Einführung in die Numerik

Jakob Zech

Universität Heidelberg

Version 17. Oktober 2023

Diese Vorlesungsunterlagen basieren hauptsächlich auf dem Skript "Numerische Mathematik 1" von Christoph Schwab (ETH Zürich)[1]. Einige Abschnitte wurden zudem aus den Skripten von Peter Bastian, Rolf Rannacher und Robert Scheichl (Universität Heidelberg) adaptiert.

Das Skript ist für eine einsemestrige Einführung in die numerische Mathematik konzipiert. Zum Verständnis werden Kenntnisse aus Grundvorlesungen in Linearer Algebra sowie in Analysis vorausgesetzt. Nicht prüfungsrelevante Abschnitte sind mit einem * gekennzeichnet. Das Skript wird während des Semesters laufend erweitert.

Anhang A, der als Referenzmaterial dient, wiederholt grundlegende Konzepte der linearen Algebra, die im Haupttext häufig referenziert werden. Er legt zudem die Konventionen für Matrix- und Vektornotation fest. Darüber hinaus enthält der Anhang Details zur Implementierung von Matrixoperationen in Python, speziell unter Verwendung der NumPy-Bibliothek. Dementsprechend ähnelt die Darstellung in Anhang A eher einer Formelsammlung, als einem Unterrichtsskript: Resultate werden nicht immer bewiesen, es werden aber Referenzen zum Nachlesen angegeben. Beispielsweise wird die in der numerischen Linearen Algebra enorm wichtige Singulärwertzerlegung als Anwendung der Matrix- und Vektornormen hergeleitet. Ohne Beweis wird dagegen die (für die Numerik weniger wichtige) Jordan'sche Normalform angegeben.

[1] Daher folgen wir auch der Schweizer Schreibweise, die insbesondere kein scharfes s verwendet.

Inhaltsverzeichnis

1 Computerarithmetik **3**
1.1 Zahldarstellungen und Gleitkommazahlen 5
1.2 Gleitkommazahlsysteme 6
1.3 Runden 9
1.4 Rundungsfehleranalyse 12
1.5 Auslöschung 14
1.6 Landau-Symbole (Wiederholung) 16
1.7 Bibliographische Bemerkungen 17

A Lineare Algebra **18**
A.1 Vektorräume 18
A.2 Matrizen 21
A.3 Matrixoperationen 22
A.4 Spezielle Matrizen 25
A.5 Eigenwerte und Eigenvektoren 27
A.6 Vektornormen 29
A.7 Matrixnormen 33
A.8 Orthogonalität 36
A.9 Singulärwertzerlegung (SVD) 41
A.10 Jordan- und Schur-Normalform* 44

Kapitel 1

Computerarithmetik

> Since none of the numbers which we take out from logarithmic and trigonometric tables admit of absolute precision, but all are to a certain extent approximate only, the results of all calculations performed by the aid of these numbers can only be approximately true.
> It may happen, that in special cases the effect of the errors of the tables is so augmented that we may be obliged to reject a method, otherwise the best, and substitute another in its place.
> — Carl F. Gauss, Theoria Motus (1809)[1]

> Matlab's creator Dr. Cleve Moler used to advise foreign visitors not to miss the country's two most awesome spectacles: the Grand Canyon, and meetings of IEEE.
> — William M. Kahan[2]

Trotz des massiven technologischen Fortschritts des letzten Jahrhunderts hat das Zitat von Gauss nichts an Bedeutung verloren. Nicht nur Logarithmentabellen sondern auch Computer haben endliche Speicherkapazitäten. Eine reelle Zahl lässt sich im allgemeinen nicht beliebig genau im Computer darstellen. Dadurch ergeben sich Rundungsfehler, die sich im Verlauf einer Rechnung verstärken können und – im schlimmsten Fall – das gewünschte Ergebnis bis zur Unkenntlichkeit verfälschen können.

Beispiel 1.1. Die folgende Darstellung der Euler'schen Zahl e ist aus der Analysis bekannt:

$$e = \lim_{n\to\infty} \left(1 + \frac{1}{n}\right)^n.$$

Man erwartet also, dass der Ausdruck $e_n = \left(1 + \frac{1}{n}\right)^n$ für grösser werdendes n immer bessere Approximationen von e liefert. Dies wäre in exakter Arithmetik auch tatsächlich zu beobachten. Für den auf dem Computer (also in inexakter Arithmetik) berechneten Wert $\hat{e}_n$ ist dies allerdings nicht mehr der Fall:

[1] Translated and quoted in Higham (2002).
[2] See http://www.cs.berkeley.edu/~wkahan/ieee754status/754story.html.

```python
import numpy as np

for i in range(1,16):
  n = 10**i
  en = (1+1/n)**n
  print(n,en,en-np.exp(1))
```

n	Berechnetes $\hat{e}_n$	Fehler $\hat{e}_n - e$
10^1	2.5937424601000023	-0.1245393683590428
10^2	2.7048138294215285	-0.0134679990375166
10^3	2.7169239322355936	-0.0013578962234515
10^4	2.7181459268249255	-0.0001359016341196
10^5	2.7182682371922975	-0.0000135912667476
10^6	2.7182804690957534	-0.0000013593632917
10^7	2.7182816941320818	-0.0000001343269633
10^8	2.7182817983473577	-0.0000000301116874
10^9	2.7182820520115603	0.0000002235525152
10^{10}	2.7182820532347876	0.0000002247757425
10^{11}	2.7182820533571102	0.0000002248980651
10^{12}	2.7185234960372378	0.0002416675781927
10^{13}	2.7161100340869009	-0.0021717943721442
10^{14}	2.7161100340870230	-0.0021717943720221
10^{15}	3.0350352065492618	0.3167533780902168

Ab $n = 10^8$ nimmt die Genauigkeit sogar wieder ab, bis bei $n = 10^{15}$ nicht eine Dezimalstelle von e richtig berechnet wird. ◇

Beispiel 1.2. Die Potenzreihe der Exponentialfunktion konvergiert für jedes $x \in \mathbb{R}$

$$e^x = \sum_{k=0}^{\infty} \frac{x^k}{k!} = 1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{24} + \dots.$$

Auf dem Rechner können wir natürlich nur eine Partialsumme

$$s_i(x) = \sum_{k=0}^{i} \frac{x^k}{k!}$$

berechnen. Aufgrund der Restglieddarstellung der Taylor-Entwicklung gibt es ein ξ mit $0 < |\xi| < |x|$, so dass

$$e^x - s_i(x) = \frac{e^\xi x^{i+1}}{(i+1)!}.$$

Wählen wir also als Abbruchkriterium $|x|^{i+1}/(i+1)! \le \texttt{tol} \cdot s_i(x)$ für eine (kleine) vorgegebene Toleranz `tol`, so folgt für negative x

$$|e^x - s_i(x)| \le \frac{|x|^{i+1}}{(i+1)!} \le \texttt{tol} \cdot s_i(x) \approx \texttt{tol} \cdot e^x.$$

Also ist der *relative Fehler* $|e^x - s_i(x)|/|e^x|$ näherungsweise durch `tol` beschränkt. Für positive x findet man eine ähnliche Schranke über eine genauere Analyse des Restglieds (Übung).

```python
def expeval(x,tol):
    s, k, term = 1, 1, 1
    while (abs(term)>tol*abs(s)):
        term = term*x/k
        s = s + term
        k = k + 1
    return s
```

Tabelle 1.1 listet die Ausgabe der Python-Funktion für `tol` $= 10^{-8}$. Im Widerspruch zu den obigen theoretischen Aussagen ist der relative Fehler für negative x in Computerarithmetik *nicht* durch `tol` beschränkt. ◇

x	Berechnetes $\hat{s}_i(x)$	$\exp(x)$	$\frac{\lvert \exp(x)-\hat{s}_i(x)\rvert}{\exp(x)}$
-20	5.6218844674e-09	2.0611536224e-09	1.7275426762011810
-18	1.5385415977e-08	1.5229979745e-08	0.0102059381875637
-16	1.1254180496e-07	1.1253517472e-07	0.0000589170202573
-14	8.3152907681e-07	8.3152871910e-07	0.0000004301769560
-12	6.1442133148e-06	6.1442123533e-06	0.0000001564807371
-10	4.5399929556e-05	4.5399929762e-05	0.0000000045444145
-8	3.3546262817e-04	3.3546262790e-04	0.0000000007889018
-6	2.4787521758e-03	2.4787521767e-03	0.0000000003333061
-4	1.8315638879e-02	1.8315638889e-02	0.0000000005306940
-2	1.3533528320e-01	1.3533528324e-01	0.0000000002736029
0	1.0000000000e+00	1.0000000000e+00	0.0000000000000000
2	7.3890560954e+00	7.3890560989e+00	0.0000000004799685
4	5.4598149928e+01	5.4598150033e+01	0.0000000019230582
6	4.0342879295e+02	4.0342879349e+02	0.0000000013442484
8	2.9809579808e+03	2.9809579870e+03	0.0000000021025838
10	2.2026465748e+04	2.2026465795e+04	0.0000000021437996
12	1.6275479114e+05	1.6275479142e+05	0.0000000017238445
14	1.2026042798e+06	1.2026042842e+06	0.0000000036341346
16	8.8861105010e+06	8.8861105205e+06	0.0000000021979895
18	6.5659968911e+07	6.5659969137e+07	0.0000000034509716
20	4.8516519307e+08	4.8516519541e+08	0.0000000048287376

Tabelle 1.1: Numerischer Fehler bei der Auswertung der Potenzreihe für e^x.

Ziel dieses Kapitels ist, die in den Beispielen 1.1 und 1.2 auftretenden Phänomene zu verstehen und vermeiden zu lernen. Dazu ist es notwendig, zunächst einmal zu klären wie eigentlich reelle Zahlen im Computer dargestellt bzw. approximiert werden.

1.1 Zahldarstellungen und Gleitkommazahlen

Die Grundlage für Zahlendarstellungen auf Rechnern bildet der folgende Satz aus der Analysis über die Darstellung reeller Zahlen in einer **Basis** $\beta \in \mathbb{N}$, $\beta \geq 2$, den wir ohne Beweis angeben.

Satz 1.3. *Jede reelle von Null verschiedene Zahl $x \in \mathbb{R}$ lässt sich in der Form*

$$x = \pm\beta^e \left(\frac{d_1}{\beta} + \frac{d_2}{\beta^2} + \frac{d_3}{\beta^3} + \cdots \right) \tag{1.1.1}$$

darstellen; mit der **Basis** $2 \leq \beta \in \mathbb{N}$, *den* **Ziffern** *(engl. "digits")* $d_1, d_2, \ldots, \in \{0, 1, 2, \ldots, \beta - 1\}$, *wobei* $d_1 \neq 0$, *und dem* **Exponenten** $e \in \mathbb{Z}$.

Die sogenannte "wissenschaftliche Darstellung" reeller Zahlen (1.1.1) wird eindeutig, wenn man zusätzlich noch fordert, dass eine unendliche Teilmenge $\mathbb{N}_1 \subset \mathbb{N}$ mit $d_k \neq \beta - 1$ für alle $k \in \mathbb{N}_1$ existiert. Diese Forderung ist notwendig, da ansonsten $x = +10^1 \cdot 0.1$ und $x = +10^0 \cdot 0.999\ldots$ zwei verschiedene Dezimaldarstellungen der gleichen reellen Zahl wären. Da wir aber auf dem Rechner sowieso nur mit einer endlichen Anzahl von Ziffern arbeiten können, ist dieser Aspekt für uns nicht weiter von Interesse.

Beispiel 1.4. (Basen)

System	β	Zifferndarstellung
Dezimal	10	$0, 1, 2, \ldots, 9$
Binär	2	$0, 1$
Oktal	8	0,1,2,...,7
Hexadezimal	16	$0, 1, 2, \ldots, 9$, A,B,C,D,E,F

Finger oder Fäuste* Das im Alltag verbreiteste Zahlensystem ist das Dezimalsystem ($\beta = 10$). Allerdings gibt es Ausnahmen, so benutzen etwa einige indigene Völker zum Zählen auch die Zehen ($\beta = 20$). Weitere in einigen Regionen der Welt gebräuchliche Basen sind $4, 8, 12$ sowie 16, siehe http://de.wikipedia.org/wiki/Zahlensystem. In der Computertechnik hat sich aus naheliegenden Gründen das Binärsystem $\beta = 2$ durchgesetzt. (Bei der *Darstellung* wird zur Übersichtlichkeit oft auf das Hexadezimalsystem zurückgegriffen.) Dem Ende der 1950-iger Jahre entwickelten auf dem Ternärsystem ($\beta = 3$) beruhenden Rechner der Moskauer Staatsuniversität war dagegen nur kurzer Erfolg beschieden.
Das Binärsystem wartet mit einigen unerwarteten Effekten auf. So ist etwa die Dezimalzahl 0.1 *nicht* durch einen endlichen Binärbruch darstellbar. In der Finanzwirtschaft setzt man bei Spezialanwendungen deswegen gelegentlich das Dezimalsystem ein, welches aber meist softwareseitig emuliert wird und dementsprechend langsam ist. Prozessoren, deren Hardware Dezimaloperationen unterstützt, sind selten; ein Beispiel ist der IBM Power6 Prozessor.

1.2 Gleitkommazahlsysteme

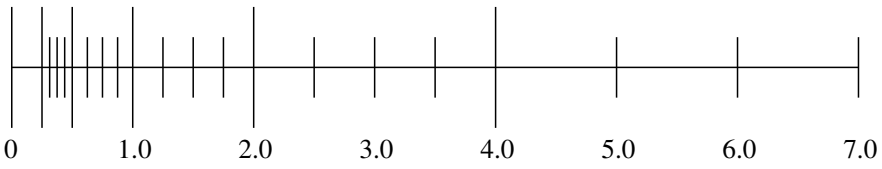
Auf Digitalrechnern können nur endlich viele Zahlen $\mathbb{F} \subset \mathbb{R}$ gespeichert und verarbeitet werden. Abgesehen von Spezialanwendungen (Bsp: Fixpunktzahlen in der Digitaltechnik) haben sich die Gleitkommazahlen durchgesetzt.[3]

Definition 1.5 (Gleitkommazahlen $\mathbb{F}(\beta, t, e_{\min}, e_{\max})$)**.** Sei $\beta \in \mathbb{N}$, $\beta \geq 2$ (**Basis**) und $t \in \mathbb{N}$ (**Mantissenlänge**), sowie $e_{\min} < 0 < e_{\max}$ mit $e_{\min}$, $e_{\max} \in \mathbb{Z}$ (**Exponentenschranken**). Dann ist die Menge $\mathbb{F} = \mathbb{F}(\beta, t, e_{\min}, e_{\max})$ wie folgt definiert:

$$\mathbb{F} := \left\{ \pm\beta^e \left(\frac{d_1}{\beta} + \frac{d_2}{\beta^2} + \cdots + \frac{d_t}{\beta^t} \right) : \begin{array}{l} d_1, \ldots, d_t \in \{0, \ldots, \beta - 1\}, \\ d_1 \neq 0, \\ e \in \mathbb{Z},\ e_{\min} \leq e \leq e_{\max}. \end{array} \right\} \cup \{0\}.$$

Offensichtlich ist $\mathbb{F}$ eine endliche Teilmenge von $\mathbb{Q}$ (und damit von $\mathbb{R}$). Es ist wichtig zu bemerken, dass die Elemente von $\mathbb{F}$ nicht gleichmässig verteilt sind.

Beispiel 1.6. Nichtnegative Gleitkommazahlen für $\beta = 2, t = 3, e_{\min} = -1, e_{\max} = 3$:



[3]Engl. "Floating Point Numbers". Das Symbol $\mathbb{F}$ ist an die englische Bezeichnung angelehnt.

Bemerkung 1.7. Alternativ könnte man mit **Festkommazahlen** arbeiten, d. h.

$$\left\{ \pm \sum_{i=e_{\min}}^{e_{\max}} d_i \beta^i : d_1, \ldots, d_t \in \{0, \ldots, \beta - 1\} \right\}.$$

Für wissenschaftliche Anwendungen ist dies aufgrund der sehr unterschiedlichen Größen vieler Zahlen in der Praxis jedoch ineffizient und wenig brauchbar:

- Ruhemasse eines Elektrons $= 0.9109384 \times 10^{-30}$ kg
- Lichtgeschwindigkeit $= 0.2997925 \times 10^{9}$ $^{\mathrm{m}}/_{\mathrm{s}}$
- Avogadrokonstante $= 0.6021415 \times 10^{24} {}^{1}/_{\mathrm{mol}}$

Um alle drei Zahlen mit ähnlicher Genauigkeit wie in der üblichen Fließkommadarstellung (z. B. "double precision" mit Basis $\beta = 2$, siehe unten) darzustellen, müsste man wesentlich mehr Werte (Ziffern) speichern.

