betragsmäßig zweitgrößten Eigenwert (sofern dieser einfach ist). Aufgrund von Rundungsfehlern wird die Komponente in Richtung v_1 nicht lange Null bleiben und das Verfahren schließlich doch gegen einen Eigenvektor zum betragsmäßig größten Eigenwert konvergieren.

Die Konvergenz wird durch den Faktor $\max_{i=2,\ldots,n} |\lambda_i/\lambda_1|$ bestimmt. Ist λ_1 nur schwach dominant, so ist die Konvergenz sehr langsam.

5. Singulärwertzerlegung

5.1. Motivation

Wir beschäftigen uns nun mit dem Problem, Gesichter in Bildern zu erkennen. Dabei möchte man zwei Fragen beantworten können:

1. Zeigt das Bild ein Gesicht?

2. Falls das Bild ein Gesicht zeigt, lässt sich das Gesicht einem aus einer Datenbank von Bildern bekannten Gesicht zuordnen.

Wir nehmen an, die Bilder sind als Grauwerte pro Pixel angegeben. Alle Bilder werden auf eine einheitliche Höhe h und Breite b gebracht. Dann lassen sich die Bilder als *Punkte* in einem n-dimensionalen Vektorraum auffassen, wobei $n = h \cdot b$ das Produkt von Höhe h und Breite b ist. Ein Bild ist also repräsentiert als $\mathbf{g} \in \mathbb{R}^n$. Wichtig ist, die Anordnung der Pixel im Vektor für alle Bilder gleich zu wählen, also eine feste Zuordnung der Bildpunkte zu den Vektorindizes zu vereinbaren.

Eine Datenbank von Bildern ist dann gegeben als eine Reihe von m Vektoren $\mathbf{g}_0, \ldots, \mathbf{g}_{m-1}$. Wir können diese Bilder auch als Matrix $\mathbf{G} = (\mathbf{g}_0, \ldots, \mathbf{g}_{m-1}) \in \mathbb{R}^{n \times m}$ schreiben.

Ein einfaches Verfahren, um zu beurteilen ob ein Bild $\mathbf{g}$ ein Gesicht ist und wenn ja welches, basiert auf der Verwendung von Vektornormen. Dazu bestimmen wir den kleinsten Abstand von $\mathbf{g}$ zu einem der Gesichtsbilder

$$\underset{i \in [0,\ldots,m-1]}{\arg\min} \|\mathbf{g} - \mathbf{g}_i\|. \tag{5.1}$$

Vergleicht man diesen Abstand mit dem Abstand zu anderen Gesichtsbildern j „in der Nähe“ $\|\mathbf{g}_i - \mathbf{g}_j\|$ so weiß man, ob $\mathbf{g}$ das Bild eines Gesichtes ist und wie groß die Ähnlichkeit von $\mathbf{g}$ zu dem Gesicht im Bild mit Index i ist.

Allerdings ist die Berechnung aller Abstände $\|\mathbf{g} - \mathbf{g}_i\|$ aufwendig, denn die Vektoren haben in unserem Anwendungsfall die „große“ Länge $n = b \cdot h$. Es stellt sich die Frage, ob man diese Operation nicht effizienter ausführen kann. Warum besteht dafür Hoffnung? Wir beobachten, dass in der Anwendung n meist viel größer als m sein wird, die Vektoren $\mathbf{g}_i$ also in einem höchstens m-dimensionalen Unterraum liegen. Darüber hinaus, haben alle Bilder in der Datenbank einen sehr ähnlichen Inhalt. Wir könnten also vermuten, dass sie alle von einer unbekannten aber einfachen Funktion f erzeugt werden, die das Konzept „Gesichtsbild“ beschreibt.

Ein gängiges Verfahren ist es hierfür eine lineare Funktion zu verwenden. Wir hatten diese Idee bereits im Kontext der linearen Regression kennengelernt, allerdings war dort das Bild der Funktion eindimensional. Jetzt ist $f : \mathbb{R}^o \mapsto \mathbb{R}^n$ offenbar eine Funktion die von o-dimensionalen Vektoren auf die n-dimensionalen Bilder abbildet. Dabei wird die Dimension o der linearen Funktion zumindest nicht größer als m sein, denn es gibt ja nur m verschiedene Gesichtsbilder.

Die Funktion setzt sich also zusammen aus n linearen Funktionen mit jeweils(!) o linearen Koeffizienten und jeweils einem Koeffizienten für den konstanten Teil:

$$\begin{aligned} f(\mathbf{x}) &= \begin{pmatrix} c_{0,0}x_0 + c_{0,1}x_1 + \dots c_{0,o-1}x_{o-1} + b_0 \\ c_{1,0}x_0 + c_{1,1}x_1 + \dots c_{1,o-1}x_{o-1} + b_1 \\ \vdots \\ c_{n-1,0}x_0 + c_{n-1,1}x_1 + \dots c_{n-1,o-1}x_{o-1} + b_{n-1} \end{pmatrix} \\ &= \begin{pmatrix} c_{0,0} & c_{0,1} & \dots & c_{0,o-1} \\ c_{1,0} & c_{1,1} & \dots & c_{1,o-1} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n-1,0} & c_{n-1,1} & \dots & c_{n-1,o-1} \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_{o-1} \end{pmatrix} + \begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_{n-1} \end{pmatrix} \\ &= \mathbf{C}\mathbf{x} + \mathbf{b}, \quad \mathbf{C} \in \mathbb{R}^{n \times o}, \mathbf{b} \in \mathbb{R}^n. \end{aligned} \tag{5.2}$$

In unserem Kontext hieße das, für jedes Gesichtsbild gibt es einen Vektor $\mathbf{x}_i \in \mathbb{R}^o$ so dass $f(\mathbf{x}_i) \approx \mathbf{g}_i$ ist. Wir überlegen uns nun, unter welchen Umständen wir durch diese Darstellung sparen können. Wir wollten Differenzen von Vektoren ausrechnen, also

$$\|f(\mathbf{x}) - f(\mathbf{x}_i)\| = \|\mathbf{C}\mathbf{x} + \mathbf{b} - (\mathbf{C}\mathbf{x}_i + \mathbf{b})\| = \|\mathbf{C}(\mathbf{x} - \mathbf{x}_i)\|. \tag{5.3}$$

Wir sehen also, dass wir nicht die Differenzen der Bilder $\mathbf{g} \in \mathbb{R}^n$ sondern der Repräsentationen $\mathbf{x} \in \mathbb{R}^o$ betrachten. Durch die Multiplikation mit einer Matrix würde das den Aufwand aber erhöhen. Außerdem müssen wir zunächst aus dem Bild $\mathbf{g}$ die Repräsentation $\mathbf{x}$ gewinnen. Wie ist das möglich? Die Gleichung $\mathbf{C}\mathbf{x} + \mathbf{b} \approx \mathbf{g}$ legt nahe, dass wir die Normalengleichung verwenden sollten, also

$$\mathbf{C}^\mathsf{T}\mathbf{C}\mathbf{x} + \mathbf{C}^\mathsf{T}\mathbf{b} = \mathbf{C}^\mathsf{T}\mathbf{g} \quad \Longleftrightarrow \quad \mathbf{x} = \left(\mathbf{C}^\mathsf{T}\mathbf{C}\right)^{-1}\mathbf{C}^\mathsf{T}(\mathbf{g} - \mathbf{b}) \tag{5.4}$$

Zwar könnten wir die Matrix $\left(\mathbf{C}^\mathsf{T}\mathbf{C}\right)^{-1}\mathbf{C}$ einmal ausrechnen, und dann auf beliebige Bilder anwenden, aber wir wissen bereits, dass dies aus numerischer Sicht keine gute Idee ist. Eine Überlegung „rettet“ unseren Ansatz: die Matrix $\mathbf{C}$ beschreibt einen linearen Unterraum. Die konkrete Wahl der Basis hat keinen Einfluss auf die Dimension o. Die Lösung besteht nun darin, die Spalten von $\mathbf{C}$ orthogonal und normiert zu wählen. Warum hilft das? Da wir in $\mathbf{C}^\mathsf{T}\mathbf{C}$ die Spalten miteinander multiplizieren erhalten wir die Einheitsmatrix! Damit wird aus dem Problem die Repräsentation $\mathbf{x}$ von $\mathbf{g}$ zu finden einfach

$$\mathbf{x} = \mathbf{C}^\mathsf{T}(\mathbf{g} - \mathbf{b}), \tag{5.5}$$

sprich man zieht vom Bild $\mathbf{g}$ den Vektor der Konstanten $\mathbf{b}$ ab und multipliziert dann mit (der gut konditionierten) Matrix $\mathbf{C}^\mathsf{T}$.

Wichtiger ist aber noch, was mit der Abstandsmessung passiert: wir hatten gesehen, dass der Abstand sich ausdrücken lässt als $\|\mathbf{C}(\mathbf{x} - \mathbf{x}_i)\|$. Für das Quadrat des Abstandes sehen wir

$$\begin{aligned} \|\mathbf{C}(\mathbf{x} - \mathbf{x}_i)\|^2 &= \left(\mathbf{C}(\mathbf{x} - \mathbf{x}_i)\right)^\mathsf{T}\left(\mathbf{C}(\mathbf{x} - \mathbf{x}_i)\right) \\ &= (\mathbf{x} - \mathbf{x}_i)^\mathsf{T}\mathbf{C}^\mathsf{T}\mathbf{C}(\mathbf{x} - \mathbf{x}_i) \\ &= (\mathbf{x} - \mathbf{x}_i)^\mathsf{T}(\mathbf{x} - \mathbf{x}_i) \\ &= \|(\mathbf{x} - \mathbf{x}_i)\|^2. \end{aligned} \tag{5.6}$$

Wir können also statt der Abstände zwischen $\mathbf{g}$ und $\mathbf{g}_i$ direkt den Abstand zwischen $\mathbf{x}$ und $\mathbf{x}_i$ ausrechnen und erhalten das gleiche Ergebnis! Dabei sparen wir jetzt erheblich, denn wir haben $o \ll n$.

Natürlich funktioniert dieser Ansatz nur dann, wenn tatsächlich $f(\mathbf{x}_i) \approx \mathbf{g}_i$, also die Bilder gut durch die lineare Funktion beschrieben werden. Wir betrachten dazu die Residuen

$$\mathbf{r}_i = f(\mathbf{x}_i) - \mathbf{g}_i = \mathbf{C}\mathbf{x}_i + \mathbf{b} - \mathbf{g}_i = \mathbf{C}\mathbf{C}^\mathsf{T}(\mathbf{g}_i - \mathbf{b}) - (\mathbf{g}_i - \mathbf{b}) = \left(\mathbf{C}\mathbf{C}^\mathsf{T} - \mathbf{I}\right)(\mathbf{g}_i - \mathbf{b}) \quad (5.7)$$

und werden ein Verfahren herleiten, mit dem wir für jede Menge von Vektoren $\mathbf{G} = (\mathbf{g}_0, \ldots, \mathbf{g}_{m-1})$ eine Matrix $\mathbf{C}$ mit orthogonalen Spalten und einen Vektor $\mathbf{b}$ bestimmen, die die Residuen minimieren.

5.2. Orthogonale Regression

Seien Vektoren $\mathbf{g}_0, \ldots, \mathbf{g}_{m-1} \in \mathbb{R}^n$ gegeben. Wir betrachten einen Vektor $\mathbf{b}$ sowie eine Matrix $\mathbf{C} = (\mathbf{c}_0, \ldots, \mathbf{c}_{o-1}) \in \mathbb{R}^{n \times o}, o \leq n$, deren Spalten normiert sowie paarweise orthogonal sind, also $\mathbf{C}^\mathsf{T}\mathbf{C} = \mathbf{I}$. Wir wollen $\mathbf{b}, \mathbf{C}$ so wählen, dass die Summe der quadrierten Längen der Residuen minimal ist:

$$\begin{aligned}
\underset{\mathbf{b},\mathbf{C}^\mathsf{T}\mathbf{C}=\mathbf{I}}{\arg\min} \sum_{i=0}^{m-1} \mathbf{r}_i^\mathsf{T}\mathbf{r} &= \underset{\mathbf{b},\mathbf{C}^\mathsf{T}\mathbf{C}=\mathbf{I}}{\arg\min} \sum_{i=0}^{m-1} (\mathbf{g}_i - \mathbf{b})^\mathsf{T} \left(\mathbf{C}\mathbf{C}^\mathsf{T} - \mathbf{I}\right)^\mathsf{T} \left(\mathbf{C}\mathbf{C}^\mathsf{T} - \mathbf{I}\right)(\mathbf{g}_i - \mathbf{b}) \\
&= \underset{\mathbf{b},\mathbf{C}^\mathsf{T}\mathbf{C}=\mathbf{I}}{\arg\min} \sum_{i=0}^{m-1} (\mathbf{g}_i - \mathbf{b})^\mathsf{T} \left(\mathbf{C}\mathbf{C}^\mathsf{T}\mathbf{C}\mathbf{C}^\mathsf{T} - \mathbf{C}\mathbf{C}^\mathsf{T} - \mathbf{C}\mathbf{C}^\mathsf{T} + \mathbf{I}\right)(\mathbf{g}_i - \mathbf{b}) \quad (5.8) \\
&= \underset{\mathbf{b},\mathbf{C}^\mathsf{T}\mathbf{C}=\mathbf{I}}{\arg\min} \sum_{i=0}^{m-1} (\mathbf{g}_i - \mathbf{b})^\mathsf{T} \left(\mathbf{I} - \mathbf{C}\mathbf{C}^\mathsf{T}\right)(\mathbf{g}_i - \mathbf{b})
\end{aligned}$$

Das Problem ist für $n = 2$ (also Punkte in der Ebene) dargestellt. Wir suchen also wie bei der linearen Regression eine Gerade, die die Punkte repräsentiert. Anders als zuvor, betrachten wir nicht den Fehler in Richtung einer der Koordinatenachsen. Vielmehr messen wir den Fehler durch Projektion (Produkt mit $\mathbf{C}^\mathsf{T}$!) auf die Gerade, also als Abstand zur Geraden. Dies macht das Problem erheblich schwieriger, denn anders als bei der linearen Regression ist die Richtung in der wir den Fehler messen nicht bekannt. Da der (kleinste) Abstand zur Geraden orthogonal steht, nennt man das Verfahren im deutschen Sprachraum *orthogonale Regression*. Die englische Bezeichnung *Total Least Squares* bezieht sich darauf, dass die Abstände zur Geraden minimiert werden.

Bevor wir den Vektor $\mathbf{b}$ und anschließend die Matrix $\mathbf{C}$ bestimmen, erinnern wir an die Eigenzerlegung und beschäftigen uns mit den Eigenwerten von Matrizen der Form $\mathbf{A}^\mathsf{T}\mathbf{A}$.

5.2.1. Eigenzerlegungen

Aus der linearen Algebra (siehe auch Anhang) wissen wir, dass jede symmetrische Matrix $\mathbf{M}^\mathsf{T} = \mathbf{M} \in \mathbb{R}^{n \times n}$ orthogonale Eigenvektoren $\mathbf{q}_i$ und reelle Eigenwerte λ_i hat. Insbesondere kann man $\mathbf{M}$ durch Übergang in die Eigenbasis $\mathbf{Q} = (\mathbf{q}_0, \ldots, \mathbf{q}_{n-1})$ *diagonalisieren*:

$$\mathbf{M}^\mathsf{T} = \mathbf{M} \quad \Longrightarrow \quad \mathbf{M} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\mathsf{T} \quad \Longleftrightarrow \quad \mathbf{Q}^\mathsf{T}\mathbf{M}\mathbf{Q} = \Lambda = \begin{pmatrix} \lambda_0 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \lambda_{n-1} \end{pmatrix}. \quad (5.9)$$

Dabei lassen sich die Spalten von $\mathbf{Q}$ so vertauschen, dass die Eigenwerte absteigend sortiert sind, also $\lambda_0 \geq \ldots \geq \lambda_{n-1}$.

Wichtig für uns sind symmetrische PSD-Matrizen der Form $\mathbf{A}^\mathsf{T}\mathbf{A}$ bzw. $\mathbf{A}\mathbf{A}^\mathsf{T}$ für rechteckige Matrizen $\mathbf{A} \in \mathbb{R}^{n\times m}$. Wir stellen fest, dass die Diagonalmatrix der Eigenwerte eine PSD-Matrix sein muss

$$\mathbf{x}^\mathsf{T}\mathbf{A}^\mathsf{T}\mathbf{A}\mathbf{x} \geq 0 \quad \Longrightarrow \quad \mathbf{x}^\mathsf{T}\mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\mathsf{T}\mathbf{x} = \mathbf{y}^\mathsf{T}\Lambda\mathbf{y} \geq 0 \tag{5.10}$$

und dadurch alle Eigenwerte nicht-negativ sind, mithin $\lambda_{n-1} \geq 0$.

Ferner ist jeder Eigenwert $\lambda_i > 0$ von $\mathbf{A}^\mathsf{T}\mathbf{A}$ auch ein Eigenwert von $\mathbf{A}\mathbf{A}^\mathsf{T}$:

$$\mathbf{A}^\mathsf{T}\mathbf{A}\mathbf{x} = \lambda\mathbf{x} \quad \Longrightarrow \quad \mathbf{A}\mathbf{A}^\mathsf{T}\,\mathbf{A}\mathbf{x} = \lambda\mathbf{A}\mathbf{x}. \tag{5.11}$$

Das heißt also die $m \times m$ Matrix $\mathbf{A}^\mathsf{T}\mathbf{A}$ und die $n \times n$ Matrix $\mathbf{A}\mathbf{A}^\mathsf{T}$ haben die selben von Null verschiedenen Eigenwerte. Ist $n > m$, dann sind also die ersten m Eigenwerte identisch, und die verbleibenden $n - m$ Eigenwerte der $n \times n$ Matrix $\mathbf{A}\mathbf{A}^\mathsf{T}$ alle Null.

Damit kennen wir die Eigenwerte der Matrix $\mathbf{C}\mathbf{C}^\mathsf{T}$. Wir wollten $\mathbf{C}$ so wählen, dass $\mathbf{C}^\mathsf{T}\mathbf{C} = \mathbf{I}$ gilt. Das heißt $\mathbf{C}^\mathsf{T}\mathbf{C}$ hat den (m-fachen) Eigenwert 1. Damit hat die Matrix $\mathbf{C}\mathbf{C}^\mathsf{T}$ also nur die Eigenwerte 1 und 0. Schreiben wir $\mathbf{C}\mathbf{C}^\mathsf{T}$ in Eigenbasis so sehen wir das

$$\mathbf{I} - \mathbf{C}\mathbf{C}^\mathsf{T} = \mathbf{I} - \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\mathsf{T} = \mathbf{Q}\mathbf{I}\mathbf{Q}^\mathsf{T} - \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\mathsf{T} = \mathbf{Q}(\mathbf{I} - \mathbf{\Lambda})\mathbf{Q}^\mathsf{T} \tag{5.12}$$

die gleiche Eigenbasis wie $\mathbf{C}\mathbf{C}^\mathsf{T}$ besitzt und ebenfalls nur die Eigenwerte 0 und 1 hat. Damit ist sie auch eine symmetrische PSD-Matrix.

5.2.2. Bestimmung der Konstanten, Schwerpunkt

Wir wollen also das Minimum für $\mathbf{b}$ in dem Minimierungsproblem

$$\underset{\mathbf{b},\mathbf{C}^\mathsf{T}\mathbf{C}=\mathbf{I}}{\arg\min} \sum_{i=0}^{m-1} (\mathbf{g}_i - \mathbf{b})^\mathsf{T} \left(\mathbf{I} - \mathbf{C}\mathbf{C}^\mathsf{T}\right) (\mathbf{g}_i - \mathbf{b}) \tag{5.13}$$

finden. Nach unseren Vorüberlegungen ist $\mathbf{I} - \mathbf{C}\mathbf{C}^\mathsf{T}$ eine PSD-Matrix, die wir mit $\mathbf{M}$ abkürzen.

Zunächst nehmen wir an, dass die Punkte $\mathbf{g}_i$ *mittelwertfrei* sind, d.h. $\sum_i \mathbf{g}_i = 0$. Wir überzeugen uns davon, dass in diesem Fall $\mathbf{b} = \mathbf{0}$ die beste Wahl ist:

$$\begin{aligned}
\sum_{i=0}^{m-1} (\mathbf{g}_i - \mathbf{b})^\mathsf{T}\mathbf{M}(\mathbf{g}_i - \mathbf{b}) &= \sum_{i=0}^{m-1} (\mathbf{g}_i^\mathsf{T}\mathbf{M}\mathbf{g}_i - \mathbf{b}^\mathsf{T}\mathbf{M}\mathbf{g}_i - \mathbf{g}_i^\mathsf{T}\mathbf{M}\mathbf{b} + \mathbf{b}^\mathsf{T}\mathbf{M}\mathbf{b}) \\
&= \sum_{i=0}^{m-1} \mathbf{g}_i^\mathsf{T}\mathbf{M}\mathbf{g}_i - \mathbf{b}^\mathsf{T}\mathbf{M}\left(\sum_{i=0}^{m-1} \mathbf{g}_i\right) - \left(\sum_{i=0}^{m-1} \mathbf{g}_i\right)^\mathsf{T} \mathbf{M}\mathbf{b} + m\mathbf{b}^\mathsf{T}\mathbf{M}\mathbf{b} \\
&= \sum_{i=0}^{m-1} \mathbf{g}_i^\mathsf{T}\mathbf{M}\mathbf{g}_i + m\mathbf{b}^\mathsf{T}\mathbf{M}\mathbf{b}
\end{aligned} \tag{5.14}$$

Den kleinstmöglichen Wert erreichen wir für die Wahl $\mathbf{b} = \mathbf{0}$ weil $\mathbf{M}$ eine PSD-Matrix ist.

Was aber ist, wenn die Bedingung $\sum_i \mathbf{g}_i = 0$ nicht erfüllt ist? Dann verschieben wir alle $\mathbf{g}_i$ um $\mathbf{b}$, verwenden also statt $\mathbf{g}_i$ den Punkt $\mathbf{g}_i - \mathbf{b}$. Wir interpretieren hier also $\mathbf{b}$ als eine Verschiebung mit dem Ziel die Punkte $\mathbf{g}_i$ mittelwertfrei zu stellen. Die gesuchte Verschiebung $\mathbf{b}$ muss erfüllen

$$\sum_{i=0}^{m-1} (\mathbf{g}_i - \mathbf{b}) = 0 \quad \Longleftrightarrow \quad \mathbf{b} = \frac{1}{m}\sum_{i=0}^{m-1} \mathbf{g}_i. \tag{5.15}$$

Die optimale Wahl für $\mathbf{b}$ ist der *Schwerpunkt* der Bilder $\mathbf{g}_i$. Diese Wahl gilt unabhängig von $\mathbf{C}$, solange $\mathbf{I} - \mathbf{C}\mathbf{C}^\mathsf{T}$ eine PSD-Matrix ist. Dies ist durch unsere Forderung nach normierten paarweise orthogonalen Spalten von $\mathbf{C}$ sichergestellt.

5.2.3. Bestimmung der Basis

Für die folgenden Überlegungen ersetzen wir die Bilder $\mathbf{g}_i$ durch die mittelwertfreien Vektoren $\mathbf{y}_i = \mathbf{g}_i - \mathbf{b}$. Nach der Variablentransformation betrachten wir jetzt die Minimierung von

$$\underset{\mathbf{C}^\mathsf{T}\mathbf{C}=\mathbf{I}}{\arg\min} \sum_{i=0}^{m-1} \mathbf{y}_i^\mathsf{T} \left(\mathbf{I} - \mathbf{C}\mathbf{C}^\mathsf{T}\right) \mathbf{y}_i = \underset{\mathbf{C}^\mathsf{T}\mathbf{C}=\mathbf{I}}{\arg\min} \sum_{i=0}^{m-1} \mathbf{y}_i^\mathsf{T}\mathbf{y}_i - \mathbf{y}_i^\mathsf{T}\mathbf{C}\mathbf{C}^\mathsf{T}\mathbf{y}_i. \tag{5.16}$$

Da die Vektoren $\mathbf{y}_i$ fest sind, brauchen wir den Teil $\sum_i \mathbf{y}_i^\mathsf{T}\mathbf{y}_i$ nicht zu betrachten. Aus der Minimierung des negativen zweiten Terms machen wir eine Maximierung mit umgekehrtem Vorzeichen:

$$\underset{\mathbf{C}^\mathsf{T}\mathbf{C}=\mathbf{I}}{\arg\max} \sum_{i=0}^{m-1} \mathbf{y}_i^\mathsf{T}\mathbf{C}\mathbf{C}^\mathsf{T}\mathbf{y}_i. \tag{5.17}$$

Wir beginnen mit der Bestimmung einer Matrix $\mathbf{C}$ mit nur einer Spalte, also dem Vektor $\mathbf{c}_0$. Von der Forderung $\mathbf{C}^\mathsf{T}\mathbf{C} = \mathbf{I}$ bleibt dann nur noch $1 = \mathbf{c}_0^\mathsf{T}\mathbf{c}_0 = \|\mathbf{c}_0\|$ übrig. Es ergibt sich

$$\underset{\mathbf{c}_0^\mathsf{T}\mathbf{c}_0=1}{\arg\max} \sum_{i=0}^{m-1} \mathbf{y}_i^\mathsf{T}\mathbf{c}\mathbf{c}^\mathsf{T}\mathbf{y}_i = \underset{\mathbf{c}_0^\mathsf{T}\mathbf{c}_0=1}{\arg\max} \sum_{i=0}^{m-1} \mathbf{c}^\mathsf{T}\mathbf{y}_i\mathbf{y}_i^\mathsf{T}\mathbf{c} = \underset{\mathbf{c}_0^\mathsf{T}\mathbf{c}_0=1}{\arg\max} \mathbf{c}^\mathsf{T} \left(\sum_{i=0}^{m-1} \mathbf{y}_i\mathbf{y}_i^\mathsf{T}\right) \mathbf{c}. \tag{5.18}$$

Schreiben wir jetzt die Vektoren $\mathbf{y}_i$ in Matrixform, also $\mathbf{Y} = \mathbf{y}_0, \ldots, \mathbf{y}_{m-1}$ dann ist $\sum_{i=0}^{m-1} \mathbf{y}_i\mathbf{y}_i^\mathsf{T} = \mathbf{Y}\mathbf{Y}^\mathsf{T}$ und wir müssen den Ausdruck $\mathbf{c}^\mathsf{T}\mathbf{Y}\mathbf{Y}^\mathsf{T}\mathbf{c}$ maximieren. Wir wissen bereits das $\mathbf{Y}\mathbf{Y}^\mathsf{T}$ eine symmetrische PSD-Matrix ist und schreiben sie in Eigenbasis

$$\mathbf{Y}\mathbf{Y}^\mathsf{T} = \mathbf{Q}\Lambda\mathbf{Q}^\mathsf{T}, \qquad \mathbf{Q}\mathbf{Q}^\mathsf{T} = \mathbf{I}, \Lambda = \begin{pmatrix} \lambda_0 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \lambda_{n-1}, \end{pmatrix} \lambda_0 \geq \cdots \geq \lambda_{n-1} \geq 0. \tag{5.19}$$

Dabei gehen wir davon aus, dass die Eigenwerte geordnet sind, also $\lambda_0 \geq \cdots \geq \lambda_{n-1} \geq 0$. Wir setzen $\mathbf{d}_0 = \mathbf{Q}^\mathsf{T}\mathbf{c}_0$. Da $\mathbf{Q}$ eine orthogonale Matrix ist, sind die Längen von $\mathbf{c}$ und $\mathbf{d}_0$ gleich, also wird aus der Bedingung $\|\mathbf{c}_0\| = 1$ die Bedingung $\|\mathbf{d}_0\| = 1$. Damit erhalten wir

$$\underset{\|\mathbf{c}_0\|=1}{\arg\max}\, \mathbf{c}_0^\mathsf{T}\mathbf{Q}\Lambda\mathbf{Q}^\mathsf{T}\mathbf{c}_0 = \underset{\|\mathbf{d}_0\|=1}{\arg\max}\, \mathbf{d}_0^\mathsf{T}\Lambda\mathbf{d}_0, \tag{5.20}$$

und es gibt sich direkt als beste Wahl $\mathbf{d}_0 = \mathbf{e}_0$ und damit $\mathbf{c}_0 = \mathbf{Q}\mathbf{e}_0 = \mathbf{q}_0$, also der Eigenvektor von $\mathbf{Y}\mathbf{Y}^\mathsf{T}$ zum größten Eigenwert.

Wir fahren fort mit dem zweiten Vektor $\mathbf{c}_1$. Jetzt muss neben $\|\mathbf{c}_1\| = 1$ auch noch die Orthogonalitätsbedingung $\mathbf{c}_0^\mathsf{T}\mathbf{c}_1 = 0$ berücksichtigt werden. Eine Ableitung analog zu der für $\mathbf{c}_0$ führt auf das Optimierungsproblem

$$\underset{\|\mathbf{c}_1\|=1, \mathbf{c}_0^\mathsf{T}\mathbf{c}_1=0}{\arg\max}\, \mathbf{c}_1^\mathsf{T}\mathbf{Q}\Lambda\mathbf{Q}^\mathsf{T}\mathbf{c}_1 \tag{5.21}$$

Man macht nun die Transformation $\mathbf{d}_1 = \mathbf{Q}^\mathsf{T}\mathbf{c}_1$ und erhält neben der Bedingung $\|\mathbf{d}_1\| = 1$ zusätzlich

$$0 = \mathbf{c}_0^\mathsf{T}\mathbf{c}_1 = \mathbf{d}_0^\mathsf{T}\mathbf{Q}^\mathsf{T}\mathbf{Q}\mathbf{d_1} = \mathbf{d}_0^\mathsf{T}\mathbf{d}_1 = \mathbf{e}_0^\mathsf{T}\mathbf{d}_1 = d_{1,0}, \tag{5.22}$$

also, dass die erste Komponente von $\mathbf{d}_1$ Null sein muss. Als Lösung für das Optimierungsproblem

$$\underset{\|\mathbf{d}_1\|=1, d_{1,0}=0}{\arg\max} \mathbf{d}_1^\mathsf{T} \Lambda \mathbf{d}_1 \tag{5.23}$$

ergibt sich damit direkt die Lösung $\mathbf{d}_1 = \mathbf{e}_1$ und damit $\mathbf{c}_1 = \mathbf{Q}\mathbf{e}_1 = \mathbf{q}_1$.

In gleicher Weise kann man für weitere Vektoren $\mathbf{c}_i, i \geq 2$ verfahren und erhält $\mathbf{c}_i = \mathbf{q}_i$. Durch Auswerten der Summe der quadrierten Residuenlängen kann man den Fehler bei der Darstellung als lineare Funktion abschätzen und das Verfahren abbrechen, wenn der Fehler klein genug ist.

5.2.4. Optimaliät der Basis

Wir haben die Spaltenvektoren der Matrix $\mathbf{C}$ nacheinander bestimmt. In der Informatik würden wir ein solches Verfahren *greedy* nennen, denn die Entscheidung ist für den einzelnen Schritt optimal, aber es nicht klar, ob sie auch für die schrittweise gewonnene Matrix $\mathbf{C}$ optimal ist. In der Tat könnten wir für eine fest vorgegebene Dimension o andere Basen mit dem gleichen Fehler finden; aber es gibt auch bei globaler Sicht auf das Problem keine bessere Lösung:

$$\begin{aligned}
\underset{\mathbf{C}^\mathsf{T}\mathbf{C}=\mathbf{I}}{\arg\max} \sum_{i=0}^{m-1} \mathbf{y}_i^\mathsf{T}\mathbf{C}\mathbf{C}^\mathsf{T}\mathbf{y}_i &= \underset{\mathbf{C}^\mathsf{T}\mathbf{C}=\mathbf{I}}{\arg\max} \sum_{i=0}^{m-1} \left(\mathbf{y}_i^\mathsf{T}\mathbf{c}_0, \mathbf{y}_i^\mathsf{T}\mathbf{c}_1, \ldots\right) \begin{pmatrix} \mathbf{c}_0^\mathsf{T}\mathbf{y}_i \\ \mathbf{c}_1^\mathsf{T}\mathbf{y}_i \\ \vdots \end{pmatrix} \\
&= \underset{\mathbf{C}^\mathsf{T}\mathbf{C}=\mathbf{I}}{\arg\max} \sum_{i=0}^{m-1} \mathbf{y}_i^\mathsf{T}\mathbf{c}_0\mathbf{c}_0^\mathsf{T}\mathbf{y}_i + \mathbf{y}_i^\mathsf{T}\mathbf{c}_1\mathbf{c}_1^\mathsf{T}\mathbf{y}_i + \ldots \\
&= \underset{\mathbf{C}^\mathsf{T}\mathbf{C}=\mathbf{I}}{\arg\max} \mathbf{c}_0^\mathsf{T}\mathbf{Y}\mathbf{Y}^\mathsf{T}\mathbf{c}_0 + \mathbf{c}_1^\mathsf{T}\mathbf{Y}\mathbf{Y}^\mathsf{T}\mathbf{c}_1 + \ldots \\
&= \underset{\mathbf{C}^\mathsf{T}\mathbf{C}=\mathbf{I}}{\arg\max} \mathbf{c}_0^\mathsf{T}\mathbf{Q}\Lambda\mathbf{Q}^\mathsf{T}\mathbf{c}_0 + \mathbf{c}_1^\mathsf{T}\mathbf{Q}\Lambda\mathbf{Q}^\mathsf{T}\mathbf{c}_1 + \ldots \\
&= \underset{\mathbf{D}^\mathsf{T}\mathbf{D}=\mathbf{I}}{\arg\max} \mathbf{d}_0^\mathsf{T}\Lambda\mathbf{d}_0 + \mathbf{d}_1^\mathsf{T}\Lambda\mathbf{d}_1 + \ldots, \qquad \mathbf{d}_i = \mathbf{Q}^\mathsf{T}\mathbf{c}_i, \\
&= \underset{\mathbf{D}^\mathsf{T}\mathbf{D}=\mathbf{I}}{\arg\max} \lambda_0(d_{00}^2 + d_{01}^2 + \ldots) + \lambda_1(d_{10}^2 + d_{11}^2 + \ldots) + \ldots
\end{aligned} \tag{5.24}$$

Wir sehen, dass λ_i mit der quadrierten Länge der i-ten *Zeile* von $\mathbf{D}$ multipliziert wird. Wegen $\lambda_0 \geq \lambda_1 \geq \ldots$ dürfen also nur die ersten o Zeilen von $\mathbf{D}$ Einträge ungleich Null haben. Um die Zeilen miteinander zu multiplizieren betrachten wir die Diagonaleinträge von $\mathbf{D}\mathbf{D}^\mathsf{T}$. Wie bereits bemerkt hat diese Matrix wegen $\mathbf{D}^\mathsf{T}\mathbf{D} = \mathbf{I}$ nur Null und Eins als Eigenwerte. Da wir die unteren Zeilen alle Null setzen wollen, ergeben sich die oberen Zeilen zwingend:

$$\mathbf{D}\mathbf{D}^\mathsf{T} = \begin{pmatrix} \mathbf{I}_{o\times p} & \mathbf{0}_{o,n-o} \\ \mathbf{0}_{n-o,o} & \mathbf{0}_{n-o,n-o.} \end{pmatrix} \tag{5.25}$$

Die ersten o Einträge der Vektoren $\mathbf{d}_i$ müssen also eine Orthogonalbasis der ersten o kanonischen Einheitsvektoren sein. Das bedeutet für die Wahl von $\mathbf{C}$, dass die o Vektoren den Unterraum der ersten o Eigenvektoren von $\mathbf{Q}$ aufspannen müssen. Unsere Wahl beim schrittweisen Vorgehen ist also eine mögliche optimale Wahl der o Vektoren in der Basis $\mathbf{C}$. Sie hat den Vorteil, für jede Wahl von weniger als o Vektoren optimal zu sein.

5.2.5. Zusammenfassung

Für gegebene Vektoren $\mathbf{g}_i \in \mathbb{R}^n, i \in [0, \ldots, m-1]$ suchen wir eine alternative Darstellung $\mathbf{x}_i \in \mathbb{R}^o, o \ll n, m$, so dass $f(\mathbf{x}_i) = \mathbf{C}\mathbf{x}_i + \mathbf{b} \approx \mathbf{g}_i$ und $\mathbf{C}^\mathsf{T}\mathbf{C} = \mathbf{I}$. In dieser Darstellung können wir Skalarprodukte und damit auch Abstände zwischen den Vektoren $\mathbf{g}_i$ auf Basis der Repräsentationen $\mathbf{x}_i$ ausrechnen.

Mögliches Vorgehen:

1. Bestimme $\mathbf{b}$ als den Schwerpunkt der Daten $\mathbf{g}_i$.
2. Befreie die Daten vom Mittelwert, rechne mit $\mathbf{y}_i = \mathbf{g}_i - \mathbf{b}$.
3. Erzeuge die Matrix $\mathbf{Y} = (\mathbf{y}_0, \ldots, \mathbf{y}_{m-1})$ und daraus $\mathbf{Y}\mathbf{Y}^\mathsf{T}$.
4. Die ersten o Eigenvektoren von $\mathbf{Y}\mathbf{Y}^\mathsf{T}$ sind die gesuchten o Spalten von $\mathbf{C}$.

Die Eigenwerte könnte man mit *Power-Iteration* (auf deutsch *von-Mises-Verfahren*) erzeugen. Wir lernen im nächten Abschnitt ein Verfahren kennen, dass die gesuchten Vektoren direkt aus der Matrix $\mathbf{Y}$ gewinnt.

5.3. Singulärwertzerlegung

Wir betrachten noch einmal, was wir im zweiten Schritt, also der Bestimmung der neuen Basisvektoren $\mathbf{C}$ gemacht haben. Dazu überspringen wir den Schritt der Mittelwertbefreiung. Wir nehmen also an, wir suchen für die gegebenen Spaltenvektoren der Matrix $\mathbf{Y}$ eine lineare Transformation in einen Unterraum mit kleinerer Dimension. Zur Unterscheidung von der orthogonalen Regression nennen wir die Matrix bzw. Vektoren $\mathbf{C} = (\mathbf{c}_0, \ldots, \mathbf{c}_{o-1})$ jetzt $\mathbf{V} = (\mathbf{v}_0, \ldots, \mathbf{v}_{o-1})$.

Wir hatten die gesuchten Vektoren $\mathbf{v}_i, \|\mathbf{v}_i\| = 1$ durch Maximierung von $\mathbf{v}_i^\mathsf{T}\mathbf{Y}\mathbf{Y}^\mathsf{T}\mathbf{v}_i$ gefunden. Wir setzen jetzt $\mathbf{X} = \mathbf{Y}^\mathsf{T}$ und sehen wegen

$$\mathbf{v}_i^\mathsf{T}\mathbf{Y}\mathbf{Y}^\mathsf{T}\mathbf{v}_i = \mathbf{v}_i^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{X}\mathbf{v}_i = (\mathbf{X}\mathbf{v}_i)^\mathsf{T}\mathbf{X}\mathbf{v}_i = \|\mathbf{X}\mathbf{v}_i\|^2, \tag{5.26}$$

und der Monotonie der Quadratfunktion, dass $\mathbf{v}_i$ auch eine Lösung des Maximierungsproblems

$$\underset{\|\mathbf{v}_i\|=1}{\arg\max} \|\mathbf{X}\mathbf{v}_i\| \tag{5.27}$$

ist. Ferner können wir für jede rechteckige Matrix $\mathbf{X}$ orthogonale und normierte Vektoren $\mathbf{v}_i$ finden, die die Summe der Längen der Produkte $\mathbf{X}\mathbf{v}_i$ maximieren.

Das interessante an den so definierten Vektoren $\mathbf{v}_i$ sind die Eigenschaften dieser Produkte! Sei

$$\sigma_i\mathbf{u}_i = \mathbf{X}\mathbf{v}_i, \qquad \|\mathbf{u}_i\| = 1, \tag{5.28}$$

wobei wir den Ergebnisvektor $\sigma_i\mathbf{u}_i$ gleich in der Form Länge σ_i mal Einheitsvektor $\mathbf{u}_i$ geschrieben haben. Nach Konstruktion sind die Vektoren $\mathbf{v}_i$ paarweise orthogonal. Aber es gilt auch

$$\begin{aligned} \sigma_i\mathbf{u}_i^\mathsf{T}\ \sigma_j\mathbf{u}_j &= (\mathbf{X}\mathbf{v}_i)^\mathsf{T}\mathbf{X}\mathbf{v}_j = \mathbf{v}_i^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{X}\mathbf{v}_j \\ &= \mathbf{v}_i^\mathsf{T}\mathbf{Y}\mathbf{Y}^\mathsf{T}\mathbf{v}_j = \mathbf{v}_i^\mathsf{T}\mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\mathsf{T}\mathbf{v}_j \\ &= \mathbf{d}_i^\mathsf{T}\mathbf{\Lambda}\mathbf{d}_j = \mathbf{e}_i^\mathsf{T}\mathbf{\Lambda}\mathbf{e}_j = \begin{cases} \lambda_i & i = j \\ 0 & i \neq j \end{cases} \end{aligned} \tag{5.29}$$

Die transformierten Vektoren $\mathbf{u}_i$ formen also genau wie die $\mathbf{v}_i$ eine orthogonale Basis. Die Längen der Vektoren ergeben sich wie folgt

$$\|\sigma_i \mathbf{u}_i\|^2 = (\sigma_i \mathbf{u}_i)^\mathsf{T}(\sigma_i \mathbf{u}_i) = \sigma_i^2 \mathbf{u}_i^\mathsf{T}\mathbf{u}_i = \sigma_i^2 = \lambda_i, \tag{5.30}$$

also $\sigma_i = \sqrt{\lambda_i}$, wobei λ_i Eigenwert der Matrix $\mathbf{X}^\mathsf{T}\mathbf{X}$ ist. Aus diesen Längen ergibt sich direkt, dass die normierten Vektoren $\mathbf{u}_i$ eine analoge Maximierungseigenschaft haben, also Lösungen der Gleichung

$$\underset{\|\mathbf{u}_i\|=1}{\arg\max} \left\| \mathbf{u}_i^\mathsf{T}\mathbf{X} \right\| \tag{5.31}$$

sind.

Um zum wesentlichen Ergebnis dieses Abschnitts zu gelangen, bestimmen wir nun statt nur weniger Vektoren $\mathbf{v}_i$ eine vollständige orthonormale Basis des $\mathbb{R}^n$. Das sind die Spalten der orthogonalen Matrix $\mathbf{Q} = \mathbf{V}$. Dabei gibt es offenbar einen Teil der Basis, die den *Kern* von $\mathbf{X}$ aufspannt. Die Transformation liefert uns auch n Vektoren $\sigma_i \mathbf{u}_i \in \mathbb{R}^m$. Für $\sigma_i = 0$ ist der normierte Vektor $\mathbf{u}_i \in \mathbb{R}^m$ allerdings unbestimmt. Wir betrachten daher nur m Vektoren der Form $\mathbf{u}_i$ und wählen die Vektoren für die $\sigma_i = 0$ gilt so, dass sie paarweise orthgonal sind. Dann ist die Matrix dieser Vektoren $\mathbf{U} \in \mathbb{R}^{m \times m}$ eine orthogonal Matrix. Jetzt können wir die Transformation in Matrixform anschaulich darstellen:

$$\boxed{\mathbf{U}} \quad \boxed{\begin{matrix} \mathbf{\Sigma} & \mathbf{0} \end{matrix}} = \boxed{\mathbf{X}} \quad \boxed{\mathbf{V}} \tag{5.32}$$

Nun können wir von rechts mit der inversen der orthogonalen Matrix $\mathbf{V}$ multiplizieren und erhalten die folgende *Faktorisierung* der beliebigen rechteckigen Matrix $\mathbf{X}$:

$$\boxed{\mathbf{X}} = \boxed{\mathbf{U}} \quad \boxed{\begin{matrix} \mathbf{\Sigma} & \mathbf{0} \end{matrix}} \quad \boxed{\mathbf{V}^\mathsf{T}} \tag{5.33}$$

Durch Transposition sehen wir, dass diese Faktorisierung auch funktioniert, wenn die Matrix $\mathbf{X}$ mehr Zeilen (als Spalten) hat. Dann ergibt sich:

$$\boxed{\mathbf{X}} = \boxed{\mathbf{U}} \quad \boxed{\begin{matrix} \mathbf{\Sigma} \\ \mathbf{0} \end{matrix}} \quad \boxed{\mathbf{V}^\mathsf{T}} \tag{5.34}$$

Die Zerlegung dieser Art heißt *Singulärwertzerlegung*. Wie wir gesehen haben, existiert sie für beliebige Matrizen (nicht nur quadratische). Die Faktoren σ_i heißen *Singulärwerte*, die Spalten der Matrix $\mathbf{U}$ heißen *linke Singulärvektoren* und die Spalten der Matrix $\mathbf{V}$ heißen rechte Singulärvektoren. Die Singulärvektoren sind normiert und paarweise orthogonal. Mit anderen Worten, jede Matrix lässt sich in das Produkt einer orthogonalen Matrix, einer Diagonalmatrix (plus Nullmatrix) und einer weiteren orthogonalen Matrix zerlegen.

5.3.1. Reduzierte Form und Rang

Betrachtet man die Produkte $(\mathbf{\Sigma}, \mathbf{0})\mathbf{V}^\mathsf{T}$ für den Fall $m > n$ bzw. $\mathbf{U}(\mathbf{\Sigma}, \mathbf{0})^\mathsf{T}$ für $m < n$ so sieht man, dass hierbei eine Reihe von Nullen entstehen, die für die weiteren Skalarprodukte keine Rolle spielen. Mann kann deshalb auch die Nullmatrix entfernen und erhält dafür rechteckige Matrizen $\mathbf{V}$ bzw. $\mathbf{U}$. Für den Fall $m > n$ ergibt sich

$$\boxed{\mathbf{X}} = \boxed{\mathbf{U}}\ \boxed{\mathbf{\Sigma}}\ \boxed{\mathbf{V}^\mathsf{T}} \tag{5.35}$$

d.h. hier rechnen wir mit einer rechteckigen Matrix $\mathbf{V} \in \mathbb{R}^{n\times m}$ mit orthogonalen Spalten. Für $m < n$ erhält man

$$\boxed{\mathbf{X}} = \boxed{\mathbf{U}}\ \boxed{\mathbf{\Sigma}}\ \boxed{\mathbf{V}^\mathsf{T}} \tag{5.36}$$

und entsprechend eine rechteckige Matrix $\mathbf{U} \in \mathbb{R}^{n\times m}$ mit orthogonalen Spalten.

Bei dieser Reduktion haben wir Teile der Basen in $\mathbf{V}$ bzw. $\mathbf{U}$ entfernt, die im Kern von $\mathbf{X}$ liegen, weil die Matrix $\mathbf{X}$ rechteckig ist und bei weniger Zeilen als Spalten nicht vollen Zeilenrang und bei weniger Spalten als Zeilen nicht vollen Spaltenrang haben. Dies heißt aber nicht, dass alle Singulärwerte größer als Null sein müssen.

5.4. Anwendungen

Die Singulärwertzerlegung hat eine große Zahl von Anwendungen. Diese Anwendungen profitieren von der expliziten Basis von Bild und Kern einer linearen Transformation sowie der expliziten numerischen Information über den Rang in Form der Singulärwerte.

Als Vorüberlegung betrachten wir, wie mit Hilfe der SVD die *Pseudoinverse* bestimmt werden kann – eine Matrix die für invertierbare Matrizen der Inversen entspricht, und für Matrizen ohne Inverse „so nah wie möglich“ an den Eigenschaften einer Inversen ist.

Sei dafür $\mathbf{X} \in \mathbb{R}^{n\times m}$ eine beliebige Matrix mit Rang r. Wir betrachten die (reduzierte) Singulärwertzerlegung $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\mathsf{T}$ und stellen fest, dass für die Singulärwerte wegen des Rangs von $\mathbf{X}$ gelten muss: $\sigma_0 \geq \ldots \geq \sigma_{r-1} > \sigma_r, \ldots = 0$. Wir können die Matrix $\mathbf{\Sigma}$ nicht invertieren – das was einer Inversen in vielerlei Hinsicht am nächsten kommt, ist die *Pseudoinverse*, die im Falle der Diagonalmatrix entsteht, wenn wir nur von Null verschiedene Elemente invertieren:

$$\mathbf{\Sigma}^+ = \mathrm{diag}(\sigma_0^{-1}, \ldots, \sigma_{r-1}^{-1}, 0, \ldots, 0) \tag{5.37}$$

Mit Hilfe dieser Konstruktion und der Singulärwertzerlegung können wir zu jeder Matrix eine Pseudoinverse konstruieren, nämlich

$$\mathbf{X}^+ = \mathbf{V}\mathbf{\Sigma}^+\mathbf{U}^\mathsf{T}. \tag{5.38}$$

Wie erwähnt, entspricht die Pseudoinverse für eine quadratische Matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ mit vollem Rang der üblichen Inversen. In diesem Fall sind $\mathbf{U}, \mathbf{V}$ orthogonale Matrizen und $\mathbf{\Sigma}^+ = \mathbf{\Sigma}^{-1}$ und damit

$$\mathbf{A}\mathbf{A}^+ = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\mathsf{T}\mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{U}^\mathsf{T} = \mathbf{U}\mathbf{\Sigma}\mathbf{\Sigma}^{-1}\mathbf{U}^\mathsf{T} = \mathbf{U}\mathbf{U}^\mathsf{T} = \mathbf{I} \tag{5.39}$$

und analog für $\mathbf{A}^+\mathbf{A}$.

5.4.1. Lineare Gleichungssystem und Ausgleichsrechnung

Wir hatten bisher unterschieden zwischen LGS $\mathbf{Ax} = \mathbf{b}$ bei denen wir genau eine Lösung erwarten und Ausgleichsproblemen $\mathbf{Ax} \approx \mathbf{b}$ für die wir $\mathbf{x}$ so bestimmen wollten, dass das Quadrat der Residuen minimiert wird. In der Praxis ist es für gegebene $\mathbf{A}$ und $\mathbf{b}$ oft allein schon aus numerischen Gründen nicht klar, ob eine Lösung des LGS zu erwarten ist. Vielmehr muss man damit rechnen, dass das LGS keine, genau eine, oder auch unendlich viele Lösungen hat. Die Verwendung der SVD (also der Pseudoinversen) zur Lösung eines LGS der Form $\mathbf{Ax} = \mathbf{b}$ hat die angenehme Eigenschaft, in allen Fällen eine plausible Antwort zu geben:

1. Hat das LGS keine Lösung so wird $\mathbf{x}$ im Sinne der Ausgleichsrechnung bestimmt.
2. Hat das LGS genau eine Lösung so wird diese Lösung bestimmt.
3. Ist das LGS unterbestimmt, so wird die Lösung $\mathbf{x}$ mit der kleinsten Norm $\|\mathbf{x}\|$ ausgewählt.