Lemma 1.8. *In* $\mathbb{F} = \mathbb{F}(\beta, t, e_{\min}, e_{\max})$ *gilt*

$$x_{\min}(\mathbb{F}) := \min\{x \in \mathbb{F} : x > 0\} = \beta^{e_{\min}-1}, \tag{1.2.1}$$
$$x_{\max}(\mathbb{F}) := \max\{x \in \mathbb{F}\} \qquad = \beta^{e_{\max}}(1 - \beta^{-t}). \tag{1.2.2}$$

Beweis. Betrachtet man Definition 1.5, so besteht die einzige Schwierigkeit im Beweis von (1.2.1)–(1.2.2) darin, den minimalen and maximal möglichen Wert der Mantisse $m = \sum_{k=1}^{t} d_k \beta^{-k}$ zu bestimmen. Es gilt wegen $d_1 \neq 0$

$$\beta^{-1} \leq m \leq \sum_{k=1}^{t} \beta^{-k}(\beta - 1) = 1 - \beta^{-t},$$

und beide Werte können angenommen werden. □

Lemma 1.9. *Der Abstand zwischen einer Gleitkommazahl* $x \neq 0$ *und der nächstgelegenen Gleitkommazahl ist* mindestens $\beta^{-1}\epsilon_M|x|$ *und* höchstens $\epsilon_M|x|$*, wobei* $\epsilon_M = \beta^{1-t}$ *der Abstand von* 1 *zur nächstgrösseren Gleitkommazahl ist.*

Beweis. Übung. □

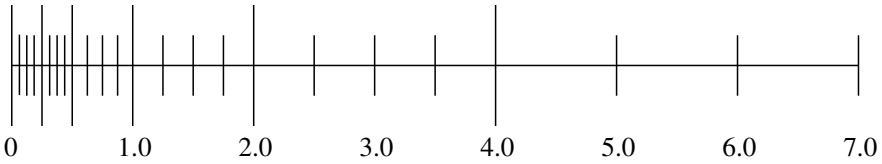
Aus Lemma 1.9 folgt, dass für eine feste Basis β die relative Genauigkeit von $\mathbb{F}$ lediglich von der Mantissenlänge t abhängt: $e_{\min}$*,* $e_{\max}$ *bestimmen nur den Bereich von* $\mathbb{F}$*, nicht aber die Genauigkeit von* $\mathbb{F}$*.*

Zwischen 0 und der kleinsten nichtnegativen Gleitkommazahl ist eine "Lücke" von $\beta^{e_{\min}-1}$, die durch Hinzufügen der sogenannten **denormalisierten Zahlen** geschlossen werden kann:

$$\widehat{\mathbb{F}} := \mathbb{F} \cup \left\{ \pm\beta^{e_{\min}} \left(\frac{d_2}{\beta^2} + \cdots + \frac{d_t}{\beta^t} \right) : \ d_2, \ldots, d_t \in \{0, \ldots, \beta - 1\} \right\}, \tag{1.2.3}$$

wobei noch der Fall $d_2 = \cdots = d_t = 0$ ausgeschlossen ist. In der Notation von Definition 1.5 werden also noch Zahlen hinzugefügt, die minimalen Exponenten $e_{\min}$ und $d_1 = 0$ haben. Gleitkommazahlen in $\mathbb{F}$ heissen **normalisiert** und in $\widehat{\mathbb{F}} \setminus \mathbb{F}$ heissen **denormalisiert**.

Beispiel 1.10. $\widehat{\mathbb{F}}$ für $\beta = 2, t = 3, e_{\min} = -1, e_{\max} = 3$:



Es ist zu beachten, dass für denormalisierte Zahlen das Lemma 1.9 *nicht* gilt; der denormalisierte Bereich hat niedrigere relative Genauigkeit.

Die im IEEE 754(r)-Standard festgelegte Gleitkommaarithmetik ist am weitesten verbreitet. Abgesehen vom oben beschriebenen Gleitkommaformat legt sie auch Regeln für Operationen auf $\mathbb{F}$ fest. Ausserdem gibt sie eine einheitliche Richtlinie zur Behandlung von Ausnahmen ($1/0, \infty \cdot \infty, \infty - \infty$) vor. Die zwei im IEEE-Standard von 1995 festgelegten Grundformate zur Basis $\beta = 2$ sind:

Typ	Grösse	Mantisse	Exponent	$x_{\min}$	$x_{\max}$
Single	32 bits	$23 + 1$ bit	8 bits	10^{-38}	10^{+38}
Double	64 bits	$52 + 1$ bit	11 bits	10^{-308}	10^{+308}

Dabei wird ein Bit der Mantisse für das Vorzeichen verwendet. Auf der anderen Seite wird d_1 *nicht* gespeichert, da im normalisierten Bereich bei $\beta = 2$ immer $d_1 = 1$ gilt. Also stehen effektiv $t = 24$ bzw. $t = 53$ Stellen für die Mantisse zur Verfügung. Doppelte Genauigkeit ist Standard auf den meisten verbreiteten Hauptprozessoren (CPU). Auf dem cell processor (Playstation 3) sowie Graphikkarten findet man hingegen nur einfache Genauigkeit. Manche Prozessoren rechnen intern im Register mit 80 bit, was bei hintereinander ausgeführten Operationen zu höherer Genauigkeit führen kann, aber auch zu recht unschönen Phänomenen (Heisenberg bug) da die Registerbelegung in höheren Programmiersprachen nicht kontrollierbar ist.

Beispiel 1.11 (Detailierte Darstellung im Rechner*)**.** Als Beispiel betrachten wir die Darstellung der Dezimalzahl $x = 6.5$ in doppelter Genauigkeit. Zunächst einmal die Binärdarstellung von x:

$$\begin{aligned} x &= (-1)^0 \cdot \left(\frac{1}{2} + \frac{1}{4} + \frac{1}{16} \right) 2^3 \\ &= (-1)^0 \cdot \left(\frac{1}{2} + \frac{1}{4} + \frac{1}{16} \right) 2^{1025-1022}. \end{aligned}$$

Die Verschiebung im Exponenten wird durchgeführt, um das Vorzeichen im Exponenten einzusparen. Abgespeichert wird das Vorzeichen, der verschobene Exponent 1025, sowie die Mantisse ohne d_1 im Binärformat:

$$0 \quad 100\ 0000\ 0001 \quad 1010\ 0000\ 0000\ \ldots\ 0000\ 0000\ 0000.$$

Python verwendet standardmäßig den Datentyp `int` für ganze Zahlen und `float` (entspricht 64-Bit doppelte Genauigkeit) für nicht-ganzzahlige numerische Werte. Bei Rechenoperationen, die sowohl `int` als auch `float` involvieren, wird der `int`-Wert automatisch zu `float` konvertiert, um die Operation in doppelter Genauigkeit abzuschliessen.

```
import numpy as np

print(np.finfo(float).tiny)  # 2.2250738585072014e-308
print(np.finfo(float).max)   # 1.7976931348623157e+308
print(1/0)                   # ZeroDivisionError: division by zero
print(0/0)                   # ZeroDivisionError: division by zero
print(3*np.inf)              # inf
print(np.inf - np.inf)       # nan
```

1.3 Runden

Eine reelle Zahl kann in $\mathbb{F}$ im allgemeinen nicht exakt dargestellt werden. Die Abbildung

$$\mathsf{rd} : \text{Bereich}(\mathbb{F}) \to \mathbb{F}$$

heisst **Runden** und ordnet der reellen Zahl $x \in \text{Bereich}(\mathbb{F})$ *eine* (nicht notwendig eindeutige) nächstgelegene Gleitkommazahl $\mathsf{rd}(x)$ in $\mathbb{F}$ zu. Dabei bezeichnet $\text{Bereich}(\mathbb{F})$ die Menge aller $x \in \mathbb{R}$, deren Betrag zwischen $x_{\min}(\mathbb{F})$ und $x_{\max}(\mathbb{F})$ (siehe Lemma 1.8) liegt:

$$\text{Bereich}(\mathbb{F}) := \{x \in \mathbb{R} : x_{\min}(\mathbb{F}) \le |x| \le x_{\max}(\mathbb{F})\} \cup \{0\}.$$

Die Abbildung rd ist nicht eindeutig bestimmt, wenn x genau in die Mitte zwischen zwei Gleitkommazahlen fällt. Es existieren verschiedene "tie breaking"-Strategien, um die Abbildung in solchen (bei $\beta = 2$ nicht seltenen) Fällen eindeutig zu machen. IEEE 754 rundet zu dem Wert, bei dem die letzte Ziffer der Mantisse gerade (bei $\beta = 2$ also Null) ist. Das folgende Beispiel verdeutlicht diese Konvention.

Beispiel 1.12. Sei $\beta = 10$, $t = 3$. Dann sind $\mathsf{rd}(0.9996) = 1.0$, $\mathsf{rd}(0.3345) = 0.334$, $\mathsf{rd}(0.3355) = 0.336$. ◇

Der folgende Satz gibt eine nützliche Abschätzung für den relativen Fehler von rd.

Satz 1.13. *Für alle $x \in \text{Bereich}(\mathbb{F})$ existiert (ein von x abhängiges) $\delta \in \mathbb{R}$ derart, so dass*

$$\mathsf{rd}(x) = x(1+\delta), \quad |\delta| \le u,$$

wobei $u = u(\mathbb{F}) := \frac{1}{2}\beta^{1-t}$ die Rundungseinheit (engl. "unit roundoff") des Gleitkommazahlsystems $\mathbb{F}$ bezeichnet.

Beweis. O.B.d.A. können wir $x > 0$ annehmen. Nach Satz 1.3 lässt sich x wie folgt darstellen:

$$x = \mu \cdot \beta^{e-t}, \quad \mu \in \mathbb{R}, \quad \beta^{t-1} \le \mu < \beta^t \,.$$

Also liegt x zwischen den benachbarten Gleitkommazahlen $y_1 = \lfloor \mu \rfloor \beta^{e-t}$, $y_2 = \lceil \mu \rceil \beta^{e-t}$, und damit ist $\mathsf{rd}(x) \in \{y_1, y_2\}$. Da x zur näheren Gleitkommazahl gerundet wird, haben wir $|\mathsf{rd}(x) - x| \leq |y_2 - y_1|/2$. Zusammen mit Lemma 1.9 ergibt sich

$$|\mathsf{rd}(x) - x| \leq \frac{1}{2}\epsilon_M |y_1| = \frac{1}{2}\beta^{1-t}|y_1| \leq \frac{1}{2}\beta^{1-t}|x|,$$

da $|y_1| \leq |x|$. Damit ist die Behauptung gezeigt. □

Die im Satz 1.13 eingeführte Rundungseinheit $u(\mathbb{F})$ zur Gleitpunktarithmetik $\mathbb{F}$ ist die prominente Grösse jeder Rundungsfehleranalyse.

Beispiel 1.14. In doppelter bzw. einfacher Genauigkeit nimmt u die folgenden Werte an:

$$\begin{aligned} \text{Double} \quad & u = 2^{-53} \approx 1.11 \cdot 10^{-16} \\ \text{Single} \quad & u = 2^{-24} \approx 5.96 \cdot 10^{-8} \end{aligned}$$

In Python kann der unit roundoff für die Datentypen `float` (doppelte Genauigkeit) und `numpy.float32` (einfache Genauigkeit) wie folgt erzeugt werden:

```
1 import numpy as np
2
3 # Doppelte Genauigkeit (float)
4 unit_roundoff_double = np.finfo(float).eps / 2
5
6 # Einfache Genauigkeit (np.float32)
7 unit_roundoff_single = np.finfo(np.float32).eps / 2
```

◇

Gelegentlich ist die folgende Variation von Satz 1.13 nützlicher.

Satz 1.15. *Für alle* $x \in \text{Bereich}(\mathbb{F})$ *existiert* $\delta \in \mathbb{R}$ *derart, dass*

$$\mathsf{rd}(x) = \frac{x}{1+\delta}, \quad |\delta| \leq u\,.$$

Beweis. Übung. □

Die Grösse δ in Satz 1.13 und Satz 1.15 hängt von x ab, ist aber nicht explizit bekannt, analog zu den Zwischenstellen in den Restgliedern von Taylorentwicklungen.

Es symbolisiere nun '$\circ$' eine der vier Grundrechenoperationen:

$$\circ \in \{+, -, *, /\}\,.$$

Setzen wir formal $\pm\infty = x/0$ für $x \in \mathbb{R}$, gilt (mit der Ausnahme $0/0$)

$$\circ : \mathbb{R} \times \mathbb{R} \to \mathbb{R} \cup \{\pm\infty\}.$$

Allerdings haben wir

$$\circ : \mathbb{F} \times \mathbb{F} \not\to \mathbb{F} \cup \{\pm\infty\}.$$

Um "zurück" nach $\mathbb{F}$ zu kommen, müssen wir runden. Eine sinnvolle Annahme wäre, dass Elementaroperationen das Ergebnis *exakt* berechnen und *danach* erst runden. In der Praxis stellt man eine leicht schwächere Forderung an die Computerarithmetik, die aus der Kombination dieser Annahme mit Satz 1.13 folgt.

Definition 1.16 (Mathematisches Standardmodell der Rundung)**.** Die Rundung zur Gleitkommaarithmetik $\mathbb{F}$ ist eine Abbildung $\mathsf{rd} : \text{Bereich}(\mathbb{F}) \to \mathbb{F}$ derart, dass für alle $x, y \in \text{Bereich}(\mathbb{F})$ ein $\delta \in \mathbb{R}$ existiert derart, dass

$$\mathsf{rd}(x \circ y) = (x \circ y)(1+\delta), \quad |\delta| \le u, \quad \circ \in \{+, -, *, /\}. \tag{1.3.1}$$

Wie in Satz 1.15 kann man alternativ zu (1.3.1) auch

$$\mathsf{rd}(x \circ y) = \frac{x \circ y}{1+\tilde{\delta}}, \quad |\tilde{\delta}| \le u, \quad \circ \in \{+, -, *, /\} \tag{1.3.2}$$

benutzen.

Beachte, dass Definition 1.16 *Nichtkommutativität* nach sich zieht: es gilt im Allgemeinen $\mathsf{rd}(x \circ y) \neq \mathsf{rd}(y \circ x)$, in der Praxis beobachtet man aber fast immer $\mathsf{rd}(x \circ y) = \mathsf{rd}(y \circ x)$. Anders sieht es mit der Assoziativität aus; diese ist selbst unter der stärkeren Annahme "exakte Rechnung, gerundetes Ergebnis" verletzt.

Beispiel 1.17. Sei $\beta = 10$, $t = 2$. Dann haben wir

$$\begin{aligned} & \mathsf{rd}(\mathsf{rd}(70+74)+74) = \mathsf{rd}(140+74) = 210 \\ \neq\ & \mathsf{rd}(70+\mathsf{rd}(74+74)) = \mathsf{rd}(70+150) = 220 \end{aligned}$$

sowie

$$\begin{aligned} & \mathsf{rd}(\mathsf{rd}(110-99)-10) = \mathsf{rd}(11-10) = 1 \\ \neq\ & \mathsf{rd}(110+\mathsf{rd}(-99-10)) = \mathsf{rd}(110-110) = 0. \end{aligned}$$

◇

Idealerweise möchte man auch für elementare Funktionen $f \in \{\exp, \sin, \cos, \tan, \ldots\}$ eine mit (1.3.1) vergleichbare Eigenschaft:

$$\mathsf{rd}(f(x)) = f(x)(1+\delta), \quad |\delta| \le u. \tag{1.3.3}$$

Die Implementierung eines solchen numerischen Verfahrens zur Berechnung von $f(x)$ ist alles andere als trivial, insbesondere weil nicht einfach vorhersagbar ist, auf wieviele Stellen genau $f(x)$ berechnet werden muss, um nach dem Runden (1.3.3) zu erfüllen (in der Literatur ist dies als *Table Maker's Dilemma* bekannt).