Im Folgenden überlegen wir uns warum das so ist.

Wir nehmen an, die Systemmatrix $\mathbf{A} \in \mathbb{R}^{n\times m}$ hat die Singulärwertzerlegung $\mathbf{A} = \mathbf{U\Sigma V}^\mathsf{T}$ mit $\mathbf{U} \in \mathbb{R}^{n\times n}, \mathbf{\Sigma} \in \mathbb{R}^{n\times m}$ und $\mathbf{V} \in \mathbb{R}^{m\times m}$. Wir nehmen zunächst an, dass das LGS keine Lösung hat und betrachten das Residuum $\mathbf{Ax} - \mathbf{b}$:

$$\begin{aligned}\|\mathbf{Ax} - \mathbf{b}\| &= \|\mathbf{U\Sigma V}^\mathsf{T}\mathbf{x} - \mathbf{b}\|, \qquad \mathbf{x} \in \mathbb{R}^m, \mathbf{b} \in \mathbb{R}^n \\ &= \|\mathbf{U}^\mathsf{T}(\mathbf{U\Sigma V}^\mathsf{T}\mathbf{x} - \mathbf{b})\| \\ &= \|\mathbf{\Sigma V}^\mathsf{T}\mathbf{x} - \mathbf{U}^\mathsf{T}\mathbf{b})\| \\ &= \|\mathbf{\Sigma x}' - \mathbf{b}'\|, \qquad \mathbf{x}' = \mathbf{V}^\mathsf{T}\mathbf{x} \in \mathbb{R}^m, \mathbf{b}' = \mathbf{U}^\mathsf{T}\mathbf{b} \in \mathbb{R}^n,\end{aligned} \tag{5.40}$$

wobei wir ausgenutzt haben, dass sich die Längen von Vektoren nicht ändern wenn wir mit einer orthogonalen Matrix (hier $\mathbf{U}^\mathsf{T}$) multiplizieren. Wir betrachten nun das quadrierte Residuum in der letzten Form und verwenden dabei, dass die Singulärwerte deren Index größer oder gleich dem Rang r sind, verschwinden:

$$\|\mathbf{\Sigma x}' - \mathbf{b}'\|^2 = \sum_{i=0}^{r-1}(\sigma_i x_i' - b_i')^2 + \sum_{i=r}^{n-1} {b_i'}^2. \tag{5.41}$$

Wir sehen, dass zur Minimierung des quadrierten Residuums nur die Wahl der x_i' für $0 \leq i < r$ relevant ist; für $r \leq i < m$ ist wegen $\sigma_i = 0$ die Wahl unerheblich. Für $i < r$ muss x_i' so gewählt werden, dass $\sigma_i x_i' - b_i' = 0$ gilt, also $x_i' = b_i'/\sigma_i$. Mit Hilfe der Pseudoinversen können wir eine mögliche Lösung für den Vektor $\mathbf{x}'$ schreiben als

$$\mathbf{x}' = \mathbf{\Sigma}^+\mathbf{b}'. \tag{5.42}$$

Dabei werden die x_i mit $i \geq r$ (die wir beliebig wählen durften) auf Null gesetzt. Für die gesuchte Lösung $\mathbf{x}$ ergibt sich damit:

$$\mathbf{x}' = \mathbf{V}^\mathsf{T}\mathbf{x} = \mathbf{\Sigma}^+\mathbf{b}' = \mathbf{\Sigma}^+\mathbf{U}^\mathsf{T}\mathbf{b} \quad \Longleftrightarrow \quad \mathbf{x} = \mathbf{V\Sigma}^+\mathbf{U}^\mathsf{T}\mathbf{b} \tag{5.43}$$

Wir bekommen also als Lösung das Produkt von $\mathbf{b}$ mit der Pseudoinversen von $\mathbf{A}$! Diese Lösung minimiert wie gezeigt das Residuum für den Fall, dass es keine Lösung gibt. Wie wir bereits von der Normalengleichung wissen, ist damit auch ein eindeutig bestimmtes LGS korrekt gelöst. Es bleibt den Fall von mehr als einer Lösung zu betrachten. Offenbar haben wir hier für die frei wählbaren $x_i, i \geq r$ eine bestimmte Wahl getroffen. Welche Eigenschaft hat $\mathbf{x}$

durch diese Wahl? Betrachten wir dazu zunächst die Länge von $\mathbf{x}'$: Die ersten r Koeffizienten von $\mathbf{x}'$ sind durch die Minimierung festgelegt. Durch die Wahl von $x'_i = 0, i \geq r$ haben wir den kürzesten Vektor $\mathbf{x}'$ aus der Lösungsmenge für das Minimierungsproblem ausgewählt. Die orthogonale Transformation $\mathbf{V}$ in $\mathbf{x} = \mathbf{V}\mathbf{x}'$ erhält diese Eigenschaft. Wie eingangs behauptet sucht die Lösung eines unterbestimmten LGS mit Hilfe der SVD also den kürzesten Vektor aus der Lösungsmenge aus.

5.4.2. Homogene LGS

Bis hierher hatten wir für LGS der Form $\mathbf{A}\mathbf{x} = \mathbf{b}$ immer (implizit) angenommen, dass die rechte Seite $\mathbf{b}$ nicht Null ist. Denn das Gleichungssystem $\mathbf{A}\mathbf{x} = \mathbf{0}$ hat immer die triviale Lösung $\mathbf{x} = 0$. Es gibt aber Anwendungen, bei denen man an einer nicht verschwindenden Lösung des *homogenen* LGS $\mathbf{A}\mathbf{x} = \mathbf{0}, \mathbf{x} \neq \mathbf{0}$ Interesse hat. Eine solche Lösung kann nur existieren, wenn $\mathbf{A}$ ein Rangdefizit hat. Dann sind alle Vektoren, die im Kern von $\mathbf{A}$ liegen eine Lösung. Die SVD bietet durch die explizite Bestimmung einer Basis des Kerns die Möglichkeit, eine Lösung zu bestimmen.

Da für jede Lösung $\mathbf{x}$ des homogenen Gleichungssystems $\mathbf{A}\mathbf{x} = \mathbf{0}$ auch $\lambda\mathbf{x}$ eine Lösung ist, fragen wir nach einer Lösung $\mathbf{x}$ mit Einheitslänge, also $\|\mathbf{x}\| = 1$. Statt nun davon auszugehen, die Matrix $\mathbf{A}$ habe ein Rangdefizit, lösen wir das allgemeinere Minimierungsproblem

$$\underset{\|\mathbf{x}\|=1}{\arg\min} \|\mathbf{A}\mathbf{x}\| = \underset{\|\mathbf{x}\|=1}{\arg\min} \|\mathbf{A}\mathbf{x}\|^2. \tag{5.44}$$

Verschwindet das Produkt $\mathbf{A}\mathbf{x}$ für geeignetes $\mathbf{x}$ so finden wir eine solche Lösung über das Minimierungsproblem. Die Minimierung ist aber ganz allgemein von Interesse und erlaubt insbesondere auch dann die Lösung des homogenen LGS, wenn keine Lösung vorhanden ist, also die Gleichung nur im Sinne eines Ausgleichsproblems gelöst werden kann.

Die SVD von $\mathbf{A}$ führt auf

$$\underset{\|\mathbf{x}\|=1}{\arg\min} \|\mathbf{U}\mathbf{\Sigma}\mathbf{V}^\mathsf{T}\mathbf{x}\|^2 = \underset{\|\mathbf{x}\|=1}{\arg\min} \|\mathbf{\Sigma}\mathbf{V}^\mathsf{T}\mathbf{x}\|^2 = \underset{\|\mathbf{x}'\|=1}{\arg\min} \|\mathbf{\Sigma}\mathbf{x}'\|^2, \qquad \mathbf{x}' = \mathbf{V}^\mathsf{T}\mathbf{x}. \tag{5.45}$$

Da für die Singulärwerte gilt $\sigma_0 \geq \sigma_1 \geq \ldots \geq \sigma_{n-1}$, hat die letzte Form die (nicht notwendigerweise eindeutige) Lösung $\mathbf{x}' = (0, \ldots, 0, 1)^\mathsf{T}$. Wegen $\mathbf{x} = \mathbf{V}\mathbf{x}'$ ist damit die beste Wahl für $\mathbf{x}$ die letzte Spalte der Matrix $\mathbf{V}$, also $\mathbf{x} = \mathbf{v}_{n-1}$.

5.5. Geometrische Intuition

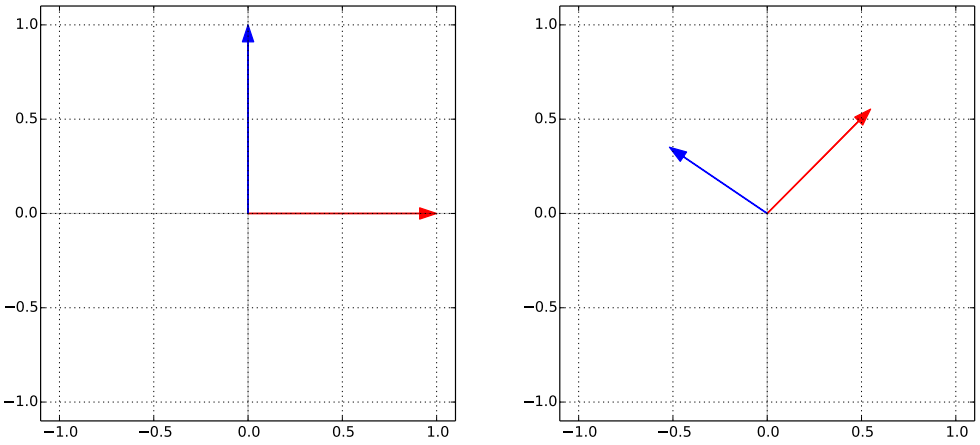
Es gibt zumindest zwei Arten, die Singulärwertzerlegung geometrisch zu interpretieren. Im ersten Fall betrachten wir die Matrix $\mathbf{X}$ als lineare Transformation, also einen Basiswechsel. Dann ist die Darstellung $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\mathsf{T}$ eine Aufteilung in drei Basiwechsel: erst eine Drehspiegelung (die also Längen und Winkel erhält), dann eine Skalierung entlang der Achsen, und dann eine weitere Drehspiegelung. Die zweite Interpretation fasst die Spalten oder Zeilen der Matrix als Punkte auf. Dann ist die Singulärwertzerlegung eine Extraktion der Hauptrichtungen (oder -komponenten) der Punktverteilung und die Singulärwerte geben die Variation der Daten entlang dieser Richtungen an. Im Folgenden erläutern wir diese zwei Sichtweisen an Beispielen in niedriger Dimension näher.

5.5.1. Transformation

Betrachten wir die 2D Transformationsmatrix:

$$\mathbf{A} = \begin{pmatrix} 0.55 & -0.52 \\ 0.55 & 0.35 \end{pmatrix}. \tag{5.46}$$

Für die Standard-Basis (links) erzeugt diese folgende Transformation (rechts):



Dies kann als eine verallgemeinerte Scherungstransformation des Einheitsquadrates verstanden werden. Nach der Anwendung von $\mathbf{A}$ erhalten wir also ein verkleinertes, verzerrtes und rotiertes Einheitsquadrat.

Betrachten wir nun die Singulärwertzerlegung der Matrix $\mathbf{A}$:

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathsf{T}} = \begin{pmatrix} 0.87 & -0.50 \\ 0.50 & 0.87 \end{pmatrix} \begin{pmatrix} 0.80 & 0.0 \\ 0.0 & 0.60 \end{pmatrix} \begin{pmatrix} 0.94 & 0.34 \\ -0.34 & 0.94 \end{pmatrix}^{T} \tag{5.47}$$

Wir können $\mathbf{A}$ also als drei einzelne Transformationen auffassen. Für $\mathbf{y} = \mathbf{A}\mathbf{x}$ wird zunächst $\mathbf{V}^{\mathsf{T}}$ auf $\mathbf{x}$ angewendet, dann $\mathbf{\Sigma}$, und zuletzt $\mathbf{U}$,

$$\mathbf{y}'' = \mathbf{V}^{\mathsf{T}}\mathbf{x} \tag{5.48a}$$
$$\mathbf{y}' = \mathbf{\Sigma}\mathbf{y}'' \tag{5.48b}$$
$$\mathbf{y} = \mathbf{U}\mathbf{y}'. \tag{5.48c}$$

Wir werden nun den Effekt der einzelnen Transformationen im Detail betrachten.

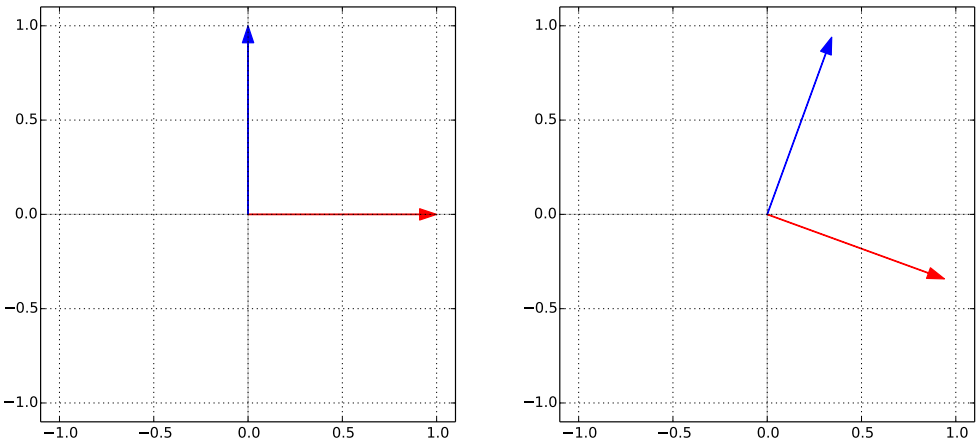
Die Matrix $\mathbf{V}$, und damit auch $\mathbf{V}^{\mathsf{T}}$, ist nach „Konstruktion“ der Singulärwertzerlegung eine orthogonale Matrix, d.h.

$$\mathbf{V}^{\mathsf{T}}\mathbf{V} = \begin{pmatrix} 0.94 & -0.34 \\ 0.34 & 0.94 \end{pmatrix} \begin{pmatrix} 0.94 & 0.34 \\ -0.34 & 0.94 \end{pmatrix} = \begin{pmatrix} 1.0 & 0.0 \\ 0.0 & 1.0 \end{pmatrix} = \mathbf{I} \tag{5.49}$$

Wenn wir das Matrixprodukt näher betrachten, dann berechnen wir in Gleichung 5.49 vier Skalarprodukte, welche die Elemente der Einhetismatrix formen:

$$\mathbf{I}_{ij} = \mathbf{v}_i^{\mathsf{T}}\mathbf{v}_j = \begin{cases} 1 & \text{wenn } i = j \\ 0 & \text{ansonsten} \end{cases} \tag{5.50}$$

wobei $\mathbf{v}_i$ die i-te Spalte von $\mathbf{V}$ ist, also $\mathbf{v}_0 = (0.94, -0.34)$ und $\mathbf{v}_1 = (0.34, 0.94)$. Da $\mathbf{v}_0^\mathsf{T}\mathbf{v}_1 = 0$ sind die Vektoren orthogonal und zwei orthogonale Vektoren formen im $\mathbb{R}^2$ notwendigerweise eine orthogonale Basis für den Vektorraum. Auf der rechten Seite in der folgenden Abbildung sind die Spalten von $\mathbf{V}$ als Vektoren dargestellt:

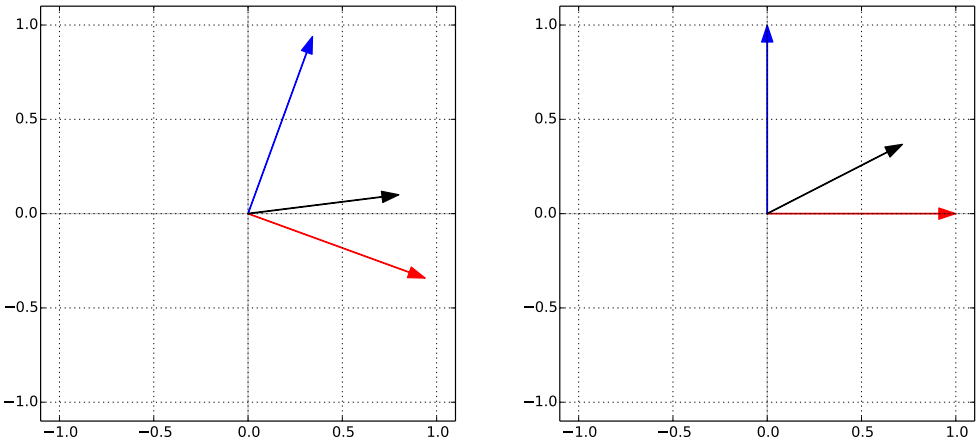


Wir sehen also, dass wir $(\mathbf{v}_0, \mathbf{v}_1)$ als eine rotierte Version der Standardbasis $(\mathbf{e}_0, \mathbf{e}_1)$ auffassen können. In der Tat haben wir für unser Beispiel (entsprechend der Konstruktion)

$$\mathbf{V}^\mathsf{T} = \begin{pmatrix} 0.94 & 0.34 \\ -0.34 & 0.94 \end{pmatrix} = \begin{pmatrix} \cos(-20.0°) & -\sin(-20.0°) \\ \sin(-20.0°) & \cos(-20.0°) \end{pmatrix} = R(-20.0°) \tag{5.51}$$

d.h. $\mathbf{V}^\mathsf{T}$ ist die Rotationsmatrix $R(-20.0°)$ um $20.0°$ im Uhrzeigersinn. Die Elemente einer Rotationsmatrix $R(\theta)$ sind also nicht nur durch $\sin(\theta), \cos(\theta)$ gegeben, sondern sie sind auch die kartesischen Koordinaten der rotierten Basis in der Standardbasis $(\mathbf{e}_0, \mathbf{e}_1)$ ausgedrückt.

Für einen gegebenen, fixen Vektor erhalten wir im rotierten Koordinatensystem also neue Koordinaten. Allerdings ist geometrisch auch einfach zu sehen, dass die Beziehung des fixen Vektors zum um den Winkel $-\theta$ rotierten Koordinatensystem die gleiche ist, wie die eines um θ gedrehten Winkels zu einem fixen Koordinatensystem:



Aber wenn die geometrische Beziehung die gleiche ist, dann müssen auch die Koordinaten in beiden Fällen übereinstimmen. Die Elemente des rotierten Vektors, oder des fixen Vektors

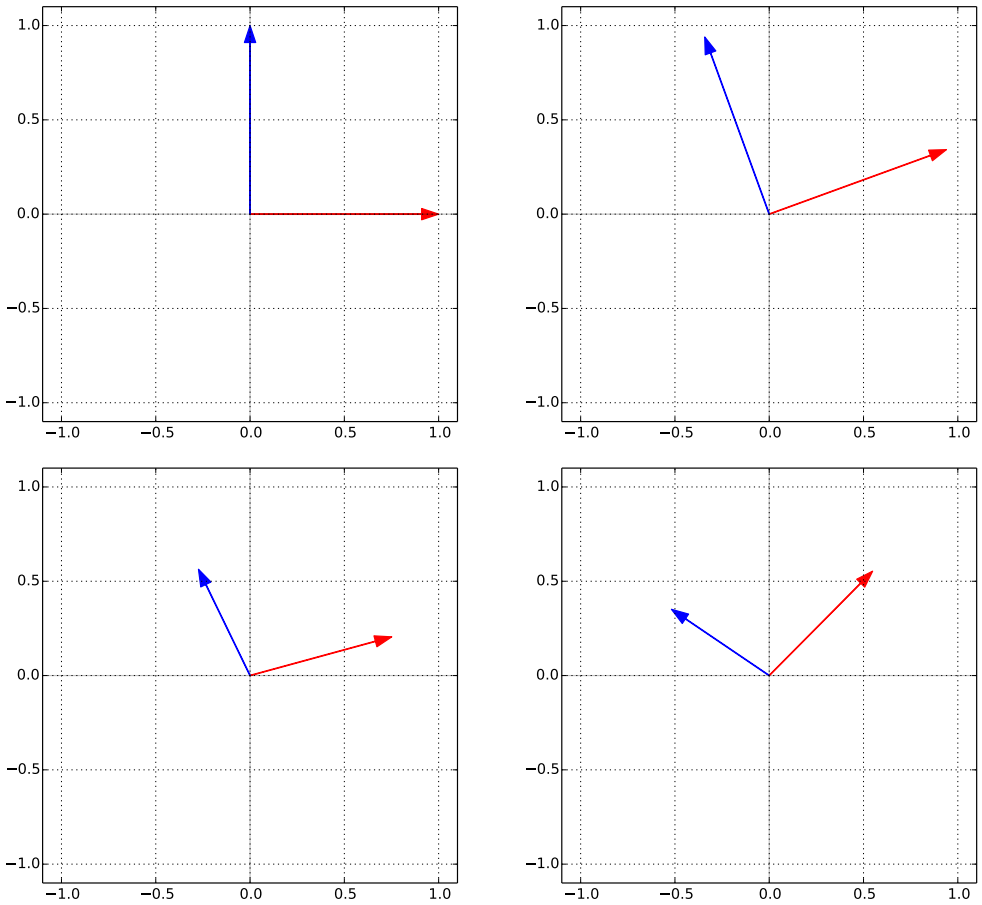


Abbildung 5.1.: *Zerlegung einer Transformation der Euklidischen Standardbasis (links oben) durch die SVD: Im ersten Schritt erfolgt eine Rotation durch V^T. Im zweiten Schritt führt S eine Skalierung der rotierten Vektoren durch. Zuletzt erfolgt eine erneute Rotation durch U.*

bezüglich des inversen rotierten Koordinatensystems, sind also durch

$$R(\theta)\vec{a} = \vec{a}_\theta \tag{5.52}$$

gegeben.

$\mathbf{\Sigma}$ ist eine Skalierungsmatrix, wie man einfach bei einer Anwendung auf die Einheitsvektoren sehen kann. Wir können uns $\mathbf{V}^\mathsf{T}$ also als die Wahl eines Koordinatensystems vorstellen, in der der Effekt der ursprünglichen Matrix $\mathbf{A}$ eine einfache Skalierung ist.

Und was ist mit $\mathbf{U}$? Diese Transformation ist wieder eine Rotation des skalierten Vektors. Natürlich können wir das auch wieder als einen Wechsel des Koordinatensystems vorstellen. Die gesamte Zerlegung ist in Abb. 5.1 dargestellt.

Diese geometrische Anschauung funktioniert konzeptionell genauso im $\mathbb{R}^n$. Für den Fall das $\mathbf{A}$ nicht quadratisch ist, wird je nach Darstellung entweder eine der orthogonalen Transformationen $\mathbf{U}, \mathbf{V}$ oder $\mathbf{\Sigma}$ rechteckig. Die rechteckige Matrix enthält dann eine Projektion auf einen Unterraum.

5.5.2. Hauptkomponentenanalyse

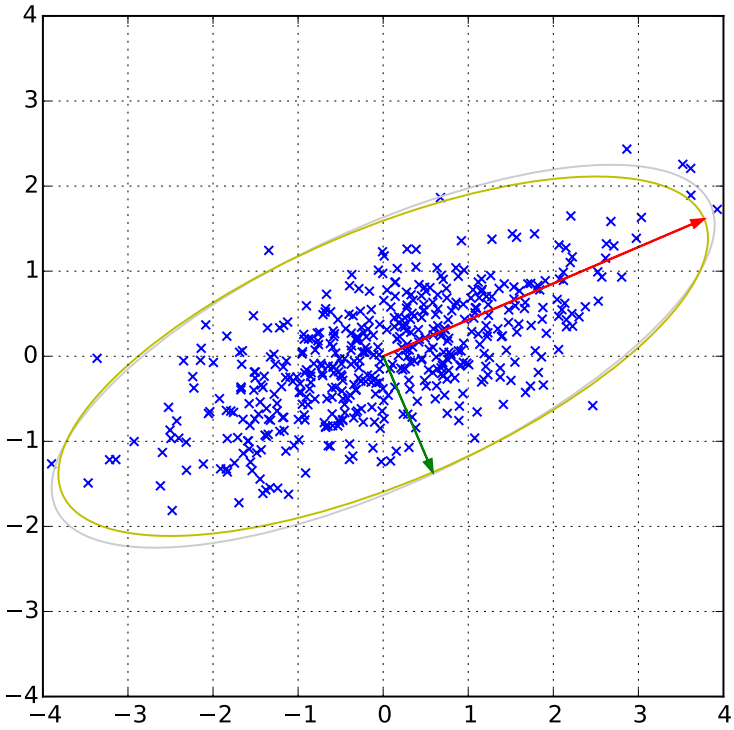
Ziel der Hauptkomponentenanalyse (engl. principal component analysis (PCA)) ist es in einer gegebenen Menge von Punkten die dominanten Richtungen zu finden. Dies ermöglicht auch, so wie im einführenden Beispiel in diesem Kapitel, eine Transformation in einen Vektorraum mit kleinerer Dimension. Die Hauptkomponenten haben genau die Eigenschaften, die wir zuvor optimiert haben, nämlich den Fehler bei einer Projektion auf den Unterraum zu minimieren.

Sei nun $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\mathsf{T}$ eine SVD von $\mathbf{X}$. Aus der Herleitung der SVD ist klar, dass $\max_{\|\mathbf{y}\|=1} \|\mathbf{X}\mathbf{y}\|^2 = \sigma_0^2$. Das Maximum wird dabei von der ersten Spalte von $\mathbf{V}$ angenommen, also $\arg\max_{\|w\|=1} \|\mathbf{X}\mathbf{w}\|^2 = \mathbf{v}_0$.

In folgendem Beispiel werden die Hauptkomponenten einer künstlich erzeugten Punktwolke berechnet. Die Erzeugung der Punktwolke erfolgt in zwei Schritten. Zunächst wird eine Matrix mit zufälligen (normalverteilten) Koordinaten erzeugt (`np.random.randn`). Durch Multiplikation der ersten und zweiten Spalte mit $\sqrt{2.0}$ und $\sqrt{0.25}$ wird dann erreicht, dass die erste und zweite Koordinate der Punkte unterschiedliche Varianzen haben (nämlich 2.0 und 0.25). Das bedeutet also, dass die Punktwolke stärker entlang der ersten Koordinatenachse ausgedehnt ist. Durch Rotation um $25°$ erhalten wir einen Datensatz der entlang der $25°$-Achse ausgedehnt ist.

Für die Bestimmung der Hauptkomponenten wird zunächst der Schwerpunkt der Punktwolke berechnet und dann abgezogen um den Schwerpunkt in den Koordinatenursprung zu verschieben. Nun wird eine SVD der Punktwolke bestimmt. Zu beachten ist hier, dass von `numpy.linalg.svd` nicht V sondern V^T zurückgegeben wird. Das bedeutet die i-te Spalte von V findet sich in `V[i]`. Die Singulärwerte stellen ein Maß für die Ausdehnung der Daten in Richtung der jeweiligen Hauptachse dar (genauer: $\sqrt{\sigma_i^2/(m-1)}$ ist die korrigierte Stichprobenvarianz). Zur besseren Visualisierung werden daher die Hauptkomponenten mit dem dreifachen der geschätzten Standardabweichungen multipliziert.

Im Folgenden ist die Ausgabe eines Programmlaufes zu sehen. Die erste Hauptachse ist in rot und die zweite Hauptachse ist in grün eingezeichnet. Zur besseren Visualisierung ist außerdem die durch die skalierten Hauptkomponenten definierte Ellipse eingezeichnet.


4
3
2
1
0
−1
−2
−3
−4
−4
−3
−2
−1
0
1
2
3
4

6. Interpolation

6.1. Motivation

Animationen sind Bildfolgen über der Zeit. Um menschlichen Betrachtern den Eindruck zu vermitteln, die Bilder würden sich kontinuierlich verändern, benötigt man mindestens 20 Bilder pro Sekunde. Bei der Erzeugung von Animationen möchte man aber nicht so viele Bilder definieren. Vielmehr werden nur einzelne Bilder spezifiziert und zwischen diesen Bildern *interpoliert*.

Als Beispiel betrachten wir eine Figur aus Liniensegmenten. Wir nehmen an, die einzelnen Segmente haben feste Länge. Der Zustand der Figur kann dann durch die Winkel der Segmente zueinander sowie eine starre Transformation (Orientierung und Position) der gesamten Figur spezifiziert werden, also ein Vektor von reellen Zahlen $\mathbf{f} \in \mathbb{R}^n$.

Nehmen wir an, die Zustände $\mathbf{f}_i$ sind für Zeitpunkte t_i gegeben. Dann suchen wir eine Funktion, die den Zustand für beliebige Zeitpunkte t angibt: $f : \mathbb{R} \mapsto \mathbb{R}^n$. Die Funktion soll dabei die gegebenen Zustände zu den gegebenen Zeiten annehmen, also $f(x_i) = \mathbf{f}_i$.

Der einfachste Ansatz dieses Ziel zu erreichen ist eine stückweise konstante Funktion, für die wir nur überlegen müssen, zu welcher Zeit der Wert von $\mathbf{f}_i$ zu $\mathbf{f}_{i+1}$ wechseln soll. Wählt man die Mitte der Intervalle so erhält man:

$$f(x) = \begin{cases} \mathbf{f}_0 & x \leq \frac{1}{2}(x_0 + x_1) \\ \mathbf{f}_1 & \frac{1}{2}(x_0 + x_1) < x \leq \frac{1}{2}(x_1 + x_2) \\ \mathbf{f}_2 & \frac{1}{2}(x_1 + x_2) < x \leq \frac{1}{2}(x_2 + x_3) \\ \vdots & \vdots \end{cases} \tag{6.1}$$

Auch wenn die Funktion die Forderung nach Interpolation erfüllt, so ist es doch unbefriedigend und in vielen Anwendungen schlecht, dass sie nicht stetig ist. Im Folgenden wollen wir uns mit „besseren“ Methoden zur Interpolation von Daten beschäftigen.

6.2. Polynominterpolation

Es liegt Nahe, statt konstanter Funktionen zur Interpolation der Daten Polynome mit höherem Grad zu verwenden. Man könnte zum Beispiel je zwei aufeinander folgende Punkte linear interpolieren, oder drei Punkte mit einer quadratischen Funktion, usw. Wir betrachten zunächst den Fall linearer Interpolation von zwei Punkten und dann den allgemeinen Fall.

6.2.1. Lineare Interpolation

Gegeben zwei Zeiten x_0, x_1 und zwei (eindimensionale) Funktionswerte f_0, f_1. Die Funktion soll im Intervall $[x_0, x_1]$ definiert und linear sein. Eine lineare Funktion lässt sich schreiben als

$$f(x) = c_0 + c_1\, x = (1, x) \begin{pmatrix} c_0 \\ c_1 \end{pmatrix}. \tag{6.2}$$

Die Interpolationsbedingungen führen auf ein LGS:

$$\left.\begin{aligned} f_0 &= c_0 + c_1\, x_0 \\ f_1 &= c_0 + c_1\, x_1 \end{aligned}\right\} \qquad \begin{pmatrix} 1 & x_0 \\ 1 & x_1 \end{pmatrix} \mathbf{c} = \mathbf{f} \tag{6.3}$$

Die Inverse der 2×2-Matrix können wir uns explizit ausrechnen:

$$\mathbf{c} = \frac{1}{x_1 - x_0} \begin{pmatrix} x_1 & -x_0 \\ -1 & 1 \end{pmatrix} \mathbf{f} \tag{6.4}$$

und erhalten dann durch Ausmultiplizieren die Funktion

$$f(x) = \frac{1}{x_1 - x_0}(1, x) \begin{pmatrix} x_1 & -x_0 \\ -1 & 1 \end{pmatrix} \mathbf{f} = \left(\frac{x_1 - x}{x_1 - x_0}, \frac{x - x_0}{x_1 - x_0} \right) \mathbf{f} = f_0 \frac{x_1 - x}{x_1 - x_0} + f_1 \frac{x - x_0}{x_1 - x_0} \tag{6.5}$$

oder

$$f(x) = (1 - u) f_0 + u\, f_1, \qquad u = \frac{x - x_0}{x_1 - x_0} \tag{6.6}$$

6.2.2. Polynominterpolation

Statt einer linearen Funktion nehmen wir nun ein Polynom mit höherem Grad:

$$f(x) = c_0 + c_1\, x + c_2\, x^2 + \ldots = (1, x, x^2, \ldots) \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \end{pmatrix} = (1, x, x^2, \ldots)\mathbf{c}. \tag{6.7}$$

Mit diesem Polynom wollen wir n Daten der Form $(x_i, f_i), 0 \leq i < n$ interpolieren. Das führt wie im Fall der linearen Interpolation auf ein LGS, in diesem Fall mit n Zeilen:

$$\left.\begin{aligned} f_0 &= c_0 + c_1\, x_0 + c_2\, x_0^2 + \ldots \\ f_1 &= c_0 + c_1\, x_1 + c_2\, x_1^2 + \ldots \\ &\vdots \\ f_{n-1} &= c_0 + c_1\, x_{n-1} + c_2\, x_{n-1}^2 + \ldots \end{aligned}\right\} \quad \begin{pmatrix} 1 & x_0 & x_0^2 & \ldots \\ 1 & x_1 & x_1^2 & \ldots \\ \vdots & \vdots & \vdots & \ddots \\ 1 & x_{n-1} & x_{n-1}^2 & \ldots \end{pmatrix} \mathbf{c} = \mathbf{f} \tag{6.8}$$

Zur Interpolation der n Punkte, also Erfüllung der n Gleichungen, muss die Matrix n (linear unabhängige) Spalten haben. Man benötigt also ein Polynom mit (mindestens) n Koeffizienten, d.h. $\mathbf{c} \in \mathbb{R}^n$ und das Polynom hat Grad $n - 1$. Die Systemmatrix

$$\mathbf{V}(x_0, \ldots, x_{n-1}) = \begin{pmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^{n-1} \\ 1 & x_1 & x_1^2 & \ldots & x_1^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n-1} & x_{n-1}^2 & \ldots & x_{n-1}^{n-1} \end{pmatrix} \in \mathbb{R}^{n \times n} \tag{6.9}$$

heißt *Vandermonde*-Matrix. Die Determinante dieser Matrix haben wir für $n = 2$ bereits ausgerechnet: $\det \mathbf{V}(x_0, x_1) = x_1 - x_0$. Mit Induktion können wir die Determinante für alle n entwickeln und erhalten:

$$\det \mathbf{V}(x_0, \ldots, x_{n-1}) = \prod_{0 \leq i < j < n} (x_j - x_i). \tag{6.10}$$

Die Determinante ist also Null g.d.w. die x_i nicht paarweise verschieden sind. Wählt man die Parameter x_i verschieden, so lässt sich die Interpolationsaufgabe mit einem Polynom vom Grad $n - 1$ eindeutig lösen (weil ein $n \times n$ LGS mit regulärer Matrix eine eindeutige Lösung hat).

6.2.3. Lagrange-Interpolation

Für den linearen Fall $n = 2$ konnten wir das LGS explizit lösen. Wir sind dabei zu einer Form gelangt, in der wir nur noch die gegebenen Funktionswerte f_0, f_1 mit den zwei *Basispolynomen* $(x_1 - x)/(x_1 - x_0)$ und $(x - x_0)/(x_1 - x_0)$ multiplizieren mussten. Können wir das LGS auch für größere n immer explizit lösen? Analog zum linearen Fall, würden wir die Lösung schreiben können als

$$f(x) = f_0 l_0(x) + f_1 l_1(x) + \dots f_{n-1} l_{n-1}(x) = \sum_{i=0}^{n-1} f_i l_i(x). \tag{6.11}$$

Diese Darstellung liefert uabhängig von den vorgegebenen Werten f_i immer das korrekte Ergebnis, wenn die Basispolynome $l_i(x)$ folgende Eigenschaft haben:

$$l_i(x_j) = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \tag{6.12}$$

Setzt man nämlich jetzt den Parameter x_j ein, so ergibt sich

$$\begin{aligned} f(x_j) &= f_0 l_0(x_j) + \dots + f_{j-1} l_{j-1}(x_j) + f_j l_j(x_j) + f_{j+1} l_{j+1}(x_j) + \dots + f_{n-1} l_{n-1}(x) \\ &= f_0 \cdot 0 + \dots + f_{j-1} \cdot 0 + f_j \cdot 1 + f_{j+1} \cdot 0 + \dots + f_{n-1} \cdot 0 \\ &= f_j, \end{aligned} \tag{6.13}$$

wie gewünscht.

Wie findet man Polynome vom Grad $n - 1$ mit dieser Eigenschaft? Das „Muster“ aus dem linearen Fall zeigt wie das funktionieren kann: hier ist der Zähler jeweils Null für den einen Wert und Zähler und Nenner identisch für den anderen Wert. Dieses Prinzip funktioniert auch für mehr als zwei Werte mit den *Lagrange*-Basispolynomen:

$$l_i(x) = \frac{x - x_0}{x_i - x_0} \cdot \frac{x - x_1}{x_i - x_1} \cdots \frac{x - x_{i-1}}{x_i - x_{i-1}} \cdot \frac{x - x_{i+1}}{x_i - x_{i+1}} \cdots \frac{x - x_{n-1}}{x_i - x_{n-1}} = \prod_{\substack{j=0 \\ j \neq i}}^{n-1} \frac{x - x_j}{x_i - x_j}. \tag{6.14}$$

Diese Polynome haben $n - 1$ lineare Faktoren, also den gewünschten Grad $n - 1$. Man kann sich leicht davon überzeugen, dass sie die erforderlichen Eigenschaften haben. Dazu setzen wir x_i in $l_i(x)$ ein und sehen

$$l_i(x_i) = \frac{x_i - x_0}{x_i - x_0} \cdot \frac{x_i - x_1}{x_i - x_1} \cdots \frac{x_i - x_{i-1}}{x_i - x_{i-1}} \cdot \frac{x_i - x_{i+1}}{x_i - x_{i+1}} \cdots \frac{x_i - x_{n-1}}{x_i - x_{n-1}} = 1 \tag{6.15}$$

sowie für $x_k, k \neq i$

$$l_i(x_k) = \prod_{\substack{j=0 \\ j \neq i}}^{n-1} \frac{x - x_j}{x_i - x_j} = \frac{x_k - x_k}{x_i - x_k} \prod_{\substack{j=0 \\ j \neq i,k}}^{n-1} \frac{x - x_j}{x_i - x_j} = 0 \cdot \prod_{\substack{j=0 \\ j \neq i,k}}^{n-1} \frac{x - x_j}{x_i - x_j} = 0. \tag{6.16}$$

Da für die Interpolation von n Punkten die Interpolation mit Polynomen vom Grad $n - 1$ eindeutig ist, haben wir mit der Darstellung also nicht nur ein Polynom gefunden, sondern genau die Lösung des im vorangegangenen Abschnitt aufgestellten LGS. Ein Beispiel für die Lagrange-Basisfunktionen mit steigendem Grad ist in Abbildung 6.1 gegeben.

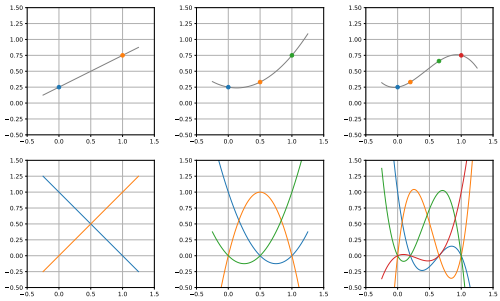


Abbildung 6.1.: *Polynominterpolation mit Lagrange-Basisfunktionen. Die gegebenen n Punkte werden eindeutig durch ein Polynom vom Grad $n-1$ interpoliert. Dieses Polynom kann in der Lagrange-Basis angegeben werden, die für die gegebenen Stützstellen unten dargestellt ist. In den Stützstellen sind die Lagrange-Basispolynome entweder 0 oder 1.*

```
import numpy as np
import matplotlib.pyplot as plt

def intpoly(x, y):
    V = np.vander(x,len(x))
    a = np.linalg.solve(V,y)
    return np.poly1d(a)

x = np.linspace(0, 4.0, 13)
y = np.sin(x * np.pi * 2.5) * np.exp(-x**2/8.0)

for k in range(1,4):
    plt.subplot(1,3,k)
    plt.plot(x, y, 'ko', zorder=0)
    for i in range(0, len(x)-k, k):
        p = intpoly(x[i:i+k+1], y[i:i+k+1])
        tx = np.linspace(x[i], x[i+k], 20)
        ty = p(tx)
        plt.plot(tx, ty, '-')

    plt.grid(True)
    plt.xlim(-0.1,4.1)
    plt.ylim(-1.1,1.1)
    plt.gca().set_aspect('equal')

plt.show()
```

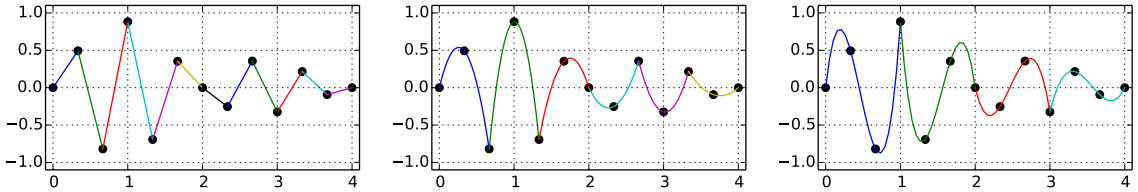


Abbildung 6.2.: *Stückweise lineare, quadratische und kubische Polynominterpolation.*

6.3. Stückweise Polynominterpolation

Da Polynome mit hohem Grad zu unerwünschtem Schwingen neigen, können nur Polynome mit kleinem Grad sinnvoll zur Interpolation eingesetzt werden. Bei mehr als 4 Datenpunkten müssen also mehrere Polynome von kleinem Grad aneinandergesetzt werden. Für die Daten (x_i, f_i) erhält man die stückweise lineare Interpolation durch

$$f(x) = \mathbf{f}_i \frac{x_{i+1} - x}{x_{i+1} - x_i} + \mathbf{f}_{i+1} \frac{x - x_i}{x_{i+1} - x_i} \quad \text{für} \quad x_i \leq x < x_{i+1} \tag{6.17}$$

Diese Funktion interpoliert zwar wie gewünscht die vorgegebenen Daten und ist stetig. Die Ableitungen sind in den Stützstellen aber nicht stetig. An diesem Defizit ändert sich auch nichts wenn man je drei aufeinander folgende Punkte quadratisch, oder allgemeiner mehr aufeinander folgende Punkte mit Polynomen höheren Grades interpoliert (Abbildung 6.2).

Die „Knicke“ ergeben sich, weil die Interpolationspolynome zweier benachbarter Abschnitte zwar am Ende des einen und am Start des anderen übereinstimmen, dort normalerweise jedoch unterschiedliche Steigungen haben. Interessant wäre es also an den Sützstellen neben den Werten auch die Steigungen vorgeben zu können.

6.3.1. Kubische Hermite-Interpolation

Die Verwendung von Funktionswerten und Ableitungen wollen wir uns nun an einem einfachen Beispiel betrachten (Abbildung 6.3). Gegeben seien zwei Datenpunkte (x_0, f_0) und (x_1, f_1) mit vorgegeben Steigungen f_0', f_1'. Gesucht ist ein kubisches Polynom

$$f(x) = c_0 + c_1\, x + c_2\, x^2 + c_3\, x^3, \tag{6.18}$$

das für die Parameter x_0, x_1 die Funktionswerte f_0, f_1 und die Steigungen f_0', f_1' annimmt. Die Steigung von $f(x)$ ist durch die Ableitung gegeben:

$$f'(x) = c_1 + 2c_2\, x + 3c_3\, x^2 \tag{6.19}$$

und da die Ableitung linear ist, ist auch $f'(x)$ wieder linear in den Parametern c_i. Zusammen mit den Bedingungen für die Funktionswerte erhalten wir damit vier Gleichungen

$$\begin{aligned} f_0 &= c_0 + c_1\, x_0 + c_2\, x_0^2 + c_3\, x_0^3 \\ f_1 &= c_0 + c_1\, x_1 + c_2\, x_1^2 + c_3\, x_1^3 \\ f_0' &= c_1 + 2c_2\, x_0 + 3c_3\, x_0^2 \\ f_1' &= c_1 + 2c_2\, x_1 + 3c_3\, x_1^2, \end{aligned} \tag{6.20}$$

die sich wie für den Fall der Polynominterpolation als LGS schreiben lassen:

$$\begin{pmatrix} 1 & x_0 & x_0^2 & x_0^3 \\ 1 & x_1 & x_1^2 & x_1^3 \\ 0 & 1 & 2x_0 & 3x_0^2 \\ 0 & 1 & 2x_1 & 3x_1^2 \end{pmatrix} \mathbf{c} = \begin{pmatrix} f_0 \\ f_1 \\ f_0' \\ f_1' \end{pmatrix}. \tag{6.21}$$

```
import numpy as np
import matplotlib.pyplot as plt


for dy in np.linspace(-2.0, 2.0, 5):
    x = np.array([0., 1.])
    y = np.array([0.75, -0.25, dy, -0.25])
    p = cubic_polynomial(x, y)
    tx = np.linspace(-1.0, 2.0, 100)
    ty = p(tx)
    plt.plot(x, y[:2], 'ko', zorder=0)
    plt.plot(tx, ty, '-')

plt.grid(True)
plt.xlim(-0.5,1.5)
plt.ylim(-0.5,1.5)
plt.gca().set_aspect('equal')

plt.show()
```

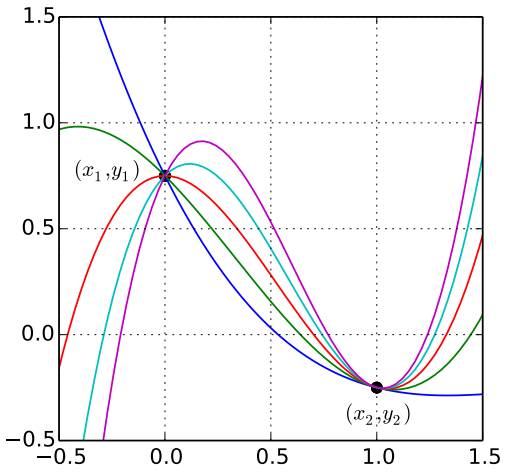


Abbildung 6.3.: *Kubische Interpolationspolynome für zwei Stützstellen mit vorgegebenen Werten und Steigungen. In der ersten Stützstelle wird die Steigungen variiert.*

Diese Matrix lässt sich nicht ohne Kenntnis von x_0, x_1 invertieren. Viele Probleme lassen sich allerdings auf die Situation $x_0 = 0, x_1 = 1$ reduzieren. Dann erhält man wegen

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 3 \end{pmatrix}^{-1} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -3 & 3 & -2 & -1 \\ 2 & -2 & 1 & 1 \end{pmatrix} \tag{6.22}$$

die Lösung

$$\begin{aligned} f(x) &= (1, x, x^2, x^3)\mathbf{c} \\ &= (1, x, x^2, x^3) \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -3 & 3 & -2 & -1 \\ 2 & -2 & 1 & 1 \end{pmatrix} \begin{pmatrix} f_0 \\ f_1 \\ f_0' \\ f_1' \end{pmatrix} \\ &= (1 - 3x^2 + 2x^3, 3x^2 - 2x^3, x - 2x^2 + x^3, -x^2 + x^3) \begin{pmatrix} f_0 \\ f_1 \\ f_0' \\ f_1' \end{pmatrix} \\ &= f_0(1 - 3x^2 + 2x^3) + f_1(3x^2 - 2x^3) + f_0'(x - 2x^2 + x^3) + f_1'(-x^2 + x^3). \end{aligned} \tag{6.23}$$

Man kann sich leicht davon überzeugen, dass dieses Polynom die gewünschten Werte und Ableitungen an den Stellen 0 und 1 annimmt.

Mit Hilfe solcher kubischer Stücke könnte man nun jede zwei aufeinanderfolgenden Punkte interpolieren. Sind in den Punkten bereits Steigungen vorgegeben, so kann man diese Steigungen interpolieren. Ansonsten setzt man die Steigungen in den Punkten im einfachsten Fall überall auf Null. Oder man schätzt die Steigungen, z.B. durch

$$\begin{aligned} f_0' &= f_{n-1}' = 0 \\ f_i' &= \frac{f_{i+1} - f_{i-1}}{x_{i+1} - x_{i-1}}. \end{aligned} \tag{6.24}$$

6.3.2. Kubischer Spline

Meistens kennt man die Steigung in den Datenpunkten nicht und die Schätzung der Steigung schränkt die verfügbaren Freiheitsgrade der Polynomstücke mehr als nötig ein. Statt Steigungen vorzugeben, können wir auch fordern, dass die Steigungen in den End- und Anfangspunkten der Polynomstücke gleich sind. Dies führt dazu, dass sich die Polynomstücke nicht mehr unabhängig voneinander bestimmen lassen.

Wie zuvor verwenden wir kubische Basispolynome $f_i(i) = (1, x, x^2, x^3)\mathbf{c}_i$, wobei wir den Index i verwenden, um das i-te Polynomstück zu identifizieren. Insgesamt gibt es bei n Punkten also $n - 1$ Polynomstücke. Die Funktion f_i soll die Funktion im Intervall $[x_i, x_{i+1}]$ interpolieren, also insbesondere die Funktionswerte f_i, f_{i+1}:

$$\begin{aligned} f_i(x_i) &= (1, x_i, x_i^2, x_i^3)\mathbf{c}_i = f_i \\ f_i(x_{i+1}) &= (1, x_{i+1}, x_{i+1}^2, x_{i+1}^3)\mathbf{c}_i = f_{i+1} \end{aligned} \tag{6.25}$$

Ferner wollen wir, dass für alle Stellen x_i außer der ersten und letzten, die Ableitung beider angrenzenden Polynomstücke übereinstimmen:

$$f_{i-1}'(x_i) = (0, 1, 2x_i, 3x_i^2)\mathbf{c}_{i-1} = (0, 1, 2x_i, 3x_i^2)\mathbf{c}_i = f_i'(x_i), \qquad 1 \leq i < n - 1. \tag{6.26}$$

Dies sind aber nicht mehr wie zuvor zwei Bedingungen pro Polynomstück, sondern $n - 2$ Bedingungen (Für Anfangs- und Endpunkt fehlt die Bedingung) für ingesamt $n - 1$ Polynomstücke.