1.4 Rundungsfehleranalyse

Das Standardmodell der Rundung erlaubt es uns, auf einfache Weise die *Fortpflanzung von Rundungsfehlern in einer Gleitkommarechnung* mathematisch abzuschätzen. Als wichtiges Beispiel betrachten wir die Berechnung des euklidischen Skalarproduktes zweier Vektoren $\underline{x}, \underline{y} \in \mathbb{R}^n$, d.h. von

$$\underline{x}^{\mathsf{T}} \underline{y} = \sum_{k=1}^{n} x_k y_k \ .$$

Für die Teilsummen $s_i = \sum_{k=1}^{i} x_k y_k$, $i = 1, 2, ...$ gilt in Gleitkommaarithmetik $\mathbb{F}$

$$\begin{aligned} \widehat{s}_1 &= \mathsf{rd}(x_1 y_1) = x_1 y_1 (1+\delta_1), \\ \widehat{s}_2 &= \mathsf{rd}(\widehat{s}_1 + x_2 y_2) = \big(\widehat{s}_1 + x_2 y_2 (1+\delta_2)\big)(1+\delta_3) \\ &= x_1 y_1 (1+\delta_1)(1+\delta_3) + x_2 y_2 (1+\delta_2)(1+\delta_3), \end{aligned}$$

wobei der konkrekte Wert von δ_i natürlich von $\underline{x}$ und $\underline{y}$ abhängt, aber für alle δ_i gilt $|\delta_i| \leq u(\mathbb{F})$. Um die Notation zu vereinfachen, lassen wir im Folgenden die Indizes weg: es steht also δ für eine beliebige reelle Zahl mit $|\delta| \leq u(\mathbb{F})$. Mit dieser Konvention erhalten wir

$$\begin{aligned} \widehat{s}_3 &= \mathsf{rd}(\widehat{s}_2 + x_3 y_3) = \big(\widehat{s}_2 + x_3 y_3 (1+\delta)\big)(1+\delta) \\ &= x_1 y_1 (1+\delta)^3 + x_2 y_2 (1+\delta)^3 + x_3 y_3 (1+\delta)^2. \end{aligned}$$

Induktiv ergibt sich also für $s_n = \underline{x}^{\mathsf{T}} \underline{y}$:

$$\widehat{s}_n = x_1 y_1 (1+\delta)^n + x_2 y_2 (1+\delta)^n + x_3 y_3 (1+\delta)^{n-1} + \cdots + x_n y_n (1+\delta)^2 \ . \tag{1.4.1}$$

Für $n \to \infty$, also für Skalarprodukte "langer" Vektoren $\underline{x}, \underline{y}$, verstärkt sich also der Einfluss von Rundungsfehlern exponentiell mit der Vektorlänge n.

Das folgende Lemma erlaubt einerseits, den Ausdruck (1.4.1) zu vereinfachen. Es zeigt aber auch, dass für einen grossen Bereich von n (der von $\mathbb{F}$ abhängt) *keine exponentielle Verstärkung* des Rundungsfehlereinflusses bezüglich der Summenlänge n, sondern *für $n < 1/u(\mathbb{F})$ nur lineare Abhängigkeit von n vorliegt.*

Lemma 1.18. *Für $|\delta_i| \leq u(\mathbb{F})$, $i = 1, \ldots, n$ und $n < 1/u(\mathbb{F})$ existiert ein $\theta_n \in \mathbb{R}$ derart, dass*

$$\prod_{i=1}^{n} (1+\delta_i) = 1 + \theta_n, \quad \textit{wobei} \quad |\theta_n| \leq \frac{n u(\mathbb{F})}{1 - n u(\mathbb{F})} =: \gamma_n(\mathbb{F}). \tag{1.4.2}$$

Beweis. Der Beweis erfolgt per Induktion. Für $n = 1$ ist die Aussage offensichtlich. Induktionsannahme: die Aussage sei für $n-1$ mit $n \geq 2$ bewiesen.

Induktionsschritt: nach Induktionsannahme gilt

$$\prod_{i=1}^{n} (1+\delta_i) = (1+\delta_n)(1+\theta_{n-1}) =: 1 + \theta_n \quad , \quad \theta_n := \delta_n + (1+\delta_n)\theta_{n-1},$$

$$|\theta_n| \leq u + (1+u)\frac{(n-1)u}{1-(n-1)u} \quad = \quad \frac{nu}{1-(n-1)u} \leq \gamma_n \ .$$

□

So gilt also z.B. für $n \leq 1/(2u(\mathbb{F}))$ die Beziehung (1.4.2) mit $|\theta_n| \leq 2nu(\mathbb{F})$.
Die Anwendung von Lemma 1.18 auf (1.4.1) ergibt

$$\widehat{s}_n = x_1y_1(1+\theta_n) + x_2y_2(1+\theta'_n) + x_3y_3(1+\theta_{n-1}) + \cdots + x_ny_n(1+\theta_2), \tag{1.4.3}$$

mit $|\theta_j| \leq \gamma_j$ und $|\theta'_n| \leq \gamma_n$. Mit der Monotonieeigenschaft $0 < \gamma_j \leq \gamma_n$ erhalten wir

$$\mathsf{rd}(\underline{x}^{\mathsf{T}}\underline{y}) = (\underline{x} + \triangle\underline{x})^{\mathsf{T}}\underline{y} = \underline{x}^{\mathsf{T}}(\underline{y} + \triangle\underline{y}), \quad |\triangle\underline{x}| \leq \gamma_n|\underline{x}|, \quad |\triangle\underline{y}| \leq \gamma_n|\underline{y}|, \tag{1.4.4}$$

wobei $|\underline{x}|$ der Vektor mit den Komponenten $|x_i|$ sei und Ungleichungen zwischen Vektoren (und Matrizen) komponentenweise zu verstehen sind. Es ist zu bemerken, dass die Schranken in (1.4.4) im Gegensatz zu (1.4.3) *unabhänging* von der Reihenfolge der Summation sind.

Das Resultat (1.4.4) gibt den sogenannten **Rückwärtsfehler** an. Es sagt aus, dass *das in* $\mathbb{F}$ *berechnete Ergebnis (Skalarprodukt) das exakte Ergebnis (in* $\mathbb{R}$*) zu gestörten Eingabedaten* ($\underline{x} + \triangle\underline{x}$ und $\underline{y} + \triangle\underline{y}$) ist. Da die Eingabedaten sowieso nie exakt sind (allein schon das Abspeichern der Daten im Rechner erzeugt einen relativen Fehler in der Grössenordnung von $u(\mathbb{F})$), führt ein Algorithmus mit kleinem Rückwärtsfehler die Rechnung im Prinzip gerade so genau durch wie es eben die Daten zulassen. Der tatsächlich erzeugte Fehler im numerisch berechneten Ergebnis heisst **Vorwärtsfehler**. Aus (1.4.4) ergibt sich sofort eine Schranke für den Vorwärtsfehler im Skalarprodukt:

$$\left|\underline{x}^{\mathsf{T}}\underline{y} - \mathsf{rd}(\underline{x}^{\mathsf{T}}\underline{y})\right| \leq \gamma_n \sum_{i=1}^{n} |x_iy_i| = \gamma_n|\underline{x}|^{\mathsf{T}}|\underline{y}| \; . \tag{1.4.5}$$

Gilt $|\underline{x}^{\mathsf{T}}\underline{y}| \approx |\underline{x}|^{\mathsf{T}}|\underline{y}|$, so ist der relative Fehler $\left|\underline{x}^{\mathsf{T}}\underline{y} - \mathsf{rd}(\underline{x}^{\mathsf{T}}\underline{y})\right| / |\underline{x}^{\mathsf{T}}\underline{y}|$ klein. Gilt allerdings $|\underline{x}^{\mathsf{T}}\underline{y}| \ll |\underline{x}|^{\mathsf{T}}|\underline{y}|$, so ist kein kleiner relativer Fehler zu erwarten.

Als erste Anwendung der erhaltenen Schranken für Rundungsfehler im Skalarprodukt betrachten wir das Matrix-Vektor-Produkt

$$\underline{y} = \boldsymbol{A}\underline{x}, \quad \underline{x} \in \mathbb{R}^n, \; \underline{y} \in \mathbb{R}^m, \; \boldsymbol{A} \in \mathbb{R}^{m\times n}.$$

Bezeichnet $\underline{a}_i^{\mathsf{T}}$ die i-te Zeile von $\boldsymbol{A}$, so haben wir $y_i = \underline{a}_i^{\mathsf{T}}\underline{x}$ und aus (1.4.4) folgt

$$\widehat{y}_i = (\underline{a}_i + \triangle\underline{a}_i)^{\mathsf{T}}\underline{x}, \quad |\triangle\underline{a}_i| \leq \gamma_n|\underline{a}_i|.$$

Der Rückwärtsfehler erfüllt also

$$\widehat{\underline{y}} = (\boldsymbol{A} + \triangle\boldsymbol{A})x, \quad |\triangle\boldsymbol{A}| \leq \gamma_n|\boldsymbol{A}|, \tag{1.4.6}$$

und der sich daraus ergebende Vorwärtsfehler

$$\left|\widehat{\underline{y}} - \underline{y}\right| \leq \gamma_n|\boldsymbol{A}||\underline{x}|.$$

Normweise Vorwärtsfehlerschranken ergeben sich direkt für die 1- und ∞-Norm:

$$\|\widehat{\underline{y}} - \underline{y}\|_p \leq \gamma_n\|\boldsymbol{A}\|_p\|\underline{x}\|_p, \quad p \in \{1, \infty\}.$$

Mit den in Abschnitt A.7 angegebenen Matrixnormäquivalenzen folgt ausserdem

$$\|\widehat{\underline{y}} - \underline{y}\|_2 \leq \min\{m,n\}^{1/2}\gamma_n\|\boldsymbol{A}\|_2\|\underline{x}\|_2.$$

1.5 Auslöschung

Numerische Auslöschung tritt beim *Subtrahieren von zwei fast gleichen, bereits rundungsfehlerbehafteten Zahlen in Gleitkommaarithmetik* auf. Auslöschung ist eine der Hauptursachen für die massive Verstärkung von Rundungsfehlern im Laufe einer Gleitkommarechnung.

Beispiel 1.19. Betrachte

$$f(x) = \frac{1 - \cos(x)}{x^2}\,, \quad x = 1.2 \times 10^{-5}.$$

Dann ist $\cos(x)$ auf 10 Dezimalen gerundet

$$c = 0.9999\ 9999\ 99\,,$$

und

$$1 - c = 0.0000\ 0000\ 01\,.$$

Damit ist $(1-c)/x^2 = 10^{-10}/1.44 \times 10^{-10} = 0.6944\ldots$. Da aber $0 \le f(x) < \frac{1}{2}$ für $x \neq 0$, ist dieses Ergebnis offensichtlich vollkommen falsch. ◇

Allgemeiner seien $a, b \in \mathbb{R}$ gegeben, und

$$\widehat{a} = \mathsf{rd}(a) = a(1+\delta_a), \quad \widehat{b} = \mathsf{rd}(b) = b(1+\delta_b)$$

mit $|\delta_a| \le u$, $|\delta_b| \le u$. Dann gilt für $x = a - b$ und $\widehat{x} = \widehat{a} - \widehat{b}$

$$\left|\frac{x - \widehat{x}}{x}\right| = \left|\frac{-a\delta_a + b\delta_b}{a-b}\right| \le u\,\frac{|a| + |b|}{|a-b|} \tag{1.5.1}$$

Der relative Fehler in der Berechnung $x = a - b$ ist gross, falls

$$|a - b| \ll |a| + |b|.$$

Es ist zu betonen, dass in obiger Analyse a und b bereits rundungsfehlerbehaftet sind. Die Subtraktion selbst geschieht bei $a \approx b$ *rundungsfehlerfrei*; sie verstärkt lediglich die bereits vorhandenen Fehler.

Rundungsfehler können sich auch wieder auslöschen, d.h., ein sehr ungenaues Zwischenergebnis führt nicht zwingend auf ein sehr ungenaues Endergebnis. Dies zeigt das folgende Beispiel.

Beispiel 1.20 (Berechnung von $(e^x - 1)/x$ für $x \downarrow 0$)**.** Wir berechnen

$$f(x) = (e^x - 1)/x = \sum_{i=0}^{\infty} \frac{x^i}{(i+1)!} \tag{1.5.2}$$

in einfacher Genauigkeit (IEEE single) auf 2 Arten.

```
# Algorithmus 1

if x == 0:
    f = 1
else:
    f = (np.exp(x)-1)/x
```

```
# Algorithmus 2
y = np.exp(x)
if y == 1:
    f = 1
else:
    f = (y-1)/np.log(y)
```

x	Alg. 1	Alg. 2
10^{-3}	*1.0005*235	*1.00050*02
10^{-4}	*1.000*1660	*1.0000500*
10^{-5}	*1.00*13581	*1.0000050*
10^{-6}	*1.0*728836	*1.0000006*
10^{-7}	*1.*1920929	*1.0000001*

Tabelle 1.2: Ergebnisse von Algorithmus 1 und 2 in einfacher Genauigkeit. Korrekt berechnete Stellen sind kursiv dargestellt.

Die erhaltenen Ergebnisse finden sich in Tabelle 1.2 Um zu verstehen, wieso Algorithmus 1 wesentlich schlechtere Resultate liefert, betrachten wir $x = 9.0 \times 10^{-8}$ und nehmen an, dass die Implementierungen von $\exp(\cdot)$ und $\log(\cdot)$ das Rundungsfehlermodell (1.3.3) erfüllen. Die ersten 9 Dezimalstellen des exakten Ergebnisses sind

$$\frac{e^x - 1}{\log e^x} = 1.00000005\,.$$

Algorithmus 1 ergibt

$$\mathsf{rd}\Big(\frac{e^x - 1}{x}\Big) = \mathsf{rd}\Big(\frac{1.19209290 \times 10^{-7}}{\overline{9.0} \qquad \times 10^{-8}}\Big) = 1.32454766\,;$$

Algorithmus 2 dagegen ergibt

$$\mathsf{rd}\Big(\frac{e^x - 1}{\log e^x}\Big) = \mathsf{rd}\Big(\frac{1.19209290 \times 10^{-7}}{1.19209282 \times 10^{-7}}\Big) = 1.00000006\,.$$

Beachte: Algorithmus 2 berechnet aufgrund von Auslöschung sehr ungenaue Werte für $e^x - 1 = 9.00000041 \times 10^{-8}$ und $\log e^x = 9 \times 10^{-8}$; die Rundungsfehler heben sich aber beim Dividieren heraus!

Dieses Phänomen kann durch die Rundungsfehleranalyse von Algorithmus 2 erklärt werden. Wir haben $\widehat{y} = e^x(1+\delta)$, $|\delta| \le u$. Falls $\widehat{y} = 1$, folgt damit

$$e^x(1+\delta) = 1 \Longleftrightarrow x = -\log(1+\delta) = -\delta + \delta^2/2 - \delta^3/3 + \dots\,,$$

so dass folgt

$$\widehat{f} = \mathsf{rd}\big(1 + x/2 + x^2/6 + \dots\big|_{x=-\delta+O(\delta^2)}\big) = 1\,. \tag{1.5.3}$$

Falls $\widehat{y} \neq 1$, folgt

$$\widehat{f} = \mathsf{rd}\big((\widehat{y}-1)/\log\widehat{y}\big) = \frac{(\widehat{y}-1)(1+\delta_1)}{\log\widehat{y}(1+\delta_2)}\,(1+\delta_3), \quad |\delta_i| \le u\,. \tag{1.5.4}$$

Definiere

$$v := \widehat{y} - 1\,.$$

Dann gilt

$$g(\widehat{y}) := \frac{\widehat{y}-1}{\log \widehat{y}} = \frac{v}{\log(1+v)} = \frac{v}{v - v^2/2 + v^3/3 - \dots}$$
$$= \frac{1}{1 - v/2 + v^2/3 - \dots} = 1 + \frac{v}{2} + O(v^2)\,.$$

Falls x klein ist, ist $y \sim 1$ und

$$g(\widehat{y}) - g(y) \approx \frac{\widehat{y}-y}{2} \approx \frac{e^x \delta}{2} \approx \frac{\delta}{2} \approx g(y)\,\frac{\delta}{2}$$

(1.5.4) $\Longrightarrow$

$$\left|\frac{\widehat{f}-f}{f}\right| \le 3.5\,u\,. \tag{1.5.5}$$

Also sind in Algorithmus 2, $\widehat{y}-1$ **und** $\log \widehat{y}$ ungenau. Aber: $(\widehat{y}-1)/\log \widehat{y}$ ist eine sehr genaue Approximation an $(y-1)/\log y$ bei $y=1$, **da** $g(y) := (y-1)/\log y$ **dort "langsam variiert"**, denn $g(\cdot)$ hat bei $y=1$ eine hebbare Singularität: es gilt $g'(1)=1$, und wir sagen, $g(\cdot)$ ist bei $y=1$ "gut konditioniert". $\diamond$

1.6 Landau-Symbole (Wiederholung)

Die **Landau-Symbole** $O(\cdot)$ und $o(\cdot)$ werden verwendet, um das asymptotische Verhalten einer Funktion $f(x)$ für $x \to a$ zu beschreiben.

Definition 1.21. Sei $f, g : \mathbb{R} \to \mathbb{R}$ und $a \in \mathbb{R} \cup \{\pm\infty\}$. Dann schreibt man

$$f(x) = O(g(x)) \text{ für } x \to a, \quad \text{wenn } \limsup_{x\to a} \left|\frac{f(x)}{g(x)}\right| < \infty,$$
$$f(x) = o(g(x)) \text{ für } x \to a, \quad \text{wenn } \lim_{x\to a} \left|\frac{f(x)}{g(x)}\right| = 0.$$

Sei $p(x) = a_l x^l + a_{l+1}x^{l+1} \cdots + a_u x^u$ Polynom mit $a_l \neq 0, a_u \neq 0$ und $u \ge l$. Dann gilt

$$\begin{aligned} p(x) &= O(x^j) \text{ für } x \to \infty && \forall j \ge u,\\ p(x) &= o(x^j) \text{ für } x \to \infty && \forall j > u,\\ p(x) &= O(x^j) \text{ für } x \to 0 && \forall j \le l,\\ p(x) &= o(x^j) \text{ für } x \to 0 && \forall j < l. \end{aligned}$$

Im allgemeinen gelten die folgenden leicht beweisbaren Rechenregeln:

$$\begin{aligned} f(x) = O(g(x)) &\implies c \cdot f(x) = O(g(x)),\\ f_1(x) = O(g_1(x)),\ f_2(x) = O(g_2(x)) &\implies f_1(x) f_2(x) = O(g_1(x) g_2(x)). \end{aligned}$$

Analoge Aussagen gelten für "o".