Man fordert deshalb für kubische Polynomstücke zusätzlich, dass die zweiten Ableitungen in den Übergangsstellen übereinstimmen:

$$f''_{i-1}(x_i) = (0,0,2,6x_i)\mathbf{c}_{i-1} = (0,0,2,6x_i)\mathbf{c}_i = f''_i(x_i), \qquad 1 \leq i < n-1. \tag{6.27}$$

Die enstehende stückweise definierte Funktion hat dann nicht nur stetige Ableitungen sondern auch stetige zweite Ableitungen und ist damit zweimal stetig differenzierbar. Sie wird durch folgendes LGS beschrieben:

$$\begin{pmatrix}
1 & x_0 & x_0^2 & x_0^3 & & & & & & & & \\
1 & x_1 & x_1^2 & x_1^3 & & & & & & & & \\
0 & 1 & 2x_1 & 3x_1^2 & 0 & -1 & -2x_1 & -3x_1^2 & & & & \\
0 & 0 & 2 & 6x_1 & 0 & 0 & -2 & -6x_1 & & & & \\
 & & & & 1 & x_1 & x_1^2 & x_1^3 & & & & \\
 & & & & 1 & x_2 & x_2^2 & x_2^3 & & & & \\
 & & & & & & & & \ddots & & & \\
 & & & & & & & & 1 & x_{n-2} & x_{n-2}^2 & x_{n-2}^3 \\
 & & & & & & & & 1 & x_{n-1} & x_{n-1}^2 & x_{n-1}^3
\end{pmatrix}
\begin{pmatrix} \mathbf{c}_0 \\ \mathbf{c}_1 \\ \vdots \\ \mathbf{c}_{n-2} \end{pmatrix} =
\begin{pmatrix} f_0 \\ f_1 \\ 0 \\ 0 \\ f_1 \\ f_2 \\ \vdots \\ f_{n-2} \\ f_{n-1} \end{pmatrix} \tag{6.28}$$

Wir haben 2 Interpolationsbedingungen für jedes Polynomstück, also $2(n-1)$, sowie $2(n-2)$ Bedingungen an die Ableitungen, also insgesamt $4n-6$ Bedingungen in Form von linearen Gleichungen. Mit anderen Worten, das LGS hat $4n-6$ Zeilen. Die $n-1$ Polynomstücke haben jeweils 4 Parameter – es gibt also $4n-4$ Unbekannte (oder Spalten in der Systemmatrix). Das resultierende LGS ist also unterbestimmt. Es ist üblich, zwei weitere lineare Gleichungen als *Randbedingungen* zu formulieren, damit die Lösung eindeutig wird. Üblich sind drei verschiedene Ansätze:

1. **Natürliche Randbedingungen:** Die zweiten Ableitungen am Anfangs- und Endpunkt des Splines werden Null gesetzt, damit die Kurve am Anfang und Ende nicht unnötigerweise gekrümmt ist

   $$f''_0(x_0) = f''_{n-2}(x_{n-1}) = 0. \tag{6.29}$$

   Dem LGS oben fügen wir also die folgenden zwei Zeilen hinzu:

   $$\left(\begin{array}{cccccccc} 0 & 0 & 2 & 6x_0 & & & & \\ & & & & 0 & 0 & 2 & 6x_{n-1} \end{array}\right) \mid \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{6.30}$$

2. **Vorgabe der Steigungen:** Die ersten Ableitungen am Anfangs- und Endpunkt des Splines werden vorgegeben:

   $$f'_0(x_0) = f'_0 \qquad f'_{n-2}(x_{n-1}) = f'_{n-1}. \tag{6.31}$$

   Dies entspricht den zusätzlichen Zeilen:

   $$\left(\begin{array}{cccccccc} 0 & 1 & 2x_0 & 3x_0^2 & & & & \\ & & & & 0 & 1 & 2x_{n-1} & 3x_{n-1}^2 \end{array}\right) \mid \begin{pmatrix} f'_0 \\ f'_{n-1} \end{pmatrix} \tag{6.32}$$

3. **Periodische Randbedingungen:** Die Funktionswerte, ersten Ableitungen und zweiten Ableitungen sollen am Anfangs- und Endpunkt des Splines übereinstimmen. Dabei geht

man davon aus, dass $f_0 = f_{n-1}$ gilt und die Periodizitätsforderung für die Funktionswerte bereits durch die Interpolationsbedingungen erfüllt wird. Zusätzlich formuliert man :

$$f_0'(x_0) = f_{n-2}'(x_{n-1}) \qquad f_0''(x_0) = f_{n-2}''(x_{n-1}). \tag{6.33}$$

Im LGS ergänzt man dann die Zeilen

$$\begin{pmatrix} 0 & 1 & 2x_0 & 3x_0^2 & \dots & 0 & -1 & -2x_{n-1} & -3x_{n-1}^2 \\ 0 & 0 & 2 & 6x_0 & \dots & 0 & 0 & -2 & -6x_{n-1} \end{pmatrix} \mid \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{6.34}$$

6.4. Approximation von Funktionen mit Polynomen

Die stückweise Polynomapproximation hatten wir damit motiviert, dass "Polynome mit hohem Grad zu unerwünschten Schwingungen neigen". Wir wollen diese Phänomen genauer untersuchen. Wie wir sehen werden, hängt die Schwingungsneigung von der Verteilung der Stützstellen x_i ab. In manchen Anwendungen kann man die Stützstellen wählen. Ein wichtiges Beispiel dafür ist die Approximation einer Funktion $f : [a, b] \in \mathbb{R} \mapsto \mathbb{R}$ durch ein Polynom $p(x) = (1, x, x^2, \dots, x^d)\mathbf{c}$ mit vorgegebem Grad d. Hierfür könnten wir $d + 1$ (paarweise verschiedene) Stützstellen x_i (typischerweise in $[a, b]$) auswählen, $f(x_i)$ ausrechnen und aus den Daten ein Polynom gewinnen, das die Funktion in den Stellen x_i interpoliert.

Natürlich ist unsere Hoffnung, dass $p(x) \approx f(x)$ für alle $x \in [a, b]$ gilt. Es zeigt sich, dass diese Eigenschaft stark von der Wahl der Stützstellen abhängt.

6.4.1. Die Sätze von Taylor und Weierstrass

Zunächst wollen wir die Hoffnung begründen, mit Polynomen beliebige Funktion $f : [a, b] \mapsto \mathbb{R}$ gut approximieren zu können. Dazu betrachten zuerst die (wahrscheinlich bekannte) *Taylor Approximation* einer Funktion f an der Stelle a. Dabei erzeugt man ein Polynom, das an der Stelle a die gleichen Ableitungen wie f hat. Wir gehen also davon aus, f sei genügend oft (stetig) differenzierbar an der Stelle a. Die einfachste, und nicht besonders nützliche, Form der Approximation ist die konstante: $f(x) \approx f(a)$. Besser wird es durch Verwendung der Tangente in a:

$$f(x) \approx f(a) + f'(a)(x - a). \tag{6.35}$$

Hier ist $f'(a)$ die Steigung in a. Durch die Verschiebung $(x-a)$ hat die Gerade mit der Steigung $f'(a)$ ihre Nullstelle bei a; Addition von $f(a)$ erzeugt dann den gewünschten Funktionswert in a. Der nächste Schritt ist die Verwendung der zweiten Ableitung, also die Addition einer Parabel mit der gewünschten zweiten Ableitung $f''(a)$:

$$f(x) \approx f(a) + f'(a)(x - a) + \frac{1}{2}f''(a)(x - a)^2. \tag{6.36}$$

Das Prinzip ist das Gleiche wie bei der Tangente: die Parabel ist so verschoben, dass die Nullstelle in a liegt. Die zweite Ableitung erfordert zusätzlich den Faktor $1/2$, denn die Ableitung einer quadratischen Funktion erzeugt den Faktor 2. In gleicher Weise kann man Ableitungen höherer Ordnung hinzufügen und gelangt zur Approximation der Funktion f durch das Taylor-Polynom vom Grad n:

$$f(x) = \sum_{i=0}^{n} \frac{f^{(i)}(a)}{i!}(x - a)^i + R_n(x). \tag{6.37}$$

Hier ist $R_n(x)$ das *Restglied*, also der Unterschied zwischen f und dem Taylor-Polynom vom Grad n. Die Approximation durch ein Taylor-Polynom ist die Grundlage vieler Algorithmen im wissenschaftlichen Rechnen: wir nehmen an, die Funktion sei lokal gut durch das Taylor-Polynom repräsentiert und machen die nächsten Schritte dann auf der Basis der Polynomdarstellung. In Kapitel 9 werden mehrere solcher Verfahren auf dieser Basis entwickelt. Um die möglichen Fehler solcher Verfahren abschätzen zu können, ist es nützlich, konkrete Formeln für das Restglied zu verwenden. Wir führen hier nur die *Lagrange*-Form ein:

$$R_n(x) = \frac{f^{(n+1)}(z)}{(n+1)!}(x-a)^{n+1}, \quad z \in [x,a] \cup [a,x]. \tag{6.38}$$

Diese Darstellung ist so zu verstehen: Für jedes x gibt es ein z zwischen x und a, so dass die Darstellung in Gl. 6.37 exakt ist. Man bezeichnet diese Aussage auch als den *Satz von Taylor*. Der Satz von Taylor in dieser (und ähnlicher) Form wird gerne als eine Verallgemeinerung des Mittelwertsatzes betrachtet. Zur Erinnerung, der Mittelwertsatz sagt, dass eine stetig differenzierbare Funktion im Intervall $[a,b]$ irgendwo die Steigung der Sekante $(f(b)-f(a))/(b-a)$ annehmen muss. Das ist identisch mit dem Satz von Taylor (und Lagrange-Restglied) für $n=0$: $f(x) = f(a) + f'(z)(x-a)$. All diese Aussagen lassen sich elementar beweisen - wir verweisen dafür auf die entsprechenden Kurse in der Analysis.

Das Lagrange-Restglied enthält unmittelbar eine Aussage darüber, welche Funktionen sich beliebig gut durch Polynome approximieren lassen. Wenn alle Ableitungen von f (im zu approximierenden Intervall) durch eine feste Konstante beschränkt sind, konvergiert das Taylor-Polynom gegen die Funktion. Nehmen wir an es gilt

$$f^{(i)}(x) \leq L, \forall x \in [a,b], i \in \mathbb{N} \tag{6.39}$$

dann kann man offenbar für jedes $\epsilon > 0$ ein n finden, so dass

$$R_n(x) \leq L\frac{(x-a)^{n+1}}{(n+1)!} \leq L\frac{(b-a)^{n+1}}{(n+1)!} \leq \epsilon, \tag{6.40}$$

denn die Fakultätsfunktion im Nenner wächst schneller als die Potenzfunktion im Zähler (sobald $(n+1)$ größer als $(b-a)$ ist).

Natürlich sind die Anforderungen an f eine erhebliche Einschränkung: die Ableitungen von f müssen nicht nur existieren, sie müssen auch beschränkt sein. Für die Konvergenz des Taylor-Polynoms sind sie nötig, aber wenn wir auch andere Methoden der Polynomapproximation zulassen, wird die Situation weniger restriktiv. Tatsächlich sagt der *Approximationssatz von Weierstrass*: Für jede stetige Funktion $f : [a,b] \mapsto \mathbb{R}$ und eine positive Schranke ϵ lässt sich ein Polynom p finden, so dass $\sup_{x\in[a,b]} |f(x) - p(x)| < \epsilon$. Und wie zu erwarten, braucht man im Allgemeinen für kleineres ϵ einen höheren Polynomgrad. Die meisten Beweise dieses Satzes sind zwar konstruktiv, aber wegen der fehlenden Anforderungen an die Differenzierbarkeit von f recht kompliziert. Wir werden uns im Folgenden weiterhin auf differenzierbare Funktionen konzentrieren und eine praktische Methode mit guten Eigenschaften entwickeln. Der Grundgedanke ist, wie zuvor, die Funktion in vorgegebenen Punkten zu interpolieren.

6.4.2. Das Runge-Phänomen

Es erscheint intuitiv einleuchtend, das Intervall $[a,b]$ gleichmäßig abzutasten und dann die Funktion f an diesen Stelle zu interpolieren. Ein schönes Beispiel dafür, dass dies tatsächlich

```
import numpy as np
import matplotlib.pyplot as plt

def runge(x):
    return 1.0/(1.0+x*x)

x = np.linspace(-5.0, 5.0, 200)
plt.plot(x, runge(x), '-', zorder=0, linewidth=3, label='$(1+x^2)^{-1}$')

for k in (3,7,11,15):
    ix = np.linspace(-5.0,5.0,k)
    p = intpoly(ix,runge(ix))
    plt.plot(x, p(x), '-', linewidth=1, label='n={}'.format(k))

plt.grid(True)
plt.xlim(-5.0,5.0)
plt.ylim(-1.5,7.5)
plt.gca().set_aspect('equal')
plt.legend()

plt.show()
```

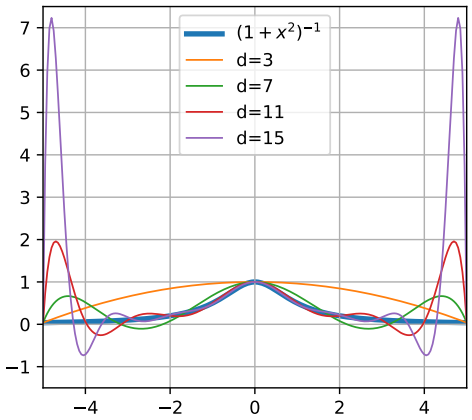


Abbildung 6.4.: *Die Funktion $f(x) = (1+x^2)^{-1}$ wird im Intervall $[-5,5]$ durch Interpolation von gleichabständigen Stützstellen approximiert. Mit steigendem Polynomgrad n wird die Approximation am Rande des Intervalls immer schlechter.*

keine gute Idee ist, stammt von Carl Runge. Wir betrachten die Funktion $f = (1+x^2)^{-1}$ auf dem Intervall $[-5,5]$. Stützstellen mit gleichen Abständen im Intervall sind definiert durch $x_i = 10i/n - 5$ für $i \in [0,1,\ldots,n], n > 0$. Diese Stützstellen definieren ein Polynom p_n vom Grad n, dass die Funktion $f(x)$ in den Stellen interpoliert – also $p_n(x_i) = f(x_i)$. Durch diese Bedingungen ist $p_n(x)$ eindeutig definiert und wir können es entweder über die Lösung des Vandermonde-Systems oder direkt durch Verwendung der Lagrange-Basis bestimmen. Abbildung 6.4 zeigt das Ergebnis der Interpolation für steigende Zahl von gleichabständigen Stützstllen. Der Abstand zwischen f und dem Interpolationspolynom $p_n(x)$ zwischen den Stütztellen an den Enden des Intervalls wird mit steigendem Grad n immer größer. In der Tat lässt sich zeigen,

dass die maximale Differenz zwischen f und p_n beliebig wächst:

$$\lim_{n\mapsto\infty} \sup_{x\in[-5,5]} |f(x) - p_n(x)| = \infty. \tag{6.41}$$

Es gibt mehrere Gründe für diese unbefriedigende Approximation. Um dies zu sehen, betrachten wir für die Lagrange-Interpolation ein Restglied:

$$f(x) = \sum_{i=0}^{n} f(x_i) l_i(x) + \frac{f^{(n+1)}(z)}{(n+1)!} \prod_{j=0}^{n} (x - x_i), \quad z \in [a, b] \tag{6.42}$$

Wie beim Taylor-Polynom gibt es also immer ein z (hier im Intervall $[a, b]$), so dass die Funktion f exakt dargestellt wird. Der Faktor $\frac{f^{(n+1)}(z)}{(n+1)!}$ ist identisch zum Lagrange-Restglied für die Taylor-Approximation. Dies mag man intuitiv so erklären, dass mit einem Polynom des Grades n genau die $n + 1$-te Ableitung nicht mehr gut approximiert werden kann. Das Polynom im Restglied

$$G_n(x) = \prod_{j=0}^{n} (x - x_i) \tag{6.43}$$

erinnert an die Zähler der Lagrange-Basispolynome. Allerdings fehlt hier keiner der Terme $(x - x_i)$, d.h. $G(x)$ hat Grad $n + 1$. Auch für diese Darstellung des Restglieds verzichten wir auf einen Beweis.

Eine einfache oberen Schranke für den Fehler ergibt sich aus oberen Schranken für beide Teile des Restglieds:

$$\sup_{x\in[a,b]} |f(x) - p_n(x)| \leq \frac{1}{(n+1)!} \left(\max_{z\in[a,b]} |f^{(n+1)}(z)| \right) \left(\max_{x\in[a,b]} |G_n(x)| \right). \tag{6.44}$$

Der erste Faktor hängt nur von der gegebenen Funktion f und dem festen Intervall $[a, b]$ ab. Hierauf haben wir keinen Einfluss. Der zweite Faktor ist bestimmt durch das betragsmäßig größte Extremum des Polynoms $G_n(x)$ (im Intervall $[a, b]$). Dieses Polynom ist nur von den Stützstellen x_i abhängig - und die können wir frei wählen. Um also das Ergebnis der Polynomapproximation der Funktion f zu verbessern, sollten wir die Stützstellen x_i so wählen, dass das Polynom $G_n(x)$ möglichst 'kleine' Extremstellen (im Intervall $[a, b]$) hat.

6.4.3. Tschebyschew-Polynome

Wir betrachten die Polynome, die durch

$$T_0(x) = 1 \tag{6.45}$$
$$T_1(x) = x \tag{6.46}$$
$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x), \quad n \geq 1 \tag{6.47}$$

gegeben sind. In der Rekursion wird der Polynomgrad von T_n durch Multiplikation mit $2x$ um eins erhöht – $T_n(x)$ ist also ein Polynom vom Grad n für alle n. Der Buchstabe T ist eine Referenz an Pafnuty Tschebyschew[1], nach dem sie benannt sind. Für die ersten Basen ergibt

[1] Der Name Tschebyshew (wissenschaftlich Transliteration Čebyšëv) wird in der Literatur, insbesondere je nach Sprache, in sehr unterschiedlicher Weise transkribiert

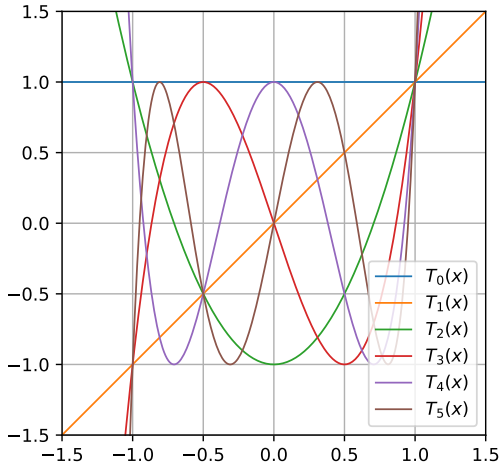


Abbildung 6.5.: *Die Tschebyshev-Polynome $T_n(x)$ bis zum Grad $n = 5$. Man sieht deutlich, dass alle Polynome ihre Nullstellen und kritischen Punkte im Intervall $[-1, 1]$ haben. Innerhalb dieses Intervalls gilt offenbar $|T_n(x)| \leq 1$. Außerhalb von $[-1, 1]$ wachsen bzw. fallen die Funktionswerte schnell.*

sich aus der Rekursionvorschrift Gl. 6.47

$$\begin{aligned} T_2 &= 2x^2 - 1 \\ T_3 &= 4x^3 - 3x \\ T_4 &= 8x^4 - 8x^2 + 1 \\ T_5 &= 16x^5 - 20x^3 + 5x \\ T_6 &= 32x^6 - 48x^4 + 18x^2 - 1 \\ T_7 &= 64x^7 - 112x^5 + 56x^3 - 7x. \end{aligned} \tag{6.48}$$

Die Tschebychev-Polynome bis zum Grad $n = 5$ sind in Abbildung 6.5 dargestellt.

Wir beschränken uns im Weiteren auf das Intervall $[-1, 1]$. In diesem Bereich haben die Tschebyschew-Polynome eine, vielleicht überraschende, explizite Darstellung:

$$T_n(x) = \cos(n \arccos x). \tag{6.49}$$

Um dies einzusehen, überprüfen wir zuerst die beiden Ansatzfunktionen $T_0(x) = \cos(0) = 1$ und $T_1(x) = \cos(\arccos x) = x$. Und dann stellen wir fest, dass sich die Rekursionsregel aus den Additionstheoremen für trigonometrische Funktionen ergeben. Wir verwenden die Regel

$$2 \cos\theta \cos\phi = \cos(\theta - \phi) + \cos(\theta + \phi) \tag{6.50}$$

und ersetzen in der Rekursion $\theta = \arccos x, x = \cos\theta$:

$$T_{n+1}(x) = \cos(n\theta + \theta) = 2\cos\theta\cos(n\theta) - \cos(n\theta - \theta) = 2xT_n(x) - T_{n-1}(x). \tag{6.51}$$

Die explizite Darstellung erleichtert es, Null- und Extremstellen der Polynome zu bestimmen. Die Kosinusfunktion hat ihre Nullstellen in $(k + \frac{1}{2})\pi$ für ganzzahlige k. Es ergibt sich

$$n \arccos x = \left(k + \frac{1}{2}\right)\pi \quad \Longrightarrow \quad x_k = \cos\left(\frac{(k + \frac{1}{2})\pi}{n}\right). \tag{6.52}$$

Wegen der Periodizität der Kosinusfunktion ist die Nullstelle für k identisch zu allen $k + 2jn, j \in \mathbb{Z}$; ferner sind wegen der Symmetrie der Kosinusfunktion auch die Nullstellen k und $-k - 1$

identisch. Es bleiben also genau n verschiedene Nullstellen, die wegen des Wertebereichs der Kosinusfunktion alle im Intervall $[-1, 1]$ liegen. Während es bereits klar war, dass Tschebyshew-Polynome vom Grad n, wie alle Polynome, genau n Nullstellen haben, zeigt unsere Untersuchung, dass alle Nullstellen reell sind und im Intervall $[-1, 1]$ liegen. Dies ist gut in Abbildung 6.5 zu sehen.

Für die Extremstellen leiten wir die explizite Darstellung der Tschebyshew-Polynome Gl. 6.49 ab und erhalten:

$$T_n'(x) = -\sin(n \arccos x) \frac{-n}{\sqrt{1-x^2}}. \tag{6.53}$$

Die Extremstellen sind Nullstellen dieser Funktion. Wir müssen uns jetzt auf das offene Intervall $]-1, 1[$ einschränken, denn für $x = \pm 1$ ist die Ableitung (in dieser Form) undefiniert. In diesem Bereich genügt es die Nullstellen von $\sin(n \arccos x)$ zu betrachten. Die Sinusfunktion ist Null für $k\pi$ und wir finden die Extremstellen

$$n \arccos x = k\pi \quad \Longrightarrow \quad x_k = \cos\left(\frac{k\pi}{n}\right). \tag{6.54}$$

Wiederum ergeben sich aus Periodizität und Symmetrie die Anzahl der verschiedenen Stellen. Diesmal sind aus Symmetriegründen die Stellen k und $-k$ identisch. Für $k = 0$ fallen diese Stellen zusammen, so dass es insgesamt $n + 1$ verschiedene Stellen gibt. Allerdings enthält diese Menge auch die Werte $x_k = \pm 1$, die nicht in dem von uns betrachteten offenen Intervall $]-1, 1[$ liegen. Da T_n ein Polynom ist, muss diese Ableitung existieren; andererseits wissen wir, dass T_n als Polynom vom Grad n nicht mehr als $n - 1$ kritische Punkte haben kann. Diese $n - 1$ kritischen Punkte haben wir gefunden.

Einsetzen der Extremstellen x_k in die explizite Darstellung ergibt

$$T_n(x_k) = \cos\left(n \arccos\cos\left(\frac{k\pi}{n}\right)\right) = \cos k\pi = (-1)^k. \tag{6.55}$$

Im Intervall $[-1, 1]$ hat die Funktion T_n also $n-1$ kritische Punkte mit den Werten ± 1 und zwei weitere Extremstellen am Rand. Im Intervall $[-1, 1]$ gilt also $|T_n(x)| \leq 1$ und die Extremwerte ± 1 werden an $n + 1$ Stellen eingenommen. Dies illustriert auch Abbildung 6.5.

Wir behaupten, dass diese Beschränkung für das Intervall $[-1, 1]$ in gewissen Sinne optimal ist. Offenbar kann man den Wertebereich jedes Polynoms durch Skalierung verändern. Um den Wertebereich von Polynomen mit Grad n zu vergleichen, fordern wir das der *führende Koeffizient*, also der Koeffizient des Terms x^n, eins ist. Wir betrachten also Polynome $p(x)$ der Form:

$$p(x) = x^n + a_{n-1}x^{n-1} + \ldots + a_1 x + a_0. \tag{6.56}$$

Die Tschebyshev-Polynom habe diese Eigenschaft nur für T_0 und T_1. Aus der Rekursionsvorschrift Gl. 6.47 ergibt sich unmittelbar als führender Koeffizient 2^{n-1} (siehe auch die expliziten Monomdarstellungen Gl. 6.48). Um die Tschebyshew-Polynome in die gewünschte Form zu bringen müssen wir also durch 2^{n-1} teilen - es ergibt sich $2^{1-n}T_n(x)$. Wegen $|T_n(x)| \leq 1$ im Intervall gilt:

$$|2^{1-n}T_n(x)| \leq 2^{1-n}, \quad x \in [-1, 1]. \tag{6.57}$$

Wir wollen nun zeigen: unter alle Polynomen vom Grad n gibt es kein Polynom $p(x)$ (mit führendem Koeffizienten eins), für das gilt $|p(x)| < 2^{1-n}$.

Hierfür gibt es einen schönen Widerspruchsbeweis. Wir betrachten die Funktion

$$q(x) = 2^{1-n}T_n(x) - p(x). \tag{6.58}$$

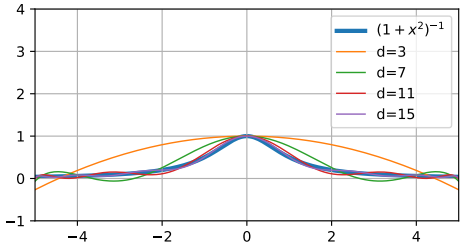


Abbildung 6.6.: *Die Runge-Funktion $f(x) = (1+x^2)^{-1}$ wird im Intervall $[-5, 5]$ durch Interpolation von in den Nullstellen eines skalierten Tschebyshew-Polynoms approximiert. Mit steigendem Polynomgrad n wird die Approximation immer besser.*

Sowohl $2^{1-n}T_n(x)$ wie auch $p(x)$ sind Polynome mit führendem Term x^n. Also ist die Differenz $q(x)$ ein Polynom mit höchstem Term x^{n-1}, ein Polynom vom Grad $n-1$. An den Stellen $\cos(k\pi/n)$ nimmt $2^{1-n}T_n(x)$ seine extremen Werte an, und zwar für steigende x-Werte immer abwechselnd Maxima und Minima. An diesen Stellen muss $q(x)$ für ein Maximum positiven Wert haben, denn $p(x)$ ist hier kleiner als 2^{1-n}; und für ein Minimum muss $q(x)$ negativ sein, denn hier ist $p(x)$ größer als -2^{1-n}. Von diesen abwechselnd positiven und negativen Werten von $q(x)$ gibt es $n+1$ im Intervall $[-1, 1]$. Als Polynom ist $q(x)$ stetig und muss zwischen den positiven und negativen Werten also n Nullstellen haben. Das ist aber für ein (nicht verschwindendes) Polynom vom Grad $n-1$ nicht möglich. Mithin gibt es keine Funktion $p(x)$, für die gilt $|p(x)| < 2^{1-n}$ und die skalierten Tschebyshev-Polynome $2^{1-n}T_n(x)$ haben unter alle Polynomen (mit führendem Koeffizienten eins) die kleinsten absoluten Werte im Intervall $[-1, 1]$.

Wir erinnern uns, dass wir auf der Suche nach einem Polynom $G(x) = \prod_{j=0}^{n}(x - x_i)$ waren, dass im Intervall $[a, b]$ möglichst klein bleibt. Offenbar hat $G(x)$ den führenden Koeffizienten eins (wenn man alle Klammern ausmultipliziert ergibt sich der Term x^n). Also haben wir (zumindest für das Intervall $[-1, 1]$) eine solches Polynom gefunden: $G(x) = 2^{1-n}T_n(x)$. Wir können sogar sofort die Nullstellen dieses Polynoms angeben, die wir für die Abtastung benötigen. Dies sind die Nullstellen des Tschebyshew-Polynoms vom Grad n, gegeben in Gl. 6.52.

Es bleibt zu überlegen, was wir für ein beliebiges Intervall $[a, b]$ machen sollten. Offenbar können wir die Nullstellen im Intervall $[-1, 1]$ einfach durch Skalierung und Verschiebung auf das Intervall $[a, b]$ abbilden, denn dies hätte keine Einfluss auf unserer Argumentation für die Optimalität der Tschebyshew-Polynome. Das heißt, im Allgemeinen wählt man als Stützstellen für Polynominterpolation im Intervall $[a, b]$:

$$x_i = \frac{b+a}{2} + \frac{b-a}{2}\cos\left(\frac{(i+\frac{1}{2})\pi}{n}\right) \tag{6.59}$$

Für diese Wahl von Sützstellen kehren wir zurück zu dem Beispiel von Runge. Wir ersetzen die gleichabständige Wahl der Stützstellen durch

```
ix = np.array([5.0*np.cos((i+0.5)*np.pi/k) for i in range(k)])
```

wobei der Faktor 5 auf das Intervall $[-5, 5]$ skaliert. Wie in Abbildung 6.6 zu sehen ist, wir nun die Approximation mit steigendem Grad immer besser.

7. Diskrete Fouriertransformation

7.1. Motivation

In diesem Abschnitt betrachten wir die Verarbeitung von *Signalen*. Unser Beispiel sollen Audiosignale sein, die wir als Amplitudenfunktion $\tilde{s} : \mathbb{R} \to \mathbb{R}$ über der Zeit verstehen könnten. Die Methoden, die wir herleiten, gelten aber in gleicher Weise auch für Signale/Funktionen, die von anderen Parametern als der Zeit abhängen, zum Beispiel dem Ort. Die Verarbeitung wird durch einen Operator $\tilde{\mathcal{A}} : (\mathbb{R} \to \mathbb{R}) \to (\mathbb{R} \to \mathbb{R})$ ausgedrückt, der ein Signal auf ein neues Signal abbildet. Speziell werden wir solche Operatoren nutzen, um Störgeräusche aus Audiosignalen zu entfernen.

Wie zuvor betrachten wir direkt eine diskrete Version des Signals / der Funktion – in diesem Fall diskretisieren wir die Zeit (äquidistant) über ein Intervall I – konkret verwenden wir $I = [0, n[$ und als Diskretisierungsschrittweite 1. So kann das Signal $\tilde{s}$ auf einem Intervall I durch den Vektor $\mathbf{s} \in \mathbb{R}^n$ von Amplituden s_t dargestellt werden. Der Index $t \in I \cap \mathbb{Z} = \{0, \dots, n-1\}$ eines Elements s_t steht für den Zeitpunkt. Das bedeutet der Vektor $\mathbf{s} \in \mathbb{R}^n$ ist eine diskrete Repräsention einer Funktion $\tilde{s} : \mathbb{R} \to \mathbb{R}$ auf dem Intervall I. Nur durch $\mathbf{s}$ ist es also unmöglich zu wissen, wie $\tilde{s}$ außerhalb von I definiert ist. Um nicht unterscheiden zu müssen, wie wir das Signal in der Nähe der Intervallgrenzen und in der Mitte des Signals behandeln, nehmen wir an, dass das Signal *periodisch* bzw. *zyklisch* fortgeführt wird. Wir gehen also davon aus, dass $\tilde{s}(t) = s_{t \bmod n}$ für alle $t \in \mathbb{Z}$.

Für den Moment mag man sich die Funktion $\tilde{s}$, bzw. die Elemente des Vektors $\mathbf{s}$, als reelle Zahlen vorstellen, aber wie wir gleich sehen werden, ist es für diesen Abschnitt sinnvoll komplexe Zahlen zu verwenden, also $\tilde{s} : \mathbb{R} \to \mathbb{C}$ und deshalb auch $\mathbf{s} \in \mathbb{C}^n$.

Für die Verarbeitung des diskreten Signals ergibt sich daher ein Operator $\mathbf{A} : \mathbb{C}^n \to \mathbb{C}^n$. Diesen beschränken wir in zweierlei Weise:

1. Wir betrachten nur lineare Operatoren $\mathbf{A}$. Das heißt, $\mathbf{A}$ lässt sich als Matrixmultiplikation umsetzen.
2. Die Verarbeitung ist *zeitinvariant* (alternativ: ortsinvariant, translationsinvariant, oder stationär, wenn das Signal vom Ort abhängt). Das bedeutet, ob wir ein Signal zunächst (periodisch) verschieben und dann $\mathbf{A}$ anwenden, oder ob wir zuerst $\mathbf{A}$ anwenden und das verarbeitete Signal dann (periodisch) verschieben ergibt das selbe Ergebnis.

Die Begründung für diese Einschränkung ergibt sich durch Systeme, wie sie in der Natur vorkommen. Die Veränderung eines Signals lässt sich in der Tat häufig ausreichend gut durch lineare Operationen beschreiben. Und Zeitinvarianz bedeutet lediglich, dass die Operation nicht vom Zeitpunkt abhängt – so wie wir bspw. für unser Ohr richtig vermuten, dass sich die Übertragung durch das Trommelfell von einem Moment zum nächsten nicht verändert. Es kommt hinzu, dass viele Operatoren in der Mathematik diese Eigenschaften besitzen. Zum Beispiel ist die Operationen „Ableiten“ linear und stationär.

Die Veränderung des Signals nennt man auch *filtern* und die Anwendung des Operators zum filtern heißt *falten*. Die Matrix $\mathbf{A}$ kann man deswegen auch auch als *Filter* betrachten und die entsprechende Matrixmultiplikation heißt entsprechend *Faltung*. Im folgenden wollen wir uns

überlegen, was diskrete lineare zeitinvariante Filter genau machen. Anschließend werden wir einen Weg finden, wir wir Signale schneller filtern können, als die oben beschriebene Operation (Matrixmultiplikation) nahelegt.

7.2. Diskrete (zyklische) Faltung

Im folgenden wollen wir den Faltungsoperator genauer betrachten und uns überlegen was es bedeutet, dass die Matrixmultiplikation in diesem Fall einen zeitinvarianten Operator darstellt. Wie bereits beschrieben bedeutet zeitinvariant, dass es keine Rolle spielt, ob wir das Signal zunächst verschieben und dann den Operator anwenden, oder ob wir zuerst den Operator anwenden und dann das Ergebnis verschieben. Im kontinuierlichen Fall heißt das

$$\tilde{A} \circ \tilde{s}(x+t) = \left(\tilde{A} \circ \tilde{s}\right)(x+t) \tag{7.1}$$

Wie könnnen wir die Verschiebung $(x+t)$ als diskreten Operator modellieren? Die „Zeit“ wird in der diskreten Version als Index der Elemente von $\mathbf{s}$ dargestellt. Wir müssen also die Koeffizienten des Vektors verschieben. Man kann sich leicht davon überzeugen, dass die *zyklische Verschiebungsmatrix*

$$\mathbf{Z} = \begin{pmatrix} 0 & 1 & 0 & 0 & \dots & 0 \\ 0 & 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 1 \\ 1 & 0 & 0 & 0 & \dots & 0 \end{pmatrix} \tag{7.2}$$

genau dies tut: es gilt $(\mathbf{Zs})_i = s_{i+1}$, wobei Indizes immer modulo n zu verstehen sind. Aus der Anschauung sollte unmittelbar klar sein, was die Matrix $\mathbf{Z}^k$ macht: sie verschiebt die Einträge des Vektors zyklisch um k Stellen, d.h. die kontinuierliche Verschiebung t wird hier durch eine diskrete Verschiebung k beschrieben. Die Matrixdarstellung der linearen, zeitinvarianten Operation mit Matrix $\mathbf{A}$ führt nun auf folgende Eigenschaft:

$$\mathbf{A}\mathbf{Z}^k\mathbf{s} = \mathbf{Z}^k\mathbf{A}\mathbf{s}. \tag{7.3}$$

Die Matrizen $\mathbf{A}$ und $\mathbf{Z}^k$ können also vertauscht werden, d.h. sie kommutieren.

Durch diese Eigenschaft können wir die Struktur von $\mathbf{A}$ bestimmen. Sei $\mathbf{a}'$ die erste Spalte aus $\mathbf{A}$. Der Vektor $\mathbf{a}'$ ergibt sich also, wenn man $\mathbf{A}$ auf den Vektor $(1,0,0,\dots,0)^\mathsf{T}$ anwendet:

$$\mathbf{a}' = \begin{pmatrix} a_0 \\ a_{n-1} \\ a_{n-2} \\ \vdots \\ a_1 \end{pmatrix} = \mathbf{A} \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}. \tag{7.4}$$

Die ungewöhnliche Indizierung in $\mathbf{a}'$ wird sich hoffentlich gleich erschließen. Wir wissen, dass $\mathbf{AZx} = \mathbf{ZAx}$. Wählen wir $\mathbf{x} = (1,0,0,\dots,0)^\mathsf{T}$, so ergibt sich

$$\mathbf{ZA} \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \mathbf{Z} \begin{pmatrix} a_0 \\ a_{n-1} \\ a_{n-2} \\ \vdots \\ a_1 \end{pmatrix} = \begin{pmatrix} a_{n-1} \\ a_{n-2} \\ \vdots \\ a_1 \\ a_0 \end{pmatrix} = \mathbf{A} \begin{pmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix} = \mathbf{AZ} \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}. \tag{7.5}$$

Somit kennen wir die letzte Spalte aus $\mathbf{A}$. Dies lässt sich genauso für die anderen Spalten weiterführen. Wir können also alle Spalten aus $\mathbf{A}$ durch Verschiebungen von $\mathbf{a}'$ beschreiben:

$$\mathbf{A} = \begin{pmatrix} a_0 & a_1 & a_2 & \dots & a_{n-2} & a_{n-1} \\ a_{n-1} & a_0 & a_1 & \dots & a_{n-3} & a_{n-2} \\ a_{n-2} & a_{n-2} & a_0 & \dots & a_{n-4} & a_{n-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ a_1 & a_2 & a_3 & \dots & a_{n-1} & a_0 \end{pmatrix}. \tag{7.6}$$

Wir stellen fest, dass in A nur die n unterschiedlichen Koeffizienten $a_0, \dots, a_{n-1}$ vorkommen. Wir haben offenbar die Indizes nach den Spalten gewählt und nicht nach den Zeilen. Der wesentliche Grund für diese Wahl ist die folgende Repräsentation als Linearkombination der Potenzen von $\mathbf{Z}$:

$$\mathbf{A} = a_0\mathbf{I} + a_1\mathbf{Z} + a_2\mathbf{Z}^2 + \ldots + a_{n-1}\mathbf{Z}^{n-1} = \sum_{k=0}^{n-1} a_k\mathbf{Z}^k. \tag{7.7}$$

Die Potenzen von $\mathbf{Z}^k$ repräsentieren eine zyklische Verschiebung des Vektors um k Stellen und bilden offenbar eine Basis aller Faltungsmatrizen. Der Vektor $\mathbf{a}'$ bestimmt die Matrix $\mathbf{A}$ vollständig. Wir können den Faltungsoperator auch als Funktion schreiben, die nur von $\mathbf{a}'$ abhängig ist:

$$\mathbf{a}' * \mathbf{s} := \mathbf{A}\mathbf{s}. \tag{7.8}$$

Der Vektor $\mathbf{a}'$ heißt daher auch *Kern* des Faltungsoperators $*$. Liegt ein unbekannter Filter vor, so kann man den Kern bestimmen, in dem man das Ergebnis des Filters für das Signal $(1, 0, \dots, 0)^\mathsf{T}$, den sog. *Einheitsimpuls* betrachtet. Das Ergebnis nennt man *Impulsantwort*.

Diese Darstellung der Faltung liefert auch gleich eine obere Schranke auf die Komplexität der Faltungsoperation für Vektoren mit n Elementen, nämlich $O(n^2)$. Eine wichtige nichttriviale Erkenntnis dieses Abschnitts wird darin liegen, dass sich die Faltung auch in $O(n \log n)$ ausrechnen lässt.

7.3. Diskrete Fouriertransformation

Um mehr über die Eigenschaften der diskreten Faltung zu erfahren wollen wir die Eigenvektoren der Matrix $\mathbf{A}$ finden. Die Eigenvektoren sind nämlich besondere Signale bzgl. der Faltung mit $\mathbf{a}$, nämlich solche, die sich bis auf Multiplikation mit einer (komplexen) Zahl nicht verändern.

Aber wie können wir die Eigenvektoren von $\mathbf{A}$ bestimmen, wenn die Matrix $\mathbf{A}$ von den Koeffizienten des Vektors $\mathbf{a}$ abhängt? Wir machen eine Beobachtung: Nehmen wir an wir kennen einen Eigenvektor $\mathbf{v}$ der Verschiebungsmatrixatrix $\mathbf{Z}$, also $\mathbf{Z}\mathbf{v} = \lambda\mathbf{v}$. Damit gilt auch

$$\mathbf{Z}^k\mathbf{v} = \mathbf{Z}^{k-1}\mathbf{Z}\mathbf{v} = \mathbf{Z}^{k-1}\lambda\mathbf{v} = \lambda\mathbf{Z}^{k-1}\mathbf{v} = \ldots = \lambda^k\mathbf{v} \tag{7.9}$$

Diese Eigenschaft können wir in der Basisdarstellung von $\mathbf{A}$ verwenden

$$\begin{aligned}\mathbf{A}\mathbf{v} &= \left(\sum_{k=0}^{n-1} a_k \mathbf{Z}^k\right)\mathbf{v} \\ &= \sum_{k=0}^{n-1} a_k \mathbf{Z}^k \mathbf{v} \\ &= \sum_{k=0}^{n-1} a_k \lambda^k \mathbf{v} \\ &= \left(\sum_{k=0}^{n-1} a_k \lambda^k\right)\mathbf{v}\end{aligned} \tag{7.10}$$

und sehen, dass $\mathbf{v}$ auch ein Eigenvektor von $\mathbf{A}$ ist! Wenn wir also n linear unabhängige Eigenvektoren von $\mathbf{Z}$ finden können, so haben wir auch (alle) n linear unabhängigen Eigenvektoren von $\mathbf{A}$ gefunden – ganz unabhängig von den konkreten Koeffizienten des Vektors $\mathbf{a}$.

7.3.1. Eigenwerte von $\mathbf{Z}$

Um die Eigenvektoren von $\mathbf{Z}$ zu bestimmen, suchen wir zunächst nach den Eigenwerten von $\mathbf{Z}$. Hierzu verwenden wir eine einfache Beobachtung: Da das Produkt mit der Matrix $\mathbf{Z}$ einen n-Vektor um ein Element zyklisch verschiebt, bildet die Matrix $\mathbf{Z}^n$ den Vektor auf sich selbst ab, also $\mathbf{Z}^n = \mathbf{I}$. Damit haben wir

$$\mathbf{Z}^n\mathbf{v} = \lambda^n\mathbf{v} = \mathbf{I}\mathbf{v} \qquad \Longrightarrow \qquad \lambda^n = 1 \tag{7.11}$$

Die Eigenwerte von $\mathbf{Z}$ sind also die sogenannten n-ten *Einheitswurzeln*

7.3.2. Einheitswurzeln

Für $n = 1$ bleibt nur $\lambda = 1$, für $n = 2$ kommt als zusätzliche Lösung $\lambda = -1$ hinzu. In den reellen Zahlen sind dies die einzigen Lösungen der Gleichung $\lambda^n = 1$. Nach dem Fundamentalsatz der Algebra hat aber das Polynom $\lambda^n - 1$ genau n Nullstellen. Dafür müssen wir allerdings auch komplexe Zahlen in Betracht ziehen. In der Tat finden wir für $n = 4$ zum Beispiel auch die Lösung $\lambda = i$, denn $i^4 = i^2 \cdot i^2 = (-1) \cdot (-1) = 1$.

Allgemein betrachten wir eine komplexe Zahl in Exponentialschreibweise $r \cdot e^{i\phi}$. Dabei ist r der Abstand der Zahl zum Ursprung und ϕ der Winkel gegenüber der reellen Achse. Aus $\lambda^n = r^n e^{i\phi n} = 1$ folgt dann direkt wegen $e^{2\pi i} = 1$ die Lösung

$$\omega_n = e^{2\pi i/n} \tag{7.12}$$

Damit sind aber auch alle ganzzahligen Potenzen ω_n^k Lösungen der Gleichung $\lambda^n = 1$. Wegen $\omega_n^n = 1$ wiederholen sich die Einheitswurzeln für höhere Potenzen: $\omega_n^{k+jn} = \omega_n^k$. Es gibt also genau n verschiedenen Einheitswurzeln der Form ω_n^k. Diese n Einheitswurzeln sind die Eigenwerte der Matrix $\mathbf{Z}$.

7.3.3. Eigenvektoren von $\mathbf{Z}$

Mit den Einheitswurzeln als Eigenwerten von $\mathbf{Z}$ bestimmen wir nun die Eigenvektoren. Es gilt $\mathbf{Z}\mathbf{v} = \omega_n^k \mathbf{v}$. Da das Produkt mit $\mathbf{Z}$ den Eigenvektor $\mathbf{v}$ zyklisch verschiebt, bedeutet das

$$v_{i+1} = \omega_n^k v_i \tag{7.13}$$

Wir setzen $v_0 = 1$ und erhalten damit direkt $v_1 = \omega_n^k$, und dann $v_2 = \omega_n^k \cdot \omega_n^k = \omega_n^{2k}$, usw. Allgemein ergeben sich damit die Eigenvektoren

$$\mathbf{v}_k = \begin{pmatrix} 1 \\ \omega_n^k \\ \omega_n^{2k} \\ \vdots \\ \omega_n^{(n-1)k} \end{pmatrix} \tag{7.14}$$

Es bleibt zu klären, ob diese Eigenvektoren eine Basis bilden. Dazu betrachten wir das (komplexe) Skalarprodukt zweier Eigenvektoren:

$$\begin{aligned} \mathbf{v}_j^* \mathbf{v}_k &= 1 + \overline{\omega_n^j}\omega_n^k + \overline{\omega_n^{2j}}\omega_n^{2k} + \ldots + \overline{\omega_n^{(n-1)j}}\omega_n^{(n-1)k} \\ &= 1 + \omega_n^{-j}\omega_n^k + \omega_n^{-2j}\omega_n^{2k} + \ldots + \omega_n^{-(n-1)j}\omega_n^{(n-1)k} \\ &= 1 + \omega_n^{k-j} + \omega_n^{2(k-j)} + \ldots + \omega_n^{(n-1)(k-j)} \end{aligned} \tag{7.15}$$

Für $j = k$ sind alle Exponenten 0 und damit $\mathbf{v}_k^* \mathbf{v}_k = n$. Die Eigenvektoren können also mit dem Faktor $1/\sqrt{n}$ in der Länge normiert werden. Für $j \neq k$ multiplizieren wir auf beiden Seiten mit ω_n^{k-j}

$$\omega_n^{k-j} \mathbf{v}_j^* \mathbf{v}_k = \omega_n^{k-j} + \omega_n^{2(k-j)} + \omega_n^{3(k-j)} \ldots + \omega_n^{n(k-j)} \tag{7.16}$$

und ziehen das Ergebnis von der Gleichung für $\mathbf{v}_j^* \mathbf{v}_k$ ab:

$$(1 - \omega_n^{k-j}) \mathbf{v}_j^* \mathbf{v}_k = 1 - \omega_n^{n(k-j)} = 0. \tag{7.17}$$

Wegen $j \neq k$ und damit $\omega_n^{k-j} \neq 1$ ist also $\mathbf{v}_j^* \mathbf{v}_k = 0$, die Eigenvektoren $\mathbf{v}_j$ und $\mathbf{v}_k$ also orthogonal.

Es lohnt sich eine Intuition für die Basis aufzubauen. Wie wir wissen, lässt sich der komplexe Einheitskreis durch $\mathrm{ccw}_1(t) = e^{\frac{2i\pi t}{n}}$ mit $t \in [0, n[$ beschreiben. Dabei dreht $\mathrm{ccw}_1(t)$ mit wachsenden t gegen den Uhrzeigersinn einmal um den Ursprung (ccw steht hier für den englischen Begriff *counterclockwise*). Erweitert man diese Funktion durch einen Faktor k, $\mathrm{ccw}_k(t) = e^{\frac{k2i\pi t}{n}}$, so dreht sich die Funktion auf dem Intervall $t \in [0, n[$ k mal um den Ursprung. Diskretisiert man ccw_k, so wie wir unsere Signale diskretisieren, ergibt sich der Eigenvektor $\mathbf{v}_k$. Man kann also $\mathbf{v}_k$ als eine diskrete Funktion ansehen, die k mal gegen den Uhrzeigersinn um den Ursprung rotiert. Möchte man im Uhrzeigersinn rotieren, so muss man nur den Exponenten negieren oder $k < 0$ wählen: $\mathrm{cw}_k(t) = e^{\frac{-k2i\pi t}{n}} = \mathrm{ccw}_{-k}(t)$. Interessanterweise erhalten wir $\mathbf{v}_{n-k}$, wenn wir $\mathrm{ccw}_{-k}(t)$ diskretisieren. Wir können also $\mathbf{v}_{n-k}$ als Funktion ansehen, die $n-k$ mal *gegen* den Uhrzeigersinn um den Ursprung rotiert; oder auch als Funktion, die k mal *im* Uhrzeigersinn um den Ursprung rotiert. Nehmen wir an, dass das Intervall $[0, n[$ einer Sekunden entspricht. Dann entsprechen k Rotationen genau Frequenzen (Schwingungen pro Sekunde). Die Eigenvektoren werden daher häufig wie folgt interpretiert: Für gerades n als

$$(\underbrace{\mathbf{v}_0}_{\text{konstant}}, \underbrace{\mathbf{v}_1}_{\text{Frequenz } 1}, \ldots, \underbrace{\mathbf{v}_{\lfloor \frac{n}{2} \rfloor}}_{\text{Frequenz } \frac{n}{2} = \text{Frequenz } -\frac{n}{2}}, \ldots, \underbrace{\mathbf{v}_{n-1}}_{\text{Frequenz } -1}), \tag{7.18}$$

bzw. für ungerades n

$$\Big(\underbrace{\mathbf{v}_0}_{\text{konstant}}, \underbrace{\mathbf{v}_1}_{\text{Frequenz } 1}, \ldots, \underbrace{\mathbf{v}_{\frac{n-1}{2}}}_{\text{Frequenz } \frac{n-1}{2}}, \underbrace{\mathbf{v}_{n-\frac{n-1}{2}}}_{\text{Frequenz } -\frac{n-1}{2}}, \ldots, \underbrace{\mathbf{v}_{n-1}}_{\text{Frequenz } -1}\Big). \tag{7.19}$$

Wir sehen eine Symmetrie zur Mitte des Vektors, wenn man das erste Element, die konstante Funktion, auslässt. Für den Fall, dass n gerade ist, fallen die Frequenzen $\frac{n}{2}$ und $-\frac{n}{2}$ zusammen. Da sich die Rotationen gegen den Uhrzeigersinn nur im Vorzeichen des imaginären Anteils von den Rotation im Uhrzeigersinn unterscheiden, sollten wir die Frequenzen am Anfang und Ende des Vektors als 'niedrig' verstehen. Die Repräsentanten der 'hohen' Frequenzen liegen also in der Mitte des Vektors (und nicht etwa am Ende). Diese Symmetrie wird uns im weiteren Verlauf noch häufiger begegnen und ist insbesondere für die praktischen Anwendung, also die Konstruktion von Filtern mit gewünschten Eigenschaften, zu beachten.