1.7 Bibliographische Bemerkungen

Die klassische Referenz zur mathematischen Analyse von Rundungsfehlern in allen wichtigen Algorithmen der numerischen linearen Algebra ist [6]. Viele Ergebnisse in diesem Werk sind seminal, und wurden in der Folgezeit nicht oder nur noch marginal verschärft.

Für eine neuere, erschöpfende Darstellung mathematischer Modelle von Gleitkommaarithmetik, deren historische Entwicklung sowie einer eingehenden Analysis von Rundungsfehlern und ihrer Fortpflanzung in den wichtigsten Algorithmen der numerischen linearen Algebra verweisen wir auf [4, Chap.1 und 2] und die Referenzen dort.

In neuerer Zeit sind, neben den IEEE Standards, eine Reihe von sogenannten "extended precision" Arithmetiken verfügbar geworden, bei denen die Mantissenlänge beliebig gross vom Benutzer eingestellt werden kann. Eine gute Übersicht, inklusive weblinks zu Public Domain codes, findet man in [1]. Diese Arithmetiken benötigen, zur Ausführung einer grossen Zahl von Gleitpunktoperationen, in der Regel deutlich mehr Rechenzeit als zB IEEE single precision. In Videospielkonsolen wird, zur Optimierung der interaktiven Spieldynamik, ebenfalls oft nur einfache Genauigkeit verwendet.

Seit 2015 ist, in Zusammenhang mit dem sogenannten "Exascale" Computing, bei dem mehr als 10^6 Prozessoren parallel zur Lösung einer Aufgabe eingesetzt werden (z.B. bei Internetserviceprovidern, und in Wetter- und Klimasimulationen) der *Strombedarf bei Höchstleistungsrechnern* in den Vordergrund gerückt. Ein wichtiger Ansatz zur Energieeinsparung (sogenannten "green computing") bei solchen Rechnern besteht auch hier darin, *mathematisch gerechtfertigt* Teile einer komplexen Rechnung nur in einfacher Genauigkeit durchzuführen; wir verweisen auf [5] und die Referenzen dort für Grundlagen, sowie auf [2] für Anwendungen von IEEE single oder sogar "half-precision" in globalen Klimamodellen.

Siehe auch "Low Precision Floating-Point Formats: The Wild West of Computer Arithmetic, Srikara Pranesh, SIAM News November 2019,[4] and N. Highham and T. Mary: Mixed precision algorithms in numerical linear algebra, Acta Numerica **31** (2022) 347-414.

Im sogenannten "Deep Learning" werden sehr grosse ("tiefe") neuronale Netze *quantisiert*, d.h., mit internen Parametern in sehr grober Gleitkommaarithmetik (z.B. $\beta = 10$, $t = 3$, $(e_{min}, e_{max}) = (-2, 2)$) trainiert und berechnet[5]; im Deep Learning kann eine reduzierte Gleitkommaarithmetik ohne Genauigkeitseinbussen für viele Anwendungen (Bildsegmentierung, Sprach- und Mustererkennung, ...) durch grössere Netzwerktiefe kompensiert werden.

[4] SIAM news: Low Precision Floating-Point Formats: The Wild West of Computer Arithmetic

[5] Vgl. https://nervanasystems.github.io/distiller/quantization.html

Anhang A

Lineare Algebra

In diesem Anhang rekapitulieren wir grundlegende Begriffe der linearen Algebra, die für die Analyse numerischer Methoden wichtig sind. Grundlegende Referenz für die *numerische lineare Algebra* ist [3]. Im folgenden verwenden wir die Konvention, dass Vektoren mit unterstrichenen Kleinbuchstaben ($\underline{v}, \underline{w}$) und Matrizen mit fettgestellten Grossbuchstaben ($\boldsymbol{A}, \boldsymbol{B}$) bezeichnet werden.

A.1 Vektorräume

Definition A.1. Ein **Vektorraum** über einem Zahlkörper $\mathbb{K}(= \mathbb{R}, \mathbb{C})$ ist eine Menge $V \neq \emptyset$ versehen mit einer **Addition "+":** $V \times V \to V$ (kommutativ + assoziativ) und einer **Skalarmultiplikation "·":** $\mathbb{K} \times V \to V$, welche zusätzlich die folgenden Eigenschaften erfüllen:

(i) $\exists \underline{0} \in V: \ \forall \underline{v} \in V: \ \underline{v} + \underline{0} = \underline{v}$, (Nullvektor)

(ii) $\forall \underline{v} \in V: \ 0 \cdot \underline{v} = \underline{0}, \ 1 \cdot \underline{v} = \underline{v}$ für $0, 1 \in \mathbb{K}$,

(iii) $\forall \underline{v} \in V: \ \exists \underline{w} \in V: \underline{v} + \underline{w} = \underline{0}$, (inverses Element)

(iv) $\forall \alpha \in \mathbb{K}: \ \forall \underline{v}, \underline{w} \in V: \ \alpha \cdot (\underline{v} + \underline{w}) = \alpha \cdot \underline{v} + \alpha \cdot \underline{w}$,
$\forall \alpha, \beta \in \mathbb{K}: \ \forall \underline{v} \in V: \ (\alpha + \beta) \cdot \underline{v} = \alpha \cdot \underline{v} + \beta \cdot \underline{v}$,

(v) $\forall \alpha, \beta \in \mathbb{K}, \forall \underline{v} \in V: \ (\alpha\beta) \cdot \underline{v} = \alpha(\beta \cdot \underline{v})$.

Hier bezeichnet das Symbol "×" das cartesische Produkt zweier Mengen, also die Menge aller geordneten Paare von Elementen aus diesen Mengen.

Im Skript wird das Symbol "·" für die Skalarmultiplikation weggelassen denn aus der Matrix- und Vektornotation ist klar, wann eine Skalarmultiplikation vorliegt.

Beispiel A.2. Abgesehen von $\mathbb{R}^n$ und $\mathbb{C}^n$ gibt es weitere kanonische Beispiele für Vektorräume, die wir im Verlaufe der Vorlesung wiederholt antreffen werden.

(i) Die Menge aller *Polynome vom Grad höchstens n*:

$$V = \mathbb{P}_n := \Big\{ p_n(x) = \sum_{k=0}^{n} a_k \, x^k \Big\}.$$

(ii) Die Menge aller auf einem beschränkten Intervall $[a, b]$ *p-mal stetig differenzierbaren Funktionen*:

$$V = C^p([a, b]) \, .$$

(Für $p = 0$ ist V die Menge der auf dem beschränkten Intervall $[a, b]$ stetigen Funktionen.)

(iii) Die Menge aller auf einem beschränkten Intervall $[a, b]$ *stückweise linearen, stetigen Funktionen*

$$V = \big\{ f \in C^0([a, b]) : \; f|_{[x_{i-1}, x_i]} \text{ ist linear} \big\}$$

bezüglich einer festen Stützstellenmenge $x_0, x_1, \ldots, x_n$ mit $a = x_0 < x_1 < \cdots < x_n = b$. Hierbei bezeichnet $f|_{[x_{i-1}, x_i]}$ die Einschränkung der Funktion f auf das Intervall $[x_{i-1}, x_i]$.

In allen obigen Beispielen ist V mit der bei Funktionen üblichen Addition und skalaren Multiplikation versehen. ◇

Definition A.3. Sei V Vektorraum. Eine Teilmenge $W \subseteq V$ heisst **Teilraum** (oder Untervektorraum) von V, wenn W selbst Vektorraum über $\mathbb{K}$ ist (mit der von V induzierten Addition und Skalarmultiplikation).

Zum Nachweis, dass eine Teilmenge $W \subseteq V$ mit $W \neq \emptyset$ ein Teilraum von V ist, muss nur die Abgeschlossenheit von W bezüglich Addition und Skalarmultiplikation sowie $\underline{0} \in V$ überprüft werden. Alle anderen Eigenschaften von Definition A.1 "erbt" W von V. Damit lassen sich mühelos die folgenden Beispiele zeigen.

Beispiel A.4.

(i) Die Menge der stückweise linearen stetigen Funktionen (siehe Beispiel A.2.(iii)) bildet einen Teilraum von $V = C^0([a, b])$.

(ii) Die Menge $\mathbb{P}_n$ (Polynome vom Grad höchstens n) bildet einen Teilraum von $C^0(\mathbb{R})$ (stetige Funktionen auf $\mathbb{R}$).

(iii) Sei V Vektorraum und $\underline{v}_1, \ldots, \underline{v}_n \in V$. Dann ist

$$W := \mathbf{span}\{\underline{v}_1, \ldots, \underline{v}_n\} \; := \{\underline{w} = \alpha_1 \, \underline{v}_1 + \cdots + \alpha_n \, \underline{v}_n : \; \alpha_i \in \mathbb{K}\}$$

Teilraum von V. Wir nennen $\underline{v}_1, \ldots, \underline{v}_n$ **erzeugendes System** von W.

(iv) Seien $W_1, \ldots, W_m \subseteq W$ Teilräume eines Vektorraums V. Dann ist

$$S := \{\underline{s} = \underline{w}_1 + \cdots + \underline{w}_m \text{ mit } \underline{w}_i \in W_i\}$$

ebenfalls ein Teilraum von V.
S heisst **direkte Summe** der W_i, symbolisch $S = W_1 \oplus \cdots \oplus W_m$, wenn die Zerlegung $\underline{s} = \underline{w}_1 + \cdots + \underline{w}_m$ jedes Vektors $\underline{s} \in S$ eindeutig ist. ◇

Definition A.5. Sei V Vektorraum. Eine Menge von Vektoren $\{\underline{v}_1, \ldots, \underline{v}_m\} \subset V$ heisst **linear unabhängig**, wenn

$$\alpha_1 \underline{v}_1 + \cdots + \alpha_m \underline{v}_m = \underline{0} \quad \Longrightarrow \quad \alpha_1 = \cdots = \alpha_m = 0\,.$$

Eine linear unabhängige Menge $\{\underline{v}_1, \ldots, \underline{v}_m\}$ heisst **Basis** von V, wenn zusätzlich $V = \text{span}\{\underline{v}_1, \ldots, \underline{v}_m\}$ gilt.

Aus dem Austauschlemma von Steinitz folgt, dass jede Basis von V die gleiche Anzahl von Basiselementen besitzt. Diese Anzahl (m in der obigen Definition) wird als **Dimension** von V mit $\dim(V)$ bezeichnet. Gibt es keine endliche Basis, so wird V als unendlichdimensional bezeichnet.

Beispiel A.6.

(i) Die **Einheitsvektoren**

$$\underline{e}_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \ \underline{e}_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \ \cdots, \ \underline{e}_n = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

bilden die kanonische Basis des $\mathbb{R}^n$ und $\mathbb{C}^n$.

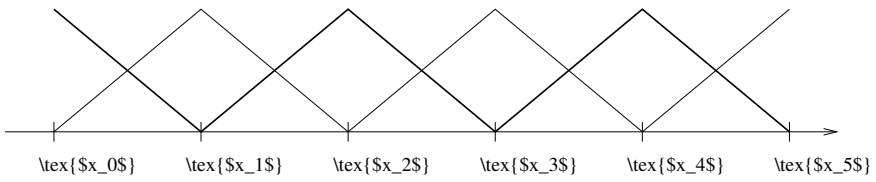
(ii) Die Monome $1, x, \ldots, x^n$ bilden eine Basis von $\mathbb{P}_n$. Damit folgt $\dim(\mathbb{P}_n) = n + 1$.

(iii) Sei $V = \{f \in C^0([a,b]) : f|_{[x_{i-1},x_i]} \text{ ist linear}\}$ mit $a = x_0 < x_1 < \cdots < x_n = b$. Dann bilden die **Hutfunktionen**

$$b_i(x) = \begin{cases} \frac{x - x_{i-1}}{x_i - x_{i-1}} & \text{für } x \in [x_{i-1}, x_i], \\ \frac{x_{i+1} - x}{x_{i+1} - x_i} & \text{für } x \in [x_i, x_{i+1}], \\ 0 & \text{sonst}, \end{cases} \qquad (i = 1, \ldots, n-1)$$

$$b_0(x) = \begin{cases} \frac{x_1 - x}{x_1 - x_0} & \text{für } x \in [x_0, x_1], \\ 0 & \text{sonst}, \end{cases} \quad b_n(x) = \begin{cases} \frac{x - x_{n-1}}{x_n - x_{n-1}} & \text{für } x \in [x_{n-1}, x_n], \\ 0 & \text{sonst}, \end{cases}$$

eine Basis von V. Es gilt also ebenfalls $\dim(V) = n + 1$. Die Funktionen $b_0, b_1, \ldots, b_n$ sind in der folgenden Abbildung für $n = 5$ dargestellt:



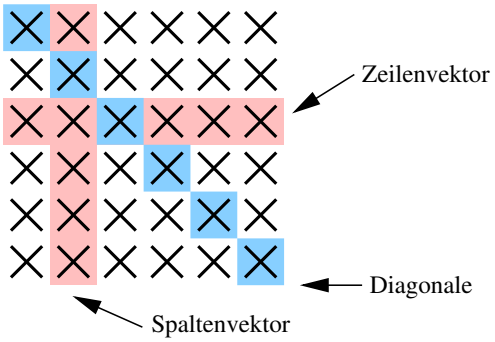
(iv) $C^p([a,b])$ besitzt keine endliche Basis, ist also unendlichdimensional. ◇

A.2 Matrizen

Sei $m, n \in \mathbb{N}$ und $\mathbb{K}$ Körper. Die mn Zahlen $a_{ij} \in \mathbb{K}$, $i = 1, \ldots, m$, $j = 1, \ldots, n$ bilden eine $m \times n$ **Matrix** mit m Zeilen und n Spalten:

$$\boldsymbol{A} = (a_{ij}) = \begin{pmatrix} a_{11} & \ldots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \ldots & a_{mn} \end{pmatrix}. \tag{A.2.1}$$

Falls $a_{ij} \in \mathbb{R}$ schreiben wir $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ und, falls $a_{ij} \in \mathbb{C}$, schreiben wir analog $\boldsymbol{A} \in \mathbb{C}^{m \times n}$. Wir benutzen die Bezeichnungen:



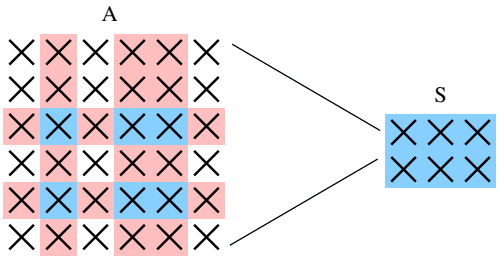

```python
# j-ter Spaltenvektor
print(A[:, j-1])
# i-ter Zeilenvektor
print(A[i-1, :])
# Diagonale
print(np.diag(A))
```

Definition A.7. Sei $\boldsymbol{A}$ eine $m \times n$ Matrix, und i_p, j_q Indizes mit

$$1 \leq i_1 < \cdots < i_k \leq m, \quad 1 \leq j_1 < \cdots < j_\ell \leq n\,.$$

Dann heisst $\boldsymbol{S} = (a_{i_p, j_q})$, $p = 1, \ldots, k$, $q = 1, \ldots, \ell$, die zugehörige **Untermatrix** von $\boldsymbol{A}$.

Die folgende Illustration veranschaulicht den Begriff der Untermatrix:




```python
S = A[np.ix_([2, 4], [1, 3, 4])]
S = A[[2, 4]][:, [1, 3, 4]]
```

Definition A.8. Eine $m \times n$ Matrix $\boldsymbol{A}$ wird als **Blockmatrix** bezeichnet, wenn sie wie folgt in Untermatrizen partitioniert ist:

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{A}_{11} & \ldots & \boldsymbol{A}_{1\ell} \\ \vdots & & \vdots \\ \boldsymbol{A}_{k1} & \ldots & \boldsymbol{A}_{k\ell} \end{pmatrix},$$

mit $m_i \times n_j$ Matrizen $\boldsymbol{A}_{ij}$ und $m_1 + \cdots + m_k = m$, $n_1 + \cdots + n_\ell = n$.

Beispiel A.9. $m = n = 3$, $k = \ell = 2$, $m_1 = n_1 = 2$, $m_2 = n_2 = 1$;

$$\left(\begin{array}{cc|c} 1 & 2 & 3 \\ 4 & 5 & 6 \\ \hline 7 & 8 & 9 \end{array}\right) = \left(\begin{array}{c|c} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} \\ \hline \boldsymbol{A}_{21} & \boldsymbol{A}_{22} \end{array}\right).$$

◇

Weitere grundlegende MATLAB-Befehle zur Manipulation von Vektoren und Matrizen sind im folgenden Beispielcode illustriert.

```
v = np.random.rand(10)     # Zufallsvektor der Laenge 10
len(v)                     # Laenge von v
v[1]                       # 2-te Komponente von v
v[1:5]                     # Komponenten 2 bis 5 von v
v[0::2]                    # alle Komponenten mit ungeradem Index
A = np.random.rand(10, 8)  # 10x8 Zufallsmatrix
m, n = A.shape             # Anzahl Zeilen (m) und Spalten (n) von A
A[0, 1]                    # Eintrag (1,2) von A
A[2:5, 3:7]                # Untermatrix aus Zeilen 3,...,5 und
   Spalten 4,...,7 von A
A[:, 2] = v                # 3-te Spalte von A mit v ueberschrieben
A[5, :] = 7                # 6-te Zeile von A mit 7 ueberschrieben
```

A.3 Matrixoperationen

Wir rekapitulieren die wichtigsten Matrixoperationen.

Definition A.10 (Matrixmultiplikation)**.** Seien Matrizen $\boldsymbol{A} = (a_{ij}) \in \mathbb{K}^{m\times n}$ und $\boldsymbol{B} = (b_{jk}) \in \mathbb{K}^{n\times q}$ gegeben. Dann ist das Produkt der Matrizen, $\boldsymbol{C} := \boldsymbol{AB} \in \mathbb{K}^{m\times q}$ definiert durch seine Komponenten $\boldsymbol{C} = (c_{ik})$ mit

$$c_{ik} = \sum_{j=1}^{n} a_{ij} b_{jk}. \tag{A.3.1}$$

Partitioniert man $\boldsymbol{B}$ in seine Spalten, $\boldsymbol{B} = \big(\underline{b}_1, \ldots, \underline{b}_q\big)$ mit $\underline{b}_j \in \mathbb{K}^n$, so lässt sich (A.3.1) spaltenweise schreiben:

$$\boldsymbol{C} = \boldsymbol{AB} = \boldsymbol{A}\big(\underline{b}_1, \ldots, \underline{b}_q\big) = \big(\boldsymbol{A}\underline{b}_1, \ldots, \boldsymbol{A}\underline{b}_q\big),$$

das heisst, die Spalte $\underline{c}_j \in \mathbb{K}^m$ in $\boldsymbol{C} = \big(\underline{c}_1, \ldots, \underline{c}_q\big)$ ist gegeben durch $\underline{c}_j = \boldsymbol{A}\underline{b}_j$.

BLAS (Basic Linear Algebra Subroutines)* Die Speicherhierarchie moderner Rechner reicht von kleinen Speichern (Register, Cache) mit sehr kurzen Zugriffszeiten bis zu grossen Speichern (Hauptspeicher, Festplatte, Netzwerkspeicher) mit vergleichsweise langen Zugriffszeiten. Um diese Hierarchie optimal auszunutzen, sollten so viele Operationen wie möglich mit Daten im schnellen Speicher (Cache) durchgeführt werden, bevor neue Daten aus dem langsamerem Speicher (Hauptspeicher) transferiert werden müssen. Eine Implementierung der Matrixmultiplikation nach (A.3.1) ist aus dieser Sicht unbefriedigend und würde zu inakzeptabel hohen Laufzeiten führen, da für jedes Element c_{ik} auf im Speicher weit auseinanderliegende Teile der Matrizen $\boldsymbol{A}$ und $\boldsymbol{B}$ zugegriffen werden muss. Um diesen Effekt zu vermeiden, werden die Matrizen $\boldsymbol{A}$ und $\boldsymbol{B}$ in Blockmatrizen partitioniert und die Matrixmultiplikation blockweise durchgeführt. Die Multiplikation der einzelnen Blöcke wird meist in hochoptimierten Assembler-Code implementiert. Viele Hardwarehersteller stellen solche effizienten Implementierungen von Vektor- und Matrixoperationen bereit und richten sich dabei nach dem sogenannten BLAS-Standard, dessen Funktionalität sich in 3 Stufen unterteilt:

BLAS level 1 enthält Vektor-Operationen, z.B. $\underline{y} \leftarrow \alpha \underline{x} + \underline{y}$.

BLAS level 2 enthält Matrix-Vektor-Operationen, z.B. $\underline{y} \leftarrow \alpha \boldsymbol{A}\underline{x} + \beta \underline{y}$.

BLAS level 3 enthält Matrix-Matrix-Operationen, z.B. $\boldsymbol{C} \leftarrow \alpha \boldsymbol{A}\boldsymbol{B} + \beta \boldsymbol{C}$.

Dieses Modulkonzept erlaubt es numerische Algorithmen effizient zu implementieren, indem der Grossteil des Algorithmus als Vektor- und Matrixoperationen (vorzugsweise level 3) formuliert wird. Die meisten numerischen Programmpakete (NumPy, MATLAB, Mathematica, ...) basieren auf BLAS.
Implementieren Sie niemals eine Matrixmultiplikation selbst.

Definition A.11. Eine $n \times n$ Matrix $\boldsymbol{A}$ heisst **invertierbar** oder regulär, wenn es eine $n \times n$ Matrix $\boldsymbol{B}$ gibt, so dass $\boldsymbol{A}\boldsymbol{B} = \boldsymbol{B}\boldsymbol{A} = \boldsymbol{I}_n$, wobei $\boldsymbol{I}_n = (\underline{e}_1, \ldots, \underline{e}_n)$ die $n \times n$ **Einheitsmatrix** bezeichnet. Eine $n \times n$ Matrix $\boldsymbol{A}$ heisst **singulär**, wenn sie nicht invertierbar ist.

Die Matrix $\boldsymbol{B}$ in Definition A.11 ist eindeutig festgelegt und wird als **Inverse** von $\boldsymbol{A}$ bezeichnet. Wir schreiben $\boldsymbol{B} = \boldsymbol{A}^{-1}$.

Proposition A.12. *Eine Matrix $\boldsymbol{A} = (\underline{a}_1, \ldots, \underline{a}_n)$ ist genau dann invertierbar, wenn ihre Spalten $\underline{a}_1, \ldots, \underline{a}_n$ linear unabhängig sind.*

Für die Zeilen von $\boldsymbol{A}$ gilt eine zu Proposition A.12 analoge Aussage.
Die beiden wichtigsten Rechenregeln bei der Arbeit mit Inversen lauten

$$(\boldsymbol{A}^{-1})^{-1} = \boldsymbol{A}, \qquad (\boldsymbol{A}\boldsymbol{C})^{-1} = \boldsymbol{C}^{-1}\boldsymbol{A}^{-1}.$$

Definition A.13. Sei $\boldsymbol{A} = (a_{ij}) \in \mathbb{K}^{m \times n}$. Dann bezeichnet $\boldsymbol{A}^\mathsf{T} = (a_{ji}) \in \mathbb{K}^{n \times m}$ die **Transponierte** von $\boldsymbol{A}$.

Beispiel:

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}^\mathsf{T} = \begin{pmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{pmatrix}.$$

Einige Rechenregeln für die Arbeit mit der Transponierten:

$$\begin{array}{ll} (\boldsymbol{A}^\mathsf{T})^\mathsf{T} = \boldsymbol{A}, & (\boldsymbol{A}+\boldsymbol{B})^\mathsf{T} = \boldsymbol{A}^\mathsf{T} + \boldsymbol{B}^\mathsf{T}, \\ (\boldsymbol{A}\boldsymbol{B})^\mathsf{T} = \boldsymbol{B}^\mathsf{T}\boldsymbol{A}^\mathsf{T}, & (\alpha\boldsymbol{A})^\mathsf{T} = \alpha\boldsymbol{A}^\mathsf{T} \quad \forall \alpha \in \mathbb{K}. \end{array} \tag{A.3.2}$$

Ist $\boldsymbol{A}$ invertierbar, so ist auch $\boldsymbol{A}^\mathsf{T}$ invertierbar und es gilt

$$(\boldsymbol{A}^\mathsf{T})^{-1} = (\boldsymbol{A}^{-1})^\mathsf{T} =: \boldsymbol{A}^{-\mathsf{T}}. \tag{A.3.3}$$

Für Matrizen mit komplexen Einträgen ist selten die Transponierte im Sinne von Definition A.13 von Interesse, sondern die folgende Variante, bei der die Einträge zusätzlich konjugiert werden.

Definition A.14. Sei $\boldsymbol{A} = (a_{ij}) \in \mathbb{C}^{m\times n}$. Dann bezeichnet $\boldsymbol{B} = \boldsymbol{A}^\mathsf{H} := \overline{\boldsymbol{A}}^\mathsf{T} = (\overline{a_{ji}}) \in \mathbb{C}^{n\times m}$ die **hermitesch Transponierte** von $\boldsymbol{A}$.

Die Rechenregeln (A.3.2) und (A.3.3) gelten gleichermassen für die hermitesch Transponierte, mit der Ausnahme

$$(\alpha\boldsymbol{A})^\mathsf{H} = \overline{\alpha}\boldsymbol{A}^\mathsf{H} \quad \forall \alpha \in \mathbb{C}.$$

Definition A.15. Die **Spur** (Englisch: trace) einer $n \times n$ Matrix $\boldsymbol{A}$ ist die Summe ihrer Diagonalelemente: $\operatorname{spur}(\boldsymbol{A}) := \sum_{i=1}^{n} a_{ii}$.

Definition A.16. Sei $\boldsymbol{A} = (a_{ij})$ eine $n \times n$ Matrix. Dann bezeichnet

$$\det \boldsymbol{A} = \sum_{\sigma\in S_n} \left(\operatorname{sgn}(\sigma) \prod_{i=1}^{n} a_{i,\sigma(i)} \right)$$

die **Determinante** von $\boldsymbol{A}$. Hierbei ist S_n die Menge aller Permutationen der Zahlen $\{1,\dots,n\}$ und sgn bezeichnet das Vorzeichen einer solchen Permutation.

Die Determinante erfüllt folgende Rechenregeln:

$$\det(\boldsymbol{A}) = \det(\boldsymbol{A}^\mathsf{T}),\ \det(\boldsymbol{A}\boldsymbol{B}) = \det(\boldsymbol{A})\det(\boldsymbol{B}),\ \det(\boldsymbol{A}^{-1}) = 1/\det(\boldsymbol{A})$$

$$\det(\boldsymbol{A}) = \overline{\det(\boldsymbol{A}^\mathsf{H})},\ \det(\alpha\boldsymbol{A}) = \alpha^n \det(\boldsymbol{A}) \quad \forall \alpha \in \mathbb{K}.$$

Desweiteren gilt $\det(\boldsymbol{A}) \neq 0$ genau dann, wenn $\boldsymbol{A}$ invertierbar ist.

```
C = A + B             # Matrix-Addition
C = np.dot(A, B)      # Matrix-Multiplikation
C = A @ B             # Matrix-Multiplikation
C = A * B             # komponentenweise Multiplikation (!)
B = np.linalg.inv(A)  # Inverse von A
B = A.conj().T        # hermitesch Transponierte von A
B = A.T               # Transponierte von A
np.trace(A)           # Spur von A
np.linalg.det(A)      # Determinante von A
```

Definition A.17. Sei $\boldsymbol{A}$ eine $m \times n$ Matrix. Die Mengen

$$\begin{aligned} \operatorname{im}(\boldsymbol{A}) &:= \{\underline{y} \in \mathbb{K}^m : \ \underline{y} = \boldsymbol{A}\underline{x}, \, \underline{x} \in \mathbb{K}^n\}, \\ \ker(\boldsymbol{A}) &:= \{\underline{x} \in \mathbb{K}^n : \ \boldsymbol{A}\underline{x} = \underline{0}\} \end{aligned}$$

heissen **Bild** bzw. **Kern** von $\boldsymbol{A}$.

Bild und Kern sind jeweils Teilräume des $\mathbb{K}^m$ bzw. $\mathbb{K}^n$.

Definition A.18. Der **Rang** einer Matrix $\boldsymbol{A}$ ist die Dimension ihres Bildes und wird mit $\operatorname{rang}(\boldsymbol{A}) := \dim(\operatorname{im}(\boldsymbol{A}))$ bezeichnet.

Aus der Definition ergibt sich, dass der Rang der Anzahl der linear unabhängigen Spalten von $\boldsymbol{A}$ entspricht. Weiterhin gelten die folgenden Eigenschaften:

$$\begin{aligned} &\operatorname{rang}(\boldsymbol{A}) = \operatorname{rang}(\boldsymbol{A}^{\mathsf{T}}) = \operatorname{rang}(\boldsymbol{A}^{\mathsf{H}}), \\ &\operatorname{rang}(\boldsymbol{A}) + \dim\big(\ker(\boldsymbol{A})\big) = n, \\ &\operatorname{rang}(\boldsymbol{A}\boldsymbol{B}) \leq \operatorname{rang}(\boldsymbol{A}) \operatorname{rang}(\boldsymbol{B}). \end{aligned}$$

Aus der letzten Eigenschaft folgt insbesondere, dass die Matrix $A = uv^{\mathsf{H}} \in \mathbb{C}^{n \times n}$ mit Vektoren $u, v \in \mathbb{C}^n$ höchstens Rang 1 hat.

A.4 Spezielle Matrizen

Die numerische lineare Algebra basiert zu grossen Teilen auf Operationen mit Matrizen mit speziellen Eigenschaften. Als einfachstes Beispiel haben wir bereits die Einheitsmatrix $\boldsymbol{I}_n$ kennengelernt. Eine allgemeine $n \times n$ **Diagonalmatrix** bezeichnen wir mit

$$\boldsymbol{D} = \mathbf{diag}(d_{11}, d_{22}, \ldots, d_{nn}) := \begin{pmatrix} d_{11} & 0 & \cdots & 0 \\ 0 & d_{22} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & d_{nn} \end{pmatrix}.$$

Wenn $d_{11}, \ldots, d_{nn}$ selber Matrizen sind, so heisst $\boldsymbol{D}$ **Blockdiagonalmatrix**.

Eine linke bzw. rechte **Dreiecksmatrix** hat die Form

$$\boldsymbol{L} = \begin{pmatrix} \ell_{11} & 0 & \cdots & 0 \\ \ell_{21} & \ell_{22} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ \ell_{n1} & \cdots & \ell_{n,n-1} & \ell_{nn} \end{pmatrix}, \quad \boldsymbol{R} = \begin{pmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ 0 & r_{22} & \ddots & \vdots \\ \vdots & \ddots & \ddots & r_{n-1,n} \\ 0 & \cdots & 0 & r_{nn} \end{pmatrix}.$$

Oft werden linke (rechte) Dreiecksmatrizen auch als untere (obere) Dreiecksmatrizen bezeichnet.

Proposition A.19. *(i) Sind $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{K}^{n\times n}$ linke (rechte) Dreiecksmatrizen, so ist $\boldsymbol{AB}$ auch eine linke (rechte) Dreiecksmatrix.*

(ii) Ist $\boldsymbol{A} \in \mathbb{K}^{n\times n}$ eine invertierbare linke (rechte) Dreiecksmatrix, so ist $\boldsymbol{A}^{-1}$ auch eine linke (rechte) Dreiecksmatrix.

Proposition A.20. *Sei $\boldsymbol{L}$ linke Dreiecksmatrix, $\boldsymbol{R}$ rechte Dreiecksmatrix. Dann gilt*

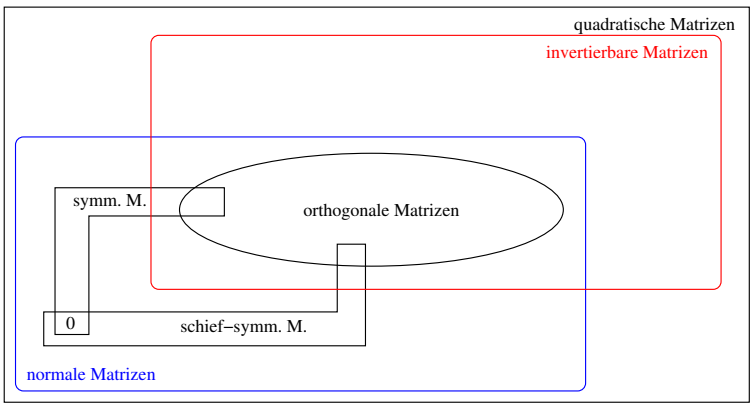
$$\det \boldsymbol{L} = \ell_{11} \cdots \ell_{nn} = \prod_{i=1}^{n} \ell_{ii}, \quad \det \boldsymbol{R} = r_{11} \cdots r_{nn} = \prod_{i=1}^{n} r_{ii}\,.$$

```
np.eye(n)          # nxn Einheitsmatrix
np.zeros((m, n))   # mxn Matrix mit allen Eintraegen Null
np.ones((m, n))    # mxn Matrix mit allen Eintraegen Eins
np.diag(v)         # Diagonalmatrix mit v auf der Diagonalen
np.tril(A)         # linke Dreiecksmatrix mit A im unteren Dreieck
np.triu(A)         # rechte Dreiecksmatrix mit A im oberen Dreieck
```

Abgesehen von Matrizen mit Nulleinträgen gibt es weitere wichtige Klassen von Matrizen, deren Einträge gewisse Beziehungen erfüllen.