7.3.4. Die DFT Matrix und die diskrete Fouriertransformation

Die normierten Eigenvektoren $\frac{1}{\sqrt{n}}\mathbf{v}_k$ bilden die DFT-Matrix.

$$\Omega_n = \frac{1}{\sqrt{n}}\left(\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_{n-1}\right) = \frac{1}{\sqrt{n}}\begin{pmatrix} 1 & 1 & 1 & 1 & \ldots & 1 \\ 1 & \omega_n & \omega_n^2 & \omega_n^3 & \ldots & \omega_n^{n-1} \\ 1 & \omega_n^2 & \omega_n^4 & \omega_n^6 & \ldots & \omega_n^{2(n-1)} \\ 1 & \omega_n^3 & \omega_n^6 & \omega_n^9 & \ldots & \omega_n^{3(n-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega_n^{n-1} & \omega_n^{2(n-1)} & \omega_n^{3(n-1)} & \ldots & \omega_n^{(n-1)(n-1)} \end{pmatrix} = \{\omega_n^{ij}\}. \tag{7.20}$$

Das Produkt eines Vektors (Signals) mit der DFT-Matrix nennen wir *Fouriertransformation*:

$$\hat{\mathbf{s}} = \Omega_n \mathbf{s}. \tag{7.21}$$

Da das Signal $\mathbf{s}$ dabei bzgl. der Eigenbasis dargestellt wird und die Eigenbasis aus Frequenzfunktionen besteht, spricht man auch von einer Darstellung im *Frequenzraum*. Auch wenn $\mathbf{s}$ bzgl. der Zeit definiert ist, spricht man hier meist vom *Ortsraum*. Die Fouriertransformation überträgt also Signale vom Ortsraum in den Frequenzraum.

Die *inverse Fouriertransformation* führt die entgegengesetzte Transformation durch, nämlich vom Frequenzraum in den Ortsraum. Sie ergibt sich direkt durch Invertieren der DFT-Matrix:

$$\mathbf{s} = \Omega_n^{-1}\hat{\mathbf{s}}. \tag{7.22}$$

Achtung: ein Teil der Literatur definiert die Forwärtstransformation ohne den Normierungsfaktor $\frac{1}{\sqrt{n}}$. Dann enthält die inverse Transformation den Faktor $\frac{1}{n}$.

Die Matrix Ω_n ist nach unseren Vorüberlegungen unitär (die komplexe Analogie zu orthogonalen Matrizen). Die Inverse von Ω_n ist daher die adjungierte Matrix Ω_n^*, also die Matrix, die durch Transposition und komplexe Konjugation der Einträge entsteht. Aus $\Omega_n = \{\omega_n^{ij}\}$ folgt direkt, dass Ω_n symmetrisch und daher identisch mit ihrer Transponierten ist. Die Inverse von Ω_n ist also die komplex Konjugierte $\overline{\Omega_n}$. Aus der Sicht der Informatik ist es interessant, die Struktur

von $\overline{\Omega_n}$ genauer zu betrachten: wegen $\overline{\omega_n^k} = \omega_n^{-k} = \omega_n^{n-k}$ geht $\overline{\Omega_n}$ aus Ω_n durch Spiegelung der Spalten 1 bis $n-1$ hervor:

$$\Omega^{-1} = \Omega^* = \overline{\Omega_n} = \frac{1}{\sqrt{n}} \left(\mathbf{v}_{0\equiv n}, \mathbf{v}_{n-1}, \ldots, \mathbf{v}_1\right) = \Omega_n \begin{pmatrix} 1 & 0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & 0 & \ldots & 0 & 1 \\ 0 & 0 & 0 & \ldots & 1 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 1 & \ldots & 0 & 0 \\ 0 & 1 & 0 & \ldots & 0 & 0 \end{pmatrix} = \Omega_n \mathbf{J} \quad (7.23)$$

Die Matrix $\mathbf{J}$ erhält bei einem Vektor das erste Element und spiegelt die Reihenfolge der verbleibenden Elemente. Diese Operation ist im Kontext der Signale als Funktionen der Zeit oder des Ortes zu sehen: Multiplikation bedeutet, dass Zeitsignal „rückwärts'" zu betrachten, oder ein Ortssignal „in die andere Richtung" zu verstehen. Offensichtlich ist $\mathbf{J}$ als Permutation eine Orthogonalmatrix. Da sie außerdem symmetrisch ist, ist sie ihre eigene Inverse $\mathbf{J} = \mathbf{J}^\mathsf{T} = \mathbf{J}^{-1}$ – eine *Involution*. Diese Eigenschaft ist auch aus dem Effekt der Operation klar: die zweimalige Spiegelung der Elemente stellt die ursprüngliche Reihenfolge wieder her. Mit Hilfe von $\mathbf{J}$ können wir nun auch auf einfache Weise beschreiben, in welchem Verhältnis die erste Zeile $\mathbf{a}$ einer Filtermatrix $\mathbf{A}$ zu deren Kern $\mathbf{a}'$ (der ersten Spalte von $\mathbf{A}$) steht:

$$\mathbf{a}' = \mathbf{J}\mathbf{a} \quad \Longleftrightarrow \quad \mathbf{a} = \mathbf{J}\mathbf{a}' \quad (7.24)$$

Man betrachte nun den unteren rechten $(n-1) \times (n-1)$-Block von Ω_n, also den Teil, der entsteht, wenn man die erste Zeile und erste Spalte entfernt. Diese quadratische Matrix ist nicht nur symmetrisch bzgl. der Hauptdiagonalen, sondern auch symmetrisch bzgl. der Gegendiagonalen, denn es gilt:

$$\omega_n^{(n-i)(n-j)} = \omega_n^{n^2-ni-nj+ij} = \omega_n^{ij}. \quad (7.25)$$

Daraus folgt, dass man sowohl die letzten $n-1$ Spalten wie auch die letzten $n-1$ Zeilen von Ω_n vertauschen kann und dabei in beiden Fällen die Inverse von Ω_n erhält:

$$\overline{\Omega_n} = \Omega_n \mathbf{J} = \mathbf{J}\Omega_n. \quad (7.26)$$

Dieses Verständnis der inversen Fouriertransformation ist für die Implementierung von Interesse. Wir brauchen also nur eine Funktion, die die Vorwärtstransformation berechnet. Die inverse Transformation erhalten wir einfach dadurch, dass wir entweder im Signal die Reihenfolge der Elemente vertauschen, oder in dem wir im Ergebnis die Reihenfolge vertauschen. Diese Symmetrie zeigt auch, dass Vorwärts- und Rückwärtstransformation austauschbar sind. Insbesondere hätten wir nach unseren Bezeichnungen statt Ω_n die Inverse $\overline{\Omega_n}$ erhalten, wenn wir z.B. die Einheitswurzeln als $\omega_n = e^{-2\pi i/n}$ statt mit positivem Vorzeichen wie in Gl. 7.12 definiert hätten.

Nochmal Achtung: In der Tat wird in der Literatur und auch in Implementierungen oft die zuletzt genannte Konvention verwendet. Offenbar macht dieser Unterschied in der Drehrichtung bei der Definition der Einheitswurzeln außer in der Reihenfolge der Eigenvektoren keinen grundsätzlichen Unterschied. Aber man sollte sich stets vergewissern, welche Definition verwendet wird.

7.4. Faltungssatz

Jetzt wollen wir uns überlegen, wie die Fouriertransformation zu einem besseren Verständis der Faltungsoperation und insbesondere bei der Berechnung der Faltung von Nutzen sein kann.

Dazu betrachten wir die Fouriertransformation der Faltung $\mathbf{a}' * \mathbf{s} = \mathbf{A}\mathbf{s}$. Die Spalten von Ω_n sind Eigenvektoren von $\mathbf{Z}$ und damit auch $\mathbf{A}$, also

$$\mathbf{A}\Omega_n = \Omega_n\Lambda \quad \Longleftrightarrow \quad \mathbf{A} = \Omega_n\Lambda\Omega_n\mathbf{J} \tag{7.27}$$

wobei die Diagonalmatrix Λ die Eigenwerte von $\mathbf{A}$ enthält. Diese Eigenwerte finden wir, in dem wir noch einmal das Produkt eines Eigenvektors von $\mathbf{Z}$ mit $\mathbf{A}$ betrachten und gleich für die Eigenwerte von $\mathbf{Z}$ die Lösung ω_n^k eintragen:

$$\mathbf{A}\mathbf{v}_k = \left(\sum_{j=0}^{n-1} a_j \left(\omega_n^k\right)^j\right)\mathbf{v}_k. \tag{7.28}$$

Wir erhalten also für die Eigenwerte von $\mathbf{A}$:

$$\lambda_k = \sum_{j=0}^{n-1} a_j\omega_n^{kj} = \mathbf{v}_k^{\mathsf{T}}\mathbf{a}. \tag{7.29}$$

Der Vektor aller Eigenwerte ist also die Fouriertransformation des Filterkerns und damit erhalten wir die Diagonalmatrix

$$\Lambda = \operatorname{diag}\left(\Omega_n\mathbf{a}\right) = \operatorname{diag}\left(\Omega_n\mathbf{J}\mathbf{a}'\right) \tag{7.30}$$

Damit ergibt sich für die Faltung der Vektoren $\mathbf{a}$ und $\mathbf{s}$:

$$\mathbf{a}' * \mathbf{s} = \mathbf{A}\mathbf{s} = \Omega_n \ \left(\operatorname{diag}\left(\Omega_n\mathbf{J}\mathbf{a}'\right) \ \Omega_n\mathbf{J}\mathbf{s}\right). \tag{7.31}$$

Diese Operationen können wir wie folgt in Worte fassen: Die Faltung von $\mathbf{a}'$ und $\mathbf{s}$ können wir berechnen, indem wir zunächst sowohl von $\mathbf{a}$ wie auch von $\mathbf{s}$ die inverse Fouriertransformation bestimmen; die resultierenden Vektoren multiplizieren wir dann elementweise und wenden die Fouriertransformation an. Bei diesem Vorgehen können wir die Rollen der Fouriertransformation und ihrer Inverse ohne Einschränkung vertauschen. Man kann also auch (und das ist das übliche Vorgehen) zunächst das Signal und den Filterkern vorwärts transformieren, dann Produkt bilden, und zuletzt die inverse Fouriertransformation durchführen. Diese Vertauschung mag auch erklären, warum in der Literatur oft die Rollen von Vorwärts- und Rückwärtstransformation gegenüber der hier gewählten Konvention vertauscht sind. So oder so, wird das Verhältnis von Faltung und Fouriertransformation typischerweise wie folgt zusammengefasst:

> Faltung im Ortsraum entspricht Produktbildung im Frequenzraum.

Unsere Überlegungen zu den Eigenschaften der inversen Transformation führen ohne Umwege auf die dazu duale Aussage

> Produktbildung im Ortsraum entspricht Faltung im Frequenzraum.

Diese Art die Faltung zu bestimmen, also ein Signal zu filtern, ist also zumindest im Frequenzraum sehr effizient, denn es müssen nur Elemente mit gleichem Index miteinander multipliziert werden. Die Komplexität der Operationen im Frequenzraum ist also $O(n)$. Um die gesamte Operation effizient zu machen, müssen wir uns nun überlegen, ob die Fouriertransformation schneller als in $O(n^2)$ durchgeführt werden kann.

7.5. Beispiele

Impuls (Abbildung 7.1).

Sinus und Kosinus (Abbildung 7.2).

Kosinus mit Phasenverschiebung Wir wollen die Fouriertransformierte des Signals

$$\mathbf{z} \in \mathbb{C}^8, \quad z_j = \cos(2\pi j/8 + \varphi) = \frac{e^{2\pi i\, j/n+\varphi} + e^{-2\pi i\, j/n+\varphi}}{2}$$

mit $\varphi \in \mathbb{R}$ bestimmen (Abbildung 7.3).

7.6. Rekursive diskrete Fouriertransformation

Als Strategie für eine schnelle Berechnung der Fouriertransformation, versuchen wir das Teile-und-Herrsche Prinzip anzuwenden: können wir die diskrete Fouriertransformation ausrechnen, indem wir das Signal aufteilen, diskrete Fouriertransformation auf den kleineren Teilen anwenden und dann die Ergebnisse geeignet kombinieren?

Wir versuchen, die diskrete Fouriertransformation für Signale der Länge $2n$ durch diskrete Fouriertransformationen der Länge n zu bestimmen. Dazu betrachten wir die DFT Matrizen der Größen $2b \times 2n$ und $n \times n$, die offenbar Potenzen der Einheitswurzeln $\omega_{2n} = e^{2\pi i/2n}$ und $\omega_n = e^{2\pi i/n}$ enthalten. Es gilt offenbar $\omega_{2n}^2 = \omega_n$, oder allgemeiner $\omega_{2n}^{2k} = \omega_n^k$. Die Matrix Ω_{2n} enthält in Zeile j und Spalte k den Eintrag ω_{2n}^{jk}. Die Einträge in allen Zeilen oder Spalten mit geradem Index führen auf einen geraden Exponenten jk und könnten auch als Potenz der Einheitswurzel ω_n ausgedrückt werden.

Jedes Element des Signals wird auf eine Spalte multipliziert. Wir können also einfach nach Spalten unterscheiden. Wir sortieren in der DFT Matrix alle Spalten mit geradem Index nach vorne. Darauf folgen die Spalten mit ungeradem Index. Das Signal sortieren wir in entsprechender Weise um, also zunächst die Elemente mit geradem Index und dann mit ungeradem Index. Diese Umordnung lässt sich ausdrücken durch Multiplikation mit einer Permutationsmatrix

$$\mathbf{P} = \begin{pmatrix} 1 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 1 & 0 \\ 0 & 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 0 & 1 \end{pmatrix} \tag{7.32}$$

Ein Signal $\mathbf{s}$ wird umgeordnet durch Multiplikation mit $\mathbf{P}$ (Anwendung der Permutation auf die Zeilen), lso erhält man das umgeordnete Signal als $\mathbf{Ps}$. Die Spalten der DFT-Matrix werden umgeordnet durch Multiplikation der transponierten Permutationsmatrix von rechts, also $\Omega_{2n}\mathbf{P}^\mathsf{T}$. Permutationsmatrizen sind orthogonal, es ergibt sich wie gewünscht $\Omega_{2n}\mathbf{P}^\mathsf{T}\,\mathbf{Ps} = \Omega_{2n}\mathbf{s}$.

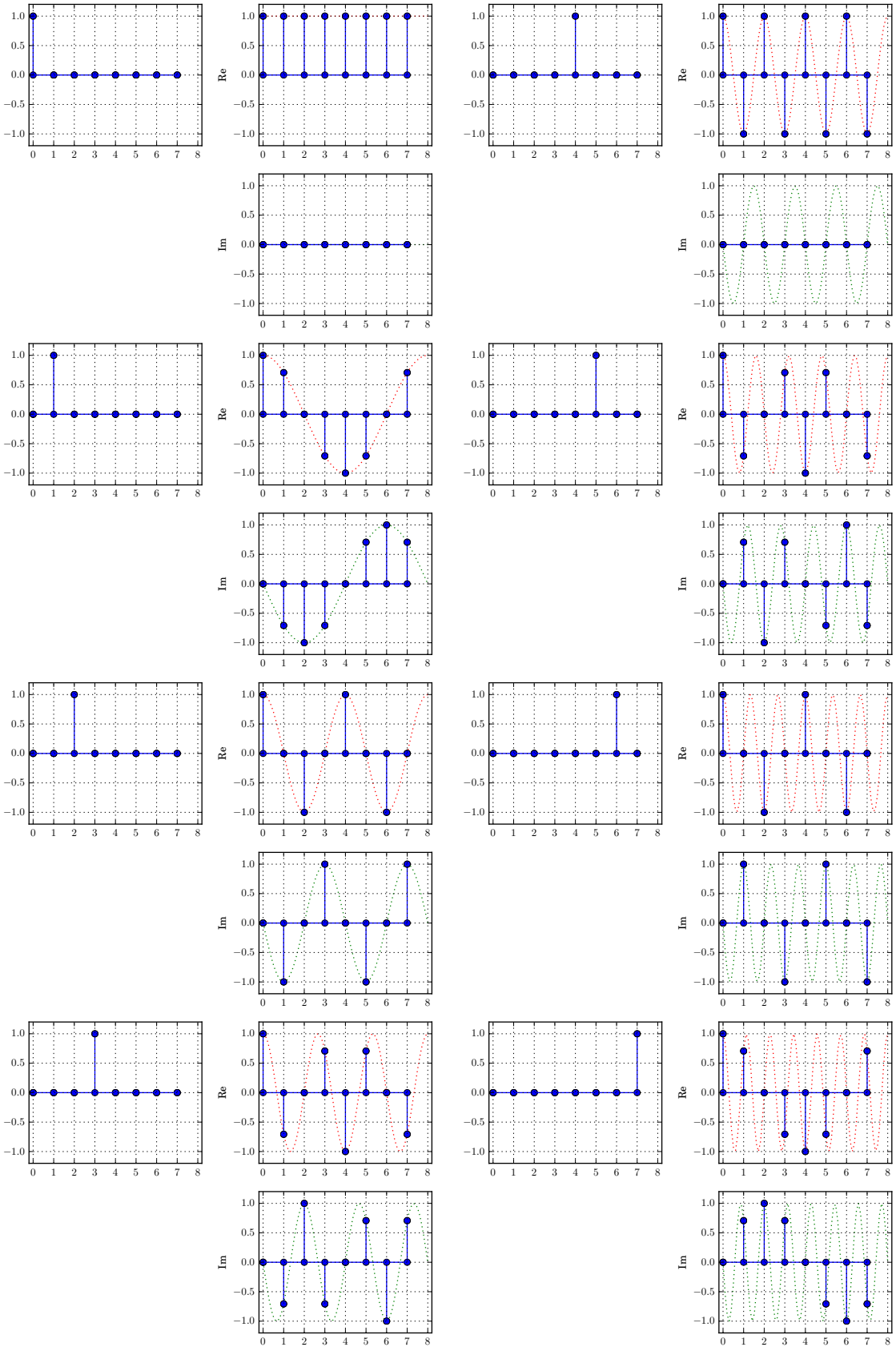


Abbildung 7.1.: *Plots der Impulsfunktionen für $nN = 8$ und seiner Fouriertransformierten.*

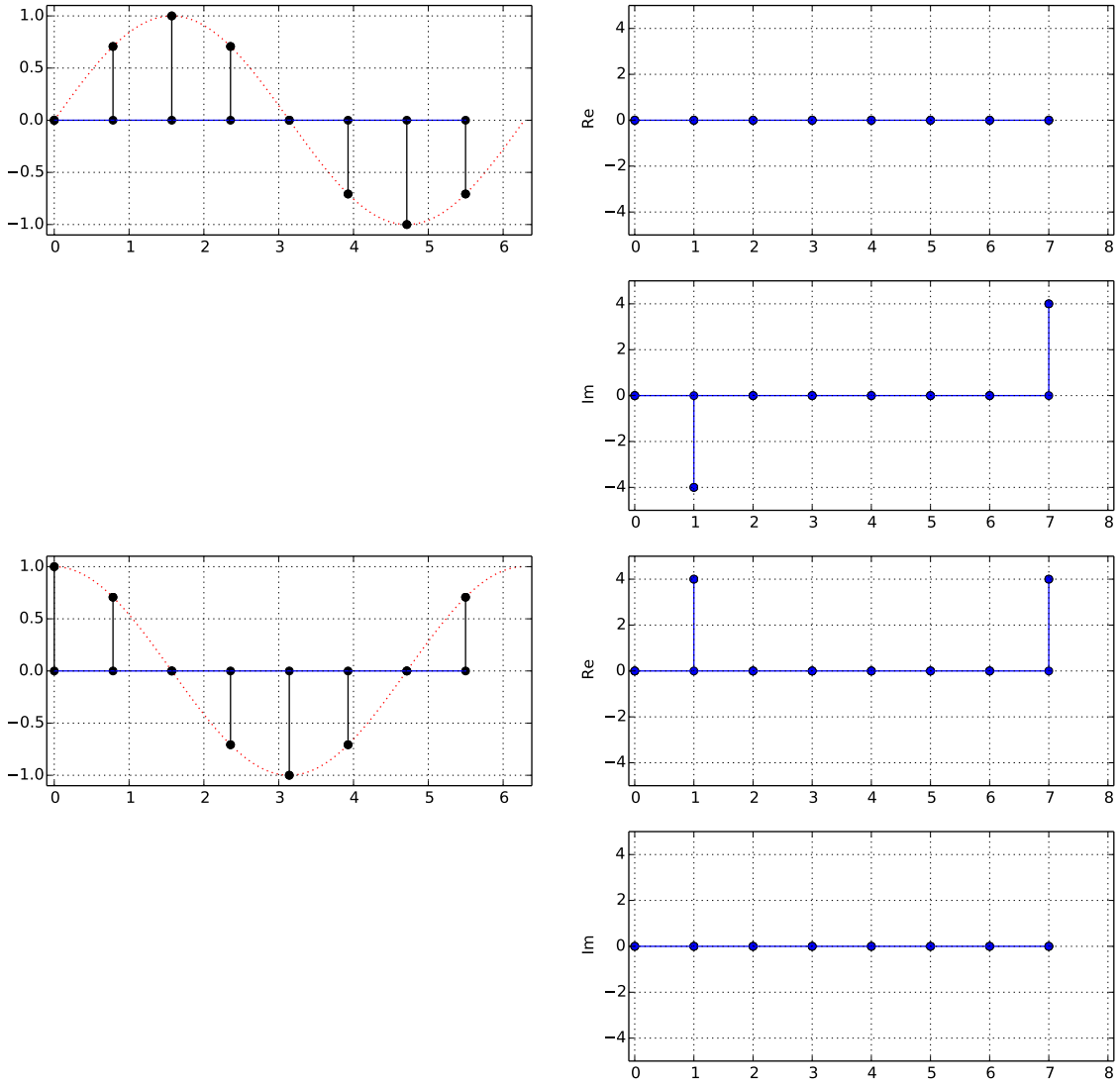


Abbildung 7.2.: *Plot von Sinus und Cosinus für $n = 8$ und seiner Fouriertransformierten.*

Die umgeordnete Matrix sieht wie folgt aus:

$$\Omega_{2n}\mathbf{P}^{\mathsf{T}} = \frac{1}{\sqrt{2n}} \begin{pmatrix} 1 & 1 & 1 & \dots & 1 & 1 & \dots \\ 1 & \omega_{2n}^{2} & \omega_{2n}^{4} & \dots & \omega_{2n} & \omega_{2n}^{3} & \dots \\ 1 & \omega_{2n}^{4} & \omega_{2n}^{8} & \dots & \omega_{2n}^{2} & \omega_{2n}^{6} & \dots \\ 1 & \omega_{2n}^{6} & \omega_{2n}^{12} & \dots & \omega_{2n}^{3} & \omega_{2n}^{9} & \dots \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots \\ 1 & \omega_{2n}^{2(n-1)} & \omega_{2n}^{4(n-1)} & \dots & \omega_{2n}^{n-1} & \omega_{2n}^{3(n-1)} & \dots \end{pmatrix}. \tag{7.33}$$

Jetzt wenden wir die Beobachtung $\omega_{2n}^{2j} = \omega_n^j$ auf die linken Spalten an. Für die rechten Spalten teilen wir die Exponenten in einen ungeraden und einen geraden Teil auf. Dabei stellen wir der Spalte (mit ursprünglichem Index) $2j + 1$ immer den Teil in ω_n dar, der in der Spalte mit Index $2j$ vorkommt. Das geht, weil die Exponenten mit dem Spaltenindex immer wachsen. Für die

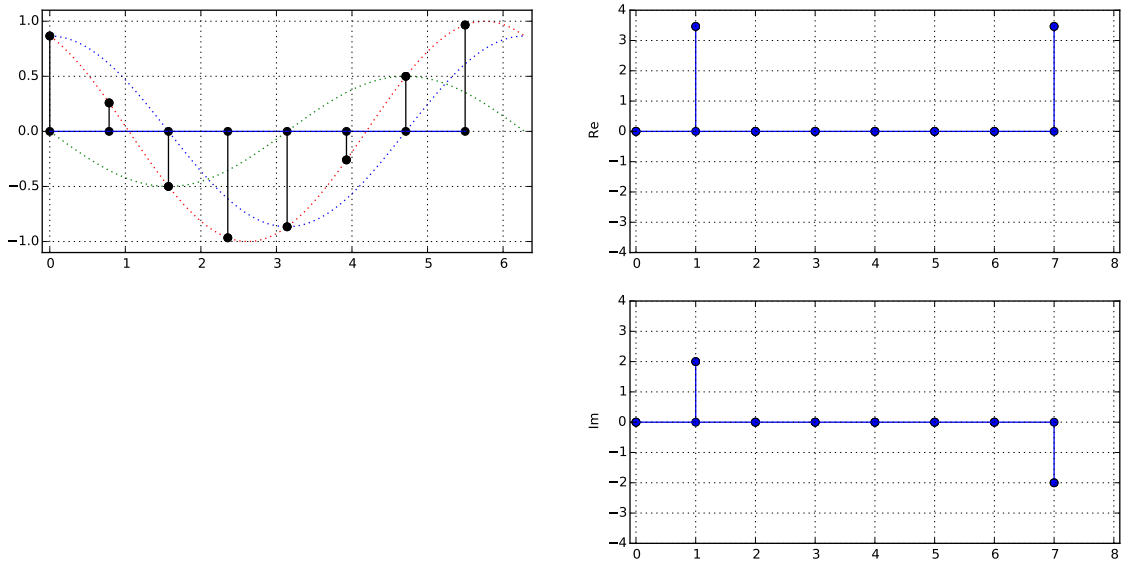


Abbildung 7.3.: *Plot von $z_j = \cos(2\pi j/N + \varphi)$ für $n = 8$ (rechts) und seiner Fouriertransformierten (links).*

unteren Zeilen verwenden wir, dass $\omega_n^{n+j} = \omega_n^j$, bzw. $\omega_{2n}^{n+j} = -\omega_{2n}^j$.

$$\Omega_{2n}\mathbf{P}^\mathsf{T} = \frac{1}{\sqrt{2n}} \begin{pmatrix} 1 & 1 & 1 & \dots & 1 & 1 & 1 & \dots \\ 1 & \omega_n & \omega_n^2 & \dots & \omega_n^{n-1} & \omega_{2n} & \omega_{2n}\omega_n & \dots \\ 1 & \omega_n^2 & \omega_n^4 & \dots & \omega_n^{2(n-1)} & \omega_{2n}^2 & \omega_{2n}^2\omega_n^2 & \dots \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots \\ 1 & \omega_n^{n-1} & \omega_n^{2(n-1)} & \dots & \omega_n^{(n-1)(n-1)} & \omega_{2n}^{n-1} & \omega_{2n}^{n-1}\omega_n^{n-1} & \dots \\ 1 & 1 & 1 & \dots & 1 & -1 & -1 & \dots \\ 1 & \omega_n & \omega_n^2 & \dots & \omega_n^{n-1} & -\omega_{2n} & -\omega_{2n}\omega_n^2 & \dots \\ 1 & \omega_n^2 & \omega_n^4 & \dots & \omega_n^{2(n-1)} & -\omega_{2n}^2 & -\omega_{2n}^2\omega_n^2 & \dots \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots \\ 1 & \omega_n^{n-1} & \omega_n^{2(n-1)} & \dots & \omega_n^{(n-1)(n-1)} & -\omega_{2n}^{n-1} & -\omega_{2n}^{n-1}\omega_n^{n-1} & \dots \end{pmatrix}. \tag{7.34}$$

Man sieht, dass in den linken Spalten zwei Kopien der DFT-Matrix Ω_n stehen. Durch unsere Wahl der Aufteilung in den rechten Spalten sehen wir die Koeffizienten dieser Matrix auch in den rechten Spalten, allerdings gibt es hier Faktoren der Form ω_{2n}^k. In der ersten Zeile ist der Faktor $1 = \omega_{2n}^0$, in der zweiten Zeile ω_{2n}, in der dritten Zeile ω_{2n}^2 usw. Die Multiplikation des Blocks mit einem konstanten Faktor pro Zeile lässt sich durch Multiplikation (von links) mit einer Diagonalmatrix der Form

$$\mathbf{F} = \operatorname{diag}(1, \omega_{2n}, \omega_{2n}^2, \dots, \omega_{2n}^{n-1}) \tag{7.35}$$

beschreiben. Jetzt können wir die umsortierte DFT-Matrix für $2n$ aus DFT-Matrizen für n „zusammenbauen“:

$$\Omega_{2n}\mathbf{P}^\mathsf{T} = \frac{1}{\sqrt{2}} \begin{pmatrix} \Omega_n & \mathbf{F}\Omega_n \\ \Omega_n & -\mathbf{F}\Omega_n \end{pmatrix}. \tag{7.36}$$

Die Fouriertransformation $\Omega_{2n}\mathbf{s}$ eines Signals $\mathbf{s}$ der Länge $2n$ können wir also ausrechnen als $\Omega_{2n}\mathbf{P}^\mathsf{T}\mathbf{P}\mathbf{s}$. In Worten lässt sich dieses Vorgehen wie folgt beschreiben: Wir teilen den Signalvektor in Elemente mit geradem und mit ungeradem Index. Von beiden Vektoren der Länge n bilden

wir die Fouriertransformation. Das Ergebnis des ungeraden Teils multiplizieren wir elementweise mit dem Vektor $(1, \omega_{2n}, \omega_{2n}^2, \ldots)$. Die erste Hälfte der gesuchten Fouriertransformation bekommen wir als Summe der beiden Vektoren, den zweiten Teil als Differenz.

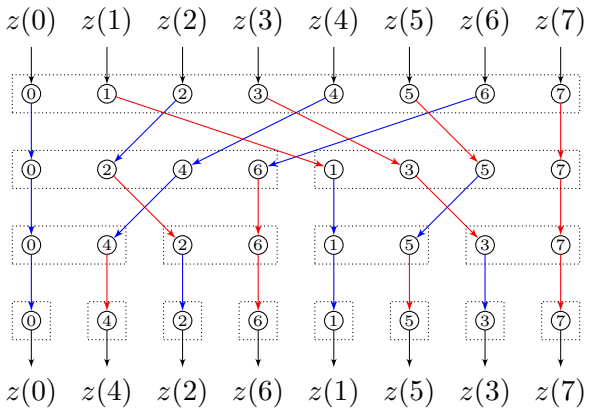
Dieses Vorgehen entspricht genau der Teile-und-Herrsche Strategie. Die Berechnung der Fouriertransformation ist durch die wiederholte rekursive Berechnung mit weniger als $O(n^2)$ Operationen möglich. Sei $F(n)$ die Anzahl der Elementaroperationen, die wir zur Berechnung der Fouriertransformation mit n Koeffizienten benötigen. Mit Hilfe des Rekursionsansatzes benötgen wir zur Berechnung der Fouriertransformation zwei mal die Fouriertransformation für $n/2$ Elemente; die Elemente müssen dann mit den Faktoren aus $\mathbf{F}$ multipliziert geeignet kombiniert werden, was in linearer Zeit in n möglich ist. Insgesamt haben wir damit

$$\begin{aligned} F(n) &= O(n) + 2F\left(\frac{n}{2}\right) \\ &= O(n) + 2\left(2\frac{n}{2} + 2F\left(\frac{n}{4}\right)\right) = O(n) + O(n) + 4F\left(\frac{n}{4}\right) \\ &= O(n) + O(n) + 4\left(2\frac{n}{4} + 2F\left(\frac{n}{8}\right)\right) = O(n) + O(n) + O(n) + 8F\left(\frac{n}{8}\right) \\ &\vdots \\ &= \sum_{k=0}^{2^k \geq n} O(n) = O(n \log n). \end{aligned} \tag{7.37}$$

7.7. Schnelle diskrete Fouriertransformation (FFT)

Die rekursive Berechnung führt direkt zu einem verbesserten Algorithmus für die diskrete Fouriertransformation. Für eine optimale Implementierung ist es wünschenswert, die in den meisten Programmiersprachen substantiellen Kosten für einen rekursiven Funktionsaufruf zu vermeiden. Im Folgenden wollen wir daher die Rekursion auflösen. Der sich dadurch ergebende Algorithmus wird als *schnelle diskrete Fouriertransformation* (*engl.* fast Fourier transform (FFT)) bezeichnet.

Um zu verstehen, welche Berechnungen durch die rekursive Zerlegung der Fouriertransformation entstehen, wollen wir den Ablauf für ein Signal der Länge $n = 8$ betrachten. Als erste Beobachtung können wir festhalten, dass die Bearbeitung in zwei Phasen aufgeteilt werden kann. In der ersten Phase (*Sortierphase*) wird das Signal nach geraden und ungeraden Indizes aufgeteilt:



Diese Phase entspricht einer Umsortierung des Signals. Stellt man die Indizes des Signals in Binärdarstellung in einer Tabelle gegenüber, so sieht man, dass sich die neuen Indizes durch Umkehrung der Bits ergeben:

$$
\begin{array}{lllllll}
0 & = & 000 & \rightarrow & 000 & = & 0 \\
1 & = & 001 & \rightarrow & 100 & = & 4 \\
2 & = & 010 & \rightarrow & 010 & = & 2 \\
3 & = & 011 & \rightarrow & 110 & = & 6 \\
4 & = & 100 & \rightarrow & 001 & = & 1 \\
5 & = & 101 & \rightarrow & 101 & = & 5 \\
6 & = & 110 & \rightarrow & 011 & = & 3 \\
7 & = & 111 & \rightarrow & 111 & = & 7
\end{array}
$$

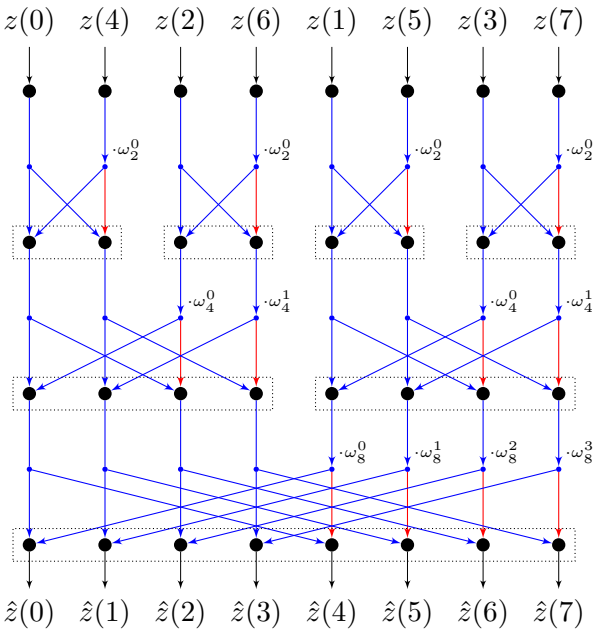
Dies entspricht einer Umwandlung der Bitwertigkeit von LSB-0 (*least significant bit*) in MSB-0 (*most significant bit*):

$$j = \sum_{k=0}^{p-1} a_k 2^k \mapsto \sum_{k=0}^{p-1} a_k 2^{p-1-k} = l \tag{7.38}$$

Mittels dieser Formel kann man für $j = 0, \ldots, 2^p - 1$ neue Indizes l bestimmen, und das Signal entsprechend umordnen.

Nebenbemerkung. *Ein effizientes Umordnen der Signalelemente nach Gleichung* (7.38) *ist wie folgt möglich. Schaut man sich obige Tabelle genauer an, so fällt zunächst auf, dass die Umordnung immer paarweise erfolgt. Es reicht somit aus, sich auf Vertauschungen mit $j < l$ zu beschränken. Bei der Berechnung von j und l kann außerdem induktiv vorgegangen werden. Begonnen wird mit dem Paar $(j, l) = (0, 0)$. In jedem Schritt wird nun i inkrementiert. Der zugehörige Index l muss auch entsprechend erhöht werden, allerdings bzgl. MSB Bitwertigkeit. Dazu werden die Bits von l vom höchsten bis zum niedrigsten Bit abgearbeitet. Gesetzte Bits werden dabei gelöscht. Wird ein nicht gesetztes Bit erreicht, so wird dieses gesetzt und der Vorgang abgeschlossen.*

Nach der Umordnung des Signals folgt die zweite Phase des Algorithmus (*Kombinationsphase*), in der die Fourierkoeffizienten bestimmt werden:

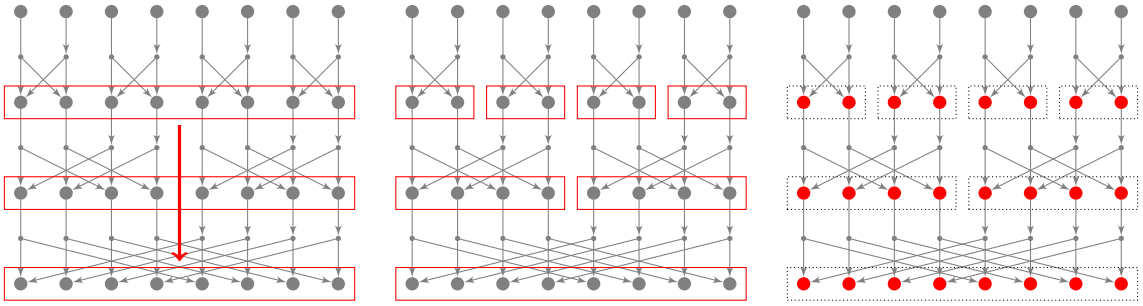


Dies entspricht einem Auflösen des Baumes von den Blättern hin zum Wurzelknoten. Zur Auflösung der Rekursion ist es vorteilhaft, die Berechnung schrittweise in der Breite auszuführen (siehe Abbildung 7.4). Dafür werden im m-ten Schritt des Verfahrens jeweils zwei Fouriertransformationen der Länge 2^m zu einer Fouriertransformation der Länge 2^{m+1} kombiniert (siehe Abbildung 7.4(b)). Für die Kombination der Fouriertransformationen $2k$ und $2k+1$ im m-ten Schritt ergibt sich

$$\begin{aligned} z\left(j+2k2^m\right) &\leftarrow z\left(j+2k2^m\right)+\omega_{2^{m+1}}^{j} z\left(j+(2k+1)2^m\right) \\ z\left(j+(2k+1)2^m\right) &\leftarrow z\left(j+2k2^m\right)-\omega_{2^{m+1}}^{j} z\left(j+(2k+1)2^m\right), \end{aligned} \tag{7.39}$$

wobei der Index j die Werte $0,\ldots,2^m-1$ und der Index k die Werte $0,\ldots,2^{logn-m-1}-1$ annimmt. Da die neuen Werte immer aus den korrespondierenden Elementen des vorherigen Schrittes gebildet werden, kann diese Operation in-place, d.h. ohne Verwendung eines weiteren Arrays, implementiert werden.

Obige Formel stellt den zentralen Rechenschritt der schnellen Fouriertransformation dar, welcher von den Blättern zur Wurzel und für jede Ebene für $k=0,1,\cdots$ ausgeführt wird. Diese Operation wird in der Literatur oft als *butterfly* bezeichnet, da die graphische Darstellung an einen Schmetterling erinnert.



(a) $M = 1, 2, 4, \ldots, N/2$ (b) $k = 0, 1, \ldots, N/2M - 1$ (c) $m = 0, 1, \ldots, M - 1$

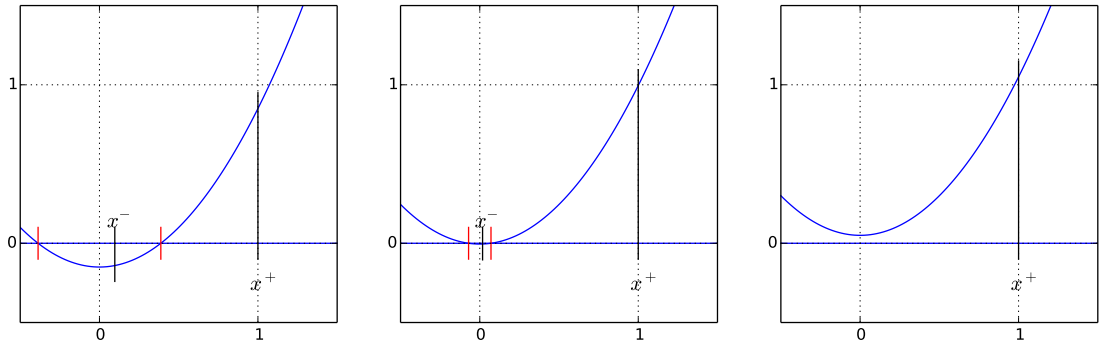
Abbildung 7.4.: *Ablauf der Kombinationsphase der schnellen diskreten Fouriertransformation für ein Signal der Länge N.*

8. Nullstellenbestimmung

Die meisten der Probleme, die wir bis hierhin betrachtet haben, konnten wir als System von linearen Gleichungen beschreiben. Nicht immer ist es angemessen oder möglich, *lineare* Gleichungen zu verwenden. Im Allgemeinen geht es also darum, ein System von nicht-linearen Gleichungen zu lösen. Diese Aufgabe ist oft schwierig.

Wir beginnen zunächst mit der Lösung *einer* (typischerweise nicht-linearen) Gleichung. Da wir Gleichungen der Form $f(x) = g(x)$ immer auf die Form $f(x) - g(x) = 0$ bringen können, und $f(x) - g(x)$ auch eine Funktion in x ist, formuliert man die Lösung von allgemeinen Gleichungen gemeinhin als Bestimmung von Nullstellen.

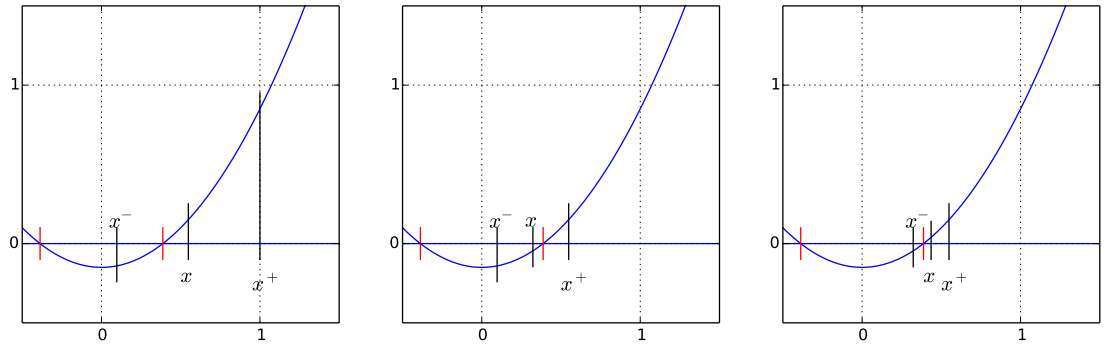
Selbst die Bestimmung einer Nullstelle einer univariaten und stetigen Funktion $f : \mathbb{R} \mapsto \mathbb{R}$ ist im Allgemeinen ein schwieriges Problem. Dies liegt daran, dass es schwierig sein kann, überhaupt eine Nullstelle zu finden. Die Existenz einer Nullstelle ist gesichert, wenn zwei Parameter $x^-, x^+ \in \mathbb{R}$ identifiziert sind, für die gilt: $f(x^-) < 0$ und $f(x^+) > 0$. Aus der Stetigkeit von f folgt dann, dass es eine Nullstelle im Interval $[x^-, x^+]$ gibt. Um zu verdeutlichen, dass es schwierig sein kann überhaupt eine Nullstelle zu finden, sprich passende Werte x^-, x^+ zu bestimmen, betrachten wir die einfache Funktion $f(x) = x^2 - \epsilon$, die für $\epsilon > 0$ Nullstellen bei $\pm\sqrt{\epsilon}$ hat. Man muss also ein Interval der Breite $2\sqrt{\epsilon}$ treffen um überhaupt einen negativen Wert von f zu erhalten.



Es ist ohne weitere Bedingungen an f nicht möglich, alle Nullstellen zu bestimmen, bzw. x^-, x^+ so zu wählen, dass im Interval $[x^-, x^+]$ genau eine Nullstelle liegt. Als einfaches Gegenbeispiel sei $f(x) = sin(\frac{1}{x})$ erwähnt. Diese Funktion hat in jedem Interval, das $x = 0$ enthält, unendlich viele Nullstellen.

8.1. Bisektion

Wir nehmen an, es sind bereits Werte x^- und x^+ gefunden, für die gilt $f(x^-) < 0$ und $f(x^+) > 0$. Dann kann man durch Bisektion eine Nullstelle im Interval $[x^-, x^+]$ beliebig genau eingrenzen: Wähle $x = (x^- + x^+)/2$. Falls $|f(x)| < \epsilon$ wäre eine Nullstelle bis auf Maschinengenauigkeit bestimmt. Ansonsten setzt man für den Fall $f(x) < 0$ eine neue untere Schranke $x^- \leftarrow x$ und für den Fall $f(x) > 0$ eine neue obere Schranke $x^+ \leftarrow x$.



Bei diesem Verfahren halbiert sich die Breite des Intervalls in jedem Schritt. Als Abbruchbedingung für das Verfahren sollte man eine Schranke für die Intervallbreite verwenden, z.B. die doppelte Maschinengenauigkeit. Eine weitere Unterteilung würde nichts mehr bringen, weil dann der Wert $x = (x^- + x^+)/2$ numerisch nicht mehr von den Intervallgrenzen unterschieden werden kann. Man darf nicht erwarten, dass $|f(x)| < \epsilon$ erreicht wird. Die Variation der Funktion um die Nullstelle wird von der ersten Ableitung dominiert, und solange diese nicht beschränkt ist, hat man auch keine Schranke für die Werte von f im Intervall $[x^-, x^+]$. Konkret sieht man das an der Taylorentwicklung von f um die Nullstelle $f(x^*) = 0$:

$$f(x) = f(x^*) + f'(x^*)(x - x^*) + \dots \tag{8.1}$$

Für $x \approx x^* \pm \epsilon$ hat man also

$$f(x) \approx \pm f'(x^*)\epsilon + \dots \tag{8.2}$$

Umgekehrt darf man auch nicht erwarten, dass die Unterteilung des Intervalls die Genauigkeit erhöht, wenn $|f(x)|$ bereits im Bereich der Maschinengenauigkeit liegt, also $|f(x)| < \epsilon$. Zusammengefasst kann man das Verfahren abbrechen wenn entweder die Intervallbreite $|x^+ - x^-|$ oder der Betrag des Funktionswerts $|f(x)|$ eine untere Schranke ϵ erreicht haben.

Die Wahl des Wertes $x = (x^- + x^+)/2$ ist sinnvoll, weil sie sicherstellt, dass das Verfahren immer nach der gleichen Zahl von Schritten konvergiert, unabhängig davon, ob nun in den einzelnen Schritten $f(x) > 0$ oder $f(x) < 0$ auftritt. In der Nähe der Nullstelle halbiert sich mit jeder Halbierung der Intervalbreite auch die Größe von $|f(x)| \approx f'(x^*)\epsilon$. Mit jeweils drei Schritten gewinnt man also in der Dezimaldarstellung eine weitere signifikante Stelle an Genauigkeit hinzu. Man nennt dieses Konvergenzverhalten *linear*, weil die Anzahl der Schritte des Algorithmus in einem linearen Zusammenhang mit der Anzahl an signifikanten Stellen der numerischen Lösung steht. Das Konvergenzverhalten gilt unabhängig von der Funktion. Damit hat das Bisektions-Verfahren das beste *worst-case* Konvergenzverhalten unter allen Verfahren zur Nullstellenbestimmung. Eine bessere Konvergenz erhält man nur durch Ausnutzung „gutmütiger“ Funktionen; und auf Kosten von langsamerer oder gar keiner Konvergenz sollte die Funktion nicht gutmütig sein.

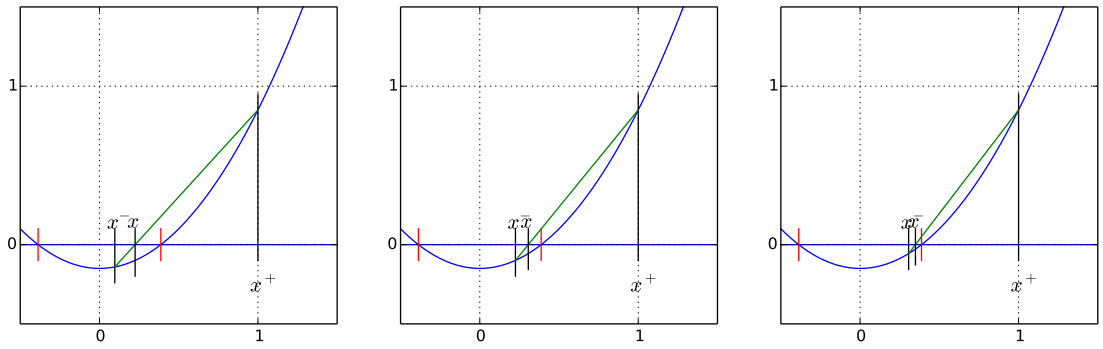
8.2. Regula falsi

Ein solches Verfahren, mit dem man die Geometrie des Problems oft gut ausnutzen kann, basiert auf der Bestimmung von *Sekanten*: Wenn wir erwarten, dass die Funktion im Kleinen ungefähr linear verläuft, sollten wir besser die Nullstelle der Gerade durch die Punkte $(x^-, f(x^-))$ und

$(x^+, f(x^+))$ zur Intervallteilung verwenden. Aus der Bedingung $(1-\lambda)f(x^-) + \lambda f(x^+) = 0$ ergibt sich für die Intervallteilung direkt $\lambda = f(x^-)/(f(x^-) - f(x^+)$ und mithin die Wahl

$$x = \left(1 - \frac{f(x^-)}{f(x^-) - f(x^+)}\right) x^- + \frac{f(x^-)}{f(x^-) - f(x^+)} x^+ = \frac{x^+ f(x^-) - x^- f(x^+)}{f(x^-) - f(x^+)}. \tag{8.3}$$

Diese Art der Berechnung ist auch aus numerischen Gesichtspunkten günstig: da die Vorzeichen von $f(x^-)$ und $f(x^+)$ unterschiedlich sind, stehen sowohl im Nenner wie auch im Zähler Summen und es besteht keine Gefahr, dass es zu Auslöschung kommt.