Definition A.21.

$\boldsymbol{A} \in \mathbb{R}^{n\times n}$ heisst	**symmetrisch**	$\iff$	$\boldsymbol{A} = \boldsymbol{A}^{\mathsf{T}}$,
	schiefsymmetrisch	$\iff$	$\boldsymbol{A} = -\boldsymbol{A}^{\mathsf{T}}$,
	orthogonal	$\iff$	$\boldsymbol{A}^{-1} = \boldsymbol{A}^{\mathsf{T}} \iff \boldsymbol{A}^{\mathsf{T}}\boldsymbol{A} = \boldsymbol{A}\boldsymbol{A}^{\mathsf{T}} = \boldsymbol{I}$.
$\boldsymbol{A} \in \mathbb{C}^{n\times n}$ heisst	**hermitesch**	$\iff$	$\boldsymbol{A}^{\mathsf{T}} = \overline{\boldsymbol{A}} \iff \boldsymbol{A}^{\mathsf{H}} = \boldsymbol{A}$,
	unitär	$\iff$	$\boldsymbol{A}^{-1} = \boldsymbol{A}^{\mathsf{H}} \iff \boldsymbol{A}^{\mathsf{H}}\boldsymbol{A} = \boldsymbol{A}\boldsymbol{A}^{\mathsf{H}} = \boldsymbol{I}$,
	normal	$\iff$	$\boldsymbol{A}\boldsymbol{A}^{\mathsf{H}} = \boldsymbol{A}^{\mathsf{H}}\boldsymbol{A}$.



A.5 Eigenwerte und Eigenvektoren

Definition A.22. Sei $\boldsymbol{A} \in \mathbb{C}^{n\times n}$. Dann ist $\lambda \in \mathbb{C}$ ein **Eigenwert** von $\boldsymbol{A}$, wenn ein von Null verschiedener Vektor $\underline{x} \in \mathbb{C}^n$ existiert, so dass

$$\boldsymbol{A}\underline{x} = \lambda \underline{x}. \tag{A.5.1}$$

Der Vektor $\underline{x}$ heisst **Rechtseigenvektor** (meist nur **Eigenvektor**) von $\boldsymbol{A}$.

Aus Gleichung (A.5.1) folgt, dass $\boldsymbol{A} - \lambda \boldsymbol{I}_n$ singulär ist, also gilt

$$p_{\boldsymbol{A}}(\lambda) := \det(\boldsymbol{A} - \lambda \boldsymbol{I}) = 0.$$

Die Funktion $p_{\boldsymbol{A}}(\lambda)$ ist ein Polynom vom Grad n in λ und heisst **charakteristisches Polynom** von $\boldsymbol{A}$. Jede Nullstelle von $p_{\boldsymbol{A}}(\lambda)$ ist ein Eigenwert von $\boldsymbol{A}$, und umgekehrt. Daraus folgt, dass $\boldsymbol{A}$ genau n Eigenwerte (inklusive ihrer Vielfachheiten) hat. Die Menge

$$\Lambda(\boldsymbol{A}) = \{\lambda \in \mathbb{C} : \ \lambda \text{ Eigenwert von } \boldsymbol{A}\}$$

heisst **Spektrum** von $\boldsymbol{A}$.

Proposition A.23. *Seien $\lambda_1, \ldots, \lambda_n$ die Eigenwerte einer Matrix $\boldsymbol{A} \in \mathbb{C}^{n\times n}$. Dann gilt:*

(i) $\det(\boldsymbol{A}) = p_{\boldsymbol{A}}(0) = \lambda_1 \lambda_2 \ldots \lambda_n$,

(ii) $\operatorname{spur}(\boldsymbol{A}) = \sum_{i=1}^{n} \lambda_i$,

(iii) $\Lambda(\boldsymbol{A}) = \Lambda(\boldsymbol{A}^{\mathsf{T}})$,

(iv) $\Lambda(\boldsymbol{A}) = \overline{\Lambda(\boldsymbol{A}^{\mathsf{H}})}$, *d.h.* $\lambda \in \Lambda(\boldsymbol{A}) \Longleftrightarrow \overline{\lambda} \in \Lambda(\boldsymbol{A}^{\mathsf{H}})$.

```python
# Eigenwerte einer Matrix A
eigenvalues = np.linalg.eigvals(A)

# V nxn Matrix mit Eigenvektoren und
# D Diagonalmatrix mit Eigenwerten von A.
# Es gilt A*V = V*D.
eigenvalues, V = np.linalg.eig(A)
D = np.diag(eigenvalues)
```

Der **Spektralradius** von $\boldsymbol{A} \in \mathbb{C}^{n\times n}$ ist der grösste Betrag aller Eigenwerte, also

$$\rho(\boldsymbol{A}) := \max_{\lambda\in\Lambda(\boldsymbol{A})} |\lambda| \,.$$

Aus Proposition A.23 folgt $\rho(\boldsymbol{A}) = \rho(\boldsymbol{A}^{\mathsf{H}})$. Die Eigenwerte von $\boldsymbol{A}^k$ sind die k-ten Potenzen der Eigenwerte von $\boldsymbol{A}$, denn $\boldsymbol{A}\underline{x} = \lambda\underline{x}$ impliziert $\boldsymbol{A}^2\underline{x} = \lambda\boldsymbol{A}\underline{x} = \lambda^2\underline{x}$, usw. Insbesondere gilt

$$\rho(\boldsymbol{A}^k) = \big(\rho(\boldsymbol{A})\big)^k,\ k \in \mathbb{N} \,.$$

In der Numerik werden Ähnlichkeitstransformationen benutzt, um eine gegebene Matrix auf eine einfachere Gestalt zu bringen. Dabei wird ausgenutzt, dass eine solche Transformation das Spektrum einer Matrix nicht verändert.

Definition A.24. Sei $\boldsymbol{A} \in \mathbb{K}^{n\times n}$, und sei $\boldsymbol{C} \in \mathbb{K}^{n\times n}$ invertierbar. Dann heisst die Matrix $\boldsymbol{C}^{-1}\boldsymbol{A}\boldsymbol{C}$ **ähnlich** zu $\boldsymbol{A}$.

Proposition A.25. *Sei $\boldsymbol{A} \in \mathbb{K}^{n\times n}$, und sei $\boldsymbol{C} \in \mathbb{K}^{n\times n}$ invertierbar. Dann gilt $p_{\boldsymbol{A}}(\lambda) = p_{\boldsymbol{C}^{-1}\boldsymbol{A}\boldsymbol{C}}(\lambda)$ und damit $\Lambda(\boldsymbol{A}) = \Lambda(\boldsymbol{C}^{-1}\boldsymbol{A}\boldsymbol{C})$.*

Beweis. Aus den Rechenregeln für die Determinante folgt

$$\begin{aligned} p_{\boldsymbol{C}^{-1}\boldsymbol{A}\boldsymbol{C}}(\lambda) &= \det(\boldsymbol{C}^{-1}\boldsymbol{A}\,\boldsymbol{C} - \lambda\underbrace{\boldsymbol{C}^{-1}\boldsymbol{C}}_{\boldsymbol{I}}) \\ &= \det(\boldsymbol{C}^{-1}(\boldsymbol{A} - \lambda\boldsymbol{I})\,\boldsymbol{C}) \\ &= \det(\boldsymbol{C}^{-1})\,\det(\boldsymbol{A} - \lambda\boldsymbol{I})\,\det(\boldsymbol{C}) \\ &= \det(\boldsymbol{C})^{-1}\,\det(\boldsymbol{A} - \lambda\boldsymbol{I})\,\det(\boldsymbol{C}) \\ &= p_{\boldsymbol{A}}(\lambda) \,. \end{aligned}$$

□

A.6 Vektornormen

Wir kommen jetzt zu den beiden wichtigsten Abschnitten in diesem Kapitel, den Vektor- und Matrixnormen. Erst diese werden es uns erlauben, den Fehler einer numerischen Berechnung sinnvoll zu quantifizieren.

Definition A.26. Sei V Vektorraum über $\mathbb{K} \in \{\mathbb{R}, \mathbb{C}\}$. Eine **Norm** auf V ist eine Abbildung $\|\cdot\| : V \to \mathbb{R}$ mit den folgenden Eigenschaften:

$$\forall \underline{x} \in V : \ \|\underline{x}\| \geq 0 \ \text{ und } \ \|\underline{x}\| = 0 \Leftrightarrow \underline{x} = \underline{0} \, . \tag{N1}$$

$$\forall \underline{x} \in V, \, \alpha \in \mathbb{K} : \ \|\alpha \underline{x}\| = |\alpha| \, \|\underline{x}\| \, . \tag{N2}$$

$$\forall \underline{x}, \underline{y} \in V : \ \|\underline{x} + \underline{y}\| \leq \|\underline{x}\| + \|\underline{y}\| \, . \tag{N3}$$

(N3) wird auch **Dreiecksungleichung** genannt. Es gilt die folgende Umkehrung.

Proposition A.27. *Sei $\|\cdot\| : V \to \mathbb{R}$ Norm auf einem Vektorraum V. Dann gilt $\big|\|\underline{x}\| - \|\underline{y}\|\big| \leq \|\underline{x} - \underline{y}\|$ für alle $\underline{x}, \underline{y} \in V$.*

Beweis. Aus (N3) ergibt sich

$$\|\underline{x}\| - \|\underline{y}\| = \|\underline{x} - \underline{y} + \underline{y}\| - \|\underline{y}\| \leq \|\underline{x} - \underline{y}\| + \|\underline{y}\| - \|\underline{y}\| = \|\underline{x} - \underline{y}\|.$$

Durch Vertauschen von $\underline{x}$ und $\underline{y}$ folgt $\|\underline{y}\| - \|\underline{x}\| \leq \|\underline{x} - \underline{y}\|$ und damit die Behauptung. □

Proposition A.27 zieht insbesondere auch **Stetigkeit der Norm**, als Abbildung von V nach $\mathbb{R}$, nach sich.

Auf $V = \mathbb{R}^n$ und $V = \mathbb{C}^n$ ist in den meisten Anwendungen nur eine Klasse von Normen von Interesse, die sogenannten ℓ^p-Vektornormen $\|\cdot\|_p$ für $1 \leq p \leq \infty$: Für einen Vektor $\underline{x} = (x_1, \ldots, x_n)^\mathsf{T} \in \mathbb{C}^n$ ist diese wie folgt definiert:

$$\|\underline{x}\|_p \ := \ \Big(\sum_{i=1}^{n} |x_i|^p \Big)^{\frac{1}{p}}, \ \ 1 \leq p < \infty, \tag{A.6.1}$$

$$\|\underline{x}\|_\infty \ := \ \max\{|x_i| : i = 1, \ldots, n\} \, . \tag{A.6.2}$$

Die wichtigsten Spezialfälle sind $p = 1$, $p = 2$ (**Euklidische Norm**), und $p = \infty$.

```
np.linalg.norm(v)              # 2-Norm eines Vektors v
np.linalg.norm(v, ord=p)       # p-Norm eines Vektors v
np.linalg.norm(v, ord=np.inf) # oo-Norm eines Vektors v
```

Satz A.28. *Die in (A.6.1) und (A.6.2) für $1 \le p \le \infty$ definierte Abbildung $\|\cdot\|_p : \mathbb{C}^n \to \mathbb{R}$ ist eine Norm auf $\mathbb{C}^n$.*

Beweis von Satz A.28* Der Beweis von (N1) und (N2) ist offensichtlich, aber um (N3) zeigen zu können, benötigen wir zunächst einige Hilfsresultate.

Lemma A.29 (Young'sche Ungleichung)**.** *Sei $1 \le p, q \le \infty$ mit $\frac{1}{p} + \frac{1}{q} = 1$. Dann gilt für alle $a, b > 0$,*

$$ab \le \frac{a^p}{p} + \frac{b^q}{q}. \tag{A.6.3}$$

Beweis. Die Abbildung $x \mapsto \log x$ ist konkav, d.h.,

$$\forall \alpha, \beta \ge 0,\ \alpha + \beta = 1 :\ \forall u, v > 0 :\quad \alpha \log u + \beta \log v \le \log(\alpha u + \beta v)$$

Setzen wir $\alpha = \frac{1}{p},\ \beta = \frac{1}{q}$ und $u = a^p,\ v = b^q$, so folgt

$$\log(ab) = \log a + \log b = \alpha \log(a^p) + \beta \log(b^q) \le \log\Big(\frac{1}{p}\, a^p + \frac{1}{q}\, b^q\Big),$$

und damit (A.6.3). □

Satz A.30 (Hölder'sche Ungleichung)**.** *Sei $1 \le p, q \le \infty$ mit $\frac{1}{p} + \frac{1}{q} = 1$. Dann gilt für alle $\underline{x}, \underline{y} \in \mathbb{C}^n$,*

$$\Big|\sum_{i=1}^n x_i\, \overline{y}_i\Big| \le \|\underline{x}\|_p\ \|\underline{y}\|_q\,.$$

Beweis. Die Fälle $\underline{x} = \underline{0}$ und $\underline{y} = \underline{0}$ sind trivial. Seien also $\underline{x} \ne \underline{0}$, $\underline{y} \ne \underline{0}$. Setze

$$\widehat{\underline{x}} = \underline{x} \,/\, \|\underline{x}\|_p,\ \ \widehat{\underline{y}} = \underline{y} \,/\, \|\underline{y}\|_q\,.$$

Aus Lemma A.29 folgt

$$|\widehat{x}_i|\ |\widehat{y}_i| \le \frac{|\widehat{x}_i|^p}{p} + \frac{|\widehat{y}_i|^q}{q},\quad i = 1, \dots, n\,,$$

$$\sum_{i=1}^n |\widehat{x}_i|\ |\widehat{y}_i| \le \frac{1}{p}\ \underbrace{\|\widehat{\underline{x}}\|_p^p}_{=1} + \frac{1}{q}\ \underbrace{\|\widehat{\underline{y}}\|_q^q}_{=1} = 1\,.$$

□

Damit können wir nun die Dreiecksungleichung (N3) für die $\|\cdot\|_p$-Vektornorm und damit Satz A.28 beweisen.

Satz A.31 (Minkowski-Ungleichung)**.** *Seien $1 \le p \le \infty$, $\underline{x}, \underline{y} \in \mathbb{C}^n$. Dann gilt*

$$\|\underline{x} + \underline{y}\|_p \le \|\underline{x}\|_p + \|\underline{y}\|_p\,.$$

Beweis.

$$\begin{aligned}\|\underline{x}+\underline{y}\|_p^p &= \sum_{i=1}^n |x_i+y_i|^p \le \sum_{i=1}^n \left(|x_i|+|y_i|\right)|x_i+y_i|^{p-1} \\ &\overset{\text{Hölder}}{\le} \left(\|\underline{x}\|_p+\|\underline{y}\|_p\right)\Big(\sum_{i=1}^n \left(|x_i+y_i|^{p-1}\right)^q\Big)^{\frac{1}{q}} \\ &= \left(\|\underline{x}\|_p+\|\underline{y}\|_p\right)\|\underline{x}+\underline{y}\|_p^{p/q} = \left(\|\underline{x}\|_p+\|\underline{y}\|_p\right)\|\underline{x}+\underline{y}\|_p^{p-1},\end{aligned}$$

wobei benutzt wurde, dass $(p-1)\,q = p$, $\frac{1}{q} = (p-1)/p = 1-\frac{1}{p}$. □

Die Euklidische Norm hat die besondere Eigenschaft, dass sie durch das Standard-Skalarprodukt (auch: **Euklidisches Skalarprodukt**)

$$(\underline{x},\,\underline{y}) = \underline{y}^{\mathsf{H}}\underline{x} = \sum_{i=1}^n x_i\,\overline{y}_i$$

auf dem $\mathbb{R}^n$ bzw. $\mathbb{C}^n$ induziert wird. Wir erinnern an die Definition eines Skalarproduktes.

Definition A.32. Sei V ein Vektorraum über $\mathbb{K}$. Eine Abbildung $(\cdot,\cdot) : V \times V \to \mathbb{K}$ heisst **Skalarprodukt** auf V, falls gilt:

$$\forall \underline{x},\underline{y},\underline{z} \in V, \quad \forall \gamma,\lambda \in \mathbb{K}:\ (\gamma\underline{x}+\lambda\underline{y},\underline{z}) = \gamma(\underline{x},\underline{z}) + \lambda(\underline{y},\underline{z})\,, \tag{S1}$$

$$(\underline{y},\underline{x}) = \overline{(\underline{x},\underline{y})} \tag{S2}$$

$$(\underline{x},\underline{x}) > 0 \quad \text{für alle } \underline{x} \neq \underline{0}\,. \tag{S3}$$

Jedes Skalarprodukt auf V induziert eine Norm durch

$$\|\underline{x}\| = \sqrt{(\underline{x},\underline{x})}\,. \tag{A.6.4}$$

Es stellt sich die Frage, *ob auf jedem Vektorraum V mit Norm $\|\circ\|$ ein dazugehöriges Skalarprodukt mit* (A.6.4) *gefunden werden kann.* Ohne Beweis geben wir dafür den folgenden Satz an.