Wenn im Verlauf des Verfahrens sowohl die untere wie auch die obere Schranke angepasst werden, konvergiert das Verfahren schneller als linear und ist somit besser als Bisektion. Für konvexe oder konkave Funktion wird allerdings immer die selbe Intervallgrenze angepasst. Das führt einerseits zu linearer Konvergenz und andererseits dazu, dass das Interval nicht mehr beliebig klein wird (siehe Abbildung oben). Man kann also nicht mehr die Intervallbreite als Konvergenzkriterium nehmen, sondern muss aus den Funktionswerten auf die erste Ableitung schließen und berücksichtigen, wenn sie größer als eins ist. Konkret bekommt man die Schätzung der Steigung direkt aus der Sekantensteigung $(f(x^-) - f(x^+))/(x^- - x^+)$ und verwendet dann als Konvergenzkriterium

$$|f(x)| < \epsilon \max\left(1, \left|\frac{f(x^-) - f(x^+)}{x^- - x^+}\right|\right). \tag{8.4}$$

Da es für das Erreichen einer solchen Schranke keine Garantie gibt sollte man immer die maximale Anzahl der Iterationen begrenzen.

8.3. Newton-Verfahren

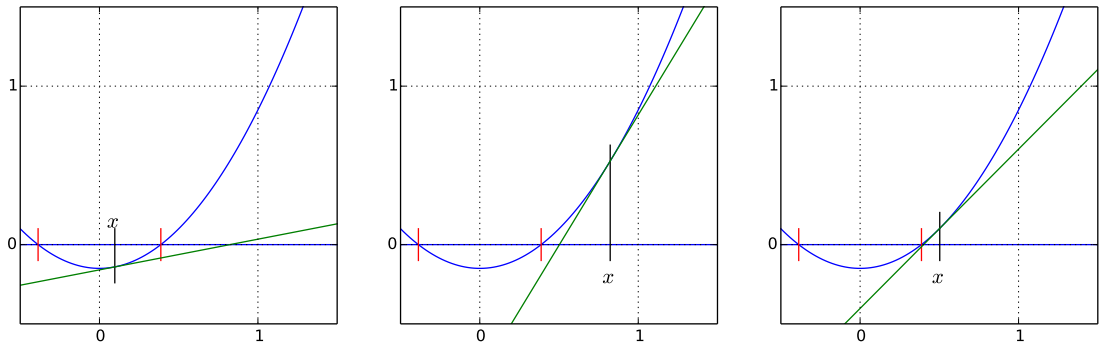
Kann man auch die Ableitung der Funktion f effizient und genau bestimmen, so legt die Taylor-Entwicklung $f(x) = f(x^*) + f'(x^*)(x - x^*) + \ldots$ um x^* nahe, die Nullstelle wie folgt zu approximieren:

$$0 = f(x^*) + f'(x^*)(x - x^*) \Longleftrightarrow x = \frac{-f(x^*) + f'(x^*)x^*}{f'(x^*)} = x^* - \frac{f(x^*)}{f'(x^*)}. \tag{8.5}$$

Die Idee des Newton-Verfahrens zur Bestimmung von Nullstellen ist es mit einem Wert x_0 in der Nähe einer Nullstelle zu beginnen und dann die Iteration

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)} \tag{8.6}$$

durchzuführen.



Wenn das Newton-Verfahren konvergiert, so ist die Konvergenz erheblich schneller als die der bisher kennengelernten Verfahren. Dafür gibt es keine Garantie, dass das Verfahren überhaupt konvergiert und, in der Tat, kommt es in der Praxis häufig zu divergentem Verhalten.

Um die schnelle Konvergenz des Newton-Verfahrens einzusehen, betrachten wir wieder die Taylor-Entwicklung, diesmal um die Stelle x_i aus dem i-ten Iterationsschritt an der Nullstelle x^*:

$$0 = f(x^*) = f(x_i) + f'(x_i)(x^* - x_i) + \frac{1}{2}f''(r_i)(x^* - x_i)^2, \quad r_i \in [x^*, x_i] \tag{8.7}$$

Hier haben wir das sogenannte Lagrange-Restglied verwendet: nach dem Mittelwertsatz existiert immer ein r_i im Interval $[x^*, x_i]$, so dass die Taylor-Entwicklung exakt wird (allerdings ist uns r_i nicht bekannt). Der Abstand zur Nullstelle in Schritt i des Verfahrens ist $|x^* - x_i|$. Wir wollen wissen wie sich der Abstand im nächsten Schritt verkleinert, also den Abstand $|x^* - x_{i+1}|$ bestimmen. Dazu ersetzen wir den ersten Wert x_i in der Taylor-Entwicklung nach Newton-Regel durch $x_{i+1} + f(x_i)/f'(x_i)$ und erhalten

$$\begin{aligned} 0 &= f(x_i) + f'(x_i)\left(x^* - x_{i+1} - \frac{f(x_i)}{f'(x_i)}\right) + \frac{1}{2}f''(r_i)(x^* - x_i)^2 \\ &= f'(x_i)\,(x^* - x_{i+1}) + \frac{1}{2}f''(r_i)(x^* - x_i)^2 \end{aligned} \tag{8.8}$$

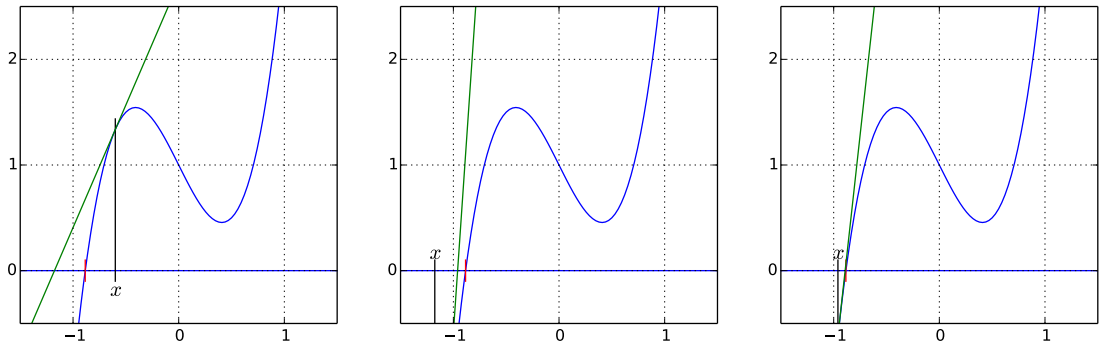
Für $f'(x_i) \neq 0$ können wir jetzt nach dem gesuchten Term umstellen:

$$|x^* - x_{i+1}| = \frac{|f''(r_i)|}{2|f'(x_i)|}(x^* - x_i)^2. \tag{8.9}$$

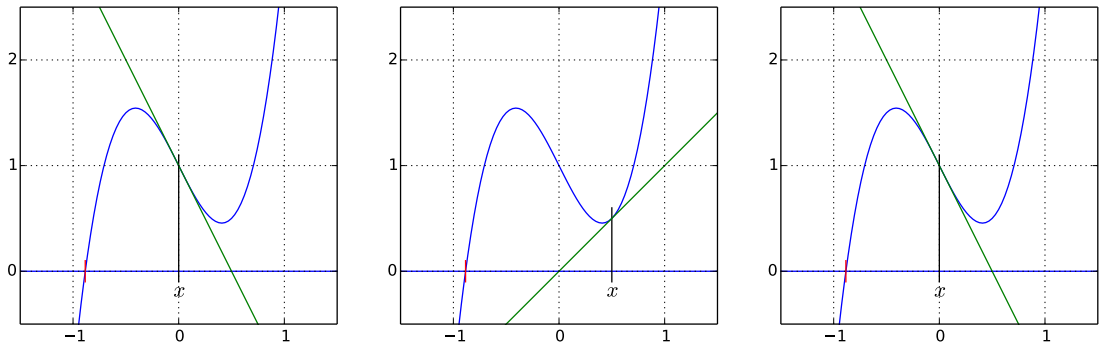
Ist also $|f'(x)|$ im Konvergenzgebiet nicht zu klein, und $|f''(x)|$ nicht zu groß, dann ist der Abstand zur Nullstelle proportional zum Quadrat des Abstandes im vorangegangenen Schritt. Man spricht hier von *quadratischer* Konvergenz. Die Anzahl der signifikanten Stellen von x wächst dann nicht mehr nur linear mit der Anzahl der Schritte, sondern quadratisch. Ein typisches Verhalten ist, dass sich die Anzahl der signifikanten Stellen mit jedem Schritt verdoppelt. Dies ist offenbar erheblich besser als das Bisektions-Verfahren.

Andererseits ist die Bedingung an $|f'(x)|$ nicht nur ein Artefakt der mathematischen Analyse. Auch in der Praxis ist die Konvergenz des Newton-Verfahrens empfindlich gegenüber der Größe der Ableitung. Wird die Ableitung zu klein, so kommt es zu Divergenz. Zur Illustration des unvorhersagbaren Verhaltens betrachten wir die Funktion $f(x) = 4x^3 - 2x + 1$ mit der Ableitung $f'(x) = 12x^2 - 2$. Diese Funktion hat eine reelle Nullstelle bei $x^* \approx -0.8846$ und ein Maximum

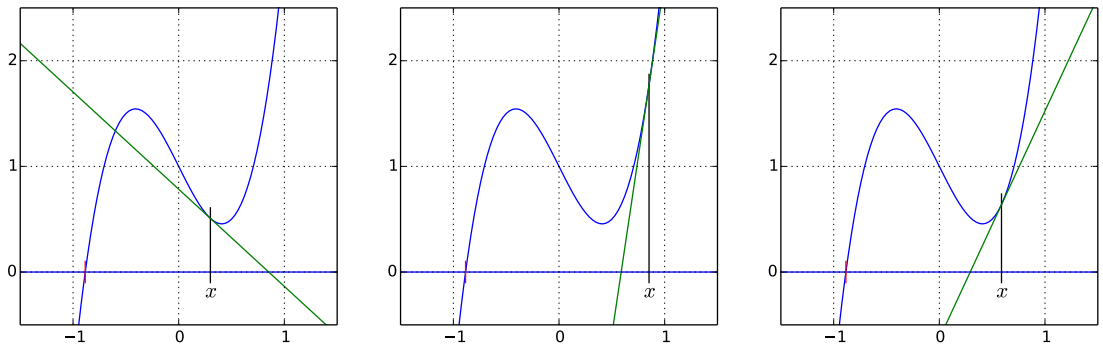
bei $\hat{x} = -\sqrt{1/6}$. Für alle Startwerte $x_0 < x^*$ konvergiert das Newtonverfahren monoton gegen die Nullstelle. Für Startwerte $x^* < x_0 < \hat{x}$ ergibt sich aus der Geometrie der Funktion im ersten Schritt $x_1 < x^*$ und dann konvergiert das Verfahren wiederum monoton, siehe folgende Abbildung:

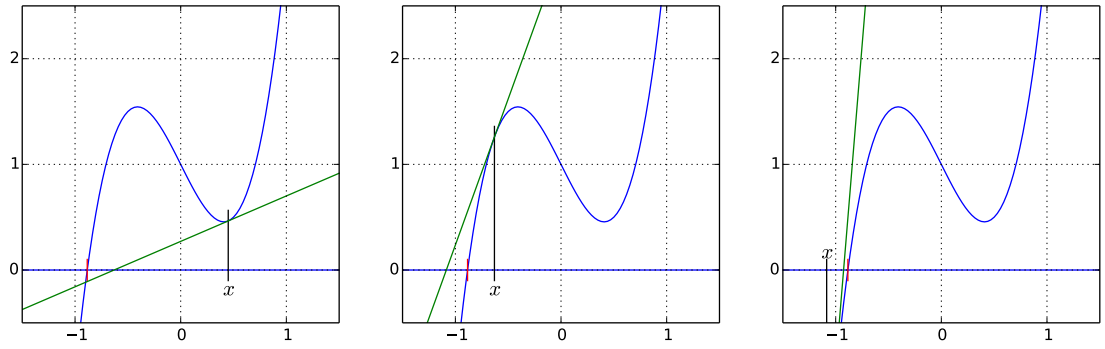


Was passiert aber für Startwerte $x_0 > \hat{x}$? Diese Frage kann man nicht pauschal beantworten. Betrachten wir z.B. den Startwert $x_0 = 0$. Wegen $f(x_0) = 1$ und $f'(x_0) = -2$ ergibt sich aus dem Newton-Schritt $x_1 = x_0 - f(x_0)/f'(x_0) = 1/2$. Jetzt haben wir $f(x_1) = 1/2$ und $f'(x_1) = 1$ und erhalten $x_2 = 0$, also den Startwert. Sprich das Newton-Verfahren pendelt zwischen den Werten $x = 0$ und $x = 1/2$ und erreicht die Nullstelle niemals. Die folgende Abbildung illustriert dieses Verhalten.



Man könnte denken, dass für Werte $x > 0$ die Nullstelle ebenfalls nicht gefunden wird. So einfach ist der Fall allerdings nicht. Die folgende Abbildung zeigt das Newton-Verfahren für die Starterte $x_0 = 6/20$ und $x_0 = 9/20$.





Im ersten Fall wird aus den ersten Iterationsschritten nicht klar, was passieren wird. Das sich aus den ersten Schritten auch nicht viel schließen lässt zeigt der zweite Fall. Hier führt der erste Schritt direkt zu $x_0 < \hat{x}$ und damit zu konvergentem Verhalten. Es gibt also auch Werte $x_0 > 0$ für die das Newton-Verfahren konvergiert.

Allgemein bezeichnet man die Menge der Startwerte $\{x\}$ für die das Newton-Verfahren gegen eine Nullstelle x^* konvergiert als *Attraktor* von x^*. Wie wir gesehen haben, ist der Attraktor einer Nullstelle nicht unbedingt ein zusammenhängendes Gebiet. Wegen der rekursiven Definition eines Attraktors ist das Attraktionsgebiet oft ein Fraktal.

8.4. Newton-Verfahren in mehreren Dimensionen

Wie bereits erwähnt ist es schwierig, die Nullstellen von Funktionen zu bestimmen, die von mehr als einer Variable abhängen. Das Newton-Verfahren lässt sich allerdings direkt erweitern.

Betrachten wir eine Funktion $\mathbf{f} : \mathbb{R}^n \mapsto \mathbb{R}^m$. Die *partielle* Ableitung der Funktion in der Koordinate c nach der Variablen x_d schreiben wir $\frac{\partial f_c}{\partial x_d}$. Wollten wir also eine Nullstelle in der Koordinate c entlang der Koordinatenachse d suchen, so ergäbe sich das Newton-Verfahren mit der Regel

$$x_d^{i+1} = x_d^i - \frac{f_c(\mathbf{x}^i)}{\frac{\partial f_c(\mathbf{x}^i)}{\partial x_d}}, \tag{8.10}$$

wobei $\mathbf{x}^i$ der gegenwärtige Vektor ist, der nur in der Komponenten x_d verändert wird. Aus der Linearität der Ableitung ergibt sich die Erweiterung dieses Verfahrens für alle Koordinaten gleichzeitig. Dazu benötigen wir die *Jacobi-Matrix*, die alle partiellen Ableitungen entält:

$$\mathbf{J}_f = \begin{pmatrix} \frac{\partial f_0}{\partial x_0} & \frac{\partial f_0}{\partial x_1} & \frac{\partial f_0}{\partial x_2} & \cdots \\ \frac{\partial f_1}{\partial x_0} & \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots \\ \frac{\partial f_2}{\partial x_0} & \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix} \in \mathbb{R}^{m \times n}. \tag{8.11}$$

Für das Newton-Verfahren ergibt sich

$$\mathbf{x}^{i+1} = \mathbf{x}^i - \mathbf{J}_f^{-1}(\mathbf{x}^i)\mathbf{f}(\mathbf{x}^i). \tag{8.12}$$

Statt tatsächlich die Inverse zu bilden, bestimmt man den Newton-Schritt $\Delta\mathbf{x} = \mathbf{x}^{i+1} - \mathbf{x}^i$ durch Lösung des lineare Gleichungssystems

$$\mathbf{J}_f(\mathbf{x}^i)\Delta\mathbf{x} = -\mathbf{f}(\mathbf{x}^i). \tag{8.13}$$

Wichtige Spezialfälle sind

1. komplexe Funktionen $f : \mathbb{C} \mapsto \mathbb{C}$: Hier kann man das Newton-Verfahren so anwenden wie im reellen Fall, da sich die Division einfach über komplexe Konjugation ausdrücken lässt:

$$z_{i+1} = z_i - \frac{f(z_i)\overline{f'(z_i)}}{|f'(z_i)|^2}. \tag{8.14}$$

2. skalare Funktionen $f : \mathbb{R}^n \mapsto \mathbb{R}$: In diesem Fall wird die Jacobi-Matrix zum Gradienten ∇f der Funktion (Achtung: der Gradient wird üblicherweise als Spaltenvektor verstanden, während die entsprechende Jacobi-Matrix ein Zeilenvektor ist). Die Pseudo-Inverse eines Vektors $\mathbf{g} \in \mathbb{R}^n$ ist $\mathbf{g}^\mathsf{T}/\|\mathbf{g}\|^2$ (das Produkt der Vektoren ergibt den Skalar 1). Damit erhält man für den Newton-Schritt

$$\mathbf{x}^{i+1} = \mathbf{x}^i - \frac{f(\mathbf{x}^i)}{\|\nabla f(\mathbf{x}^i)\|^2}\nabla f(\mathbf{x}^i). \tag{8.15}$$

9. Optimierung

Viele Probleme aus Technik und Wirtschaft lassen sich als Optimierungsaufgaben formulieren. Dabei geht es in der Regel darum, optimale Werte für unbekannte Parameter zu finden. Das was zu optimieren ist wird dabei durch eine *Zielfunktion* beschrieben, welche von einem oder mehreren unbekannten Parameter(n) abhängt. Die Zielfunktion beschreibt oft einen Fehler oder bei physikalisch motivierten Problemen eine Energie. Anstatt Zielfunktion werden daher auch oft die Begriffe Fehlerfunktion oder Energiefunktion verwendet. Bei ökonomischen Prozessen beschreibt die Zielfunktion Größen wie Kosten, Gewinne oder Produktionskapazitäten. Bei manchen Anwendungen sind nicht alle Parameterwerte zulässig. Wird die Menge der zulässigen Parameterwerte eingeschränkt, so spricht man von einem Optimierungsproblem mit *Nebenbedingungen*. Ergibt sich ein Teil der Parameter aus der speziellen Aufgabenstellung und wird fixiert spricht man von *Randbedingungen*. Je nachdem ob die Zielfunktion minimiert oder maximiert werden soll, spricht man von einem *Minimierungs-* oder *Maximierungsproblem*.

Ein Beispiel für ein Minimierungsproblem hatten wir im Rahmen der Ausgleichsrechnung kennen gelernt. Die Summe der Fehlerquadrate $\|Ax - b\|^2$ definierte hier die zu minimierende Zielfunktion. Da alle $x \in \mathbb{R}^n$ in Betracht gezogen werden gibt es keine Nebenbedingungen. Ein weiteres Beispiel ist die Charakterisierung der Singulärwerte und Singulärvektoren. Die zu maximierende Zielfunktion ist hier gegeben durch $\|Ax\|$. Als Nebenbedingung werden nur Einheitsvektoren zugelassen. In beiden Fällen kann die Lösung der Optimierungsaufgabe vergleichsweise einfach berechnet werden. Im Allgemeinen ist dies jedoch nicht möglich und spezielle Verfahren sind je nach Typ des Optimierungsproblems notwendig um dieses zu lösen.

9.1. Vorüberlegungen

Wir betrachten zunächst eine Funktion die nur von einer (reellen) Variablen abhängt: $f : \mathbb{R} \mapsto \mathbb{R}$. Wir beschränken uns auf die Bestimmung eines Minimums – Maximierungsprobleme können durch die Betrachtung der Funktion $-f$ in Minimierungsprobleme übersetzt werden. Die Funktion f hat ein Minimum in x^* wenn gilt:

$$\exists \delta > 0 : f(x^*) < f(x), \quad x \in [x^* - \delta, x^* + \delta]. \tag{9.1}$$

Diese Bedingung beschreibt ein *lokales* Minimum. Ein lokales Minimum von f in x^* ist auch ein *globales* Minimum im Wertebereich $\Omega = [\alpha, \omega]$ wenn zusätzlich gilt: $f(x^*) < f(x), x \in \Omega$. Wir gehen im Folgenden davon aus, dass die Funktion f ein eindeutiges globales Minimum hat. Dies muss in der Praxis nicht unbedingt so sein: die Funktion könnte mehrere Parameterwerte haben, an denen sie ihren kleinsten Wert annimmt.

Wie wir aus der Analysis wissen, können wir lokale Minima gut charakterisieren, wenn die Funktion zweimal stetig differenzierbar ist. Dann ist eine notwendige Bedingung für eine *Extremstelle* oder einen *kritischen Punkt* das Verschwinden der ersten Ableitung: $f'(x^*) = 0$. Auch für das algorithmische Auffinden von kritischen Punkten wird diese Bedingung verwendet, zumindest wenn sich die Ableitung der Funktion einfach ausrechnen lässt. Ob ein kritischer Punkt ein Minimum ist sehen wir an der zweiten Ableitung. Wenn diese positiv ist, also $f''(x^*) > 0$, dann ist die Funktion in der Nähe von x^* *konvex* und f hat ein lokales Minimum in x^*.

Der Begriff Konvexität spielt für die Minimierung von Funktionen eine zentrale Rolle. Eine Funktion ist konvex im Gebiet Ω wenn für jede zwei Stellen x, y die Verbindungsstrecke zwischen den Punkten nicht unterhalb der Funktion liegt:

$$f((1-\lambda)x + \lambda y) \leq (1-\lambda)f(x) + \lambda f(y), \quad \lambda \in [0,1], x, y \in \Omega \tag{9.2}$$

Wichtig ist nun folgende Beobachtung: Ist eine Funktion im Gebiet Ω konvex, so hat sie in diesem Gebiet nur ein Minimum, sprich ein lokales Minimum ist immer auch das globale Minimum. Dieses globale Minimum können wir finden, in dem wir wiederholt einen Schritt in Richtung der Werte machen, die kleinere Funktionswerte ergeben. Gibt es aber mehrere lokale Minima, so können wir das globale Minimum nicht mit Sicherheit bestimmen. Die meisten Algorithmen bestimmen ausgehend von einem Startwert das nächste lokale Minimum.

Wie übertragen sich diese Bedingungen nun auf Funktionen mit mehreren Veränderlichen? Wir wollen also eine Funktion $f : \mathbb{R}^n \mapsto \mathbb{R}$ minimieren. Nehmen wir an es gibt ein Gebiet $\Omega \subseteq \mathbb{R}^n$ in dem f konvex ist. Dabei ist die Bedingung für Konvexität identisch zum eindimensionalen Fall. Diese Bedingung hat allerdings im mehrdimensionalen Fall weitere Konsequenzen: für zwei Punkte $\mathbf{x}, \mathbf{y} \in \Omega$ soll auch die Verbindungsstrecke $(1-\lambda)\mathbf{x} + \lambda\mathbf{y}, \lambda \in [0,1]$ im Parametergebiet Ω liegen; ansonsten wäre die Definition von Konvexität nicht anwendbar. Dies bedeutet, dass Ω eine konvexe Menge sein muss. Um diese Bedingung zu überprüfen sind folgende Regeln für konvexe Mengen nützlich:

- Der gesamte Vektorraum über den reellen Zahlen $\mathbb{R}^n$ ist konvex.
- Lineare Unterräume $\{\mathbf{x} \in \mathbb{R}^n : \mathbf{A}\mathbf{x} = \mathbf{b}, \mathbf{A} \in \mathbb{R}^{m \times n}, \mathbf{b} \in \mathbb{R}^m\}$ sind konvex.
- Halbräume $\{\mathbf{x} \in \mathbb{R}^n : \mathbf{a}^\mathsf{T}\mathbf{x} < b, \mathbf{a} \in \mathbb{R}^n, b \in \mathbb{R}\}$ sind konvex.
- Der Schnitt $\Omega \cap \Omega'$ von konvexen Mengen Ω, Ω' ist konvex.

Ist also f in einem konvexen Gebiet Ω konvex, so gibt es in Ω nur ein lokales und damit globales Minimum.

Um wie im eindimensionalen Fall Ableitungen zu verwenden, müssen wir den Ableitungsbegriff auf Funktionen mit mehreren Veränderlichen erweitern. Die Ableitung beschreibt, wie sich die Funktion verändert, wenn man den Parameter ändert. Hier kommt es offenbar darauf an, *wie* man den Parameterwert $\mathbf{x} = (x_0, x_1, x_2, \ldots) \in \mathbb{R}^n$ verändert, also in welche *Richtung* man sich bewegen möchte. Ein spezieller Fall ist die Bewegung entlang der Koordinatenachsen. Bewegt man $\mathbf{x}$ entlang der i-ten Koordinatenachse, so verändert sich nur die Komponente x_i – alle anderen Komponenten bleiben fest. Dies bedeutet, wir können die Ableitung entlang der Koordinatenachse bilden, indem wir f als Funktion von nur einer Veränderlichen, nämlich x_i, betrachten; alle anderen Komponenten von $\mathbf{x}$ werden wie andere Koeffizienten behandelt. Man schreibt diese *partielle* Ableitung als $\frac{\partial f}{\partial x_i}$.

Was passiert, wenn wir entlang einer beliebigen Richtung $\mathbf{r}$ ableiten wollen? Ableiten ist eine lineare Operation, also können wir diese *Richtungsableitung* gewinnen, indem wir die einzelnen Komponenten von $\mathbf{r}$ als Gewichtungsfaktoren für die partiellen Ableitungen nehmen:

$$D_\mathbf{r} f = r_0 \frac{\partial f}{\partial x_0} + r_1 \frac{\partial f}{\partial x_1} + \ldots = \mathbf{r}^\mathsf{T} \nabla f. \tag{9.3}$$

Dabei ist der *Gradient* ∇f der *Spaltenvektor* der partiellen Ableitungen von f. Der Gradient spielt eine wichtige Rolle für die Optimierung. Dies hat zumindest zwei Gründe: 1. ist die Bedingung $\nabla f = \mathbf{0}$ die zum eindimensionalen Fall äquivalente notwendige Bedingung für

einen kritischen Punkt und 2. zeigt der Gradient in die Richtung des steilsten Anstiegs. Diese Behauptungen wollen wir im Folgenden näher betrachten.

Betrachten wir f als Funktion der einzelnen Variablen x_0, dann muss offenbar analog zum eindimensionalen Fall in einem Minimum $\mathbf{x}^*$ gelten: $\frac{\partial \mathbf{x}^*}{\partial x_0} = 0$. Andernfalls könnte man durch Bewegung entlang x_0 den Funktionswert kleiner machen. Diese Bedingung gilt aber gleichermaßen für alle Koordinatenachsen, also ist $\nabla f = \mathbf{0}$ eine notwendige Bedingung für ein Minimum. Nehmen wir nun an es gilt $\nabla f(\mathbf{x}) \neq \mathbf{0}$. In welche Richtung $\mathbf{q}$ müsste man sich bewegen, um den Funktionswert so stark wie möglich zu verändern? Die Größe der Veränderung ist $\|\mathbf{q}^\mathsf{T} \nabla f(\mathbf{x})\|$. Da es nur um die Richtung von $\mathbf{q}$ und nicht die Länge geht fordern wir $\|\mathbf{q}\| = 1$. Mit dieser Bedingung sehen wir sofort, dass $\mathbf{q} = \nabla f(\mathbf{x}) / \|\nabla f(\mathbf{x})\|$ die beste Wahl ist. Der Gradient zeigt also in Richtung der stärksten Veränderung von f. Wollten wir f mit nur einem Schritt minimieren, so erscheint ein Schritt in Richtung $-\nabla f(\mathbf{x})$ eine gute Wahl zu sein. Wir kommen auf diese Idee später zurück.

Im eindimensionalen Fall konnten wir in einem kritischen Punkt die zweite Ableitung verwenden, um Minima von Sattelpunkten und Maxima zu unterscheiden. Auch bei mehreren Veränderlichen helfen die zweiten Ableitungen, allerdings wird die Situation komplizierter. Bei n Veränderlichen gibt es n partielle Ableitungen. Jede dieser partiellen Ableitungen können wir nach allen n Veränderlichen ableiten. Diese n^2 partiellen Ableitungen schreiben wir in Form der *Hesse-Matrix*:

$$\mathbf{H}_f = \begin{pmatrix} \frac{\partial^2 f}{\partial x_0^2} & \frac{\partial^2 f}{\partial x_0 \partial x_1} & \frac{\partial^2 f}{\partial x_0 \partial x_2} & \cdots \\ \frac{\partial^2 f}{\partial x_1 \partial x_0} & \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots \\ \frac{\partial^2 f}{\partial x_2 \partial x_0} & \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix} \tag{9.4}$$

Beschränken wir uns auf zweimal stetig differenzierbare Funktionen f (was im Kontext der Anwedungen plausibel ist) so ist die Hesse-Matrix symmetrisch. Es ist also unerheblich, ob man erst nach x_i und dann nach x_j oder umgekehrt erst nach x_j und dann nach x_i ableitet. Die Hesse-Matrix wird auch verwendet, um die Taylor-Entwicklung einer Funktion in mehreren Veränderlichen bis zum quadratischen Term zu beschreiben:

$$f(\mathbf{x}) \approx f(\mathbf{a}) + (\mathbf{x} - \mathbf{a})^\mathsf{T} \nabla f(\mathbf{a}) + \frac{1}{2} (\mathbf{x} - \mathbf{a})^\mathsf{T} \mathbf{H}_f(\mathbf{a}) (\mathbf{x} - \mathbf{a}). \tag{9.5}$$

Diese Darstellung ergibt sich genau wie die eindimensionale Taylor-Entwicklung. Mithilfe des Gradienten und der Hesse-Matrix erhält man lediglich eine kompakte Darstellung der Summen.

Welche Eigenschaft muss $\mathbf{H}_f$ in einem kritischen Punkt $\mathbf{x}^*$ von f haben, damit $\mathbf{x}^*$ ein Minimum ist? Betrachten wir dazu die Taylor-Entwicklung um $\mathbf{x}^*$ für kleine Veränderungen $\mathbf{d}$:

$$f(\mathbf{x}^* + \mathbf{d}) \approx f(\mathbf{x}^*) + \mathbf{d}^\mathsf{T} \underbrace{\nabla f(\mathbf{x}^*)}_{=\mathbf{0}} + \frac{1}{2} \mathbf{d}^\mathsf{T} \mathbf{H}_f(\mathbf{x}^*) \mathbf{d} = f(\mathbf{x}^*) + \frac{1}{2} \mathbf{d}^\mathsf{T} \mathbf{H}_f(\mathbf{x}^*) \mathbf{d} \tag{9.6}$$

Wir sehen, dass f in $\mathbf{x}^*$ ein Minimum hat, wenn $\mathbf{d}^\mathsf{T} \mathbf{H}_f(\mathbf{x}^*) \mathbf{d}$ immer positiv ist. Mit anderen Worten, die Hesse-Matrix $\mathbf{H}_f(\mathbf{x}^*)$ muss positiv definit sein, damit f im kritischen Punkt $\mathbf{x}^*$ ein Minimum annimmt. Ist die Hesse-Matrix für alle $\mathbf{x} \in \Omega$ PD, dann ist f in diesem Gebiet konvex, hat also nur ein lokales und globales Minimum.

Nach diesen Vorüberlegungen betrachten wir nun konkrete Verfahren zur Bestimmung von Minimalstellen.

9.2. Goldener-Schnitt-Suche

Wir beginnen mit einem Verfahren zur Bestimmung von Extremstellen für Funktionen in einer Veränderlichen und ohne die Verwendung von Ableitungen. Das Verfahren stellt eine Analogie zur Bisektion für die Nullstellenbestimmung dar. Brauchten wir für die Bisektion zwei Werte, die eine Nullstelle eingrenzen, so brauchen wir nun drei: Intervallgrenzen x^l und x^r sowie eine Stelle x^*, die im Intervall liegt $x^l < x^* < x^r$ und deren Funktionswert kleiner ist als die Funktionswerte in den Intervallgrenzen $f(x^*) < f(x^l), f(x^r)$.

Im Verlauf des Verfahren bestimmt man neue Werte an der Stelle x' die im größeren der beiden Intervalle $[x^l, x^*]$ und $[x^*, x^r]$ liegt. Nehmen wir an es gilt $x^l < x^* < x' < x^r$. Wir unterscheiden danach, ob der neue Funktionswert $f(x')$ kleiner oder größer als $f(x^*)$ (dem aktuell kleinsten Wert) ist. Es ergibt sich sofort die folgende Regel:

$$\begin{aligned} f(x') < f(x^*) &\Longrightarrow x^l \leftarrow x^*, x^* \leftarrow x' \\ f(x') \geq f(x^*) &\Longrightarrow x^r \leftarrow x' \end{aligned} \tag{9.7}$$

Sollte das Intervall $[x^l, x^*]$ größer als das Intervall $[x^*, x^r]$ gewesen sein und hätte man $x' \in [x^l, x^*]$ gewählt, so müssen die Regeln sinngemäß angewendet werden.

Es bleibt die Frage, wie man x' wählt. Wir bezeichnen die Breite des linken Intervalls mit $l = x^* - x^l$ und des rechten mit $r = x^r - x^*$. Nehmen wir wieder an das rechte Intervall ist größer als das linke und platzieren x' im Abstand s von x^*. Im Laufe des Verfahrens möchten wir die Intervalle immer auf die gleiche Weise unterteilen. Dies bedeutet das Verhältnis der ursprünglichen Intervallbreiten l zu r entspricht dem Verhältnis der neuen Intervallteilung s zu $r - s$:

$$\frac{l}{r} = \frac{s}{r-s} \tag{9.8}$$

Aber welche Intervallteilung l ist optimal? Bei der Bisektion ergab die hälftige Teilung das beste worst-case Verhalten: unabhängig von den Funktionswerten wurde das Intervall in jedem Schritt halbiert. Im Fall den wir betrachten ergeben sich als mögliche neue Intervalle entweder $[x^*, x^r]$ oder $[x^l, x']$. Für das beste worst-case Verhalten sollten diese beiden Intervalle die gleiche Breite haben, also $r = l + s$. Zusammen mit der Bedingung oben ergibt sich

$$\frac{l}{r} = \frac{r-l}{r-(r-l)} = \frac{r}{l} - 1 \Longleftrightarrow 1 + \frac{l}{r} = \frac{r+l}{r} = \frac{r}{l}. \tag{9.9}$$

Das bedeutet, das Verhältnis der Intervallbreiten enstpricht dem Goldenen Schnitt.

Analog zum Bisektionsverfahren hat die Goldene-Schnitt-Suche unter allen eindimensionalen Optimierungsverfahren das beste worst-case Verhalten. Nur mit zusätzlicher Kenntnis über die Eigenschaften von f kann man eine Extremstelle schneller bestimmen.

9.3. Newton-Verfahren

Jetzt betrachten wir die Minimierung einer zweimal stetig differenzierbaren Funktion $f : \mathbb{R} \mapsto \mathbb{R}$ unter Verwendung der ersten und zweiten Ableitung f' und f''. Die Idee ist einfach: ein kritischer Punkt x^* is gekennzeichnet durch $f'(x^*) = 0$. Wir verwenden das Newton-Verfahren

zur Bestimmung von Nullstellen, um diese Nullstelle zu finden. Da $f''(x)$ die Ableitung der Funktion f' an der Stelle x ist, ergibt sich sofort der Newton-Schritt:

$$x_{i+1} = x_i - \frac{f'(x_i)}{f''(x_i)} \tag{9.10}$$

Das Newton-Verfahren zur Optimierung hat die gleichen guten und schlechten Eigenschaften wie das Verfahren zur Nullstellenbestimmung. Gleichermaßen lässt sich das Verfahren auf die Behandlung von Funktionen in mehreren Veränderlichen verallgemeinern. Betrachten wir also eine Funktion $f : \mathbb{R}^n \mapsto \mathbb{R}$. Dann suchen wir die Nullstelle der Funktion $\nabla f : \mathbb{R}^n \mapsto \mathbb{R}^n$. Für das Newton-Verfahren in mehreren Veränderlichen benötigten wir die Jacobi-Matrix, die Matrix der ersten Ableitungen. Da unsere Funktion die ersten Ableitungen enthält, wird die Rolle der Jacobi-Matrix jetzt durch die Matrix der zweiten Ableitungen übernommen, also der Hesse Matrix. Man erhält damit den Newton-Schritt $\Delta\mathbf{x}$ zur Optimierung von $f(\mathbf{x})$ als Lösung des linearen Gleichungssystems:

$$\mathbf{H}_f \Delta\mathbf{x} = -\nabla f(\mathbf{x}). \tag{9.11}$$

9.4. Gradientenabstieg

In vielen Fällen ist die Hesse-Matrix einer Funktion $f : \mathbb{R}^n \mapsto \mathbb{R}$ nicht oder zumindest nicht mit vertretbarem Aufwand zu bestimmen. Das Newton-Verfahren ist dann nicht anwendbar. Würden wir annehmen, f wäre um den kritischen Punkt ungefähr isotrop (würde also seine Steigung in alle Richtung ähnlich verändern), so ergäbe sich als Hesse-Matrix $s\mathbf{I}$. Damit hätte man als Iterationsschritt:

$$\mathbf{x}^{i+1} = \mathbf{x}^i - \frac{1}{s}\nabla f(\mathbf{x}^i) = \mathbf{x}^i - \delta\nabla f(\mathbf{x}^i) \tag{9.12}$$

Dieses Vorgehen erscheint intuitiv plausibel: Der Gradient $\nabla f(\mathbf{x})$ zeigt in die Richtung des steilsten Anstieges der Funktion f. Wenn wir ein Minimum von f suchen, sollten wir offenbar einen Schritt in die entgegengesetzte Richtung machen. Verfahren, die auf dieser Grundidee beruhen, heißen *Gradientenabstiegsverfahren*. Zentrales Problem dieser Verfahren ist die Bestimmung der Schrittgröße δ.

Ein klassischer Ansatz ist die Bestimmung der Schrittgröße als Optimierungsproblem in einer Variablen aufzufassen. Man möchte also die Funktion $f(\mathbf{x}_i - \delta\nabla f(\mathbf{x}^i)) = g(\delta)$ durch Variation von δ minimieren. Dazu kann man das bereits vorgestellte Verfahren des Goldenen Schnitts verwenden. Dafür braucht man drei Werte von δ, so dass eine Minimalstelle eingegrenzt wird. Typischerweise lässt man das Intervall bei $\delta = 0$ beginnen. Dann sucht man ausgehend von einem festen Wert (bspw. $\delta = 1$) zunächst eine Stelle mit $g(\delta^r) > g(0)$ und dann eine Stelle $g(\delta^*) < g(0)$. Dies kann z.B. durch die Ersetzung $\delta \leftarrow \alpha\delta$ geschehen, wobei $\alpha > 1$ zum Finden einer rechten Schranke und dann $\alpha < 1$ zum Finden eines Minimums verwendet wird. Ist das Tripel bestimmt kann das Verfahren des Goldenen Schnitts wie beschrieben durchgeführt werden.

Es zeigt sich allerdings in der Praxis, dass der Rechenaufwand für die exakte Bestimmung des Minimums meist nicht lohnt. Es genügt vielmehr, bei der Wahl von δ gewisse Bedingungen zu erfüllen. Wir gehen hier nicht auf die Details ein und fordern stattdessen einfach, dass der Funktionswert kleiner wird: $f(\mathbf{x}_i - \delta\nabla f(\mathbf{x}^i)) < f(\mathbf{x}_i)$. Die Schrittweite δ kann man dann bestimmen wie oben angegeben, nämlich indem man mit einem festen Wert beginnt, und dann solange kleinere Werte wählt bis die Bedingung erfüllt ist.

9.5. Konjugierte Gradienten

Warum ist es eigentlich keine gute Idee, entlang des Gradienten zu minimieren? Oder anders, unter welchen Umständen führt diese Verfahren schnell zum Minimum? Um das zu verstehen, approximieren wir die Funktion $f : \mathbb{R}^n \mapsto \mathbb{R}$ lokal durch eine quadratische Funktion

$$f(\mathbf{x}) = \mathbf{c} - \mathbf{x}^\mathsf{T}\mathbf{b} + \frac{1}{2}\mathbf{x}^\mathsf{T}\mathbf{A}\mathbf{x}, \quad \mathbf{A} \in \mathbb{R}^{n\times n}, \mathbf{A}^\mathsf{T} = \mathbf{A}, \mathbf{b}, \mathbf{c} \in \mathbb{R}^n. \tag{9.13}$$

Wir können uns diese Darstellung als eine lokale Taylor-Approximation vorstellen – dabei spielt die symmetrische Matrix $\mathbf{A}$ offenbar die Rolle der Hesse-Matrix. Die besondere Schreibweise wird plausibel wenn man den Gradienten dieser Funktion bildet

$$\nabla f(\mathbf{x}) = \mathbf{A}\mathbf{x} - \mathbf{b} \tag{9.14}$$

und dann sieht, dass kritische Punkte die linearen Bedingungen $\mathbf{A}\mathbf{x} = \mathbf{b}$ erfüllen müssen.

Was passiert in diesem Fall bei Gradientenabstieg? Die einfache Form von f erlaubt uns, die Schrittweite zum Minimum entlang des Gradienten explizit auszurechnen. Wir suchen das Minimum der Funktion

$$g(\delta) = f(\mathbf{x} - \delta\nabla f(\mathbf{x})) = \mathbf{c} - (\mathbf{x} - \delta\nabla f(\mathbf{x}))^\mathsf{T}\mathbf{b} + \frac{1}{2}(\mathbf{x} - \delta\nabla f(\mathbf{x}))^\mathsf{T}\mathbf{a}(\mathbf{x} - \delta\nabla f(\mathbf{x})). \tag{9.15}$$

Ableiten (nach δ) und Nullsetzen ergibt

$$\begin{aligned} 0 = g'(\delta) &= \nabla f(\mathbf{x})^\mathsf{T}\mathbf{b} - \nabla f(\mathbf{x})^\mathsf{T}\mathbf{A}(\mathbf{x} - \delta\nabla f(\mathbf{x})) \\ &= \nabla f(\mathbf{x})^\mathsf{T}(\mathbf{A}\mathbf{x} - \nabla f(\mathbf{x})) - \nabla f(\mathbf{x})^\mathsf{T}\mathbf{A}(\mathbf{x} - \delta\nabla f(\mathbf{x})) \\ &= -\nabla f(\mathbf{x})^\mathsf{T}\nabla f(\mathbf{x})) + \delta\nabla f(\mathbf{x})^\mathsf{T}\mathbf{A}\nabla f(\mathbf{x})) \end{aligned} \tag{9.16}$$

und damit

$$\delta = \frac{\nabla f(\mathbf{x})^\mathsf{T}\nabla f(\mathbf{x})}{\nabla f(\mathbf{x})^\mathsf{T}\mathbf{A}\nabla f(\mathbf{x})}. \tag{9.17}$$

Der Gradient für den nächsten Schritt des Verfahrens ist dann

$$\mathbf{A}\left(\mathbf{x} - \frac{\nabla f(\mathbf{x})^\mathsf{T}\nabla f(\mathbf{x})}{\nabla f(\mathbf{x})^\mathsf{T}\mathbf{A}\nabla f(\mathbf{x})}\nabla f(\mathbf{x})\right) - \mathbf{b} = \left(\mathbf{I} - \frac{\nabla f(\mathbf{x})^\mathsf{T}\nabla f(\mathbf{x})}{\nabla f(\mathbf{x})^\mathsf{T}\mathbf{A}\nabla f(\mathbf{x})}\mathbf{A}\right)\nabla f(\mathbf{x}) \tag{9.18}$$

und man sieht, dass er orthogonal auf $\nabla f(\mathbf{x})$ steht:

$$\nabla f(\mathbf{x})^\mathsf{T}\left(\mathbf{I} - \frac{\nabla f(\mathbf{x})^\mathsf{T}\nabla f(\mathbf{x})}{\nabla f(\mathbf{x})^\mathsf{T}\mathbf{A}\nabla f(\mathbf{x})}\mathbf{A}\right)\nabla f(\mathbf{x}) = \nabla f(\mathbf{x})^\mathsf{T}\nabla f(\mathbf{x}) - \frac{\nabla f(\mathbf{x})^\mathsf{T}\nabla f(\mathbf{x})}{\nabla f(\mathbf{x})^\mathsf{T}\mathbf{A}\nabla f(\mathbf{x})}\nabla f(\mathbf{x})^\mathsf{T}\mathbf{A}\nabla f(\mathbf{x}) = 0. \tag{9.19}$$

Diese Beobachtung ist auch intuitiv einleuchtend. Läuft man entlang einer Richtung bis zum Minimum der Funktion, muss der Gradient, also die Richtung des steilsten Anstiegs, orthogonal auf der Bewegungsrichtung stehen.

Es mag zunächst vernünftig erscheinen, in jedem Schritt eine Richtung zu wählen, die orthogonal auf der vorher gewählten Richtung steht. Man überzeugt sich aber schnell an Beispielen, dass dies im Allgemeinen keine gute Idee ist. Wir machen allerdings folgende Beobachtung: Wenn die Matrix $\mathbf{A}$ ein Vielfaches der Identität ist, also der Gradient der Funktion in alle Richtungen gleichermaßen wächst, dann wäre die Wahl von orthogonalen Richtungen gut. Genauer: wählt

man als Richtungen im $\mathbb{R}^n$ irgendeine Orthogonalbasis und minimiert nacheinander entlang der Basisvektoren, so ist das Minimum nach n Schritten gefunden.

Nun hat aber in der Praxis die Matrix $\mathbf{A}$ (die Hesse-Matrix der lokalen Taylor-Approximation) typischerweise nicht die Form $s\mathbf{I}$. Wir könnten aber versuchen, die Koordinaten $\mathbf{x}$ durch eine Koordinaten-Transformation in ein Koordinatensystem zu übertragen, indem diese Situation gegeben ist. Dazu verwenden wir die Diagonalisierung der (symmetrischen) Matrix $\mathbf{A} = \mathbf{Q}^\mathsf{T}\Lambda\mathbf{Q}$. Dann sehen wir durch

$$\mathbf{x}^\mathsf{T}\mathbf{A}\mathbf{x} = \mathbf{x}^\mathsf{T}\mathbf{Q}^\mathsf{T}\Lambda\mathbf{Q}\mathbf{x} = \left(\mathbf{x}^\mathsf{T}\mathbf{Q}^\mathsf{T}\Lambda^{\frac{1}{2}}\right)\mathbf{I}\left(\Lambda^{\frac{1}{2}}\mathbf{Q}\mathbf{x}\right) = \left(\Lambda^{\frac{1}{2}}\mathbf{Q}\mathbf{x}\right)^\mathsf{T}\mathbf{I}\left(\Lambda^{\frac{1}{2}}\mathbf{Q}\mathbf{x}\right), \tag{9.20}$$

die gewünschte Transformation

$$\mathbf{y} = \Lambda^{\frac{1}{2}}\mathbf{Q}\mathbf{x}. \tag{9.21}$$

In den transformierten Koordinaten $\mathbf{y}$ hätten wir gerne, dass aufeinanderfolgende Richtungen orthogonal sind. Für die Variablen $\mathbf{x}$ bedeutet das

$$0 = \mathbf{y}^\mathsf{T}\mathbf{y}' = \left(\mathbf{x}^\mathsf{T}\mathbf{Q}^\mathsf{T}\Lambda^{\frac{1}{2}}\right)\left(\Lambda^{\frac{1}{2}}\mathbf{Q}\mathbf{x}'\right) = \mathbf{x}^\mathsf{T}\mathbf{Q}^\mathsf{T}\Lambda\mathbf{Q}\mathbf{x}' = \mathbf{x}^\mathsf{T}\mathbf{A}\mathbf{x}' \tag{9.22}$$

Wenn wir also zwei Vektoren $\mathbf{x}$ und $\mathbf{x}'$ so wählen wollen, dass sie im transformierten Fall orthogonal wären, dann müssen sie bezüglich des durch $\mathbf{A}$ gegebenen Innenproduktes orthogonal sein. Mann nennt die Vektoren $\mathbf{x}, \mathbf{x}'$ *konjugiert*, falls dies der Fall ist, also $\mathbf{x}^\mathsf{T}\mathbf{A}\mathbf{x}' = 0$.

Die Idee eines verbesserten Gradientenabstiegsverfahrens ist nun die folgende: Man bestimmt in jedem Schritt des Verfahrens den Gradienten. Man verwendet aber nicht den Gradienten als Richtung für die Minimierung sondern modifiziert ihn so, dass der Richtungsvektor zu allen vorher verwendeten Richtungsvektoren konjugiert ist, also orthogonal bzgl. des durch $\mathbf{A}$ gegebenen Innenproduktes . Nennen wir die Richtung im Schritt i des Verfahrens $\mathbf{r}^i$, dann soll also gelten

$$\mathbf{r}^{i^\mathsf{T}}\mathbf{A}\mathbf{r}^j, \quad j < i. \tag{9.23}$$

Wie erreichen diese Eigenschaft, indem wir aus dem Gradienten im Schritt i die Anteile aller bisher verwendeten Richtungen herausprojizieren. Der Anteil von $\mathbf{r}^j$ in $\nabla f(\mathbf{x}^i)$ ist gegeben durch das Innenprodukt – man erhält also

$$\mathbf{r}^i = \nabla f(\mathbf{x}^i) - \sum_{j=0}^{i-1} \frac{\mathbf{r}^{j^\mathsf{T}}\mathbf{A}\nabla f(\mathbf{x}^i)}{\mathbf{r}^{j^\mathsf{T}}\mathbf{A}\mathbf{r}^j}\mathbf{r}^j \tag{9.24}$$

Diese Konstruktion ist analog zum Gram-Schmidt-Verfahren zur Orthogonalisierung von Vektoren. Durch Nachrechnen ließe sich auch leicht überprüfen, dass die gewünschten Bedinungen alle erfüllt sind.

Ein praktisches Verfahren hat man mit dieser Rechnung allerdings nicht gewonnen. Dies hat zwei Gründe: einerseits ist der Ausgangspunkt gewesen, dass die Hesse-Matrix sich auch nur schwierig bestimmen lässt – in der Anwendung ist aber $\mathbf{A}$ genau die Hesse-Matrix. Andererseits wollen wir für eine sehr große Zahl von Variablen möglichst vermeiden, alle Vektoren $\mathbf{r}^i$ die im Laufe des Verfahrens entstehen zu speichern.

Die Verwendung von Gradienten als Ausgangspunkt erlaubt eine erhebliche Vereinfachung, die die Eigenschaften der konjugierten Richtungen $\{\mathbf{r}^i\}$ und die Orthogonalität der Gradienten ausnutzt. Die Rechnung ist langwierig (weswegen wir sie hier nicht wiedergeben) aber nicht schwierig. Als Ergebnis erhält man:

$$\mathbf{r}^i = \nabla f(\mathbf{x}^i) + \frac{\nabla f(\mathbf{x}^i)^\mathsf{T}\nabla f(\mathbf{x}^i)}{\nabla f(\mathbf{x}^{i-1})^\mathsf{T}\nabla f(\mathbf{x}^{i-1})}\mathbf{r}^{i-1}. \tag{9.25}$$

Um die Richtung in Schritt i zu bestimmen, braucht man also nur die Richtung des vorangegangenen Schrittes, sowie den aktuellen und letzten Gradienten. Das komplette Verfahren ergibt sich durch die Wahl eines Startpunktes $\mathbf{x}^0$, den Gradienten als erste Minimierungsrichtung $\mathbf{r}^0 = \nabla f(\mathbf{x}^0)$ und im Folgenden die Schritte

$$\mathbf{x}^i = \mathbf{x}^{i-1} - \delta \mathbf{r}^{i-1}, \quad i > 0. \tag{9.26}$$

Dabei ist es wichtig, dass in jedem Schritt δ die Funktion f tatsächlich minimiert. Ist dies der Fall, dann wissen wir aus unseren Überlegungen, dass nach n Schritten das Minimum jeder quadratischen Funktion f gefunden wäre. Für Funktionen, die komplizierter als ein quadratisches Polynom sind, hängt die Anzahl der Schritte davon ab, wie gut die Funktion durch eine quadratische Funktion approximiert wird.

9.6. Lagrange-Multiplikatoren

Jetzt wollen wir uns Optimierungsproblemen mit Nebenbedingungen widmen. Dabei richten wir unser Augenmerk zunächst auf Probleme der Form

$$\arg\min_{\mathbf{x}} f(\mathbf{x}), \quad g(\mathbf{x}) = c. \tag{9.27}$$

Wir suchen also nicht mehr das Minimum von $f(\mathbf{x})$ unter allen $\mathbf{x}$, sondern betrachten nur solche $\mathbf{x}$, die die Bedingung $g(\mathbf{x}) = c$ erfüllen.

Die Menge $\{\mathbf{x} : g(\mathbf{x}) = c\}$ beschreibt die erlaubten Punkte. Zur Veranschaulichung betrachten wir zunächst Funktionen, die zwei Veränderliche haben, also $f, g : \mathbb{R}^2 \mapsto \mathbb{R}$. Dann können wir uns g als Höhenfunktion über der Ebene vorstellen. Die Menge $\{\mathbf{x} : g(\mathbf{x}) = c\}$ ist dann eine Höhenlinie, also eine ebene Kurve. Bewegen wir uns entlang der Kurve, so ändert sich der Wert von g nicht (das ist ja gerade die Definition). Die Veränderung von g entlang irgendeiner Richtung $\mathbf{r}$ ist $\mathbf{r}^\mathsf{T} \nabla g$. Da sich g in Bewegungsrichtung gar nicht ändert ist dieses Produkt also 0 oder, mit andern Worten, die Bewegungsrichtung steht orthogonal auf dem Gradienten ∇g.

Unser Ziel ist es, das Minimum von f auf der Kurve zu finden. Wir bewegen uns also entlang der Kurve so, dass der Wert von f fällt. Ein kritischer Punkt ist erreicht, wenn sich der Wert von f nicht mehr verändert. In diesem Punkt ist die Bewegungsrichtung also auch orthogonal zum Gradienten ∇f von f.

Ein Minimum $\mathbf{x}^*$ von f auf der Kurve $\{\mathbf{x} : g(\mathbf{x}) = c\}$ ist also dadurch charakterisiert, dass die Bewegungsrichtung orthogonal zu sowohl $\nabla f(\mathbf{x}^*)$ wie auch $\nabla g(\mathbf{x}^*)$ ist. Damit sind die Vektoren $\nabla f(\mathbf{x}^*)$ und $\nabla g(\mathbf{x}^*)$ parallel – sie zeigen in die selbe (oder entgegengesetzte) Richtung haben aber verschiedene Länge. Diese Beobachtung machen wir gleichermaßen für Funktionen $f, g : \mathbb{R}^n \mapsto \mathbb{R}$ – die Menge $\{\mathbf{x} : g(\mathbf{x}) = c\}$ erlaubt dann im Allgemeinen $n - 1$ Bewegungsrichtungen, die alle orthogonal auf beiden Gradienten stehen. Wir können die notwendige Bedingung für einen kritischen Punkt ganz Allgemein also wie folgt schreiben:

$$\nabla f(\mathbf{x}^*) = \lambda \nabla g(\mathbf{x}^*). \tag{9.28}$$

Wie hilft uns diese Beobachtung, um das ursprüngliche Minimierungsproblem zu lösen? Die Idee ist eine Funktion zu konstruieren, deren kritische Punkte die Parallelitätsbedingung erfüllen. Eine solche Funktion ist gegeben durch die *Lagrange-Funktion*

$$L(\mathbf{x}, \lambda) = f(\mathbf{x}) - \lambda(g(\mathbf{x}) - c). \tag{9.29}$$

Betrachten wir nämlich alle partiellen Ableitungen bzgl. $\mathbf{x}$

$$\nabla_{\mathbf{x}} L = \nabla f - \lambda \nabla g \tag{9.30}$$

und setzen dies gleich Null, so ergibt sich direkt die oben formulierte Bedingung. Zusätzlich ergibt das Nullsetzen der partiellen Ableitung nach λ

$$\frac{\partial L}{\partial \lambda} = g(\mathbf{x}) - c \tag{9.31}$$

die gewünschte Bedingung $g(\mathbf{x}) = c$.

Wir haben also die Minimierung einer Funktion $f : \mathbb{R}^n \mapsto \mathbb{R}$ mit Nebenbedingung in die Minimierung einer neuen Funktion $L : \mathbb{R}^{n+1} \mapsto \mathbb{R}$ *ohne Nebenbedingung* übersetzt. Diese neue Funktion können wir nun mit einem der bereits vorgestellten Verfahren lösen. Der zusätzliche Parameter λ wird *Lagrange-Multiplikator* genannt.

Die Grundidee lässt sich einfach auf mehr als eine Nebenbedingung erweitern. Allgemein ergibt sich aus der Problemstellung

$$\arg\min_{\mathbf{x}} f(\mathbf{x}), \quad g_i(\mathbf{x}) = c_i, i \in \{0, 1, \ldots\} \tag{9.32}$$

die Lagrange-Funktion

$$L(\mathbf{x}, \lambda_0, \lambda_1, \ldots) = f(\mathbf{x}) - \sum_i \lambda_i (g_i(\mathbf{x}) - c_i). \tag{9.33}$$

10. Numerische Integration

10.1. Motivation

Eine Vielzahl von Problemen in den Ingenieursdisziplinen, aber auch Simulation von Phänomenen aus der Physik, Chemie, Biologie erfordert *Integration*. Dies geht von der Auswertung von bestimmten Integralen in einer oder mehreren Dimension(en) bis zur Behandlung von *Differentialgleichungen*.

Maße – Quadratur Integrale bestimmen *Maße*. Immer wenn man Längen, Flächen, Volumen, etc. bestimmen will, kann dies durch Integrale ausgedrückt werden. In einigen Fällen lassen sich diese Integrale geschlossen ausdrücken, z.B. der Flächeninhalt eines Dreiecks. Möchte man aber den Flächeninhalt einer krummlinig begrenzten Figur ausrechnen, so gibt es zum Integranden meist keine einfache Stammfunktion und es bleibt nur die numerische Approximation des Integrals. Diesen Vorgang nennt man auch *Quadratur*, weil man die Seitenlänge eines flächengleichen Quadrates bestimmt.

Ein überraschend einfaches Beispiel für die Notwendigkeit von numerischen Verfahren ist das *elliptische* Integral: im einfachsten Fall ist das die Länge eines Sektors auf einer Ellipse. Eine Ellipse lässt sich parametrisch beschreiben durch die Funktion $(x(t), y(t)) = (a\cos t, b\sin t)$. Zur Bestimmung der Länge eines elliptischen Bogens integriert man die Länge des Tangentenvektors, also

$$L = \int_{t_0}^{t_1} \left((-a\sin t, b\cos t)(-a\sin t, b\cos t)^{\mathsf{T}} \right)^{\frac{1}{2}} \, dt = \int_{t_0}^{t_1} \sqrt{a^2\sin^2 t + b^2\cos^2 t}\, dt.$$

Bereits für dieses 'harmlos' aussehende Integral gibt es (außer für den Spezialfall $a = b$) keine geschlossene Lösung.

Integralfunktionen Viele wichtige Funktionen sind über Integrale definiert. Eine zentrale Rolle für Prozesse mit einem einem zufälligen Ausgang spielt die Gaußsche Normalverteilung, die im wesentlichen durch die Funktion e^{-x^2} gegeben ist. Will man bestimmen, wie wahrscheinlich es ist, dass ein bestimmtes, normalverteiltes Ereignis eintritt, so muss man die Fläche unter dieser Kurve bestimmen. Dies ist die (Gaußsche) *Fehlerfunkion*

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2}\, dt.$$

Da diese Funktion für viele Anwendungen von zentraler Bedeutung ist, ist bereits in jeder mathematischen Bibliothek ein numerisches Verfahren implementiert. Aber nicht jeder Prozess folgt einer Normalverteilung. Wenn wir die Fehlerfunktion für eine weniger übliche Verteilung bestimmen wollen bleibt oft nur die Bestimmung des Integrals mit numerischer Integration.

Integralfunktionen werden auch unabhängig von Zufallsprozessen überall in den Naturwissenschaften verwendet. So ist z.B. der *Integrallogarithmus* $\operatorname{li}(x) = \int_0^x (\ln t)^{-1}\, dt$ nicht nur in der Physik wichtig, sondern auch in der Form $\operatorname{Li}(x) = \operatorname{li}(x) - \operatorname{li}(2)$ eine bekannte Abschätzung der Anzahl von Primzahlen bis zum Wert x.

Differentialgleichungen t.b.c.

10.2. Von Interpolation zu Integration

Zur numerischen Bestimmung des Integrals

$$\int_a^b f(x)\,dx \tag{10.1}$$

haben wir bereits ein Verfahren kennengelernt: Interpolation! Bei der Interpolation haben wir die Funktion $f(x)$ aus Paaren $x_i, f_i = f(x_i)$ durch eine (stückweise) Polynomfunktion approximiert – und Polynome sind einfach zu integrieren. Wir könnten also Stellen x_i wählen, die gegebene Funktion f an diesen Stellen auswerten, dann die stückweise polynomielle Interpolationsfunktion bestimmen und diese dann integrieren.

Wir unterscheiden dabei wieder zwei Aspekte: die Unterteilung des Intervals $[a, b]$ in Teilintervalle und dann die Bestimmung der Polynomfunktion in jedem Teilintervall. Da wir im Allgemeinen wenig über die Funktion f sagen können, wird das Intervall $[a, b]$ zumeist in Teile gleicher Breite unterteilt. Dies soll so geschehen, dass f in jedem Teilstück gut durch ein Polynom approximiert werden kann. Anders als bei Interpolation ist für die Integration der Übergang der Funktion von einem Teilstück in das Nächste nicht von Bedeutung. Wir erlauben also, dass die Approximation von f an den Grenzen der Teilstücke nicht stetig oder nicht stetig differenzierbar ist.

Je nach Eigenschaften von f mag bereits das gegebene Intervall $[a, b]$ ausreichend klein sein, oder aber es ist eine Teilung in sehr viele Stücke nötig. Ohne Kenntnis der Funktion f und der Breite des Intervalls lässt sich hierzu nichts Generelles sagen. Im Folgenden gehen wir davon aus, dass eine Unterteilung bereits stattgefunden hat und das Intervall $[a, b]$ ausreichend klein gewählt ist.

Interpolation mit Lagrange-Polynomen Wir erinnern uns daran, dass die Lagrange-Basispolynome zu einer einfachen Lösung des Interpolationsproblems führten: Gegeben Paare $(x_i, f(x_i))$ ergibt sich das Interpolationspolynom einfach als $f(x) \approx \sum_i f(x_i) l_i(x)$. Die Lagrange-Polynome $l_i(x)$ sind dabei abhängig von den Stützstellen x_i, siehe Abschnitt 6.2.3 und dort Gl. 6.14 für die Definition von l_i.

Setzen wir diese Approximation ein, so erhalten wir

$$\begin{aligned}\int_a^b f(x)\,dx \approx \int_a^b \sum_i f(x_i) l_i(x)\,dx = \sum_i f(x_i) \int_a^b l_i(x)\,dx = \\ (b-a) \sum_i f(x_i) \frac{1}{b-a} \int_a^b l_i(x)\,dx = (b-a) \sum_i w_i f(x_i).\end{aligned} \tag{10.2}$$

Doe Approximation des Integrals besteht also aus einer gewichteten Summe der Funktionswerte von f an den Stellen x_i mit den Gewichten

$$w_i = \frac{1}{b-a} \int_a^b l_i(x)\,dx. \tag{10.3}$$

Da die Funktionen l_i Polynome sind, lassen sich die Gewichte auf einfache Weise ausrechnen, z.B. in dem man die Lagrange-Polynome in die Monombasis überführt und dann in der

Monombasis nach den bekannten Regeln die Stammfunktion ausrechnet. Verschiedene Verfahren der numerischen Integration unterscheiden sich danach, wie die Stützstellen x_i ausgewählt werden.

10.2.1. Newton-Cotes Formeln

Es liegt Nahe, die Stützstellen x_i *gleichabständig* zu wählen, also mit einem konstanten Abstand $h = x_{i+1} - x_i$. Dieser Ansatz führt auf Gewichte w_i, die unabhängig vom Intervall $[a, b]$ ausgerechnet werden können. Man unterscheidet danach, ob die erste und letzte Stützstelle auf dem Rand des Intervalls liegen, oder nicht. Man unterscheidet typischerweise drei Fälle:

1. Man setzt x_0 und $h = (b - a)/n$. Daraus ergeben sich $n + 1$ Stützstellen $x_i = a + ih$, die das Intervall $[a, b]$ in n Stücke teilen.
2. Man teilt das Intervall in $n + 2$ Stücke der Breite $h = (b - a)/(n + 2)$ und setzt $x_i = a + (i + 1)h$. Die erste Stützstelle liegt also bei $x_0 = a + h$.
3. Man teilt das Intervall in $n + 1$ Stücke der Breite $h = (b - a)/(n + 1)$ und setzt $x_i = a + (i + 1/2)h$. Die erste Stützstelle liegt dann bei $x_0 = a + h/2$.

Die ersten beiden Fälle unterscheiden sich in der Anordnung der Stützstellen nur dahingehend, ob die Intervallgrenzen Teil der Punkte sind, an denen die Funktion ausgewertet wird. Diese beiden Fälle führen auf die *Newton-Cotes* Formeln für die Gewichte. Die erste Variante heißt *abgeschlossen*, die zweite *offen*. Der dritte Fall führt auf die *Maclaurin* Formeln.

Man kann für feste Wahl der Stützstellen und gegebenes n nun die Gewichte ausrechnen und gelangt zu folgendem Ergebnis:

n	Abgeschlossene NC	Offene NC	Maclaurin
0	-	1	1
1	$\frac{1}{2}, \frac{1}{2}$	$\frac{1}{2}, \frac{1}{2}$	$\frac{1}{2}, \frac{1}{2}$
2	$\frac{1}{6}, \frac{4}{6}, \frac{1}{6}$	$\frac{2}{3}, -\frac{1}{3}, \frac{2}{3}$	$\frac{2}{8}, \frac{2}{8}, \frac{3}{8}$
3	$\frac{1}{8}, \frac{3}{8}, \frac{3}{8}, \frac{1}{8}$	$\frac{11}{24}, \frac{1}{24}, \frac{1}{24}, \frac{11}{24}$	$\frac{13}{48}, \frac{11}{48}, \frac{11}{48}, \frac{12}{48}$

Wegen der Schwingungsneigung von Polynominterpolation werden größere n kaum verwendet. Von den offenen Newton-Cotes Formeln hat nur der Fall $n = 0$, die *Mittelpunktsregel* praktische Bedeutung. Für höheres n, sind die offenen Newton-Cotes Formeln wegen der negativen (oder sehr kleinen) Gewichte nicht zu empfehlen.

10.3. Gauß-Integration

Im letzten Abschnitt haben wir gesehen, dass sich das bestimmte Integral $\int_a^b f(x)\,dx$ mit $n+1$ Stützstellen exakt bestimmen lässt, wenn $f(x)$ ein Polynom mit Grad n ist. Dabei ist die Wahl der Stützstellen egal und man kann z.B. die Newton-Cotes Formeln verwenden. Man könnte sich die Frage stellen, ob eine andere, besonders geschickte Wahl der Stützstellen zu einem besseren Verfahren führt. Inwiefern könnte das Verfahren besser werden? Wie schon Gauß festgestellt hat, ist es möglich mit $n + 1$ Stützstellen Polynome höheren Grades exakt zu integrieren, und zwar bis zum Grad von $2n + 1$. Das aus dieser Beobachtung entstehende

Verfahren ist tatsächlich nicht nur besser für entsprechende Polynome geeignet, sondern ergibt auf vielen 'gutmütigen' Funktionen einen geringeren Fehler.

Um die Voraussetzungen für dieses Verfahren zu verstehen, nehmen wir an $f(x)$ sei ein Polynom vom Grad $2n+1$. Dann können wir schreiben $f(x) = p_{n+1}(x)q(x) + r(x)$. Dabei ist $p_{n+1}(x)$ ein Polynom mit Grad $n+1$, für das wir uns im Folgenden besondere Eigenschaften überlegen werden. Die Funktionen $q(x), r(x)$ sind Polynome vom Grad höchstens n, die sich aus $f(x)$ und $p_{n+1}(x)$ ergeben (man teilt $f(x)$ durch $p_{n+1}(x)$ mit Rest). Wir wollen, dass die Approximation des bestimmten Integrals durch $\sum_i w_i f(x_i)$ für besondere Stützstellen x_i exakt wird, also:

$$\begin{aligned}\int_a^b p_{n+1}(x)q(x) + r(x)\,dx = \int_a^b p_{n+1}(x)q(x) + \int_a^b r(x)\,dx = \\ \sum_i w_i f(x_i) = \sum_i w_i p_{n+1}(x_i)q(x_i) + \sum_i w_i r(x_i).\end{aligned} \tag{10.4}$$

Betrachten wir jeweils nur den zweiten Teil der Summen, also $\int_a^b r(x)\,dx$ und $\sum_i w_i r(x_i)$, so wird das Integral eines Polynoms vom Grad n durch die gewichtete Summe von Funktionswerten ausgedrückt. Wir wissen bereits, dass dies für jede Wahl von Stützstellen möglich ist. Die Idee von Gauß-Integration ist nun, durch die Wahl von p_{n+1} und x_i die anderen Terme zu *eliminieren*.

Wenn p_{n+1} gegeben ist, werden wir die Terme der Art $w_i p_{n+1}(x_i)q(x_i)$ los, indem wir x_i als Nullstelle des Polynoms p_{n+1} wählen – dann ist offenbar $p_{n+1}(x_i) = 0$. Allerdings müssen dann auch alle $n+1$ Nullstellen von p_{n+1} reell sein und im Intervall $[a, b]$ liegen. Die Idee, Nullstellen eines Polynoms als Stützstellen zu verwenden, sollte uns nicht neu vorkommen: wir hatten dies bereits für die Polynominterpolation zur Vermeidung von ungewünschten Oszillation verwendet und dort für Tschebychev-Polynome gezeigt, dass alle Nullstellen reell sind und im gewünschten Intervall liegen (siehe Abschnitt 6.4.3). Hier werden wir diese Eigenschaft als einen Spezialfall für eine große Klasse von Polynom-Basen wiederfinden.

Für die Eigenschaft $\int_a^b p_{n+1}(x)q(x) = 0$ werden wir das Integral des Produkts als *Skalarprodukt* auf den Polynomen interpretieren. Die beiden Polynome sollen also *orthogonal* sein – mit dieser Eigenschaft von Polynombasen beschäftigen wir uns als nächstes.

10.3.1. Orthogonale Polynome

Wir hatten bereits bei der Interpolation gesehen, dass Polynome einen Vektorraum bilden. Es ist üblich, allgemeiner von *Funktionsräumen* zu sprechen. Man definiert dazu ein Skalarprodukt für Funktionen wie folgt:

$$\langle f, g\rangle = \int_a^b f(x)g(x)\,dx \tag{10.5}$$

und überzeugt sich direkt davon, dass dieses Skalarprodukt die gewohnten Eigenschaften hat. Zwei Funktionen f, g für die gilt $\langle f, g\rangle = 0$ heißen – wie wir das von Vektoren gewöhnt sind – *orthogonal*. Genau wie orthogonale Basen von Vektorräumen haben orthogonale Basen von Funktionsräumen oft nützliche Eigenschaften. Hier interessieren wir uns für orthogonale Basen von Polynomen p_i mit Grad i.

Eine solche Basis lässt sich durch Orthogonalisierung der Monombasis erzeugen, z.B. mit dem Gram-Schmidt-Verfahren (siehe B.3.4). Wir beginnen also mit $p_0 = 1$ und setzen dann

$$p_1 = x - \frac{\langle p_0, x\rangle}{\langle p_0, p_0\rangle} p_0 \tag{10.6}$$

$$p_2 = x^2 - \frac{\langle p_0, x^2\rangle}{\langle p_0, p_0\rangle} p_0 - \frac{\langle p_1, x^2\rangle}{\langle p_1, p_1\rangle} p_1 \tag{10.7}$$

$$\vdots$$

$$p_i = x^i - \sum_{j=0}^{i-1} \frac{\langle p_j, x^i\rangle}{\langle p_j, p_j\rangle} p_j. \tag{10.8}$$

Um die Polynome auszurechnen, können wir sie in Monombasis schreiben: $p_i = (1, x, \ldots, x^i)\mathbf{c}_i$. Dann ergibt das Gram-Schmidt-Verfahren unmittelbar die Koeffizienten: $\mathbf{c}_0 = \mathbf{e}_0 = (1, 0, \ldots)^\mathsf{T}$ und

$$\mathbf{c}_i = \mathbf{e}_i - \sum_{j=0}^{i-1} \frac{\langle p_j, x^i\rangle}{\langle p_j, p_j\rangle} \mathbf{c}_j. \tag{10.9}$$

Neben dem Rechnen mit Vektoren brauch man hier also 'nur' das Skalarprodukt von Polynomen (in Monombasis). Nach Definition ist das ein Integral über das Produkt der Polynome. Produkte von Polynomen sind wieder Polynome – und die lassen sich einfach durch die Bildung von Stammfunktionen integrieren. Konkret ist der Integrand im Ausdruck $\langle p_i, p_j\rangle = \langle p_j, p_i\rangle, j < ik$ gegeben durch

$$\left(1, x, \ldots, x^j\right) \mathbf{c}_j \left(1, x, \ldots, x^i\right) \mathbf{c}_i = \mathbf{c}_j^\mathsf{T} \begin{pmatrix} 1 & x & \ldots & x^{i-1} & x^i \\ x & x^2 & \ldots & x^i & x^{i+1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x^{j-1} & x^j & \ldots & x^{j+i-2} & x^{j+i-1} \\ x^j & x^{j+1} & \ldots & x^{j+i-1} & x^{j+i} \end{pmatrix} \mathbf{c}_i \tag{10.10}$$

Die Stammfunktion dieses Ausdrucks ergibt sich direkt durch die Stammfunktionen der Monome – man ersetzt also einfach in der Matrix x^i durch $\frac{1}{i+1}x^{i+1}$. Wenn wir in diese Matrizen die Intervallgrenzen a, b einsetzen, ergeben sich die Matrizen

$$\begin{aligned} \mathbf{A}_{ij} &= \left(\frac{1}{k+l+1} a^{k+l+1}\right)_{kl}, \quad 0 \le k \le j, \quad 0 \le l \le i, \\ \mathbf{B}_{ij} &= \left(\frac{1}{k+l+1} b^{k+l+1}\right)_{kl}, \quad 0 \le k \le j, \quad 0 \le l \le i. \end{aligned} \tag{10.11}$$

Damit erhält man für

$$\langle p_i, p_j\rangle = \int_a^b p_i(x) p_j(x)\, dx = \int_a^b \left(1, x, \ldots, x^j\right) \mathbf{c}_j \left(1, x, \ldots, x^i\right) \mathbf{c}_i\, dx = \mathbf{c}_i \left(\mathbf{B}_{ij} - \mathbf{A}_{ij}\right) \mathbf{c}_j. \tag{10.12}$$

Für jedes endliche Intervall $[a, b]$ lässt sich also nur durch Rechnen mit Vektoren und Auswerten von Polynomen eine orthogonale Polynombasis p_i entwickeln. Dabei hat das Polynom p_i den Grad i (weil die Monombasis x^i Grad i hat) und ist zu allen Polynomen mit Grad kleiner i orthogonal (da es zu den Basiselementen einer Polynombasis vom Grad $i-1$ orthogonal ist).

Nullstellen Für die Integration wollten wir als Stützstellen die Nullstellen von p_i verwenden. Das geht aber nur dann, wenn alle Nullstellen von p_i reell sind und im Intervall $[a, b]$ liegen. Dies ist in der Tat der Fall!

Um das einzusehen, betrachten wir die m *verschiedene* Stellen $v_j, 0 \leq j < m$ im Intervall $[a, b]$, in denen p_i das Vorzeichen wechselt. Offenbar ist jede Stelle v_j eine Nullstelle von p_i. Umgekehrt muss p_j aber nicht notwendigerweise an jeder Nullstelle das Vorzeichen wechseln; außerdem könnte p_i mehrfache Nullstellen haben, die wir nur einmal zählen, sowie weitere reelle Nullstellen außerhalb von $[a, b]$.

Wir konstruieren das Polynom

$$p^*(x) = (x - v_0)(x - v_1) \cdots (x - v_{m-1}). \tag{10.13}$$

Dieses Polynom hat offenbar Grad m und m verschiedene reelle Nullstellen. Jetzt bilden wir das Skalarprodukt mit p_i, also

$$\langle p^*, p_i \rangle = \int_a^b p^*(x) p_i(x)\, dx. \tag{10.14}$$

Sowohl p_i wie auch p^* wechseln das Vorzeichen genau in den Stellen v_j. Also ist in allen Intervallen $[v_j, v_{j+1}]$ das Vorzeichen von $p^*(x)p_i(x)$ entweder immer positiv oder immer negativ und es gilt entweder $\langle p^*, p_i \rangle > 0$. Da p_i zu allen Polynomen mit Grad kleiner i orthogonal ist, muss p^* Grad i haben. Es gibt also $m = i$ Stellen v_i. Da p_i nach dem Fundamentalsatz der Algebra aber nicht mehr als i Nullstellen haben kann, sind die v_j genau die Nullstellen von p_i.

10.3.2. Gauß-Integration

Für ein gegebenes Intervall $[a, b]$ können wir ein Polynom p_{n+1} konstruieren, das orthogonal bzgl. des Skalarproduktes auf dem Intervall zu allen Polynomen mit kleinerem Grad ist. Ein Polynom $f(x)$ mit Grad nicht höher als $2n + 1$ schreiben wir als $f(x) = p_{n+1}q(x) + r(x)$, wobei q und r Polynome mit Grad nicht größer als n sind. Dann ist

$$\int_a^b f(x)\, dx = \int_a^b p_{n+1}q(x) + r(x)\, dx = \int_a^b r(x)\, dx. \tag{10.15}$$

Das Polynom $r(x)$ lässt sich in jeder Polynombasis für den Grad n darstellen, also auch mit den Lagrange-Polynomen für $n + 1$ Stützstellen x_i:

$$r(x) = \sum_i r(x_i) l_i(x), \quad l_i(x_j) = \delta_{ij}. \tag{10.16}$$

Damit erhalten wir

$$\int_a^b f(x)\, dx = \int_a^b r(x)\, dx = \int_a^b \sum_i r(x_i) l_i(x) = \sum_i r(x_i) \int_a^b l_i(x). \tag{10.17}$$

Wir wählen die Stützstellen als Nullstellen von p_{n+1} – wie gerade gezeigt sind alle Nullstellen reell und liegen im Intervall $[a, b]$. Damit ergibt sich weiter

$$\sum_i r(x_i) \int_a^b l_i(x) = \sum_i \underbrace{(p_{n+1}(x_i)q(x_i) + r(x_i))}_{=0} \int_a^b l_i(x) = \sum_i f(x_i) \int_a^b l_i(x). \tag{10.18}$$

Damit haben wir Gewichte

$$w_i = \int_a^b l_i(x) \tag{10.19}$$

gefunden, für die sich jedes Polynom $f(x)$ bis zum Grad $2n+1$ exakt integrieren lässt durch die gewichtete Summe

$$\int_a^b f(x)\,dx = \sum_i w_i f(x_i). \tag{10.20}$$

Voraussetzung ist 'lediglich', dass die x_i Nullstellen eines Polynoms sind, dass orthogonal auf allen Polynomen mit kleinerem Grad im Intervall $[a,b]$ ist. Da die Lagrange-Funktionen l_i ebenfalls Polynome sind, ist die Bestimmung der Gewichte w_i nicht schwierig.

Offenbar muss man für die Gauß-Integration einige Vorberechnungen durchführen. Das lohnt sich immer dann, wenn viele verschiedene Funktionen in einem festen Intervall integriert werden müssen. Die Ergebnisse der Gauß-Integration sind immer dann gut, wenn sich f gut durch ein Polynom des Grads $2n+1$ annähren lässt – mit anderen Worten, wenn die höheren Ableitungen von f klein sind. Die wahre Stärke der Gauß-Intergation liegt in der im folgenden diskutierten Verallgemeinerung, die diese Einschränkung weitgehend aufhebt.

10.3.3. Gewichtete Gauß-Integration

Nicht alle Funktionen lassen sich gut durch Polynome approximieren – das wissen wir bereits aus dem Kapitel zur Interpolation. Aber Gauß-Integration lässt sich erheblich verallgemeinern: wir nehmen an, die zu integrierende Funktion $f(x)$ ist ein Polynom (vom Grad höchstens $2n+1$), jetzt aber gewichtet (multipliziert) mit einer nicht-negativen, aber ansonsten praktisch beliebigen Funktion w. Das Polynom schreiben wir wie zuvor als $p_{n+1}q(x) + r(x)$, dann ist

$$f(x) = w(x)(p_{n+1}q(x) + r(x)) \tag{10.21}$$

und wir wollen wieder das Integral $\int_a^b f(x)\,dx$ als gewichtete Summe von Funktionswerten $f(x_i)$ darstellen. Offenbar ist die Klasse der Funktionen $f(x)$ jetzt viel allgemeiner, aber die Ideen der Gauß-Integration führen wiederum darauf, dass $n+1$ Stützstellen für eine *exakte* Auswertung des Integrals genügen.

In der Praxis wird die Funktion f keine einfache Zerlegung in ein Polynom und eine nicht-negative Gewichtsfunktion haben. Aber oft können wir eine Gewichtsfunktion wählen, so dass der 'Rest' gut durch ein Polynom zu approximieren sein wird. In einer solchen Situation kennen wir zwar w, weil wir es selbst gewählt haben, aber nicht q, r. Sprich, für eine Wahl von Stützstellen x_i können wir $f(x_i)$ und $w(x_i)$ bestimmen, nicht aber $q(x_i)$ oder $r(x_i)$. Wie wir sehen werden ist dies aber auch nicht nötig, wenn wir die Stützstellen x_i wider als Nullstellen des Polynoms p_{n+1} wählen.

Wir betrachten nun das gesuchte Integral und setzen es gleich der gewichteten Summe:

$$\begin{aligned} \int_a^b f(x)\,dx = \int_a^b w(x)p_{n+1}(x)q(x) + \int_a^b w(x)r(x)\,dx = \\ \sum_i w_i f(x_i) = \underbrace{\sum_i w_i w(x_i)p_{n+1}(x_i)q(x_i)}_{=0} + \sum_i w_i w(x_i)r(x_i). \end{aligned} \tag{10.22}$$

Die Bedingung an das Polynom p_{n+1} ist nun, dass der Ausdruck $\int_a^b w(x)p_{n+1}(x)q(x)$ verschwindet. Diese Bedingung ist eine Verallgemeinerung der orthogonalen Polynome auf w-orthogonale Polynome. Dazu definieren wir das Skalarprodukt

$$\langle f, g\rangle_w = \int_a^b w(x)f(x)g(x)\,dx, \qquad w : [a,b] \mapsto \mathbb{R}^{+,0}. \tag{10.23}$$

Diese Konstruktion kann man sich in Vektoren so vorstellen, dass man statt des Skalarproduktes $\mathbf{a}^\mathsf{T}\mathbf{b}$ das Skalarprodukt $\mathbf{a}^\mathsf{T}\mathbf{D}\mathbf{b}$ für eine Diagonalmatrix $\mathbf{D}$ mit nicht-negativen Einträgen betrachtet. Wie zuvor, kann man w-orthogonale Polynome p_i einfach durch das Gram-Schmidt-Verfahren konstruieren. Allerdings müssen wir jetzt im Allgemeinen numerisch integrieren, da w eine beliebige Funktion sein kann und das Produkt mit einem Polynom im Allgemeinen keine analytische Stammfunktion haben wird. Es gilt allerdings nach wie vor, dass alle Nullstellen von p_{n+1} reell sind und in das Intervall $[a,b]$ fallen. Eine Voraussetzung dafür ist unsere Wahl von w als nicht-negative Funktion auf dem Intervall $[a,b]$. Dies führt dazu, dass die Argumente aus dem vorangegangenen Abschnitt nach wie vor gültig sind. Und wir haben bereits eine Polynom-Basis dieser Art kennengelernt: Tschebychev-Polynome sind orthogonal in $[-1,1]$ bzgl. der Gewichtsfunktion $(1-x^2)^{-1/2}$.

Für die entsprechend konstruierte Polynomfunktion p_{n+1} haben wir also

$$\int_a^b f(x)\,dx = \int_a^b w(x)r(x)\,dx = \sum_i w_i w(x_i) r(x_i) = \sum_i w_i f(x_i). \tag{10.24}$$

Die Gewichte w_i ergeben sich wieder durch Repräsentation von $r(x)$ in Lagrange-Basis:

$$\int_a^b w(x)r(x)\,dx = \int_a^b w(x)\left(\sum_{i=0}^n r(x_i)l_i(x)\right) = \sum_{i=0}^n r(x_i)\int_a^b w(x)l_i(x)\,dx = \sum_i w_i w(x_i) r(x_i) \tag{10.25}$$

und wir sehen, dass die Gewichte w_i bestimmt werden können, ohne dass wir die (unbekannten) Funktionswerte $r(x_i)$ benötigen:

$$w_i = \frac{1}{w(x_i)}\int_a^b w(x)l_i(x). \tag{10.26}$$

Zur Bestimmung der Gewichte muss man allerdings (typischerweise numerisch) die Integrale der gewichteten Lagrange-Polynome integrieren.

Zusammenfassung: Gauß-Integration ist immer dann interessant, wenn man viele, unterschiedliche aber verwandte Funktionen f über demselben Intervall $[a,b]$ integrieren muss. In diesem Fall wählt man eine geeignete Gewichtsfunktion w und die gewünschte Zahl von Stützstellen $n+1$.

1. Man erzeugt das w-orthogonale Polynom p_{n+1} mit dem Gram-Schmidt-Verfahren aus einer beliebigen Polynombasis, z.B. der Monom-Basis. Die dabei auftretenden Skalarprodukte $\langle p_i, p_j\rangle_w$ können je nach Wahl von w eventuell nur numerisch bestimmt werden. Da dieser Vorgang nur einmal durchgeführt werden muss, kann man dies mit dem Newton-Cotes Verfahren für feine Unterteilung des Intervalls $[a,b]$ machen.

2. Nun bestimmt man die Nullstellen von p_{n+1}, z.B. mit Newton-Verfahren oder mit Bisektion. Eventuell teilt man das Polynom nach der Bestimmung einer Nullstelle x_i durch x_i, um zu vermeiden, dieselbe Nullstelle mehrfach zu finden.

3. Aus den Nullstellen ergeben sich die Lagrange-Polynome l_i und man bestimmt nun die Gewichte w_i wiederum mit dem Newton-Cotes Verfahren auf einer Unterteilung des Intervalls $[a, b]$.

Neben der genaueren Bestimmung von Integralen mit wenigen Stützstellen für viele Funktionen, lässt sich Gauß-Integration auch zur Approximation von uneigentlichen Integralen, also z.B. $\int_{-\infty}^{\infty} f(x)\,dx$ verwenden. Dazu wählt man eine Gewichtsfunktion, deren Integral im unbeschränkten Intervall endlich ist, z.B. $w(x) = e^{-x^2}$.

A. Komplexe Zahlen

A.1. Komplexe Zahlen

Die Menge der geordneten Paare (a, b) von reellen Zahlen zusammen mit den beiden Verknüpfungen

$$(a, b) + (c, d) := (a + c, b + d) \tag{A.1a}$$
$$(a, b) \cdot (c, d) := (ac - bd, ad + bc) \tag{A.1b}$$

definiert den Körper $\mathbb{C}$ der *komplexen Zahlen*. Ist $z = (a, b)$ eine komplexen Zahl, so bezeichnet man mit $\Re(z) := a$ den *Realteil* und mit $\Im(z) := b$ den *Imaginärteil* von z. Identifiziert man jede reelle Zahl $x \in \mathbb{R}$ mit der komplexen Zahl $(x, 0)$, so kann man $\mathbb{R}$ als Teilkörper der komplexen Zahlen auffassen. Bezeichnet man nun die komplexe Zahl $(0, 1)$ mit i, so gilt $i^2 = -1$ und jede komplexe Zahl $z = (a, b)$ kann eindeutig in der Form $z = a + bi$ geschrieben werden.

Rechnen mit komplexen Zahlen Beim Rechnen mit komplexen Zahlen kann man (meistens) wie im Reellen verfahren und i^2 durch -1 ersetzen, wenn es auftritt:

Addition:

$$(a + bi) + (c + di) = a + bi + c + di = (a + c) + (b + d)i \tag{A.2a}$$

Subtraktion

$$(a + bi) - (c + di) = a + bi - c + di = (a - c) + (b - d)i \tag{A.2b}$$

Multiplikation

$$(a + bi)(c + di) = ac + adi + bci + bdi^2 = (ac - bd) + (ad + bc)i \tag{A.2c}$$

Division

$$\frac{a + bi}{c + di} = \frac{a + bi}{c + di} \cdot \frac{c - di}{c - di} = \frac{ac - adi + bci - bdi^2}{c^2 - (di)^2} = \frac{ac + bd}{c^2 + d^2} + \frac{bc - ad}{c^2 + d^2} i \tag{A.2d}$$

Absolutbetrag Der *Absolutbetrag* einer komplexen Zahl $z = a + bi$ ist definiert als:

$$|z| = |a + bi| := \sqrt{a^2 + b^2} \tag{A.3}$$

Für alle $z \in \mathbb{C}$ ist also $|z| \geq 0$ und es gilt $|z| = 0$ genau dann wenn $z = 0$ ist. Außerdem gelten die folgenden Rechenregeln:

1. $|z \cdot w| = |z| \cdot |w|$
2. $|\frac{z}{w}| = \frac{|z|}{|w|}$ (für $w \neq 0$)
3. $|z + w| \leq |z| + |w|$

Konjugation Die zu einer komplexen Zahl $z = a + bi$ *konjugierte* komplexe Zahl ist definiert als:

$$\overline{z} = \overline{a + bi} := a - bi \tag{A.4}$$

Es gelten die folgenden Rechenregeln:

1. $\overline{z + w} = \overline{z} + \overline{w}$
2. $\overline{z \cdot w} = \overline{z} \cdot \overline{w}$
3. $\overline{\overline{z}} = z$
4. $\overline{z} \cdot z = |z|^2$

Mit Hilfe der konjugierten einer komplexen Zahl $z \neq 0$, lässt sich deren Inverses besonders einfach darstellen:

$$z^{-1} = \frac{1}{z} = \frac{\overline{z}}{\overline{z}z} = \frac{\overline{z}}{|z|^2} \tag{A.5}$$

Außerdem gilt:

$$\Re(z) = \frac{z + \overline{z}}{2} \quad \text{und} \quad \Im(z) = \frac{z - \overline{z}}{2}\,, \tag{A.6}$$

woraus insbesondere folgt, dass $z = \overline{z}$ genau dann gilt, wenn die komplexe Zahl z reell ist, d.h., wenn $\Im(z) = 0$ ist.

Für eine komplexwertige Matrix mit m Zeilen und n Spalten

$$A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix} \in \mathbb{C}^{m \times n} \tag{A.7}$$

wird mit

$$A^* = \overline{A}^T = \begin{pmatrix} \overline{a_{11}} & \cdots & \overline{a_{1m}} \\ \vdots & \ddots & \vdots \\ \overline{a_{n1}} & \cdots & \overline{a_{mn}} \end{pmatrix} \in \mathbb{C}^{n \times m} \tag{A.8}$$

die *konjugiert-transponierte Matrix* von A bezeichnet. Für eine komplexe Zahl z wird daher auch oft die Schreibweise z^* für die Konjugierte verwendet.

A.2. Exponentialfunktion und trigonometrische Funktionen

Die Exponentialfunktion und die trigonometrischen Funktionen Sinus und Kosinus werden für komplexe Zahlen wie im Reellen mittels Reihendarstellung definiert:

$$\exp(z) := \sum_{k=0}^{\infty} \frac{z^k}{k!} \qquad \sin(z) := \sum_{k=0}^{\infty} (-1)^k \frac{z^{2k+1}}{(2k+1)!} \qquad \cos(z) := \sum_{k=0}^{\infty} (-1)^k \frac{z^{2k}}{(2k)!} \tag{A.9}$$

Im Reellen konvergieren die Reihen absolut. Für beliebige komplexen Zahlen konvergieren somit auch die Reihen der Real- und Imaginärteile absolut, woraus die absolute Konvergenz

im komplexen folgt. Es gilt $\exp(0) = 1$, $\exp(1) = e$, wobei $e = \sum_{k=0}^{\infty} \frac{1}{k!}$ die Eulerschen Zahl bezeichnet, und

$$\exp(z) \cdot \exp(w) = \left(\sum_{k=0}^{\infty} \frac{z^k}{k!}\right) \cdot \left(\sum_{k=0}^{\infty} \frac{w^k}{k!}\right) =^{1} \sum_{k=0}^{\infty} \sum_{\nu=0}^{k} \frac{z^\nu}{\nu!} \frac{w^{k-\nu}}{(k-\nu)!} \tag{A.10a}$$

$$= \sum_{k=0}^{\infty} \frac{1}{k!} \sum_{\nu=0}^{k} \binom{k}{\nu} z^\nu w^{k-\nu} = \sum_{k=0}^{\infty} \frac{(z+w)^k}{k!} = \exp(z+w) \tag{A.10b}$$

für alle $z, w \in \mathbb{C}$. Für $\exp(z)$ schreibt man daher auch oft e^z.

Zwischen der komplexen Exponentialfunktion und den komplexen Sinus- und Kosinusfunktionen besteht ein Zusammenhang. Es gilt die *Eulersche Formel*:

$$\exp(iz) = \sum_{k=0}^{\infty} \frac{(iz)^k}{k!} = \sum_{k=0}^{\infty} \frac{i^k z^k}{k!} \tag{A.11a}$$

$$= \sum_{k=0}^{\infty} \frac{i^{2k} z^{2k}}{(2k)!} + \sum_{k=0}^{\infty} \frac{i^{2k+1} z^{2k+1}}{(2k+1)!} = \sum_{k=0}^{\infty} (-1)^k \frac{z^{2k}}{(2k)!} + i \sum_{k=0}^{\infty} (-1)^k \frac{z^{2k+1}}{(2k+1)!} \tag{A.11b}$$

$$= \cos(z) + i \sin(z) \tag{A.11c}$$

Aus der Reihendarstellung (A.9) folgt, dass der Sinus eine ungerade Funktion (d.h., $\sin(z) = -\sin(z)$) und der Kosinus eine gerade Funktion (d.h., $\cos(z) = \cos(-z)$) ist. Entsprechend folgt:

$$\exp(-iz) = \cos(z) - i \sin(z) \tag{A.12}$$

Wir erhalten somit durch Addition bzw. Subtraktion von (A.11) und (A.12) die folgenden Darstellungen der Sinus- und Kosinusfunktionen:

$$\sin(z) = \frac{\exp(iz) - \exp(-iz)}{2i} \tag{A.13a}$$

$$\cos(z) = \frac{\exp(iz) + \exp(-iz)}{2} \tag{A.13b}$$

Aus der Eulersche Formel ergibt außerdem eine weitere wichtige Eigenschaft der Exponentialfunktion. Die komplexe Exponentialfunktion ist periodisch mit der komplexen Periode $2\pi i$, denn für $k \in \mathbb{Z}$ gilt:

$$\exp(z + 2\pi i k) = \exp(z) \cdot \exp(2\pi i k) \tag{A.14a}$$

$$= \exp(z) \cdot \big(\cos(2\pi k) + i \sin(2\pi k)\big) = \exp(z) \cdot (1 + i \cdot 0) = \exp(z) \tag{A.14b}$$

A.3. Komplexe Zahlenebene und Polarkooridinaten-Darstellung

Das es sich bei den komplexen Zahlen um Paare von reellen Zahlen handelt, können wir diese als Punkte (bzw. besser Vektoren) in einem zwei-dimensionalen Koordinatensystem graphisch darstellen. Dies erlaubt eine geometrische Interpretation der komplexen Zahlen.

In vielen Fällen ist es dabei einfacher die komplexen Zahlen durch Länge und Winkel, anstatt durch Real- und Imaginärteil, zu beschreiben. Sei $z \neq 0$ eine von Null verschiedene komplexe Zahl und

$$\alpha + \beta i = \frac{z}{|z|}\,. \tag{A.15}$$

[1] Satz vom Cauchy-Produkt: Königsberger, Konrad. 2001. Analysis 1. 5th ed. Springer. p. 72.

Dann gibt es einen eindeutig bestimmen Winkel $\varphi \in [0, 2\pi)$, so dass $\alpha = \cos(\varphi)$ und $\beta = \sin(\varphi)$ ist. Der Winkel φ wird als das *Argument* der komplexen Zahl z bezeichnet. Wir erhalten so die *Polarkoordinaten*-Darstellung von z:

$$z = |z| \cdot (\alpha + \beta i) \tag{A.16a}$$

$$= |z| \cdot \big(\cos(\varphi) + i \sin(\varphi)\big) = |z| \cdot \exp(i\varphi) = |z|\, e^{i\varphi} \tag{A.16b}$$

Für den Winkel φ gilt

$$\tan \varphi = \frac{\Im(z)}{\Re(z)} = \frac{\beta}{\alpha} \tag{A.17}$$

und kann daher mit der Umkehrfunktion des Tangents berechnet werden. Dabei muss allerdings darauf geachtet werden die richtige Lösung auszuwählen. Viele Programmiersprachen definieren hierfür die sogenannte `atan2` Funktion, die abhängig vom Quadranten die richtige Lösung auswählt:

$$\operatorname{atan2}(y, x) = \begin{cases} \arctan\left(\frac{y}{x}\right) & x > 0 \\ \arctan\left(\frac{y}{x}\right) + \pi & y \geq 0,\ x < 0 \\ \arctan\left(\frac{y}{x}\right) - \pi & y < 0,\ x < 0 \\ +\frac{\pi}{2} & x = 0,\ y > 0 \\ -\frac{\pi}{2} & x = 0,\ y < 0 \\ \textit{undefiniert} & x = 0,\ y = 0 \end{cases} \tag{A.18}$$

Addition und Subtraktion: Die Addition und Subtraktion von komplexen Zahlen entspricht geometrisch der normalen Addition bzw. Subtraktion von Vektoren (Abbildung A.1(b)).

Multiplikation: Sei $z = |z|\, e^{i\varphi}$ und $w = |w|\, e^{i\psi}$. Dann erhalten wir:

$$zw = |z| \cdot |w| \cdot e^{i\varphi} \cdot e^{i\psi} = |z| \cdot |w| \cdot e^{i(\varphi+\psi)} \tag{A.19}$$

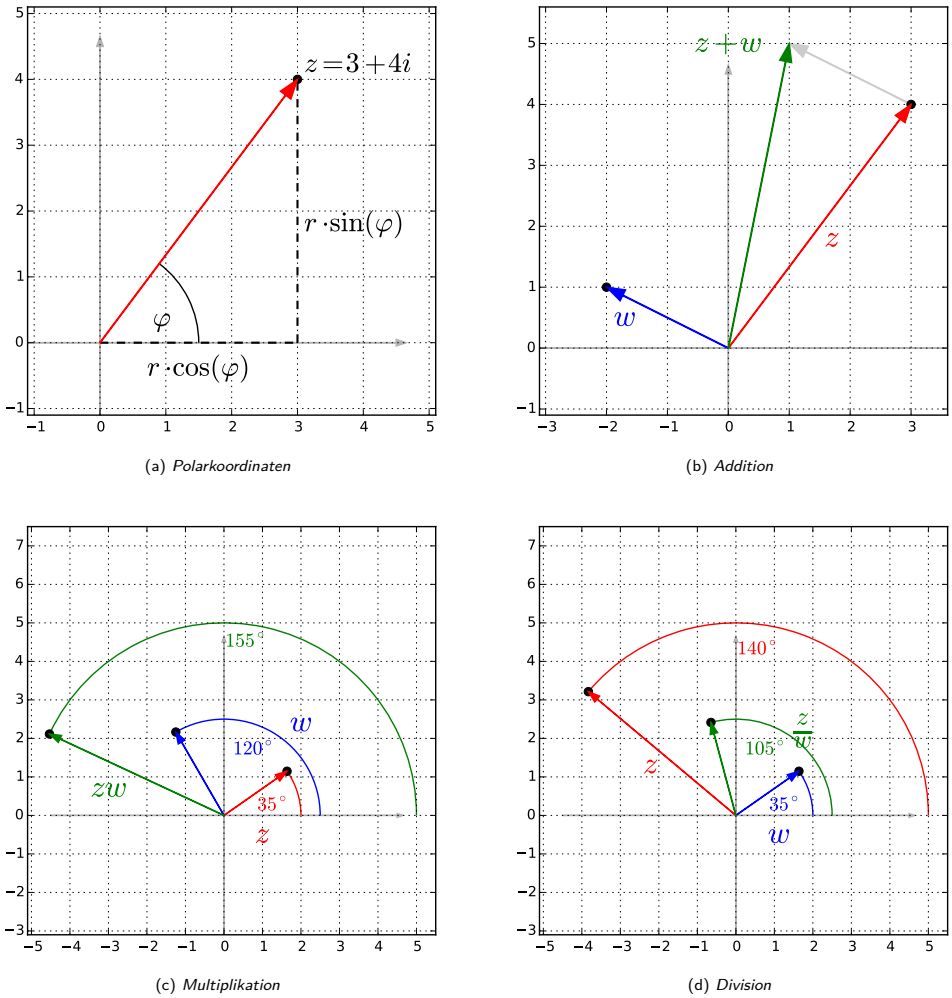
Bei der Multiplikation werden also die Längen multipliziert und die Winkel addiert (Abbildung A.1(c)).