Satz A.33 (Parallelogrammgleichung)**.** *Auf dem normierten Vektorraum V über dem Körper $\mathbb{K}$ kann ein zur Norm $\|\circ\|$ kompatibles Skalarprodukt $(\cdot,\cdot)$ genau dann eingeführt werden, wenn die Norm $\|\circ\|$ die* Parallelogrammgleichung

$$\|\underline{x}+\underline{y}\|^2 + \|\underline{x}-\underline{y}\|^2 = 2(\|\underline{x}\|^2+\|\underline{y}\|^2) \quad \forall \underline{x},\underline{y} \in V \tag{A.6.5}$$

erfüllt.

Für $\mathbb{K} = \mathbb{R}$ ist das zugehörige Skalarprodukt gegeben durch die Polargleichung

$$(\underline{x},\underline{y}) = \frac{1}{4}\left(\|\underline{x}+\underline{y}\|^2 - \|\underline{x}-\underline{y}\|^2\right) \quad \forall \underline{x},\underline{y} \in V\,. \tag{A.6.6}$$

Falls $\mathbb{K} = \mathbb{C}$ ist das zugehörige Skalarprodukt gegeben durch

$$(\underline{x}, \underline{y}) = \frac{1}{4}\left(\|\underline{x}+\underline{y}\|^2 - \|\underline{x}-\underline{y}\|^2 + \mathrm{i}\|\underline{x}+\mathrm{i}\underline{y}\|^2 - \mathrm{i}\|\underline{x}-\mathrm{i}\underline{y}\|^2\right) \quad \forall \underline{x}, \underline{y} \in V\ . \tag{A.6.7}$$

Satz A.34 (Cauchy-Schwarzsche Ungleichung)**.** *Sei V Vektorraum mit Skalarprodukt $(\cdot,\cdot) : V \times V \to \mathbb{K}$ und der dadurch induzierten Norm $\|\cdot\| : V \to \mathbb{K}$. Dann gilt für jede $\underline{x}, \underline{y} \in V$:*

$$|(\underline{x}, \underline{y})| \leq \|\underline{x}\|\,\|\underline{y}\|\ .$$

Gleichheit gilt genau dann, wenn $\underline{x}$ und $\underline{y}$ linear abhängig sind.

Beweis. Hier nur der Beweis der Ungleichung für $V = \mathbb{C}^n$ mit dem Euklidischen Skalarprodukt: Entspricht der Hölderschen Ungleichung (Satz A.30) für den Spezialfall $p = q = 2$. □

Lemma A.49 zeigt, dass orthogonale und unitäre

In der Praxis ist die Wahl einer geeigneten Norm oft entscheidend für eine Quantifizierung des Fehlers. Vom Standpunkt der Theorie hingegen gibt es keinen signifikanten mathematischen Unterschied zwischen zwei zueinander *äquivalenten* Normen.

Definition A.35. Sei V Vektorraum. Zwei Normen $\|\cdot\|_V : V \to \mathbb{R}$ und $\|\cdot\|_W : V \to \mathbb{R}$ heissen **äquivalent**, wenn Konstanten $c > 0$ und $C > 0$ existieren, so dass

$$c\|\underline{x}\|_W \leq \|\underline{x}\|_V \leq C\|\underline{x}\|_W, \qquad \forall \underline{x} \in V. \tag{A.6.8}$$

Es ist aus der Analysis (Widerspruchsargument beruhend auf Kompaktheit abgeschl. und beschränkter Teilmengen des $\mathbb{R}^n$ [→ Satz von Heine-Borel]) bekannt, dass auf einem *endlichdimensionalen* Vektorraum *alle* Normen zueinander äquivalent sind, insbesondere sind also alle $\|\cdot\|_p$-Vektornormen auf $\mathbb{C}^n$ äquivalent (allerdings mit Konstanten c und C in (A.6.8), die von n und von p abhängen). In der Numerik sind wir an der Dimensionsabhängigkeit der Äquivalenzkonstanten c, C in (A.6.8) interessiert.

Beispiel A.36. Für alle $\underline{x} \in \mathbb{C}^n$ und alle $1 \leq p_1 \leq p_2 \leq \infty$ gilt

$$\|\underline{x}\|_{p_2} \leq \|\underline{x}\|_{p_1} \leq n^{\left(\frac{1}{p_1}-\frac{1}{p_2}\right)}\|\underline{x}\|_{p_2}\ .$$

Als Spezialfälle folgen

$$\begin{aligned}\|\underline{x}\|_2 \leq \|\underline{x}\|_1 &\leq \sqrt{n}\,\|\underline{x}\|_2,\\ \|\underline{x}\|_\infty \leq \|\underline{x}\|_2 &\leq \sqrt{n}\,\|\underline{x}\|_\infty,\\ \|\underline{x}\|_\infty \leq \|\underline{x}\|_1 &\leq n\,\|\underline{x}\|_\infty\ .\end{aligned}$$

◇

Verlust der Normäquivalenz im Unendlichdimensionalen* Sei $V = C^0([0,1])$, der Raum aller auf dem Intervall $[0,1]$ stetigen Funktionen. V ist ein unendlichdimensionaler Vektorraum, auf dem die folgenden beiden Normen definiert werden können:

$$\|f\|_\infty = \sup_{x\in[0,1]} |f(x)|, \quad \|f\|_2 = \left(\int_0^1 f(x)^2 \, \mathrm{d}x\right)^{1/2}$$

für eine Funktion $f \in C^0([0,1])$. Betrachte nun

$$f_\theta(x) = \begin{cases} (\theta - x)/\theta & \text{für } x \in [0,\theta], \\ 0 & \text{sonst,} \end{cases}$$

für ein $\theta \in (0,1]$. Dann ist

$$\|f_\theta\|_\infty = 1, \quad \|f_\theta\|_2 = \sqrt{\frac{\theta}{3}}.$$

Also gibt es keine Konstanten $c, C > 0$ unabhängig von θ, so dass (A.6.8) für diese beiden Normen erfüllt sein könnte.

A.7 Matrixnormen

Analog zur Länge eines Vektors messen wir auch die Grösse von Matrizen durch Normen, den sogenannten Matrixnormen.

Definition A.37. Eine Abbildung $\|\cdot\|_M : \mathbb{C}^{m\times n} \to \mathbb{R}$ heisst **Matrixnorm**, falls sie die folgenden Eigenschaften erfüllt:

$$\|\boldsymbol{A}\|_M \geq 0 \text{ und } \|\boldsymbol{A}\|_M = 0 \Leftrightarrow \boldsymbol{A} = \boldsymbol{0}\,, \tag{N1}$$

$$\|\alpha\boldsymbol{A}\|_M = |\alpha| \, \|\boldsymbol{A}\|_M \quad \forall \alpha \in \mathbb{C}, \quad \forall \boldsymbol{A} \in \mathbb{C}^{m\times n}\,, \tag{N2}$$

$$\|\boldsymbol{A} + \boldsymbol{B}\|_M \leq \|\boldsymbol{A}\|_M + \|\boldsymbol{B}\|_M \quad \forall \boldsymbol{A}, \boldsymbol{B} \in \mathbb{C}^{m\times n}\,. \tag{N3}$$

In der Numerik sind meist nur Matrixnormen von Interesse, mit denen sich obere Schranken für die Vektornorm eines Matrix-Vektor-Produkts angeben lassen.

Definition A.38. Eine Matrixnorm $\|\cdot\|_M$ auf $\mathbb{C}^{m\times n}$ heisst **konsistent** (oder **verträglich**) mit den Vektornormen $\|\cdot\|_V : \mathbb{C}^n \to \mathbb{R}$ und $\|\cdot\|_W : \mathbb{C}^m \to \mathbb{R}$, falls

$$\|\boldsymbol{A}\underline{x}\|_W \leq \|\boldsymbol{A}\|_M \|\underline{x}\|_V \quad \forall \underline{x} \in \mathbb{C}^n\,.$$

Beispiel A.39. Die **Frobeniusnorm** auf $\mathbb{C}^{m\times n}$ ist definiert durch

$$\|\boldsymbol{A}\|_F := \left(\sum_{i=1}^m \sum_{j=1}^n |a_{ij}|^2\right)^{\frac{1}{2}} = \operatorname{tr}\left(\boldsymbol{A}\boldsymbol{A}^{\mathsf{H}}\right)^{\frac{1}{2}}.$$

Die Frobenius Matrixnorm $\|\cdot\|_F$ ist konsistent mit der Euklidischen Vektornorm $\|\cdot\|_2$ auf $\mathbb{C}^n$ und $\mathbb{C}^m$:

$$\|\boldsymbol{A}\underline{x}\|_2^2 = \sum_{i=1}^{m} \Big|\sum_{j=1}^{n} a_{ij}\, x_j\Big|^2 \le \sum_{i=1}^{m} \Big(\sum_{j=1}^{n} |a_{ij}|^2 \sum_{j=1}^{n} |x_j|^2\Big) = \|\boldsymbol{A}\|_F^2\, \|\underline{x}\|_2^2,$$

wobei die Cauchy-Schwarzsche Ungleichung (Satz A.34) angewandt wurde. ◇

Definition A.40. Seien $\|\cdot\|_V$, $\|\cdot\|_W$ Vektornormen auf $\mathbb{C}^n$ bzw. $\mathbb{C}^m$. Dann heisst

$$\|\boldsymbol{A}\|_{VW} := \sup_{\underline{x}\neq\underline{0}} \frac{\|\boldsymbol{A}\underline{x}\|_W}{\|\underline{x}\|_V} = \sup_{\|\underline{x}\|_V=1} \|\boldsymbol{A}\underline{x}\|_W. \tag{A.7.1}$$

die durch $\|\cdot\|_V$, $\|\cdot\|_W$ **induzierte Matrixnorm** auf $\mathbb{C}^{m\times n}$.

Dass die in (A.7.1) definierte Funktion $\|\cdot\|_{VW}$ tatsächlich eine Matrixnorm im Sinne von Definition A.37 ist, folgt aus den Eigenschaften (N1), (N2), (N3) der Vektornorm $\|\cdot\|_W$. Da $\|\boldsymbol{A}\|_{VW}$ nicht über die Einträge von $\boldsymbol{A}$ sondern mittels der *Operation* Matrix-Vektor-Produkt definiert wird, nennt man eine induzierte Norm auch **Operatornorm**. Aus (A.7.1) folgt sofort, dass $\|\cdot\|_{VW}$ mit den Vektornormen $\|\cdot\|_V$ und $\|\cdot\|_W$ konsistent ist, d.h.

$$\forall \underline{x} \in \mathbb{C}^n: \qquad \|\boldsymbol{A}\underline{x}\|_W \le \|\boldsymbol{A}\|_{VW}\, \|\underline{x}\|_V,$$

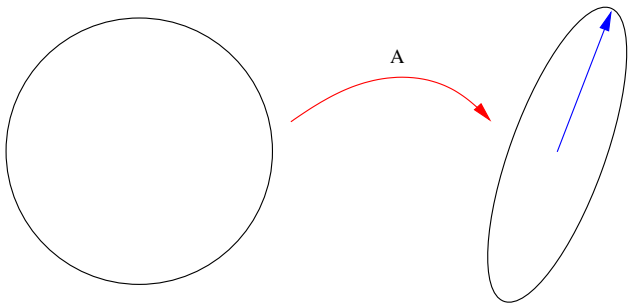
Bemerkung A.41. Für jede induzierte Norm $\|\cdot\|_{VW}$ mit $V = W$ hat die Einheitsmatrix $\boldsymbol{I}_n$ Norm 1. Auf der anderen Seite gilt $\|\boldsymbol{I}_n\|_F = \sqrt{n}$. Also ist die Frobeniusnorm *keine* induzierte Norm.

Beispiel A.42 ($\|\cdot\|_p$ - Matrixnormen)**.** Sei $1 \le p \le \infty$. Dann ist die von der $\|\cdot\|_p$ Vektornorm induzierte Matrixnorm definiert durch

$$\|\boldsymbol{A}\|_p := \max_{\underline{x}\neq\underline{0}} \frac{\|\boldsymbol{A}\underline{x}\|_p}{\|\underline{x}\|_p} = \max_{\|x\|_p=1} \|\boldsymbol{A}\underline{x}\|_p. \tag{A.7.2}$$

Übung: Zeige, dass die $\|\cdot\|_p$ - Matrixnorm wohldefiniert ist (warum kann man sup in (A.7.1) durch max in (A.7.2) ersetzen?).

Die rechte Seite von (A.7.2) lässt sich für $\boldsymbol{A} \in \mathbb{R}^{2\times 2}$ geometrisch gut veranschaulichen. Für $p = 2$ bildet die Menge $\{\underline{x} \in \mathbb{R}^2 : \|\underline{x}\|_2 = 1\}$ den Einheitskreis. Dieser wird von A auf eine Ellipse abgebildet. Der maximale Radius dieser Ellipse ist das Supremum von $\{\|\boldsymbol{A}\underline{x}\|_2 : x \in \mathbb{R}^2, \|\underline{x}\|_2 = 1\}$ und damit die 2-Norm von $\boldsymbol{A}$:



Analog bildet für $p \in \{1, \infty\}$ die Matrix $\boldsymbol{A} \in \mathbb{R}^{2\times 2}$ ein Quadrat auf ein Parallelogram ab. Nur für $p \in \{1, \infty\}$ lassen sich auch einfache Formeln für die Berechnung von $\|\boldsymbol{A}\|_p$ angeben:

$$\|\boldsymbol{A}\|_1 = \max_{j=1,\dots,n} \sum_{i=1}^{m} |a_{ij}| \text{ (Spaltensummennorm)}$$

$$\|\boldsymbol{A}\|_\infty = \max_{i=1,\dots,m} \sum_{j=1}^{n} |a_{ij}| \text{ (Zeilensummennorm)}$$

```
np.linalg.norm(A, ord=1)       # 1-Norm
np.linalg.norm(A, ord=2)       # 2-Norm
np.linalg.norm(A, ord=np.inf) # oo-Norm
np.linalg.norm(A, ord='fro')  # Frobeniusnorm
```

Eine in der Praxis überaus wichtige Eigenschaft von Matrixnormen ist *Submultiplikativität.*

Definition A.43. Betrachte eine Klasse von Matrixnormen $\|\cdot\|_M : \mathbb{C}^{k\times l} \to \mathbb{R}$, die für alle $k, l \geq 1$ definiert ist. Dann heisst $\|\cdot\|_M$ **submultiplikativ**, falls

$$\forall \boldsymbol{A} \in \mathbb{C}^{m\times n},\ \boldsymbol{B} \in \mathbb{C}^{n\times q} :\ \|\boldsymbol{AB}\|_M \leq \|\boldsymbol{A}\|_M \, \|\boldsymbol{B}\|_M \,.$$

Die meisten, aber nicht alle Matrixnormen sind submultiplikativ.