Division: Sei $z = |z|\, e^{i\varphi}$ und $w = |w|\, e^{i\psi}$. Dann erhalten wir:

$$\frac{z}{w} = \frac{|z|}{|w|} \cdot \frac{e^{i\varphi}}{e^{i\psi}} = \frac{|z|}{|w|}\, e^{i(\varphi-\psi)} \tag{A.20}$$

Bei der Division werden also die Längen durcheinander dividiert und die Winkel von einander subtrahiert (Abbildung A.1(d)).

Konjugation: Die Konjugation entspricht einer Spiegelung an der reellen Achse.



(a) *Polarkoordinaten* (b) *Addition*

(c) *Multiplikation* (d) *Division*

Abbildung A.1.: *Geometrische Veranschaulichung der komplexen Zahlen.*

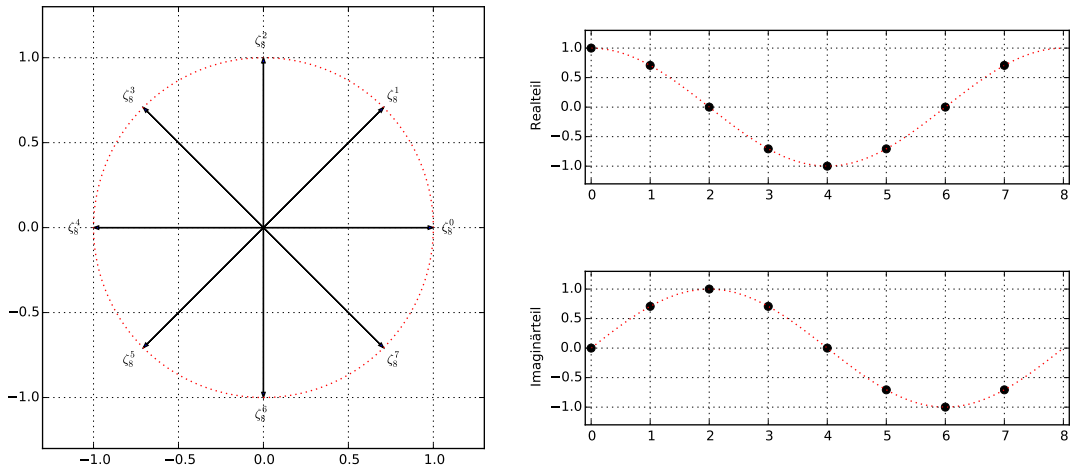


Abbildung A.2.: *Visualisierung der* 8*-ten Einheitswurzel* $\zeta_8^0, \ldots, \zeta_8^8$.

A.4. Einheitswurzeln

Sei $n \in \mathbb{N}$ und bezeichne

$$\zeta_n = e^{2\pi i/n} = \cos\left(\frac{2\pi}{n}\right) + i\sin\left(\frac{2\pi}{n}\right). \tag{A.21}$$

Die Gleichung $z^n = 1$ hat in den komplexen Zahlen genau n Lösungen, welche durch die *n-ten Einheitswurzeln*,

$$\zeta_n^0, \zeta_n^1, \zeta_n^2, \ldots, \zeta_n^n \tag{A.22}$$

gegeben sind.

Offensichtlicherweise sind die Zahlen ζ_n^i paarweise verschieden. Außerdem gilt:

$$\left(\zeta_n^i\right)^n = \left(e^{2\pi i/n}\right)^n = e^{2\pi i} = \cos(2\pi) + i\sin(2\pi) = 1\,.$$

Durch die ζ_n^i sind also n Lösungen der Gleichung $z^n = 1$ gegeben. Sei nun umgekehrt w irgendeine Lösung der Gleichung $z^n = 1$. Dann ist $|w|^n = 1$ und damit $|w| = 1$. Das bedeutet, dass es ein $\varphi \in [0, 2\pi)$ gibt mit $w = \cos\varphi + i\sin\varphi = e^{i\varphi}$. Nach Voraussetzung ist $w^n = e^{in\varphi} = \cos n\varphi + i\sin n\varphi = 1$. Folglich ist also $\cos n\varphi = 1$ und $\sin n\varphi = 0$ und daher gibt es ein $k \in \mathbb{Z}$ mit $n\varphi = k \cdot 2\pi$. Weil $0 \leq \varphi < 2\pi$ ist, folgt somit, dass $0 \leq n\varphi < 2\pi n$ ist. Für k kommen also nur die Werte $0, \ldots, n-1$ in Frage.

A.5. Kanonisches Skalarprodukt im $\mathbb{C}^n$

Der $\mathbb{C}^n$ ist mit den üblichen Definitionen für Addition und Skalarmultiplikation ein $\mathbb{C}$-Vektorraum. Das *kanonische Skalarprodukt* im $\mathbb{C}^n$ ist definiert als die Abbildung $\mathbb{C}^n \times \mathbb{C}^n \to \mathbb{C}$, welche durch

$$\langle z, w\rangle = z_1\bar{w}_1 + \ldots + z_n\bar{w}_n = \sum_{k=1}^{n} z_k\bar{w}_k \tag{A.23}$$

gegeben ist. Das kanonische Skalarprodukt im komplexen unterscheidet sich also durch die komplexe Konjugation im zweiten Argument vom Skalarprodukt im reellen und hat für $z, z', w, w' \in \mathbb{C}^n$ und $\lambda \in \mathbb{C}$ folgende Eigenschaften:

1. $\langle z + z', w\rangle = \langle z, w\rangle + \langle z', w\rangle$
2. $\langle \lambda z, w\rangle = \lambda\langle z, w\rangle$
3. $\langle z, w\rangle = \overline{\langle w, z\rangle}$
4. $\langle z, z\rangle \in \mathbb{R}_{\geq 0}$
5. $\langle z, z\rangle = 0 \Leftrightarrow z = 0$

Aus c) mit a) und b) folgen außerdem:

6. $\langle z, w + w'\rangle = \langle z, w\rangle + \langle z, w'\rangle$
7. $\langle z, \lambda w\rangle = \bar{\lambda}\langle z, w\rangle$

Wegen d) induziert das kanonische Skalarprodukt eine Norm $\mathbb{C}^n \to \mathbb{R}_{\geq 0}$ auf dem $\mathbb{C}^n$:

$$z \mapsto \|z\| = \sqrt{\langle z, z \rangle}\,.$$

Diese ist identisch mit der Norm, welche durch das euklidische Skalarprodukt auf dem $\mathbb{R}^{2n}$ induziert wird.

B. Basiswissen Lineare Algebra

Im wissenschaftlichen Rechnen modellieren wir die Realiät als eine Menge von Zahlen. Es ist oft besonders übersichtlich, die Zahlen als Vektor zu schreiben. Lineare Operationen sind dann Produkte mit Matrizen. Allgemein dienen uns Methoden und Konzepte der linearen Algebra als prägnante Notation, die es vermeidet, alle Zahlen einzeln ausschreiben zu müssen. Hier wiederholen wir kurz Notation, Rechenregeln und einige Fakten aus der linearen Algebra. Wir beschränken uns auf reelle Zahlen und verzichten bewusst auf die Konsequenzen der Repräsentation von reellen Zahlen auf dem Rechner. Mit diesen Konsequenzen beschäftigen wir uns dann eingehend im Rahmen der Veranstaltung.

B.1. Notation

B.1.1. Matrix

Eine Matrix mit n Zeilen und m Spalten von nm reellen Zahlen schreiben wir mit (oft lateinischen, manchmal griechischen) Großbuchstaben. Die Elemente werden mit den entsprechenden Kleinbuchstaben und zwei Indizes geschrieben. Der erste Index gibt die Zeile und der zweite Index die Spalte an. Die Indizierung beginnt bei 0. Der größte Zeilenindex ist $n-1$, der größte Spaltenindex $m-1$. Wenn aus dem Kontext klar ist, was Zeilen- und was Spaltenindex ist, verzichten wir auf ein Komma zwischen den Indizes.

$$\mathbf{A} \in \mathbb{R}^{n\times m}, \qquad \mathbf{A} = \left.\underbrace{\begin{pmatrix} a_{00} & a_{01} & \dots & a_{0,m-1} \\ a_{10} & a_{11} & \dots & a_{1,m-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n-1,0} & a_{n-1,1} & \dots & a_{n-1,m-1} \end{pmatrix}}_{m \text{ Spalten}}\right\} n \text{ Zeilen} \tag{B.1}$$

Eine Matrix ergibt sich oft aus den Einträgen:

$$\mathbf{A} = \{a_{ij}\}, \qquad \text{Zeilenindex } i\text{, Spaltenindex } j. \tag{B.2}$$

Die Elemente a_{ii} mit gleichem Zeilen- und Spaltenindex heißen *Diagonalelemente* der Matrix. Alle Elemente dieser Art nennt man auch *Diagonale* der Matrix. Eine Matrix, deren Anzahl von Zeilen n und Anzahl von Spalten m identisch ist (also $n = m$) heißt *quadratisch*. Quadratische Matrizen, die nur auf der Diagonale von Null verschiedene Werte haben heißen *Diagonalmatrix*. Die *Einheitsmatrix* ist eine Diagonalmatrix, deren Einträge alle eins sind:

$$\mathbf{I}_n \in \mathbb{R}^{n\times n}, \qquad \mathbf{I}_n = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{pmatrix}. \tag{B.3}$$

B.1.2. Transposition

Die *Transposition* vertauscht Zeilen und Spalten einer Matrix. Die Transponierte von $\mathbf{A}$ schreiben wir $\mathbf{A}^\mathsf{T}$. Dabei vertauschen sich auch die Anzahlen der Zeilen und Spalten, sprich aus $\mathbf{A} \in \mathbb{R}^{n\times m}$ wird $\mathbf{A}^\mathsf{T} \in \mathbb{R}^{m\times n}$. Ausgeschrieben bedeutet das:

$$\mathbf{A} = \begin{pmatrix} a_{00} & a_{01} & \dots & a_{0,m-1} \\ a_{10} & a_{11} & \dots & a_{1,m-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n-1,0} & a_{n-1,1} & \dots & a_{n-1,m-1} \end{pmatrix} \quad \Rightarrow \quad \mathbf{A}^\mathsf{T} = \begin{pmatrix} a_{00} & a_{10} & \dots & a_{n-1,0} \\ a_{01} & a_{11} & \dots & a_{n-1,1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{0,m-1} & a_{1,m-1} & \dots & a_{m-1,n-1} \end{pmatrix}. \tag{B.4}$$

Beim Transponieren verändert sich die Diagonale einer Matrix nicht. Eine Matrix, die sich bei Transposition nicht verändert heißt *symmetrisch*. Symmetrische Matrizen sind offenbar quadratisch und es gilt

$$\mathbf{A} = \mathbf{A}^\mathsf{T} \in \mathbb{R}^{n\times n}, \qquad \mathbf{A} = \{a_{ij}\}, \quad a_{ij} = a_{ji}. \tag{B.5}$$

B.1.3. Vektor

Als Vektor verstehen wir eine Matrix mit nur einer Spalte. Vektoren sind also (zumindest im Kontext der Lehrveranstaltung Wissenschaftliches Rechnen) immer Spaltenvektoren. Wir schreiben Vektoren mit (meist lateinischen, selten griechischen) Kleinbuchstaben. Die Elemente des Vektors werden mit einem Index geschrieben. Die Indizierung beginnt bei 0.

$$\mathbf{a} \in \mathbb{R}^{n\times 1} = \mathbb{R}^n, \qquad \mathbf{a} = \left.\begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{n-1} \end{pmatrix}\right\} n \text{ Zeilen} \tag{B.6}$$

Wenn wir ausdrücken wollen, dass ein Vektor Zeilenform hat, transponieren wir einen Spaltenvektor:

$$\mathbf{a}^\mathsf{T} \in \mathbb{R}^{1\times n}, \qquad \mathbf{a}^\mathsf{T} = (\underbrace{a_0, a_1, \dots, a_{n-1}}_{n \text{ Spalten}}). \tag{B.7}$$

Matrizen lassen sich auch durch Vektoren beschreiben, und zwar sowohl durch Spaltenvektoren

$$\mathbf{A} \in \mathbb{R}^{n\times m}, \qquad \mathbf{A} = (\mathbf{a}_0, \mathbf{a}_1, \dots, \mathbf{a}_{m-1}), \qquad \mathbf{a}_i \in \mathbb{R}^n \tag{B.8}$$

wie auch durch Zeilenvektoren

$$\mathbf{A} \in \mathbb{R}^{n\times m}, \qquad \mathbf{A} = \begin{pmatrix} \mathbf{a}_0^\mathsf{T} \\ \mathbf{a}_1^\mathsf{T} \\ \vdots \\ \mathbf{a}_{n-1}^\mathsf{T} \end{pmatrix} \qquad \mathbf{a}_i \in \mathbb{R}^m. \tag{B.9}$$

Achtung: die Vektoren $\mathbf{a}_i$ in den vorangegangenen Beschreibungen sind nicht gleich. Im ersten Fall gibt es m Spaltenvektoren mit n Einträgen; im zweiten Fall n Zeilenvektoren mit jeweils m Einträgen.

B.2. Operationen

B.2.1. Addition

Matrizen (und damit auch Vektoren) können addiert werden, wenn sie die gleichen Dimensionen haben. Die Addition wird elementweise durchgeführt:

$$\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n \times m}, \qquad \mathbf{A} + \mathbf{B} = \{a_{ij} + b_{ij}\}. \tag{B.10}$$

Die Addition ist (zumindest für reelle Matrizen) kommutativ und assioziativ.

$$\mathbf{A} + \mathbf{B} = \mathbf{B} + \mathbf{A}, \qquad \mathbf{A} + (\mathbf{B} + \mathbf{C}) = (\mathbf{A} + \mathbf{B}) + \mathbf{C}. \tag{B.11}$$

Die Addition hat eine eindeutige Inverse, die *Subtraktion*, die sich direkt aus der Definition der Addition ergibt („Man kann immer Minus rechnen“).

B.2.2. Multiplikation

Zwei Matrizen $\mathbf{A}, \mathbf{B}$ können multipliziert werden, wenn die Anzahl der Spalten von $\mathbf{A}$ der Anzahl der Zeilen von $\mathbf{B}$ entspricht; insbesondere können quadratische Matrizen mit gleicher Dimension multipliziert werden. Die Matrixmultiplikation folgt folgender Regel:

$$\mathbf{A} \in \mathbb{R}^{n \times m}, \mathbf{B} \in \mathbb{R}^{m \times o}, \qquad \mathbf{AB} = \left\{ \sum_{k=0}^{m-1} a_{ik} b_{kj} \right\} \in \mathbb{R}^{n \times o}. \tag{B.12}$$

Das Produkt $\mathbf{AB}$ hat also die gleiche Zahl an Zeilen wie die Matrix $\mathbf{A}$ und genauso viele Spalten wie $\mathbf{B}$. Die Beschreibung des Produktes in der oben angegebenen Form ist wenig anschaulich. Betrachten wir zunächst wie dieses Produkt auf Vektoren funktioniert.

B.2.3. Inneres Produkt / Skalarprodukt, Matrix-Vektor-Produkt

Wir können einen Zeilenvektor $\mathbf{a}^\mathsf{T}$ mit einem Spaltenvektor $\mathbf{b}$ multiplizieren, wenn die beiden Vektoren gleich Länge haben, also $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$. Nach der oben angegeben Regel erhalten wir eine 1×1-Matrix, also einen Skalar, wovon man sich leicht überzeugen kann:

$$\mathbf{a}^\mathsf{T}\mathbf{b} = a_0 b_0 + a_1 b_1 + \ldots + a_{n-1} b_{n-1} = \sum_{i=0}^{n-1} a_i b_i. \tag{B.13}$$

Diese Form des Produktes zwischen zwei Vektoren heißt *inneres Produkt* oder *Skalarprodukt*. Es ist wichtig, Produkte der Form $\mathbf{a}^\mathsf{T}\mathbf{b}$ als Skalar „zu sehen“. Mit dem Skalarprodukt können wir das allgemeine Produkt zwischen Matritzen $\mathbf{A} \in \mathbb{R}^{n \times m}$ und $\mathbf{B} \in \mathbb{R}^{m \times o}$ sehr übersichtlich schreiben:

$$\mathbf{AB} = \begin{pmatrix} \mathbf{a}_0^\mathsf{T} \\ \mathbf{a}_1^\mathsf{T} \\ \vdots \\ \mathbf{a}_{n-1}^\mathsf{T} \end{pmatrix} (\mathbf{b}_0, \mathbf{b}_1, \ldots, \mathbf{b}_{o-1}) = \begin{pmatrix} \mathbf{a}_0^\mathsf{T}\mathbf{b}_0 & \mathbf{a}_0^\mathsf{T}\mathbf{b}_1 & \ldots & \mathbf{a}_0^\mathsf{T}\mathbf{b}_{o-1} \\ \mathbf{a}_1^\mathsf{T}\mathbf{b}_0 & \mathbf{a}_1^\mathsf{T}\mathbf{b}_1 & \ldots & \mathbf{a}_1^\mathsf{T}\mathbf{b}_{o-1} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{a}_{n-1}^\mathsf{T}\mathbf{b}_0 & \mathbf{a}_{n-1}^\mathsf{T}\mathbf{b}_1 & \ldots & \mathbf{a}_{n-1}^\mathsf{T}\mathbf{b}_{o-1} \end{pmatrix}. \tag{B.14}$$

Der Eintrag in der i-ten Zeile und j-ten Spalte des Produktes $\mathbf{AB}$ ist also das Skalarprodukt der i-ten Zeile $\mathbf{a}_i^\mathsf{T}$ von $\mathbf{A}$ mit der j-ten Spalte $\mathbf{b}_j$ von $\mathbf{B}$. Als wichtige Spezialfälle ergeben sich die Produkte zwischen Matrizen und Vektoren. Multipliziert man eine Matrix $\mathbf{A} \in \mathbb{R}^{n\times m}$ von rechts mit einem Spaltenvektor $\mathbf{b}$ der Länge m so erhält man wiederum einen Spaltenvektor, allerdings mit der Länge n. Für diesen Fall ist es günstig, die Matrix in Zeilenform zu schreiben:

$$\mathbf{b} \in \mathbb{R}^m, \qquad \mathbf{Ab} = \begin{pmatrix} \mathbf{a}_0^\mathsf{T} \\ \mathbf{a}_1^\mathsf{T} \\ \vdots \\ \mathbf{a}_{n-1}^\mathsf{T} \end{pmatrix} \mathbf{b} = \begin{pmatrix} \mathbf{a}_0^\mathsf{T}\mathbf{b} \\ \mathbf{a}_1^\mathsf{T}\mathbf{b} \\ \vdots \\ \mathbf{a}_{n-1}^\mathsf{T}\mathbf{b} \end{pmatrix} \in \mathbb{R}^n. \tag{B.15}$$

Man kann die Matrix auch von links mit einem Zeilenvektor $\mathbf{b}^\mathsf{T}$ der Länge n multiplizieren. Dann erhält man einen Zeilenvektor der Länge m und betrachtet die Matrix in Spaltenform:

$$\mathbf{b} \in \mathbb{R}^n, \qquad \mathbf{b}^\mathsf{T}\mathbf{A} = \mathbf{b}^\mathsf{T}\left(\mathbf{a}_0, \mathbf{a}_1, \ldots, \mathbf{a}_{m-1}\right) = \left(\mathbf{b}^\mathsf{T}\mathbf{a}_0, \mathbf{b}^\mathsf{T}\mathbf{a}_1, \ldots, \mathbf{b}^\mathsf{T}\mathbf{a}_{m-1}\right) \in \mathbb{R}^{1\times m}. \tag{B.16}$$

Multipliziert man die Matrix von links mit einem Zeilenvektor $\mathbf{b}^\mathsf{T}$ und von rechts mit einem Spaltenvektor $\mathbf{c}$ (so dass die Dimensionen passen) so erhält man wieder einen Skalar:

$$\mathbf{b}^\mathsf{T}\mathbf{Ac} = \left(\mathbf{b}^\mathsf{T}\mathbf{a}_0, \mathbf{b}^\mathsf{T}\mathbf{a}_1, \ldots, \mathbf{b}^\mathsf{T}\mathbf{a}_{m-1}\right)\mathbf{c} = \mathbf{b}^\mathsf{T}\begin{pmatrix} \mathbf{a}_0^\mathsf{T}\mathbf{c} \\ \mathbf{a}_1^\mathsf{T}\mathbf{c} \\ \vdots \\ \mathbf{a}_{n-1}^\mathsf{T}\mathbf{c} \end{pmatrix} = \sum_{i=0}^{n-1}\sum_{j=0}^{m-1} b_i a_{ij} c_j. \tag{B.17}$$

Hier werden also jeweils das i-te Element aus dem Zeilenvektor $\mathbf{b}^\mathsf{T}$ mit dem j-ten Element aus dem Spaltenvektor $\mathbf{c}$ und dem Matrixelement a_{ij} multipliziert und dann alle Produkte dieser Art aufsummiert.

B.2.4. Äußeres Produkt, Spur

Man kann zwei Vektoren nach den Regeln der Matrixmultiplikation auch so multiplizieren, dass der Spaltenvektor links und der Zeilenvektor rechts steht. Hat der Spaltenvektor $\mathbf{a}$ die Länge n und der Zeilenvektor $\mathbf{b}^\mathsf{T}$ die Breite m, so ergibt sich eine Matrix mit n Zeilen und m Spalten, die alle möglichen Produkte zwischen den Koeffizienten der beiden Vektoren enthält:

$$\mathbf{ab}^\mathsf{T} = \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{n-1} \end{pmatrix}(b_0, b_1, \ldots, b_{m-1}) = \begin{pmatrix} a_0b_0 & a_0b_1 & \ldots & a_0b_{m-1} \\ a_1b_0 & a_1b_1 & \ldots & a_1b_{m-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n-1}b_0 & a_{n-1}b_1 & \ldots & a_{n-1}b_{m-1} \end{pmatrix} \in \mathbb{R}^{n\times m}. \tag{B.18}$$

Für zwei Vektoren gleicher Länge, hätte man auch die Möglichkeit das innere Produkt zu bilden (das einen Skalar liefern würde). Zur Unterscheidung nennt man das Produkt nach Gl. B.18 *äußeres Produkt*. Eine Beziehung zwischen innerem und äußerem Produkt liefert die *Spur* (engl. *trace*). Die Spur einer quadratischen Matrix ist die Summe ihrer Diagonalelemente. Für zwei Vektoren $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$ gleicher Länge sieht man leicht:

$$tr(\mathbf{ab}^\mathsf{T}) = tr\begin{pmatrix} a_0b_0 & a_0b_1 & \ldots & a_0b_{m-1} \\ a_1b_0 & a_1b_1 & \ldots & a_1b_{m-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n-1}b_0 & a_{n-1}b_1 & \ldots & a_{n-1}b_{n-1} \end{pmatrix} = a_0b_0 + a_1b_1 + \ldots + a_{n-1}b_{n-1} = \mathbf{a}^\mathsf{T}\mathbf{b}. \tag{B.19}$$

Transponiert man das äußere Produkt so vertauschen sich die Vektoren; insbesondere ist das äußere Produkt eines Vektors mit sich selbst eine symmetrische Matrix:

$$\left(\mathbf{a}\mathbf{b}^\mathsf{T}\right)^\mathsf{T} = \mathbf{b}\mathbf{a}^\mathsf{T}, \qquad \left(\mathbf{a}\mathbf{a}^\mathsf{T}\right)^\mathsf{T} = \mathbf{a}\mathbf{a}^\mathsf{T}. \tag{B.20}$$

Auch mit dem äußeren Produkt können wir die Matrixmultiplikation schreiben. Diese Form ist weniger üblich, aber genau wie die Darstellung mit dem inneren Produkt eine einfache Anwendung der Rechenregeln für Produkte zwischen Vektoren:

$$\mathbf{A} \in \mathbb{R}^{n\times m}, \mathbf{B} \in \mathbb{R}^{m\times o}, \qquad \mathbf{A}\mathbf{B} = (\mathbf{a}_0, \mathbf{a}_1, \dots \mathbf{a}_{m-1}) \begin{pmatrix} \mathbf{b}_0^\mathsf{T} \\ \mathbf{b}_1^\mathsf{T} \\ \vdots \\ \mathbf{b}_{m-1}^\mathsf{T} \end{pmatrix} = \sum_{i=0}^{m-1} \mathbf{a}_i \mathbf{b}_i^\mathsf{T}. \tag{B.21}$$

An dieser Darstellung sieht man sofort, dass die Beobachtung bzgl. Transposition von äußeren Produkten und Symmetrieeigenschaften aus Gl. B.20 auch für Matrizen gelten, d.h.

$$\left(\mathbf{A}\mathbf{B}^\mathsf{T}\right)^\mathsf{T} = \mathbf{B}\mathbf{A}^\mathsf{T}, \qquad \left(\mathbf{A}\mathbf{A}^\mathsf{T}\right)^\mathsf{T} = \mathbf{A}\mathbf{A}^\mathsf{T}. \tag{B.22}$$

Die Spur einer quadratischen Matrix ist gegenüber Transposition invariant:

$$\mathbf{A} \in \mathbb{R}^{n\times n}, \qquad tr\left(\mathbf{A}^\mathsf{T}\right) = tr(\mathbf{A}). \tag{B.23}$$

B.2.5. Weitere Rechenregeln

Das Matrixprodukt ist assoziativ und distributiv:

$$\mathbf{A}(\mathbf{B}\mathbf{C}) = (\mathbf{A}\mathbf{B})\mathbf{C}, \qquad \mathbf{A}(\mathbf{B}+\mathbf{C}) = \mathbf{A}\mathbf{B} + \mathbf{A}\mathbf{C}. \tag{B.24}$$

Die Einheitsmatrix (Gl. B.3) ist Neutralelement der Matrixmultiplikation:

$$\mathbf{A} \in \mathbb{R}^{n\times m}, \qquad \mathbf{A} = \mathbf{A}\mathbf{I}_m = \mathbf{I}_n\mathbf{A}. \tag{B.25}$$

Sie heißt deswegen auch *Identiät*. Das Matrixprodukt ist im allgemeinen *nicht kommutativ*

$$\mathbf{A}\mathbf{B} \neq \mathbf{B}\mathbf{A}. \tag{B.26}$$

Das sieht man für rechteckige Matrizen schon allein daran, dass niemals beide Produkte den Regeln der Matrixmultiplikation entsprechen. Aber auch quadratische Matrizen darf man nicht einfach vertauschen. Allerdings gibt es Paare $\mathbf{A}, \mathbf{B}$ von Matrizen die *kommutieren*. Zum Beispiel ist dies der Fall, wenn (mindestens) eine der beiden Matrizen die Einheitsmatrix ist (siehe Gl. B.25). Es ist nicht einfach, die Paare kommutierender Matrizen zu charakterisieren. Geschickte Rechenarbeit zeigt uns allerdings, dass die Spur eines Matrixproduktes invariant gegenüber Kommutation ist. Für die Spur des Produktes einer Matrix $\mathbf{A}$ in Zeilendarstellung und einer Matrix $\mathbf{B}$ in Spaltendarstellung

$$\mathbf{A} = \begin{pmatrix} \mathbf{a}_0^\mathsf{T} \\ \mathbf{a}_1^\mathsf{T} \\ \vdots \\ \mathbf{a}_{n-1}^\mathsf{T} \end{pmatrix}, \qquad \mathbf{B} = (\mathbf{b}_0, \mathbf{b}_1, \dots, \mathbf{b}_{n-1})$$

finden wir durch Anwendung einer Reihe der oben angegebenen Identitäten für Matrixprodukte, Spur und Transposition

$$tr(\mathbf{AB}) \underset{\text{Gl. B.14}}{=} \sum_{i=0}^{n-1} \mathbf{a}_i^\mathsf{T} \mathbf{b}_i. \underset{\text{Gl. B.19}}{=} \sum_{i=0}^{m-1} tr\left(\mathbf{a}_i \mathbf{b}_i^\mathsf{T}\right) \underset{\text{Gl. B.20}}{=} \sum_{i=0}^{m-1} tr\left(\left(\mathbf{b}_i \mathbf{a}_i^\mathsf{T}\right)^\mathsf{T}\right) \tag{B.27}$$

$$= \quad tr\left(\left(\sum_{i=0}^{m-1} \mathbf{b}_i \mathbf{a}_i^\mathsf{T}\right)^\mathsf{T}\right) \underset{\text{Gl. B.21}}{=} tr\left((\mathbf{BA})^\mathsf{T}\right) \underset{\text{Gl. B.23}}{=} tr(\mathbf{BA}). \tag{B.28}$$

Es ist ferner nützlich, sich zu vergegenwärtigen, dass viele Eigenschaften von Matrizen bei der Bildung von Produkten verloren gehen. So ist das Produkt von symmetrischen Matrizen nicht symmetrisch, was man z.B. an folgendem einfachen Gegenbeispiel sieht:

$$\begin{pmatrix} 0 & a \\ a & 0 \end{pmatrix} \begin{pmatrix} b & 0 \\ 0 & c \end{pmatrix} = \begin{pmatrix} 0 & ac \\ ab & 0 \end{pmatrix}. \tag{B.29}$$

B.2.6. Multplikative Inverse, lineare Gleichungssysteme

Auch für die Matrixmultiplikation kann man die Frage stellen, ob es zum Produkt eine inverse Operation gibt, sprich ob Gleichungen der Form

$$\mathbf{AX} = \mathbf{B}, \qquad \mathbf{A} \in \mathbb{R}^{n \times m}, \mathbf{X} \in \mathbb{R}^{m \times o}, \mathbf{B} \in \mathbb{R}^{n \times o} \tag{B.30}$$

für gegebene Matrizen $\mathbf{A}, \mathbf{B}$ eine Lösung in $\mathbf{X}$ haben. Strenggenommen müssten wir uns auch den Fall $\mathbf{XA}$ anschauen, aber alle folgenden Aussagen lassen sich auf diesen Fall ausdehnen, wenn man die Begriffe Zeile und Spalte vertauscht. Durch die übliche Darstellung von $\mathbf{A}$ als Zeilenvektoren und $\mathbf{X}, \mathbf{B}$ als Spaltenvektoren sehen wir, dass wir es mit o unabhängigen Problemen zu tun haben:

$$\mathbf{AX} = \begin{pmatrix} \mathbf{a}_0^\mathsf{T} \\ \mathbf{a}_1^\mathsf{T} \\ \vdots \\ \mathbf{a}_{n-1}^\mathsf{T} \end{pmatrix} (\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{o-1}) = (\mathbf{b}_0, \mathbf{b}_1, \ldots, \mathbf{b}_{o-1}). \tag{B.31}$$

Alle Eigenschaften dieses Problems können wir also auch analysieren, wenn wir uns auf einen Spaltenvektor $\mathbf{x}$ und einen Spaltenvektor $\mathbf{b}$ beschränken – sprich die Lösbarkeit von $\mathbf{Ax} = \mathbf{b}$ diskutieren. Probleme dieser Art nennen wir *lineare Gleichungssysteme (LGS)* und die Existenz und Anzahl der Lösungen hängt von $\mathbf{A}$ und $\mathbf{b}$ ab. Die praktische Bestimmung von Lösungen ist ein zentrales Thema des wissenschaftlichen Rechnens.

Für quadratische Matrizen beobachten wir, dass das Produkt zweier Matrizen $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n \times n}$ wieder eine quadratische Matrix mit $n \times n$ Elemente ist. Wir sagen, die quadratischen Matrizen sind gegenüber dem Matrixprodukt abgeschlossen. Die Situation ist analog zu den reellen Zahlen, wo für $a, b \in \mathbb{R}$ auch $ab \in \mathbb{R}$ folgt. Zu jeder reellen Zahl außer der Null gibt es ein inverses Element a^{-1}. Dieses Element ist definiert als die Zahl, die man mit a multiplizieren muss, um das multiplikative Neutralelement zu erhalten, also $a \cdot a^{-1} = 1$. Analog kann man für quadratische Matrizen die Inverse definieren:

$$\mathbf{I}_n = \mathbf{A}^{-1}\mathbf{A} = \mathbf{A}\mathbf{A}^{-1}. \tag{B.32}$$

Genau wie bei den reellen Zahlen gibt es nicht zu jeder quadratischen Matrix $\mathbf{A}$ ein inverses Element. Wir werden im Weiteren erst ein wenig geometrische Intuition aufbauen, bevor wir die Frage nach der Existenz weiter diskutieren. Existiert allerdings ein inverses Element, so führt dies direkt auf die Lösung von linearen Gleichungssystemen:

$$\mathbf{A}\mathbf{x} = \mathbf{b} \quad \Longleftrightarrow \quad \mathbf{A}^{-1}\mathbf{A}\mathbf{x} = \mathbf{A}^{-1}\mathbf{b} \quad \underset{\text{Gl. B.32}}{\Longleftrightarrow} \quad \mathbf{x} = \mathbf{A}^{-1}\mathbf{b}. \tag{B.33}$$

Diese Gleichung beschreibt zwar die Lösung $\mathbf{x}$, aber aus Sicht des wissenschafltichen Rechnens ist es selten sinnvoll, die Inverse $\mathbf{A}^{-1}$ explizit auszurechnen und dann mit dem Vektor $\mathbf{b}$ zu multiplizieren.

B.3. Geometrische Intuition, euklidische Räume

Viele Begriffe und Eigenschaften für das Rechnen mit Matrizen und Vektoren haben einen geometrischen Hintergrund. Den meisten Menschen fällt auch das Verstädnis auf Basis dieser geometrischen Anschauung leichter. Wir betrachten dazu die Vektoren als Punkte in einem n-dimensionalen euklidischen Raum – zur Anschauung zeigen wir Bilder im zweidimensionalen Raum, also der Ebene.

B.3.1. Vektoren und Längen und Winkel – Skalarprodukt

Mit jedem Punkt im euklidischen Raum verbinden wir den Ursprungsvektor, also die Verbindung zwischen Urpsrung und Punkt. Dann können wir über Längen von und Winkel zwischen den Vektoren reden. Das Skalarprodukt ist unser „Messwerkzeug“. Die Länge der Strecke vom Ursprung bis zum Punkt $\mathbf{x} \in \mathbb{R}^n$ ergibt sich aus der Beobachtung

$$\|\mathbf{x}\|^2 = \mathbf{x}^\mathsf{T}\mathbf{x}, \tag{B.34}$$

also als Quadratwurzel des inneren Produktes eines Vektors mit sich selbst. Wir sagen ein Urpsrungsvektor $\mathbf{x}$ hat *Einheitslänge*, falls gilt $\mathbf{x}^\mathsf{T}\mathbf{x} = 1$; wir sagen dann auch der Vektor $\mathbf{x}$ ist *normiert*. Zwischen normierten Vektoren misst das Skalarprodukt den Kosinus des Winkels:

$$\mathbf{x}, \mathbf{y} \in \mathbb{R}^n, \quad \mathbf{x}^\mathsf{T}\mathbf{x} = \mathbf{y}^\mathsf{T}\mathbf{y} = 1, \qquad \mathbf{x}^\mathsf{T}\mathbf{y} = \cos(\phi). \tag{B.35}$$

Haben die Vektoren keine Einheitslänge, so könnte man sie normieren, indem man durch ihre Länge teilt. Multipliziert man das Produkt der Längen auf beiden Seiten in Gl. B.35 so ergibt sich

$$\mathbf{x}, \mathbf{y} \in \mathbb{R}^n, \qquad \mathbf{x}^\mathsf{T}\mathbf{y} = \|\mathbf{x}\|\|\mathbf{y}\| \cos(\phi). \tag{B.36}$$

Man beachte, dass diese Darstellung auch richtig bleibt, wenn einer der Vektoren die Länge Null hat, also verschwindet. Dann ist das Produkt auf beiden Seiten Null. Darüberhinaus ist das Produkt Null, wenn $\cos(\phi) = 0$, also wenn die Vektoren einen Winkel von $\pi/2$ einschließen. Man sagt dann die Vektoren sind *orthogonal*.

B.3.2. Basis, Basiswechsel

Sei $\mathbf{x} \in \mathbb{R}^n$, dann sind die Koeffizienten x_i Koordinaten bzgl. der *Standardbasis* des Raumes:

$$\mathbf{x} = x_0\mathbf{e}_0 + x_1\mathbf{e}_1 + \ldots + x_{n-1}\mathbf{e}_{n-1}, \qquad \mathbf{e}_0 = \begin{pmatrix}1\\0\\\vdots\\0\end{pmatrix}, \mathbf{e}_1 = \begin{pmatrix}0\\1\\\vdots\\0\end{pmatrix}, \ldots, \mathbf{e}_{n-1} = \begin{pmatrix}0\\0\\\vdots\\1\end{pmatrix}. \quad \text{(B.37)}$$

Diese Darstellung ist trivial und drückt vergleichweise umständlich die Tatsache $\mathbf{Ix} = \mathbf{x}$ aus. Wichtig ist aber dabei eine allgemeine Idee zu sehen: Die Elemente von $\mathbf{x}$ beschreiben einen Punkt bzgl. eines Koordinatensystems. Das Koordinatensystem wird durch Basisvektoren beschrieben. Die Koeffizienten von $\mathbf{x}$ geben an, „wieviel“ von jedem der Basisvektoren verwendet werden muss, oder „wie weit“ man sich in Richtung jedes Basisvektors vom Ursprung aus bewegen muss. Schreibt man die Basisvektoren als Spaltenvektoren in eine Matrix, dann ergibt sich der Ort aus dem Koeffizientenvektor $\mathbf{x} \in \mathbb{R}^n$ durch Bildung eines Matrix-Vektor-Produktes:

$$(\mathbf{a}_0, \mathbf{a}_1, \ldots, \mathbf{a}_{n-1})\,\mathbf{x} = \mathbf{Ax}. \quad \text{(B.38)}$$

Dabei müssen die Basisvektoren $\mathbf{a}_i$ nicht unbedingt Vektoren mit n Elementen sein – $\mathbf{x}$ könnte auch ein Ort in einem Raum mit mehr oder weniger als n Dimensionen ausdrücken. Das Matrix-Vektor-Produkt beschreibt in jedem Fall einen Wechsel der Basis, und zwar von der Basis, die durch die Spalten der Matrix $\mathbf{A}$ gegeben ist, in die Einheits- oder Standardbasis.

Damit können wir die Frage nach der Lösung (und Lösbarkeit) von Gleichungssystemen $\mathbf{Ax} = \mathbf{b}$ wie folgt geometrisch interpretieren: gegeben einen Punkt $\mathbf{b} \in \mathbb{R}^n$, lässt sich der Punkt in der durch die Spalten von $\mathbf{A}$ gegebenen Basis darstellen und, wenn ja, was sind die Koordinaten dieses Punktes bzgl. dieser Basis?

B.3.3. Orthogonalbasis und orthogonale Matrizen

Besonders praktisch ist es, wenn die Basisvektoren orthogonal zueinander stehen. Warum? Betrachten wir einen beliebigen Punkt $\mathbf{b} \in \mathbb{R}^n$, den wir in der Basis $\mathbf{a}_0, \mathbf{a}_1, \ldots$ darstellen wollen. Wenn wir $\mathbf{b}$ in Richtung $\mathbf{a}_0$ bewegen, dann ändert dies nichts an den Koordinaten bzgl. $\mathbf{a}_1, \ldots$ – nur die Koordinate bzgl. $\mathbf{a}_0$ wird verändert. Um zu sehen, wie man die Koordinate bzgl. $\mathbf{a}_i$ bestimmt, verschieben wir den Punkt $\mathbf{b}$ entlang der anderen Basisrichtungen so, dass nur noch der Anteil bzgl. $\mathbf{a}_i$ übrig bleibt. Dann hat der Punkt also die Darstellung $\mathbf{b} = x_i\mathbf{a}_i$ (alle anderen $x_j, j \neq i$ sind Null) und x_i ist direkt der gesuchte Koeffizient. Man erhält x_i indem man von links mit $\mathbf{a}^\mathsf{T}$ multipliziert und so eine skalare Gleichung erzeugt:

$$\mathbf{a}_i^\mathsf{T}\mathbf{b} = x_i\mathbf{a}_i^\mathsf{T}\mathbf{a}_i \quad \Longleftrightarrow \quad x_i = \frac{\mathbf{a}_i^\mathsf{T}\mathbf{b}}{\mathbf{a}_i^\mathsf{T}\mathbf{a}_i}. \quad \text{(B.39)}$$

Ganz besonders bequem ist die Bestimmung der Koeffizienten bzgl. von Basisvektoren $\mathbf{a}_i$ mit Einheitslänge. Dann verschwindet der Nenner in Gl. B.39. Wir fassen unsere Beobachtungen zusammen: die Darstellung eines Punktes $\mathbf{b}$ in den Spaltenvektoren der Matrix $\mathbf{A}$ entspricht der Lösung des linearen Gleichungssystems $\mathbf{Ax} = \mathbf{b}$. Wenn die Spaltenvektoren $\mathbf{a}_i$ der Matrix Einheitslänge haben (also $\mathbf{a}_i^\mathsf{T}\mathbf{a}_i = 1$) und je zwei unterschiedliche Spaltenvektoren orthogonal zu einander stehen (also $\mathbf{a}_i^\mathsf{T}\mathbf{a}_j = 0, i \neq j$), dann ergeben sich die Koeffizienten $\mathbf{x}$ als $x_i = \mathbf{a}_i^\mathsf{T}\mathbf{b}$. Ob wir den Punkt $\mathbf{b}$ auf diese Weise tatsächlich darstellen können, hängt davon ab, ob

„genug“ Basisvektoren $\mathbf{a}_i$ zur Verfügung stehen. Es ist plausibel, dass man zur Darstellung eines Punktes im n-dimensionalen Raum auch n Basisvektoren braucht (schon eine Teilmenge der Standardbasis liefert direkt einen Widerspruch). Dieses Ergebnis können wir in Matrixschreibweise wie folgt notieren:

$$\begin{aligned}
&\mathbf{A} = (\mathbf{a}_0, \mathbf{a}_1, \ldots, \mathbf{a}_{n-1}), \quad \mathbf{a}_i \in \mathbb{R}^n, \quad \mathbf{a}_i^\mathsf{T}, \mathbf{a}_i = 1, \quad \mathbf{a}_i^\mathsf{T}\mathbf{a}_j = 0, i \neq j \\
&\mathbf{A}\mathbf{x} = \mathbf{b} \quad \Longrightarrow \quad \mathbf{x} = \begin{pmatrix} \mathbf{a}_0^\mathsf{T}\mathbf{b} \\ \mathbf{a}_1^\mathsf{T}\mathbf{b} \\ \vdots \end{pmatrix} = \mathbf{A}^\mathsf{T}\mathbf{b}.
\end{aligned} \tag{B.40}$$

Die beschriebenen Matrizen mit orthogonalen und normierten Spaltenvektoren führen also auf eine besonders einfache Lösung von linearen Gleichungssystemen. Setzen wir $\mathbf{b} = \mathbf{e}_i$ so ergibt sich direkt

$$\mathbf{A}\,(\mathbf{x}_0, \mathbf{x}_1, \ldots) = (\mathbf{e}_0, \mathbf{e}_1, \ldots) = \mathbf{I}_n \quad \Longrightarrow \quad (\mathbf{x}_0, \mathbf{x}_1, \ldots) = \mathbf{A}^\mathsf{T}\mathbf{I}_n = \mathbf{A}^\mathsf{T} \tag{B.41}$$

und damit $\mathbf{A}\mathbf{A}^\mathsf{T} = \mathbf{I}_n$. Diese Eigenschaft kann man auch direkt aus den Anforderungen an die Spaltenvektoren ableiten. Offensichtlich sind nicht nur die Spaltenvektoren paarweise orthogonal sondern auch die Zeilenvektoren. Matrizen mit dieser Eigenschaft nennt man *Orthogonalmatrizen*. Man beachte, dass Orthogonalmatrizen immer normierte Spaltenvektoren (und damit auch normierte Zeilenvektoren) haben, während die Länge der Basisvektoren einer Orthogonalbasis beliebig ist. Der Vergleich mit der Definition von multiplikativer Inverse (Gl. B.32) ergibt sofort, dass die Inverse einer Orthogonalmatrix ihre Transponierte ist:

$$\mathbf{A}\mathbf{A}^\mathsf{T} = \mathbf{I}_n \quad \Longleftrightarrow \quad \mathbf{A}^{-1} = \mathbf{A}^\mathsf{T}. \tag{B.42}$$

Basiswechsel mit Orthogonalmatrizen haben wichtige Eigenschaften. Zum Beispiel erhalten sie Längen von und Winkel zwischen Vektoren. Etwas allgemeiner kann man sagen, dass Basiswechsel mit orthogonalen Matrizen das Skalarprodukt nicht verändern. Betrachtet man zwei Vektoren $\mathbf{x}$ und $\mathbf{y}$ in Standardbasis, so ergibt sich die Darstellung nach der Transformation als $\mathbf{A}^\mathsf{T}\mathbf{x}$ und $\mathbf{A}^\mathsf{T}\mathbf{y}$. Für das Skalarprodukt der transformierten Vektoren erhält man

$$\left(\mathbf{A}^\mathsf{T}\mathbf{x}\right)^\mathsf{T}\left(\mathbf{A}^\mathsf{T}\mathbf{y}\right) = \left(\mathbf{x}^\mathsf{T}\mathbf{A}\right)\left(\mathbf{A}^\mathsf{T}\mathbf{y}\right) = \mathbf{x}^\mathsf{T}\left(\mathbf{A}\mathbf{A}^\mathsf{T}\right)\mathbf{y} = \mathbf{x}^\mathsf{T}\mathbf{I}\mathbf{y} = \mathbf{x}^\mathsf{T}\mathbf{y}. \tag{B.43}$$

Der Basiswechsel in eine orthogonale Basis erhält zumindest Winkel, aber keine Längen.

B.3.4. Orthogonalisierung – Gram-Schmidt

Wie beschrieben haben orthogonale Basen nützliche Eigenschaften. Aber wie gelangt man an eine Orthogonalbasis $\mathbf{a}_0, \mathbf{a}_1, \ldots$? Jede Basis $\mathbf{b}_0, \mathbf{b}_1, \ldots$ des $\mathbb{R}^n$ lässt sich, zumindest konzeptionell, recht einfach *orthogonalisieren*. Dabei baut man die Orthogonalbasis schrittweise auf und entfernt aus dem Basiselement $\mathbf{b}_i$ die Anteile der bereits bestimmten orthogonalen Vektoren $\mathbf{a}_j, j < i$. Diese Vorgehen wird als *Gram-Schmidt-Verfahren* bezeichnet.

Man beginnt mit $\mathbf{a}_0 = \mathbf{b}_0$. Das nächste Element $\mathbf{a}_1$ soll orthogonal auf $\mathbf{a}_0$ stehen. Man nimmt $\mathbf{b}_1$ und entfernt den Anteil von $\mathbf{a}_0$, also

$$\mathbf{a}_1 = \mathbf{b}_1 - \frac{\mathbf{a}_0^\mathsf{T}\mathbf{b}_1}{\mathbf{a}_0^\mathsf{T}\mathbf{a}_0}\mathbf{a}_0. \tag{B.44}$$

Eine kurze Rechnung zeigt, dass in der Tat $\mathbf{a}_1$ orthogonal auf $\mathbf{a}_0$ steht:

$$\mathbf{a}_0^\mathsf{T}\mathbf{a}_1 = \mathbf{a}_0^\mathsf{T}\mathbf{b}_1 - \frac{\mathbf{a}_0^\mathsf{T}\mathbf{b}_1}{\mathbf{a}_0^\mathsf{T}\mathbf{a}_0}\mathbf{a}_0^\mathsf{T}\mathbf{a}_0 = \mathbf{a}_0^\mathsf{T}\mathbf{b}_1 - \mathbf{a}_0^\mathsf{T}\mathbf{b}_1 = 0. \tag{B.45}$$

Der nächste Basisvektor $\mathbf{a}_2$ soll nun orthogonal auf sowohl $\mathbf{a}_0$ wie auch $\mathbf{a}_1$ stehen. Man entfernt aus $\mathbf{b}_2$ die Anteile beider Vektoren:

$$\mathbf{a}_2 = \mathbf{b}_2 - \frac{\mathbf{a}_0^\mathsf{T}\mathbf{b}_2}{\mathbf{a}_0^\mathsf{T}\mathbf{a}_0}\mathbf{a}_0 - \frac{\mathbf{a}_0^\mathsf{T}\mathbf{b}_2}{\mathbf{a}_1^\mathsf{T}\mathbf{a}_1}\mathbf{a}_1. \tag{B.46}$$

Allgemein entfern man für das Basiselement $\mathbf{a}_i$ alle Anteile der vorangegangenen Basiselement und erhält die Rechenvorschrift:

$$\mathbf{a}_i = \mathbf{b}_i - \sum_{j=0}^{i-1} \frac{\mathbf{a}_j^\mathsf{T}\mathbf{b}_i}{\mathbf{a}_j^\mathsf{T}\mathbf{a}_j}\mathbf{a}_j. \tag{B.47}$$

Möchte man normierte orthogonal Basisvektoren $\tilde{\mathbf{a}}_i$, also eine Orhtogonalbasis, so kann man einfach nach jedem Schritt den Vektor normieren. Durch die Normierung entfallen die Nenner der Form $\mathbf{a}_j^\mathsf{T}\mathbf{a}_j$, die diese für normierte Vektoren alle 1 sind. Konkret:

$$\mathbf{a}_i = \mathbf{b}_i - \sum_{j=0}^{i-1} (\tilde{\mathbf{a}}_j^\mathsf{T}\mathbf{b}_i)\tilde{\mathbf{a}}_j \tag{B.48}$$

$$\tilde{\mathbf{a}}_i = \frac{\mathbf{a}_i}{\|\mathbf{a}_i\|}. \tag{B.49}$$

Man kann sich leicht überlegen, dass die Vektoren $\mathbf{a}_0, \ldots, \mathbf{a}_j$ (bzw. $\tilde{\mathbf{a}}_0, \ldots, \tilde{\mathbf{a}}_j$) den gleichen Unterraum aufspannen wie $\mathbf{b}_0, \ldots, \mathbf{b}_j$. Da $\mathbf{b}_{j+1}$ kein Element dieses Raumes ist, kann $\mathbf{a}_{j+1}$ nicht verschwinden. Zumindest in reellen Zahlen erzeugt das Gram-Schmidt-Verfahren also aus jeder Basis eine Orthogonalbasis. Für die Praxis des wissenschaftlichen Rechnens ist das Verfahren in der oben genannten Weise aber weniger geeignet – es gibt verschiedene gute Alternativen.

B.3.5. Fläche, Volumen – Determinante

Den Flächeninhalt eines Parallelograms in der Ebene berechnen wir nach der Regel „Grundseite mal Höhe“. Nehmen wir an das Parallelogram wird von zwei Vektoren $\mathbf{x}, \mathbf{y} \in \mathbb{R}^2$ aufgespannt. Wir können $\mathbf{x}$ als Grundseite nehmen. Dann ist die Höhe $|\sin(\phi)|\|\mathbf{y}\|$, wobei ϕ der Winkel zwischen $\mathbf{x}$ und $\mathbf{y}$ ist. Verwenden wir in der Rechnung statt $\mathbf{y}$ den um $90°$ gedrehten Vektor $\mathbf{y}^\perp$ so ergibt sich

$$A(\mathbf{x}, \mathbf{y}) = \|\mathbf{x}\|\|\mathbf{y}\||\sin(\phi)| = \|\mathbf{x}\|\|\mathbf{y}^\perp\||\cos(\phi)| = |\mathbf{x}^\mathsf{T}\mathbf{y}^\perp| = |x_0 y_1 - x_1 y_0|. \tag{B.50}$$

Wir haben also offenbar eine Regel gefunden, um von einer 2×2-Matrix zum Flächeninhalt des von den Spaltenvektoren aufgespannten Parallelograms zu gelangen. Wir nennen die Abbildung

$$\mathbf{A} = \begin{pmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{pmatrix}, \qquad |\mathbf{A}| = a_{00} \cdot a_{11} - a_{01} \cdot a_{10} \tag{B.51}$$

Determinante der Matrix $\mathbf{A}$. Wichtiger noch als die Volumenberechnung eines Parallelograms ist die Interpretation der Determinante wenn $\mathbf{A}$ einen Basiswechsel beschreibt: die Determinante $|\mathbf{A}|$ misst die *Veränderung* des Flächeninhalts jeder ebenen geometrischen Figur beim Wechsel in die durch $\mathbf{A}$ definierte Basis (die Spalten von $\mathbf{A}$). Man beachte, dass die Determinante ein Vorzeichen hat, also nicht nur den Flächeninhalt sondern auch die Orientierung beschreibt. Was bedeutet es, wenn die Determinante Null ist? Jede Figur (bspw. ein Dreieck) hat nach der Transformation in die Basis die durch die Spalten von $\mathbf{A}$ aufgespannt wird, keinen Flächeninhalt mehr, sprich die Figuren degenerieren zu Linien oder Punkten.

Der Determinantenbegriff lässt sich auf alle quadratischen Matrizen ausdehnen. Für 3×3 Matrizen misst die Determinante das Volumen des von den Spaltenvektoren aufgespannten sog. Parallelepipeds, bzw. die Veränderung des Volumens bei einem Basiswechsel. Sie kann wie folgt berechnet werden:

$$\mathbf{A} = \begin{pmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{pmatrix}, \qquad \begin{aligned} |\mathbf{A}| = {} & a_{00}a_{11}a_{22} + a_{01}a_{12}a_{20} + a_{02}a_{10}a_{21} \\ & - a_{00}a_{12}a_{21} - a_{01}a_{10}a_{22} - a_{02}a_{11}a_{20}. \end{aligned} \tag{B.52}$$

Für größere Matrizen gibt es zwei bekannte Formeln, die Laplace und Leibnitz zugeschrieben werden. Wir geben sie hier nicht wieder, weil sie für die Berechnung der Determinante wenig nützlich sind. Stattdessen nennen wir einige Eigenschaften, die wir verwenden werden, und versuchen sie plausibel zu machen.

Wie erwähnt erhalten orthogonale Matrizen alle Längen und Winkel. Damit erhalten sie auch Fläche, Volumen, und alle analogen Begriffen in höheren Dimensionen. Es gilt also:

$$\mathbf{A}^\mathsf{T}\mathbf{A} = \mathbf{I} \quad \Longrightarrow \quad |\mathbf{A}| = \pm 1. \tag{B.53}$$

Insbesondere gilt für die Identität $|\mathbf{I}| = 1$. Man beachte, dass orthogonale Matrizen spiegeln können und dass dabei das Volumen sein Vorzeichen, nicht aber den Betrag, ändert. Hat eine orthogonale Matrix $\mathbf{A}$ positive Determinante, also $|\mathbf{A}| = +1$ so spricht man aus naheliegenden Gründen von einer *Rotation*. Rotationen erhalten neben Längen und Winkeln auch die Orientierung. Ferner ist es wichtig einzusehen, dass volumenerhaltende Abbildungen (also Abbildungen $\mathbf{A}$ mit $|\mathbf{A}| = 1$) nicht notwendigerweise orthogonal sind. Ein Beispiel für eine solche Matrix ist $\begin{pmatrix} 2 & 0 \\ 1 & 1/2 \end{pmatrix}$ – wie man leicht ausrechnet ist die Determinante 1, aber keine der Spalten normiert und die Spalten nicht orthogonal zu einander.

Eine Diagonalmatrix skaliert entlang der Achsen des Koordinatensystems. Die Volumenänderung ist also genau das Produkt der Diagonalelemente:

$$\mathbf{D} = \begin{pmatrix} d_0 & 0 & \dots & 0 \\ 0 & d_1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & d_{n-1} \end{pmatrix} \quad \Longrightarrow \quad |\mathbf{D}| = d_0 \cdot d_1 \cdot \ldots \cdot d_{n-1}. \tag{B.54}$$

Skaliert man entlang aller Achsen mit dem gleichen Faktor s, kann man die Wirkung auch einfach durch Multiplikation aller Elemente mit dem Skalar s beschreiben. Man hat also:

$$\mathbf{A} \in \mathbb{R}^n, \qquad |s\mathbf{A}| = s^n|\mathbf{A}|. \tag{B.55}$$

Betrachtet man die Wirkung von zwei aufeinanderfolgenden Basiswechseln, so muss man die Volumenveränderungen offenbar mit einander multiplizieren (man kann sich das in der Ebene oder im Raum leicht plausibel machen):

$$|\mathbf{AB}| = |\mathbf{A}||\mathbf{B}| = |\mathbf{B}||\mathbf{A}| = |\mathbf{BA}|. \tag{B.56}$$

Die Determinante ist also nach der Spur eine weitere Funktion auf quadratischen Matrizen, die gegenüber Kommutation invariant ist. Die Beobachtung für die Skalierung in Gl. B.55 ist nur ein Spezialfall, der allgemeineren Regel in Gl. B.56. Es folgt eine weitere Beobachtung aus der Definition der multiplikativen Inversen

$$\mathbf{I} = \mathbf{A}\mathbf{A}^{-1} \qquad 1 = |\mathbf{I}| = |\mathbf{A}\mathbf{A}^{-1}| = |\mathbf{A}||\mathbf{A}^{-1}| \quad \Longleftrightarrow \quad |\mathbf{A}^{-1}| = |\mathbf{A}|^{-1}. \tag{B.57}$$

Eine multiplikative Inverse für $\mathbf{A}$ kann also offenbar nur dann existieren, wenn die Determinante nicht verschwindet, also $|\mathbf{A}| \neq 0$. Dies entsprich auch der geometrischen Intuition von Basiswechseln: wenn bei einem Basiswechsel das Volumen vollständig verloren geht, kann dieser Prozess durch einen weiteren Basiswechseln nicht „rückgängig“ gemacht werden.

B.3.6. Eigenvektoren und Diagonalisierung

Nachdem wir uns überlegt haben, wie ein Basiswechsel auf Längen, Winkel und Volumen wirkt, betrachten wir nun, was er mit einzelnen Vektoren macht. Die Idee ist, nach Vektoren zu „suchen“, die sich beim Basiswechsel nur in der Länge verändern, nicht aber in der Richtung. Diese Vektoren sind weitgehend invariant gegenüber dem Basiswechsel und helfen uns, die Wirkung besser zu verstehen. Ein solcher Vektor erfüllt offenbar folgende Gleichung:

$$\mathbf{A} \in \mathbb{R}^{n\times n} \qquad \mathbf{A}\mathbf{x} = \lambda\mathbf{x}. \tag{B.58}$$

Ein Vektor $\mathbf{x}$ der Gl. B.58 erfüllt heißt *Eigenvektor* von $\mathbf{A}$. Der Faktor λ heißt *Eigenwert* zum Eigenvektor $\mathbf{x}$ von $\mathbf{A}$. Da mit $\mathbf{x}$ auch $s\mathbf{x}$ ein Eigenvektor von $\mathbf{A}$ ist (und zwar zum selben Eigenwert), wählen wir den Eigenvektor immer normiert, also so dass $\mathbf{x}^\mathsf{T}\mathbf{x} = 1$. Die Bestimmung der Eigenvektoren und/oder Eigenwerte für eine gegebene Matrix $\mathbf{A}$ ist nicht einfach und Thema des wissenschaftlichen Rechnens.

Für symmetrische Matrizen haben Eigenvektoren eine besondere Eigenschaft: sie sind orthogonal (bzw. können orthgonal gewählt werden). Das ist leicht zu sehen. Wir nehmen an

$$\mathbf{A} = \mathbf{A}^\mathsf{T}, \quad \mathbf{A}\mathbf{x} = \lambda\mathbf{x}, \quad \Longleftrightarrow \quad \mathbf{x}^\mathsf{T}\mathbf{A}^\mathsf{T} = \lambda\mathbf{x}^\mathsf{T}, \quad \mathbf{A}\mathbf{y} = \mu\mathbf{y}$$

und betrachten

$$\lambda \cdot \mathbf{x}^\mathsf{T}\mathbf{y} = \left(\lambda\mathbf{x}^\mathsf{T}\right)\mathbf{y} = \left(\mathbf{x}^\mathsf{T}\mathbf{A}^\mathsf{T}\right)\mathbf{y} = \mathbf{x}^\mathsf{T}\mathbf{A}\mathbf{y} = \mathbf{x}^\mathsf{T}\left(\mathbf{A}\mathbf{y}\right) = \mathbf{x}^\mathsf{T}\mu\mathbf{y} = \mu \cdot \mathbf{x}^\mathsf{T}\mathbf{y}. \tag{B.59}$$

Sind die Eigenwerte λ und μ verschieden, dann folgt direkt $\mathbf{x}^\mathsf{T}\mathbf{y} = 0$, und damit sind die Eigenvektoren $\mathbf{x}$ und $\mathbf{y}$ orthogonal, wie behauptet. Es bleibt den (Spezial-)Fall $\lambda = \mu$ zu betrachten – wir haben also zwei unterschiedliche normierte Eigenvektoren $\mathbf{x}, \mathbf{y}$ zu einem Eigenwert λ. Wie bereits bemerkt, ist die Eigenvektorgleichung für alle $s\mathbf{x}$ erfüllt. Wir addieren die beiden linken und rechten Seiten der Eigenvektorgleichungen für $s\mathbf{x}$ und $\mathbf{y}$ und sehen

$$\mathbf{A}\ s\mathbf{x} + \mathbf{A}\mathbf{y} = \mathbf{A}(s\mathbf{x} + \mathbf{y}) = \lambda(s\mathbf{x} + \mathbf{y}), \tag{B.60}$$

sprich, jeder Vektor der Form $s\mathbf{x}+\mathbf{y}$ ist ebenfalls Eigenvektor zum Eigenwert λ. Ausgehend von $\mathbf{x}$ können wir nun statt $\mathbf{y}$ den Eigenvektor $\mathbf{z}=-\mathbf{x}^\mathsf{T}\mathbf{y}\cdot\mathbf{x}+\mathbf{y}$ konstruieren, der auf $\mathbf{x}$ wie gewünscht orthogonal steht:

$$\begin{aligned}\mathbf{x}^\mathsf{T}\mathbf{z} &= \mathbf{x}^\mathsf{T}(-\mathbf{x}^\mathsf{T}\mathbf{y}\cdot\mathbf{x})+\mathbf{x}^\mathsf{T}\mathbf{y}\\ &= -\mathbf{x}^\mathsf{T}(\mathbf{x}\cdot\mathbf{x}^\mathsf{T}\mathbf{y})+\mathbf{x}^\mathsf{T}\mathbf{y}\\ &= -(\mathbf{x}^\mathsf{T}\mathbf{x})\cdot\mathbf{x}^\mathsf{T}\mathbf{y}+\mathbf{x}^\mathsf{T}\mathbf{y}\\ &= -1\cdot\mathbf{x}^\mathsf{T}\mathbf{y}+\mathbf{x}^\mathsf{T}\mathbf{y}\\ &= 0.\end{aligned} \tag{B.61}$$

Wenn es also mehrere (trotz Normierung unterschiedliche) Eigenvektoren mit demselben Eigenwert gibt, kann man sie stets orthogonal wählen. Man kann zeigen, dass es für eine symmetrische Matrix $\mathbf{A}\in\mathbb{R}^{n\times n}$ immer n orthogonale Eigenvektoren gibt (die zudem alle reellwertig sind). Wir bezeichnen die n Eigenvektoren mit $\mathbf{x}_0,\mathbf{x}_1,\ldots$. Jetzt schreiben wir alle Eigenvektorgleichungen in Matrixform. Da die Eigenvektoren in Spaltenform dargestellt sind, multiplizieren wir die Eigenwerte in einer Diagonalmatrix von rechts:

$$\mathbf{A}\mathbf{X}=\mathbf{A}\left(\mathbf{x}_0,\mathbf{x}_1,\ldots,\mathbf{x}_{n-1}\right)=\left(\mathbf{x}_0,\mathbf{x}_1,\ldots,\mathbf{x}_{n-1}\right)\begin{pmatrix}\lambda_0 & 0 & \ldots & 0\\ 0 & \lambda_1 & \ldots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \ldots & \lambda_{n-1}\end{pmatrix}=\mathbf{X}\Lambda. \tag{B.62}$$

Die Spalten der Matrix $\mathbf{X}$ sind nach Konstruktion alle normiert und paarweise orthogonal sind. Also ist die Inverse von $\mathbf{X}$ ihre Transponierte und wir erhalten:

$$\mathbf{A}\mathbf{X}=\mathbf{X}\Lambda \quad\Longleftrightarrow\quad \mathbf{A}=\mathbf{X}\Lambda\mathbf{X}^\mathsf{T}. \tag{B.63}$$

Das heißt also, jede symmetrische Matrix lässt sich als Produkt dreier Matrizen darstellen, nämlich eine orthogonalen Matrix, die die Eigenvektoren von $\mathbf{A}$ als Spalten enthält, einer Diagonalmatrix mit den Eigenwerten von $\mathbf{A}$ auf der Diagonale, und einer orthogonalen Matrix, bei der die Eigenvektoren in den Zeilen stehen. Da mit $\mathbf{x}$ auch $-\mathbf{x}$ ein Eigenvektor ist, kann man die orthogonale Eigenbasis $\mathbf{X}$ immer so wählen, dass sie nicht spiegelt, also als Rotation.

Wir sehen direkt, dass die Determinante eine symmetrischen Matrix dem Produkt ihrer Eigenwerte entspricht:

$$\mathbf{A}=\mathbf{A}^\mathsf{T},\qquad |\mathbf{A}| \underset{\text{Gl. B.63}}{=} |\mathbf{X}\Lambda\mathbf{X}^\mathsf{T}| \underset{\text{Gl. B.56}}{=} |\mathbf{X}||\Lambda||\mathbf{X}^\mathsf{T}| \underset{\text{Gl. B.53}}{=} |\Lambda| \underset{\text{Gl. B.54}}{=} \lambda_0\cdot\lambda_1\cdot\ldots\cdot\lambda_{n-1}. \tag{B.64}$$

C. Matrixfaktorisierungen

C.1. LR Zerlegung[1].

In der Praxis will man häufig ein Gleichungssystem $Ax = b$ für verschiedene Vektoren b lösen. Das Gauß-Verfahren, wie wir es bis jetzt betrachtet haben, erfordert jedoch für jeden Vektor b eine vollständig neue Lösung des Gleichungssystems. Die LR Zerlegung bietet hierfür eine Lösung und ermöglicht es, ein Gleichungssystem effizient für verschiedene "Eingabedaten" b zu lösen. Einen Teil der LR Zerlegung haben wir bereits kennengelernt: R steht für die obere Dreiecksmatrix, welche das Ergebnis der Gauß-Elimination ist. Was für eine bereits berechnete Matrix R fehlt, um diese für eine beliebige rechte Seite b verwenden zu können, sind die Transformationen, Zeilenaddition und Zeilentausch, welche von der Eingabematrix A zu R geführt haben. Der zweite Teil der LR Zerlegung, die untere Dreiecksmatrix L, enthält diese Informationen. Damit haben wir $A = L\,R$ und L und R repräsentieren zusammen wieder das ursprüngliche lineare Gleichungssystem. Da L und R beide Dreiecksmatrizen sind, ist eine sehr effiziente Lösung des Gleichungssystems mit Vorwärts- und Rückwärtseinsetzen möglich. Betrachten wir nun näher, wie L konstruiert werden kann und wie damit eine effiziente Lösung eines Gleichungssystems möglich ist.

Betrachten wir ein 2×2 Gleichungssystem, welche in Matrixform als

$$A \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} a & b \\ -a & c \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} e \\ f \end{pmatrix} \tag{C.1}$$

gegeben ist. Für die Gauß-Elimination müssen wir $-a$ in der zweiten Zeile durch die Addition der beiden Zeile eliminieren. Wenn wir das Produkt von A mit einer speziellen Matrix

$$L_1 = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} = \mathrm{Id} + \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \tag{C.2}$$

betrachten, so haben wir

$$L_1\,A = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} a & b \\ -a & c \end{pmatrix} = \begin{pmatrix} a & b \\ a-a & b+c \end{pmatrix} \tag{C.3}$$

Wir sehen also, dass $L_1\,A$ genau die gewünschten Operationen implementiert: die erste Zeile des Produkts ist wieder die erste Zeile von A und die zweite Zeile ist die Summe der Zeilen in A.

Wenn wir mehrere Zeilen-Transformationen haben, dann kann können diese als Produkt von mehreren L_a Matrizen ausgedrückt werden. Zum Beispiel, für ein 3×3 Gleichungssystem drücken

$$L_1 = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad L_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \tag{C.4}$$

[1] LR steht für „links-rechts". Im englischen spricht man von der *LU-Decomposition* für „lower-upper"

die Addition der ersten mit der zweiten und der ersten mit der dritten Zeile aus. Das Produkt der Matrizen

$$L_2\, L_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \tag{C.5}$$

entspricht, entsprechend der Definition der Matrixmultiplikation, der aufeinanderfolgenden Umsetzung der beiden Transformationen. An den vorhergehenden Beispielen sehen wir auch, dass L_a die Summe aus der Einheitsmatrix und einer Matrix $\overline{L_a}$ ist, deren (i, j)-te Element 1 ist, wenn wir die j-te Zeile der Matrix zur i-ten addieren. Im Allgemeinen, wenn wir Vielfache von Zeilen addieren, gilt für das (i, j)-te Element der Matrix $\overline{L}_a$ das

$$\overline{L}^a_{ij} = \begin{cases} c & \text{wenn } c \text{ multipliziert mit } j\text{-ten Zeile zur } i\text{-ten addiert wird} \\ 0 & \text{ansonsten} \end{cases} \tag{C.6}$$

und $L_a = \mathrm{Id} + \overline{L}_a$.

Betrachten wir nun, wie wir mit Hilfe der Matrizen L_a die LR Zerlegung von A erhalten können. Wenn wir Gauß-Elimination (zunächst ohne Pivoting) durch die Matrizen L_a ausdrücken, dann haben wir

$$L_k \cdots L_1 A = R. \tag{C.7a}$$

wobei R die obere Dreiecksmatrix ist, mit welcher das Gleichungssystem effizient mit Rückwertseinsetzen gelöst werden kann. Bezeichnen wir nun die akkumulierten Transformationen mit $L^{-1} = L_k \cdots L_1$ (die Bezeichnung L^{-1} ist unintuitiv aber Konvention), dann können wir die Matrix A, welche das ursprüngliche Gleichungssystem repräsentiert, als

$$L^{-1} A = R \quad \Longrightarrow \quad A = L\, R \tag{C.7b}$$

schreiben. Damit haben wir die LR Zerlegung von A erhalten. Um die Matrix L zu bestimmen, muss L^{-1} invertiert werden. Durch die spezielle Struktur von $L^{-1} = L_k \cdots L_1$ ist die Invertierung sehr einfach möglich. Betrachten wir dafür zunächst eine einzelne Matrix L_1, z.B.

$$L_1 = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{C.8}$$

welche, wie bereits besprochen, die Addition der ersten und zweiten Zeile der Matrix A darstellt. Die umgekehrte Operation ist die Subtraktion der ersten Zeile von der zweiten. Die entsprechende Matrix ist

$$L'_1 = \begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{C.9}$$

Für die Anwendung von L_1 gefolgt von L'_1 haben wir also

$$L'_1\, L_1\, A = A \tag{C.10}$$

da L_1 und L'_1 inverse Operationen darstellen. Aber dies bedeutet auch, dass

$$L'_1\, L_1 = \mathrm{Id} \quad \rightarrow \quad L'_1 = L_1^{-1}, \tag{C.11}$$

d.h. die Matrix L_1' ist die Inverse von L_1. Man kann natürlich auch per Hand leicht nachrechnen, dass $L_1' L_1 = I$ gilt. Mit anderen Worten: es gilt

$$L_a^{-1} = \mathrm{Id} - \overline{L_a}. \tag{C.12}$$

Mit den Inversen der L_a Matrizen können wir die gesuchte Matrix L bestimmen:

$$L = \left(L^{-1}\right)^{-1} = (L_k \cdots L_1)^{-1} \tag{C.13a}$$

und mit $(A\,B)^{-1} = B^{-1}A^{-1}$ erhalten wir

$$L = L_1^{-1} \cdots L_k^{-1}. \tag{C.13b}$$

Es muss beachtet werden, dass die Matrizen L_a^{-1} in der umgekehrten Reihenfolge multipliziert werden, wie die L_a Matrizen, d.h.

$$L_k \cdots L_1\, A = R \quad \Longrightarrow \quad A = L_1^{-1} \cdots L_k^{-1}\, R. \tag{C.14}$$

In der Praxis wird für eine Computerimplementierung folgende Formel zur Bestimmung von L verwendet:

$$L = \mathrm{Id} - \sum_{a=1}^{k} \overline{L_a}. \tag{C.15}$$

Diese zeigt auch explizit, dass sowohl das Produkt als auch die Inverse von unteren Dreiecksmatrizen wieder eine untere Dreiecksmatrize ist.

Beispiel Schauen wir uns noch einmal das einführende Beispiel von Aufgabe 1 an. Um die Matrix umzuformen, haben wir zunächst die erste Zeile auf die zweite und dritte addiert, und dann das doppelte der zweiten Zeile zur dritten addiert. Wir erhalten damit in Matrixform

$$L_3\, L_2\, L_1\, A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 2 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} A. \tag{C.16a}$$

Als nächstes müssen die Matrizen auf die rechte Seite gebracht werden. Damit erhalten wir

$$A = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}^{-1} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}^{-1} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 2 & 1 \end{pmatrix}^{-1} R. \tag{C.16b}$$

Wie im Vorausgehenden beschrieben sind die Matrizen L_a einfach zu invertieren, in dem wir das Vorzeichen des nicht-trivialen Matrixelementes umdrehen, was dem Wechsel zwischen Addition und Subtraktion einer Zeile entspricht. Damit erhalten wir

$$A = \begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -2 & 1 \end{pmatrix} R. \tag{C.16c}$$

Durch formelles Multiplizieren der L_a Matrizen, welches wie in Eq. C.40 durch Addieren der nicht-diagonalen Einträge möglich ist, erhält man

$$A = \begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ -1 & -2 & 1 \end{pmatrix} R \tag{C.16d}$$

welches die gesuchte Faktorisierung $A = LR$ ist. Explizit haben wir also für die Zerlegung von A:

$$\begin{pmatrix} 1 & 0 & -1 \\ -1 & -1 & 2 \\ -1 & 2 & -3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ -1 & -2 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & -1 \\ 0 & -1 & 1 \\ 0 & 0 & -2 \end{pmatrix} \tag{C.17}$$

wobei die Matrix R in der Praxis durch das Ausführen von Gauss-Elimination bestimmt wird und L durch das Akkumlieren der Transformationen.

Berechnung der Lösung eines Gleichungssystems mit Hilfe der LR Zerlegung Mithilfe der LR Zerlegung lässt sich ein Gleichungssystem leicht und effizient für beliebige rechte Seiten lösen. Durch Einsetzen der Faktorisierung erhalten wir zunächst:

$$A\,x = (LR)\,x = L\,R\,x = b. \tag{C.18}$$

Auf der linken Seite des Systems stehen jetzt zwei Matrizen, L und R, so dass keine direkte Lösung möglich ist. Zunächst lösen wir deshalb für einen temporären Vektor

$$y = R\,x \tag{C.19}$$

mit welchem Eq. C.43 ein reguläres Gleichungssystem

$$L\,y = b \tag{C.20}$$

ist. Darüberhinaus ist L eine untere Dreiecksmatrix, so dass das System effizient durch Vorwärtseinsetzen beginnend von der ersten Zeile gelöst werden kann. Ist y bestimmt so kann das Gleichungssystem in Gleichung C.44 gelöst werden, was auch wieder effizient möglich ist, da R eine obere Dreiecksmatrix ist, so dass Rückwärtseinsetzen verwendet werden kann. Die vollständige Lösung des Gleichungssystems erhalten wir als mit einmaligen Vorwärtseinsetzen zur Bestimmung des Zwischenergebnisses y in Gleichung C.45 und dann der Lösung nach x in Gleichung C.44 mit Rückwärtseinsetzen.

Bei der Spaltenpivotsuche werden die Zeilen von A neu angeordnet. Diese Vertauschung der Zeilen lässt sich formal durch die Multiplikation von links mit einer Permutationsmatrix beschreiben. Eine solche Matrix hat die Eigenschaft, dass in jeder Zeile und Spalte nur eine 1 auftritt und alle anderen Element in jeder Spalte und Zeile null sind. Zum Beispiel

$$P = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{C.21}$$

ist eine Permuationsmatrix. Wenn wir dies auf eine Matrix A anwenden dann erhalten wir

$$P\,A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} = \begin{pmatrix} A_{21} & A_{22} \\ A_{11} & A_{12} \end{pmatrix} \tag{C.22}$$

und wir sehen, dass P die zwei Zeilen von A vertauscht. Formell lässt sich eine Permutationsmatrix als eine Permutation der Einheitsvektoren $e_i = \delta_{ii}$ beschreiben, wenn diese als Spalten in einer Matrix angeordnet werden. Ist $\pi\colon \{1, \ldots, n\} \to \{1, \ldots, n\}$ eine Permutation von n Elementen, so ist die zugehörige Permutationsmatrix gegeben durch:

$$P_\pi = (e_{\pi(1)}, \ldots, e_{\pi(n)}) \tag{C.23}$$

Es sein nun P die Permutationsmatrix, welche die Vertauschungen bei der Durchführung des Gaußschen Eliminationsverfahrens beschreibt. Dann gilt: $PA = LR$. Um eine Lösung für $Ax = b$ können wir also $PAx = LRx = Pb$ mittels Vorwärts-/Rückwärtseinsetzen lösen:

$$Ly = Pb \qquad \text{(Vorwärtseinsetzen)} \tag{C.24a}$$
$$Rx = y \qquad \text{(Rückwärtseinsetzen)} \tag{C.24b}$$

Beispiel Gegeben sei die Matrix

$$A = \begin{pmatrix} -2 & 2 & -2 \\ -1 & -1 & 2 \\ 1 & 2 & -1 \end{pmatrix}. \tag{C.25}$$

Um die LR-Zerlegung mit Pivoting durchzuführen, starten wir mit den drei Matritzen $P = \mathrm{Id}, L = 0$ und $R = A$. Es ist zu beachten, dass die Matrix L beim Pivoting auch permutiert werden muss, da sich die hier gespeicherten Operationen ja auf die unpermutierte Matrix beziehen.

R	L	P	Beschreibung
$\begin{pmatrix} -2 & 2 & -2 \\ -1 & -1 & 1 \\ 1 & 3 & -1 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	Inititalisierung.
$\begin{pmatrix} -2 & 2 & -2 \\ 0 & -2 & 2 \\ 0 & 4 & -2 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0.5 & 0 & 0 \\ -0.5 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	Erste Spalte eliminieren.
$\begin{pmatrix} -2 & 2 & -2 \\ 0 & 4 & -2 \\ 0 & -2 & 2 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ -0.5 & 0 & 0 \\ 0.5 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$	Pivoting: Zeile 2 und 3 tauschen.
$\begin{pmatrix} -2 & 2 & -2 \\ 0 & 4 & -2 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ -0.5 & 1 & 0 \\ 0.5 & -0.5 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$	Zweite Spalte eliminieren.

Nachdem wir nun das Gaußsche Eliminationsverfahren mit Pivoting durchgeführt haben, erhalten wir die gewünschte Zerlegung $PA = LR$.

C.2. LR-Zerlegung

LR steht für „links-rechts“. Im englischen spricht man von der *LU-Decomposition* für „lower-upper“.

In der Praxis will man häufig ein Gleichungssystem $Ax = b$ für verschiedene Vektoren b lösen. Das Gauß-Verfahren, wie wir es bis jetzt betrachtet haben, erfordert jedoch für jeden Vektor b eine vollständig neue Lösung des Gleichungssystems. Die LR Zerlegung bietet hierfür eine Lösung und ermöglicht es, ein Gleichungssystem effizient für verschiedene “Eingabedaten” b zu lösen. Einen Teil der LR Zerlegung haben wir bereits kennengelernt: R steht für die obere Dreiecksmatrix, welche das Ergebnis der Gauß-Elimination ist. Was für eine bereits berechnete Matrix R fehlt, um diese für eine beliebige rechte Seite b verwenden zu können, sind die

Transformationen, Zeilenaddition und Zeilentausch, welche von der Eingabematrix A zu R geführt haben. Der zweite Teil der LR Zerlegung, die untere Dreiecksmatrix L, enthält diese Informationen. Damit haben wir $A = L\,R$ und L und R repräsentieren zusammen wieder das ursprüngliche lineare Gleichungssystem. Da L und R beide Dreiecksmatrizen sind, ist eine sehr effiziente Lösung des Gleichungssystems mit Vorwärts- und Rückwärtseinsetzen möglich. Betrachten wir nun näher, wie L konstruiert werden kann und wie damit eine effiziente Lösung eines Gleichungssystems möglich ist.

C.2.1. Einführendes Beispiel

Betrachten wir ein 2×2 Gleichungssystem, welche in Matrixform als

$$A \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} a & b \\ -a & c \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} e \\ f \end{pmatrix} \tag{C.26}$$

gegeben ist. Für die Gauß-Elimination müssen wir $-a$ in der zweiten Zeile durch die Addition der beiden Zeile eliminieren. Wenn wir das Produkt von A mit einer speziellen Matrix

$$L_1 = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} = \mathrm{Id} + \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \tag{C.27}$$

betrachten, so haben wir

$$L_1\,A = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} a & b \\ -a & c \end{pmatrix} = \begin{pmatrix} a & b \\ a-a & b+c \end{pmatrix} \tag{C.28}$$

Wir sehen also, dass $L_1\,A$ genau die gewünschten Operationen implementiert: die erste Zeile des Produkts ist wieder die erste Zeile von A und die zweite Zeile ist die Summe der Zeilen in A.

Wenn wir mehrere Zeilen-Transformationen haben, dann kann können diese als Produkt von mehreren L_a Matrizen ausgedrückt werden. Zum Beispiel, für ein 3×3 Gleichungssystem drücken

$$L_1 = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad L_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \tag{C.29}$$

die Addition der ersten mit der zweiten und der ersten mit der dritten Zeile aus. Das Produkt der Matrizen

$$L_2\,L_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \tag{C.30}$$

entspricht, entsprechend der Definition der Matrixmultiplikation, der aufeinanderfolgenden Umsetzung der beiden Transformationen. An den vorhergehenden Beispielen sehen wir auch, dass L_a die Summe aus der Einheitsmatrix und einer Matrix $\overline{L_a}$ ist, deren (i,j)-te Element 1 ist, wenn wir die j-te Zeile der Matrix zur i-ten addieren. Im Allgemeinen, wenn wir Vielfache von Zeilen addieren, gilt für das (i,j)-te Element der Matrix $\bar{L}_a$ das

$$\overline{L}^a_{ij} = \begin{cases} c & \text{wenn } c \text{ multipliziert mit } j\text{-ten Zeile zur } i\text{-ten addiert wird} \\ 0 & \text{ansonsten} \end{cases} \tag{C.31}$$

und $L_a = \mathrm{Id} + \overline{L}_a$.

Betrachten wir nun, wie wir mit Hilfe der Matrizen L_a die LR Zerlegung von A erhalten können. Wenn wir Gauß-Elimination (zunächst ohne Pivoting) durch die Matrizen L_a ausdrücken, dann haben wir

$$L_k \cdots L_1 A = R. \tag{C.32a}$$

wobei R die obere Dreiecksmatrix ist, mit welcher das Gleichungssystem effizient mit Rückwertseinsetzen gelöst werden kann. Bezeichnen wir nun die akkumulierten Transformationen mit $L^{-1} = L_k \cdots L_1$ (die Bezeichnung L^{-1} ist unintuitiv aber Konvention), dann können wir die Matrix A, welche das ursprüngliche Gleichungssystem repräsentiert, als

$$L^{-1}A = R \quad \Longrightarrow \quad A = L\,R \tag{C.32b}$$

schreiben. Damit haben wir die LR Zerlegung von A erhalten. Um die Matrix L zu bestimmen, muss L^{-1} invertiert werden. Durch die spezielle Struktur von $L^{-1} = L_k \cdots L_1$ ist die Invertierung sehr einfach möglich. Betrachten wir dafür zunächst eine einzelne Matrix L_1, z.B.

$$L_1 = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{C.33}$$

welche, wie bereits besprochen, die Addition der ersten und zweiten Zeile der Matrix A darstellt. Die umgekehrte Operation ist die Subtraktion der ersten Zeile von der zweiten. Die entsprechende Matrix ist

$$L_1' = \begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{C.34}$$

Für die Anwendung von L_1 gefolgt von L_1' haben wir also

$$L_1'\, L_1\, A = A \tag{C.35}$$

da L_1 und L_1' inverse Operationen darstellen. Aber dies bedeutet auch, dass

$$L_1'\, L_1 = \mathrm{Id} \quad \rightarrow \quad L_1' = L_1^{-1}, \tag{C.36}$$

d.h. die Matrix L_1' ist die Inverse von L_1. Man kann natürlich auch per Hand leicht nachrechnen, dass $L_1' L_1 = I$ gilt. Mit anderen Worten: es gilt

$$L_a^{-1} = \mathrm{Id} - \overline{L_a}. \tag{C.37}$$

Mit den Inversen der L_a Matrizen können wir die gesuchte Matrix L bestimmen:

$$L = \left(L^{-1}\right)^{-1} = (L_k \cdots L_1)^{-1} \tag{C.38a}$$

und mit $(A\,B)^{-1} = B^{-1}A^{-1}$ erhalten wir

$$L = L_1^{-1} \cdots L_k^{-1}. \tag{C.38b}$$

Es muss beachtet werden, dass die Matrizen L_a^{-1} in der umgekehrten Reihenfolge multipliziert werden, wie die L_a Matrizen, d.h.

$$L_k \cdots L_1\, A = R \quad \Longrightarrow \quad A = L_1^{-1} \cdots L_k^{-1}\, R. \tag{C.39}$$

In der Praxis wird für eine Computerimplementierung folgende Formel zur Bestimmung von L verwendet:

$$L = \mathrm{Id} - \sum_{a=1}^{k} \overline{L_a}. \tag{C.40}$$

Diese zeigt auch explizit, dass sowohl das Produkt als auch die Inverse von unteren Dreiecksmatrizen wieder eine untere Dreiecksmatrize ist.

Beispiel Schauen wir uns noch einmal das einführende Beispiel von Aufgabe 1 an. Um die Matrix umzuformen, haben wir zunächst die erste Zeile auf die zweite und dritte addiert, und dann das doppelte der zweiten Zeile zur dritten addiert. Wir erhalten damit in Matrixform

$$L_3\, L_2\, L_1\, A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 2 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} A. \tag{C.41a}$$

Als nächstes müssen die Matrizen auf die rechte Seite gebracht werden. Damit erhalten wir

$$A = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}^{-1} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}^{-1} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 2 & 1 \end{pmatrix}^{-1} R. \tag{C.41b}$$

Wie im Vorausgehenden beschrieben sind die Matrizen L_a einfach zu invertieren, in dem wir das Vorzeichen des nicht-trivialen Matrixelementes umdrehen, was dem Wechsel zwischen Addition und Subtraktion einer Zeile entspricht. Damit erhalten wir

$$A = \begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -2 & 1 \end{pmatrix} R. \tag{C.41c}$$

Durch formelles Multiplizieren der L_a Matrizen, welches wie in Eq. C.40 durch Addieren der nicht-diagonalen Einträge möglich ist, erhält man

$$A = \begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ -1 & -2 & 1 \end{pmatrix} R \tag{C.41d}$$

welches die gesuchte Faktorisierung $A = LR$ ist. Explizit haben wir also für die Zerlegung von A:

$$\begin{pmatrix} 1 & 0 & -1 \\ -1 & -1 & 2 \\ -1 & 2 & -3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ -1 & -2 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & -1 \\ 0 & -1 & 1 \\ 0 & 0 & -2 \end{pmatrix} \tag{C.42}$$

wobei die Matrix R in der Praxis durch das Ausführen von Gauss-Elimination bestimmt wird und L durch das Akkumulieren der Transformationen.

C.2.2. Lösen eines Gleichungssystems mit der LR-Zerlegung

Mithilfe der LR Zerlegung lässt sich ein Gleichungssystem leicht und effizient für beliebige rechte Seiten lösen. Wir betrachten zunächst den Fall ohne Pivoting. Durch Einsetzen der Faktorisierung erhalten wir zunächst:

$$A\,x = (LR)\,x = L\,R\,x = b. \tag{C.43}$$

Auf der linken Seite des Systems stehen jetzt zwei Matrizen, L und R, so dass keine direkte Lösung möglich ist. Zunächst lösen wir deshalb für einen temporären Vektor

$$y = R\,x \tag{C.44}$$

mit welchem Eq. C.43 ein reguläres Gleichungssystem

$$L\,y = b \tag{C.45}$$

ist. Darüber hinaus ist L eine untere Dreiecksmatrix, so dass das System effizient durch Vorwärtseinsetzen beginnend von der ersten Zeile gelöst werden kann. Ist y bestimmt so kann das Gleichungssystem in Gleichung C.44 gelöst werden, was auch wieder effizient möglich ist, da R eine obere Dreiecksmatrix ist, so dass Rückwärtseinsetzen verwendet werden kann. Die vollständige Lösung des Gleichungssystems erhalten wir als mit einmaligen Vorwärtseinsetzen zur Bestimmung des Zwischenergebnisses y in Gleichung C.45 und dann der Lösung nach x in Gleichung C.44 mit Rückwärtseinsetzen.

Betrachten wir nun den Fall mit Pivoting. Bei der Spaltenpivotsuche werden die Zeilen von A neu angeordnet. Diese Vertauschung der Zeilen lässt sich formal durch die Multiplikation von links mit einer Permutationsmatrix beschreiben. Eine solche Matrix hat die Eigenschaft, dass in jeder Zeile und Spalte nur eine 1 auftritt und alle anderen Element in jeder Spalte und Zeile null sind. Zum Beispiel

$$P = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{C.46}$$

ist eine Permuationsmatrix. Wenn wir dies auf eine Matrix A anwenden dann erhalten wir

$$P\,A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} = \begin{pmatrix} A_{21} & A_{22} \\ A_{11} & A_{12} \end{pmatrix} \tag{C.47}$$

und wir sehen, dass P die zwei Zeilen von A vertauscht. Formell lässt sich eine Permutationsmatrix als eine Permutation der Einheitsvektoren $e_i = \delta_{ii}$ beschreiben, wenn diese als Spalten in einer Matrix angeordnet werden. Ist $\pi\colon \{1,\ldots,n\} \to \{1,\ldots,n\}$ eine Permutation von n Elementen, so ist die zugehörige Permutationsmatrix gegeben durch:

$$P_\pi = (e_{\pi(1)}, \ldots, e_{\pi(n)}) \tag{C.48}$$

Es sein nun P die Permutationsmatrix, welche die Vertauschungen bei der Durchführung des Gaußschen Eliminationsverfahrens beschreibt. Dann gilt: $PA = LR$. Um eine Lösung für $Ax = b$ können wir also $PAx = LRx = Pb$ mittels Vorwärts-/Rückwärtseinsetzen lösen:

$$Ly = Pb \qquad \text{(Vorwärtseinsetzen)} \tag{C.49a}$$

$$Rx = y \qquad \text{(Rückwärtseinsetzen)} \tag{C.49b}$$

Beispiel Gegeben sei die Matrix

$$A = \begin{pmatrix} -2 & 2 & -2 \\ -1 & -1 & 2 \\ 1 & 2 & -1 \end{pmatrix}. \tag{C.50}$$

Um die LR-Zerlegung mit Pivoting durchzuführen, starten wir mit den drei Matritzen $P = \mathrm{Id}, L = 0$ und $R = A$. Es ist zu beachten, dass die Matrix L beim Pivoting auch permutiert werden muss, da sich die hier gespeicherten Operationen ja auf die unpermutierte Matrix beziehen.

R	L	P	Beschreibung
$\begin{pmatrix} -2 & 2 & -2 \\ -1 & -1 & 1 \\ 1 & 3 & -1 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	Inititalisierung.
$\begin{pmatrix} -2 & 2 & -2 \\ 0 & -2 & 2 \\ 0 & 4 & -2 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0.5 & 0 & 0 \\ -0.5 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	Erste Spalte eliminieren.
$\begin{pmatrix} -2 & 2 & -2 \\ 0 & 4 & -2 \\ 0 & -2 & 2 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ -0.5 & 0 & 0 \\ 0.5 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$	Pivoting: Zeile 2 und 3 tauschen.
$\begin{pmatrix} -2 & 2 & -2 \\ 0 & 4 & -2 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ -0.5 & 1 & 0 \\ 0.5 & -0.5 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$	Zweite Spalte eliminieren.

Nachdem wir nun das Gaußsche Eliminationsverfahren mit Pivoting durchgeführt haben, erhalten wir die gewünschte Zerlegung $PA = LR$.

C.2.3. Herleitung der LR-Zerlegung ohne Pivoting

Im Folgenden wollen wir die Gaußsche Eliminationsmethode mittels geeigneter Matrizen beschreiben. Als Ergebnis erhalten wir, dass sich eine Matrix A, für welche die Gaußsche Eliminationsmethode durchführbar ist, als Produkt einer unteren Dreiecksmatrix L und einer oberen Dreiecksmatrix R schreiben lässt.

Für den Schritt von $k \to k+1$ muss die k-te Zeile von $A^{(k)}$ mit dem Faktor l_{ik} multipliziert und dann zur i-te Zeile von $A^{(k)}$ addiert werden. Dies ergibt:

$$A^{(k+1)} = \begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & \cdots & \cdots & a_{1n}^{(1)} \\ & a_{21}^{(2)} & \cdots & \cdots & \cdots & a_{2n}^{(2)} \\ & & \ddots & & & \vdots \\ & & & a_{kk}^{(k)} & \cdots & a_{kn}^{(k)} \\ & & & a_{k+1,k}^{(k)} - l_{k+1,k}\, a_{kk}^{(k)} & \cdots & a_{k+1,n}^{(k)} - l_{k+1,k}\, a_{k,n}^{(k)} \\ & & & & \vdots & \\ & & & a_{nk}^{(k)} - l_{nk}\, a_{kk}^{(k)} & \cdots & a_{nn}^{(k)} - l_{nk}\, a_{kn}^{(k)} \end{pmatrix} \tag{C.51}$$

$$A^{(k+1)} = \begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & \cdots & \cdots & a_{1n}^{(1)} \\ & a_{21}^{(2)} & \cdots & \cdots & \cdots & a_{2n}^{(2)} \\ & & \ddots & & & \vdots \\ & & & a_{kk}^{(k)} & \cdots & a_{kn}^{(k)} \\ & & & a_{k+1,k}^{(k)} & \cdots & a_{k+1,n}^{(k)} \\ & & & \vdots & & \vdots \\ & & & a_{nk}^{(k)} & \cdots & a_{nn}^{(k)} \end{pmatrix} - \begin{pmatrix} 0 & 0 & \cdots & \cdots & \cdots & 0 \\ & 0 & \cdots & \cdots & \cdots & 0 \\ & & \ddots & & & \vdots \\ & & & 0 & \cdots & 0 \\ & & & l_{k+1,k}\, a_{kk}^{(k)} & \cdots & l_{k+1,k}\, a_{kn}^{(k)} \\ & & & \vdots & & \vdots \\ & & & l_{nk}\, a_{kk}^{(k)} & \cdots & l_{nk}\, a_{kn}^{(k)} \end{pmatrix} \tag{C.52}$$

$$= A^{(k)} - \underbrace{\begin{pmatrix} 0 \\ \vdots \\ 0 \\ l_{k+1,k} \\ \vdots \\ l_{n,k} \end{pmatrix}}_{=\tau_k} \underbrace{\left(0, \ldots, 0, a_{kk}^{(k)}, \ldots, a_{kn}^{(k)}\right)}_{=e_k^T A^{(k)}} \tag{C.53}$$

$$= A^{(k)} - \tau_k\, e_k^T A^{(k)} = \left(I_n - \tau_k\, e_k^T\right) A^{(k)} \tag{C.54}$$

Definieren wir nun

$$L_k = I_n - \tau_k\, e_k^T = \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & -l_{k+1,k} & 1 & \\ & & \vdots & & \ddots \\ & & -l_{nk} & & 1 \end{pmatrix}, \tag{C.55}$$

so erhalten wir also für einen Schritt der Gaußschen Eliminationsmethode die folgende kompakte Darstellung:

$$A^{(k+1)} = L_k\, A^{(k)} \tag{C.56}$$

Zusammengefasst erhalten wir für die von der Gaußschen Eliminationsmethode bestimmte untere Dreiecksmatrix:

$$R = A^{(n)} = L_{n-1}\, A^{(n-1)} = L_{n-1}\, L_{n-2} A^{(n-2)} = \ldots = L_{n-1} \cdots L_1 A^{(1)} = L_{n-1} \cdots L_1 A \tag{C.57}$$

Bringen wir nun die Faktoren $L_{n-1} \cdots L_1$ durch Invertieren auf die linke Seite und fassen wir die Faktoren als

$$L = \left(L_{n-1} \cdots L_1\right)^{-1} = L_1^{-1} \cdots L_{n-1}^{-1} \tag{C.58}$$

zusammen, so erhalten wir eine Zerlegung der Matrix A in das Produkt der Matrizen L und R:

$$A = LR \tag{C.59}$$

Eine explizite Darstellung der Koeffizienten von L ist gegeben durch:.

$$L = I_n + \sum_{k=1}^{n} \tau_k \, e_k^T = \begin{pmatrix} 1 & & & \\ l_{2,1} & \ddots & & \\ \vdots & & \ddots & \\ l_{n,1} & \dots & l_{n,n-1} & 1 \end{pmatrix} \tag{C.60}$$

Diese ergibt sich aus den beiden folgenden Eigenschaften der L_k:

$$L_k^{-1} = I_n + \tau_k e_k^T \tag{C.61}$$

$$\prod_{k=1}^{n} L_k^{-1} = I_n + \sum_{k=1}^{n} \tau_k \, e_k^T \tag{C.62}$$

Die erste Aussage lässt sich leicht durch Nachrechnen verifizieren:

$$\begin{aligned} L_k L_k^{-1} &= \left(I_n - \tau_k \, e_k^T\right)\left(I_n + \tau_k \, e_k^T\right) \\ &= I_n I_n + I_n \, \tau_k \, e_k^T - \tau_k \, e_k^T I_n - \tau_k \underbrace{e_k^T \tau_k}_{=0} e_k^T = I_n \end{aligned} \tag{C.63}$$

Für den Beweis der zweite Aussage betrachten wir $\prod_{k=1}^{m} (I + \tau_k e_k^T) = I_n + \sum_{k=1}^{m} \tau_k \, e_k^T$. Für $m = 1$ ist dies offensichtlich richtig. Nehmen wir nun an die Aussage sei für $m \geq 1$ richtig, dann gilt für $m + 1$:

$$\begin{aligned} \prod_{k=1}^{m+1} \left(I_n + \tau_k \, e_k^T\right) &= \prod_{k=1}^{m} \left(I_n + \tau_k \, e_k^T\right)\left(I_n + \tau_{m+1} \, e_{m+1}^T\right) \\ &= \left(I_n + \sum_{k=1}^{m} \tau_k \, e_k^T\right)\left(I_n + \tau_{m+1} \, e_{m+1}^T\right) \\ &= I_n I_n + I_n \, \tau_{m+1} \, e_{m+1}^T + \sum_{k=1}^{m} \tau_k \, e_k^T \, I_n + \sum_{k=1}^{m} \tau_k \underbrace{e_k^T \tau_{m+1}}_{=0} e_{m+1}^T \\ &= I_n + \sum_{k=1}^{m+1} \tau_k \, e_k^T \end{aligned} \tag{C.64}$$

C.2.4. Herleitung der LR-Zerlegung mit Pivoting

TBD