Beispiel A.44. Für eine $m \times n$ Matrix $\boldsymbol{A}$, sei $\|\boldsymbol{A}\|_{\max} := \max_{i=1,\dots,m} \max_{j=1,\dots,n} |a_{ij}|$. Seien

$$\boldsymbol{A} = \boldsymbol{B} = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}, \quad \boldsymbol{AB} = \begin{pmatrix} 2 & 2 \\ 2 & 2 \end{pmatrix}.$$

Also haben wir $\|\boldsymbol{AB}\|_{\max} = 2$, $\|\boldsymbol{A}\|_{\max}\|\boldsymbol{B}\|_{\max} = 1$, und damit ist $\|\cdot\|_{\max}$ nicht submultiplikativ. $\diamond$

Proposition A.45. *Sei wieder $1 \leq p \leq \infty$. Dann ist die von der $\|\cdot\|_p$-Vektornorm induzierte Matrixnorm submultiplikativ.*

Beweis.

$$\begin{aligned} \|\boldsymbol{AB}\|_p &= \max_{\underline{x}\neq\underline{0}} \frac{\|\boldsymbol{AB}\underline{x}\|_p}{\|\underline{x}\|_p} = \max_{\substack{\underline{x}\neq\underline{0} \\ \boldsymbol{B}\underline{x}\neq\underline{0}}} \frac{\|\boldsymbol{AB}\underline{x}\|_p}{\|\boldsymbol{B}\underline{x}\|_p} \frac{\|\boldsymbol{B}\underline{x}\|_p}{\|\underline{x}\|_p} \\ &\leq \max_{\substack{\underline{x}\neq\underline{0} \\ \boldsymbol{B}\underline{x}\neq\underline{0}}} \frac{\|\boldsymbol{AB}\underline{x}\|_p}{\|\boldsymbol{B}\underline{x}\|_p} \max_{\underline{x}\neq\underline{0}} \frac{\|\boldsymbol{B}\underline{x}\|_p}{\|\underline{x}\|_p} \leq \|\boldsymbol{A}\|_p \, \|\boldsymbol{B}\|_p \,. \end{aligned}$$

□

Proposition A.46. *Die Frobeniusnorm $\|\cdot\|_F$ ist submultiplikativ.*

Beweis. Sei

$$\boldsymbol{A} = (\underline{a}_1, \ldots, \underline{a}_n) \in \mathbb{C}^{m\times n},\ \boldsymbol{B} = \begin{pmatrix} \underline{b}_1^{\mathsf{T}} \\ \vdots \\ \underline{b}_n^{\mathsf{T}} \end{pmatrix} \in \mathbb{C}^{n\times k}\,.$$

Dann hat das Matrixprodukt $\boldsymbol{AB}$ folgende Darstellung als Summe von Rang-1 Matrizen, die durch Multiplikation von Spaltenvektoren $\underline{a}_i$ von $\boldsymbol{A}$ mit Zeilenvektoren $\underline{b}_j^{\mathsf{T}}$ von $\boldsymbol{B}$ gewonnen werden:

$$\boldsymbol{AB} = \underline{a}_1\,\underline{b}_1^{\mathsf{T}} + \cdots + \underline{a}_n\,\underline{b}_n^{\mathsf{T}}.$$

Anwendung der Dreiecksungleichung (N3) für die Frobeniusnorm ergibt

$$\begin{aligned}
\|\boldsymbol{AB}\|_F &= \|\underline{a}_1\,\underline{b}_1^{\mathsf{T}} + \cdots + \underline{a}_n\,\underline{b}_n^{\mathsf{T}}\|_F \\
&\le \|\underline{a}_1\,\underline{b}_1^{\mathsf{T}}\|_F + \cdots + \|\underline{a}_n\,\underline{b}_n^{\mathsf{T}}\|_F \\
&= \|\underline{a}_1\|_2\,\|\underline{b}_1\|_2 + \cdots + \|\underline{a}_n\|_2\,\|\underline{b}_n\|_2 \\
&\le \Big(\sum_{i=1}^{n} \|\underline{a}_i\|_2^2\Big)^{\frac{1}{2}} \Big(\sum_{i=1}^{n} \|\underline{b}_i\|_2^2\Big)^{\frac{1}{2}} \\
&= \|\boldsymbol{A}\|_F\,\|\boldsymbol{B}\|_F\,.
\end{aligned}$$

□

Wie bei den Vektornormen lassen sich auch bei den Matrixnormen gewisse Normäquivalenzen zeigen. Zum Beispiel gilt für $\boldsymbol{A} \in \mathbb{C}^{m\times n}$:

$$\begin{aligned}
\|\boldsymbol{A}\|_2 \le \|\boldsymbol{A}\|_F &\le \sqrt{\operatorname{rang}(\boldsymbol{A})}\,\|\boldsymbol{A}\|_2, \\
\frac{1}{\sqrt{n}}\|\boldsymbol{A}\|_\infty \le \|\boldsymbol{A}\|_2 &\le \sqrt{m}\,\|\boldsymbol{A}\|_\infty, \\
\frac{1}{\sqrt{m}}\|\boldsymbol{A}\|_1 \le \|\boldsymbol{A}\|_2 &\le \sqrt{n}\,\|\boldsymbol{A}\|_1.
\end{aligned}$$

A.8 Orthogonalität

Definition A.47. Sei V Vektorraum mit Skalarprodukt $(\cdot,\cdot)$. Zwei Vektoren $\underline{x}$ und $\underline{y}$ heissen **orthogonal** (oder senkrecht) bezüglich $(\cdot,\cdot)$, falls

$$(\underline{x}, \underline{y}) = 0$$

gilt. Symbolisch schreiben wir $\underline{x} \perp \underline{y}$.

Im folgenden werden wir uns auf den Fall beschränken, dass V endlichdimensional ist. Analog zu Definition A.47 heisst eine Basis $\{\underline{x}_1, \ldots, \underline{x}_n\}$ von V **Orthogonalbasis**, falls

$$\underline{x}_i \perp \underline{x}_j, \quad \forall i \neq j.$$

Gilt zusätzlich $\|\underline{x}_i\| = \sqrt{(\underline{x}_i, \underline{x}_i)} = 1$ für alle i, dann heisst $\{\underline{x}_1, \ldots, \underline{x}_n\}$ **Orthonormalbasis**.

Lemma A.48. Sei $\boldsymbol{Q} = (\underline{q}_1, \ldots, \underline{q}_n) \in \mathbb{C}^{n\times n}$. $\boldsymbol{Q}$ ist genau dann unitär, wenn die Spalten $\underline{q}_1, \ldots, \underline{q}_n$ eine Orthonormalbasis von $\mathbb{C}^n$ bezüglich dem Euklidischen Skalarprodukt bilden.

Beweis. Folgt sofort aus der Beziehung $\boldsymbol{Q}^{\mathsf{H}}\boldsymbol{Q} = \boldsymbol{I}_n$. □

Die für die Numerik wichtigste Eigenschaft unitärer Matrizen ist, dass sie die Euklidische Norm eines Vektors $\underline{x}$ nicht verändern.

Lemma A.49. *Sei $\boldsymbol{Q} \in \mathbb{C}^{n\times n}$ unitär. Dann gilt $\|\boldsymbol{Q}\underline{x}\|_2 = \|\underline{x}\|_2$ für alle $\underline{x} \in \mathbb{C}^n$.*

Beweis. $\|\boldsymbol{Q}\underline{x}\|_2 = \sqrt{(\boldsymbol{Q}\underline{x})^{\mathsf{H}}\boldsymbol{Q}\underline{x}} = \sqrt{\underline{x}^{\mathsf{H}}\boldsymbol{Q}^{\mathsf{H}}\boldsymbol{Q}\underline{x}} = \sqrt{\underline{x}^{\mathsf{H}}\underline{x}} = \|\underline{x}\|_2$. □

Als Korollar folgt, dass unitäre Matrizen die 2- und Frobeniusnorm einer Matrix erhalten.[1]

Proposition A.50. *Seien $\boldsymbol{U} \in \mathbb{C}^{m\times m}$ und $\boldsymbol{V} \in \mathbb{C}^{n\times n}$ unitär. Dann gilt für alle $\boldsymbol{A} \in \mathbb{C}^{m\times n}$*

$$\|\boldsymbol{U}\boldsymbol{A}\|_2 = \|\boldsymbol{A}\boldsymbol{V}\|_2 = \|\boldsymbol{A}\|_2, \text{ und} \quad \|\boldsymbol{U}\boldsymbol{A}\|_F = \|\boldsymbol{A}\boldsymbol{V}\|_F = \|\boldsymbol{A}\|_F .$$

Beweis. Das Resultat für die 2-Norm von $\boldsymbol{U}\boldsymbol{A}$ folgt mit Lemma A.49 direkt aus der Definition:

$$\|\boldsymbol{U}\boldsymbol{A}\|_2 = \max_{\|\underline{x}\|_2=1} \|\boldsymbol{U}\boldsymbol{A}x\|_2 = \max_{\|\underline{x}\|_2=1} \|\boldsymbol{A}x\|_2 = \|\boldsymbol{A}\|_2.$$

Für die Frobeniusnorm partitionieren wir $\boldsymbol{A} = (\underline{a}_1, \ldots, \underline{a}_n)$ und erhalten mit Lemma A.49

$$\|\boldsymbol{U}\boldsymbol{A}\|_F^2 = \|\boldsymbol{U}\underline{a}_1\|_2^2 + \cdots + \|\boldsymbol{U}\underline{a}_n\|_2^2 = \|\underline{a}_1\|_2^2 + \cdots + \|\underline{a}_n\|_2^2 = \|\boldsymbol{A}\|_F^2.$$

Der Beweis für $\boldsymbol{A}\boldsymbol{V}$ erfolgt analog (nachrechnen!). □

A.8.1 Orthogonales Komplement und Projektoren

Seien im folgenden V und W Vektorräume endlicher Dimension.

[1] Matrixnormen mit dieser Eigenschaft heissen **unitär invariant**.

Definition A.51. Sei W Teilraum eines Vektorraums V. Dann bezeichnet

$$W^\perp = \{\underline{y} \in V : \underline{y} \perp \underline{x} \text{ für alle } \underline{x} \in W\}$$

das **orthogonale Komplement** von W (in V).

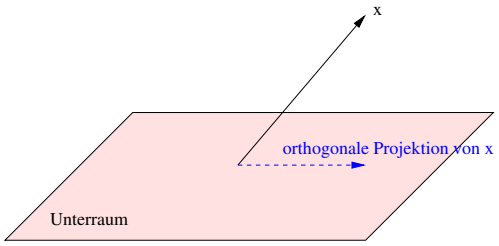
Jeder Vektor $\underline{x} \in V$ lässt sich als Summe von Vektoren des Teilraums und dessen orthogonalen Komplement schreiben:

$$\underline{x} = \underline{x}_W + \underline{x}_{W^\perp}, \quad \underline{x}_W \in W, \quad \underline{x}_{W^\perp} \in W^\perp. \tag{A.8.1}$$

Diese Zerlegung ist eindeutig (Beweis Übung). Damit lässt sich V als direkte Summe (siehe Beispiel A.4.4) schreiben:

$$V = W \oplus W^\perp.$$

Der Vektor $\underline{x}_W$ heisst *orthogonale Projektion* von $\underline{x}$ auf W. Dieser hat von allen Vektoren aus W den geringsten Abstand zu $\underline{x}$.



Lemma A.52. *Sei W Teilraum eines Vektorraums V und sei $\underline{x} \in V$. Dann gilt für die orthogonale Projektion $\underline{x}_W$ von $\underline{x}$ auf W:*

$$\|\underline{x} - \underline{x}_W\|_2 = \min_{\underline{w} \in W} \|\underline{x} - \underline{w}\|_2.$$

Beweis. Aus der Zerlegung (A.8.1) folgt

$$\|\underline{x} - \underline{w}\|_2^2 = \|(\underline{x}_W - \underline{w}) + \underline{x}_{W^\perp}\|_2^2 = \|\underline{x}_W - \underline{w}\|_2^2 + \|\underline{x}_{W^\perp}\|_2^2 \geq \|\underline{x}_{W^\perp}\|_2^2$$

für alle $w \in W$. Dabei wurde ausgenutzt, dass die $\underline{x}_W - \underline{w} \in W$ und $\underline{x}_{W^\perp} \in W^\perp$ orthogonal zueinander stehen (Satz des Pythagoras). Auf der anderen Seite gilt $\|\underline{x} - \underline{x}_W\|_2 = \|\underline{x}_{W^\perp}\|_2$. Damit minimiert $\underline{x}_W$ den Ausdruck $\|\underline{x} - \underline{w}\|_2$. □

Um $\underline{x}_W$ zu berechnen, benötigt man lediglich eine Orthonormalbasis für W.

Satz A.53. *Sei W k-dimensionaler Teilraum von $\mathbb{C}^n$ und $\underline{x} \in \mathbb{C}^n$. Bilden die Spalten von $\boldsymbol{W} \in \mathbb{C}^{n \times k}$ eine Orthonormalbasis für W, so ist $\boldsymbol{W}\boldsymbol{W}^{\mathsf{H}}x$ die orthogonale Projektion von $\underline{x}$ auf W und $(\boldsymbol{I}_n - \boldsymbol{W}\boldsymbol{W}^{\mathsf{H}})\underline{x}$ die orthogonale Projektion von $\underline{x}$ auf $W^\perp$.*

Beweis. Für einen Vektor $\underline{w} \in W$ gibt es $\widetilde{\underline{w}} \in \mathbb{C}^k$, so dass $\underline{w} = \boldsymbol{W}\widetilde{\underline{w}}$. Also gilt

$$\boldsymbol{W}\boldsymbol{W}^{\mathsf{H}}\underline{w} = \boldsymbol{W}\underbrace{\boldsymbol{W}^{\mathsf{H}}\boldsymbol{W}}_{=\boldsymbol{I}_k}\widetilde{\underline{w}} = \underline{w},$$

und damit

$$(\underline{w}, (\boldsymbol{I}_n - \boldsymbol{W}\boldsymbol{W}^{\mathsf{H}})\underline{x}) = \underline{w}^{\mathsf{H}}(\boldsymbol{I}_n - \boldsymbol{W}\boldsymbol{W}^{\mathsf{H}})\underline{w} = 0,\ \forall \underline{w} \in W \ \Rightarrow\ (\boldsymbol{I}_n - \boldsymbol{W}\boldsymbol{W}^{\mathsf{H}})\underline{x} \in W^{\perp}.$$

Mit $\boldsymbol{W}\boldsymbol{W}^{\mathsf{H}}\underline{x} \in \mathrm{im}(\boldsymbol{W}) = W$ folgt

$$\underline{x} = \boldsymbol{W}\boldsymbol{W}^{\mathsf{H}}\underline{x} + (\boldsymbol{I}_n - \boldsymbol{W}\boldsymbol{W}^{\mathsf{H}})\underline{x},$$

eine Zerlegung von $\underline{x}$ im Sinne von (A.8.1). □

A.8.2 Gram-Schmidt-Algorithmus

Seien $k \geq 1$ linear unabhängige Vektoren

$$\underline{x}_1, \ldots, \underline{x}_k \in \mathbb{C}^n$$

gegeben. (Die lineare Unabhängigkeit impliziert $k \leq n$.) Der Gram-Schmidt-Algorithmus erzeugt aus diesen Vektoren eine neue Familie von k Vektoren

$$\underline{q}_1, \ldots, \underline{q}_k \in \mathbb{C}^n$$

derart, so dass $\|\underline{q}_i\|_2 = 1$ und

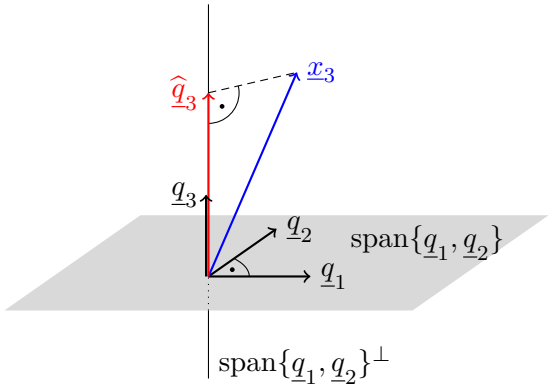
$$\mathrm{span}\{\underline{x}_1, \ldots, \underline{x}_\ell\} = \mathrm{span}\{\underline{q}_1, \ldots, \underline{q}_\ell\}, \quad \forall 1 \leq \ell \leq k. \tag{A.8.2}$$

Weiterhin sind die $\underline{q}_i$ *paarweise orthogonal*:

$$\underline{q}_i \perp \underline{q}_j, \quad i \neq j\,. \tag{A.8.3}$$

Die **Idee des Gram-Schmidt-Algorithmus** im ℓ-ten Schritt ist wie folgt: Man erhält den ℓ-ten Vektor $\underline{q}_\ell$ durch orthogonale Projektion von $\underline{x}_\ell$ auf das orthogonale Komplement von $\mathrm{im}(\underline{q}_1, \ldots, \underline{q}_{\ell-1})$ und Normalisierung:

$$\begin{aligned} \widehat{\underline{q}}_\ell &= (\boldsymbol{I} - (\underline{q}_1, \ldots, \underline{q}_{\ell-1})(\underline{q}_1, \ldots, \underline{q}_{\ell-1})^{\mathsf{H}})\underline{x}_\ell, \\ \underline{q}_\ell &= \widehat{\underline{q}}_\ell / \|\widehat{\underline{q}}_\ell\|_2. \end{aligned}$$



Dies führt zu dem folgenden Algorithmus.

Algorithmus A.54 (Gram-Schmidt)**.**

Input: Linear unabhängige Vektoren $\underline{x}_1, \ldots, \underline{x}_k \in \mathbb{C}^n$.
Output: Orthonormalbasis $\underline{q}_1, \ldots, \underline{q}_k$, so dass (A.8.2) erfüllt ist.
for $\ell = 1, \ldots, k$ **do**
$\widehat{\underline{q}}_\ell := \underline{x}_\ell - \sum_{j=1}^{\ell-1} (\underline{q}_j, \underline{x}_\ell)\, \underline{q}_j$
$\underline{q}_\ell := \frac{\widehat{\underline{q}}_\ell}{\|\widehat{\underline{q}}_\ell\|_2}$
end for

In Python lässt sich der Algorithmus wie folgt implementieren:

```
# Die Spalten von A enthalten
# x_1, ..., x_k und werden mit
# q_1, ..., q_k ueberschrieben.
k = A.shape[1]

for l in range(k):
    sl = A[:, :l].T @ A[:, l]
    qlhut = A[:, l] - A[:, :l] @ sl
    rll = np.linalg.norm(qlhut)
    A[:, l] = qlhut / rll
```

Bemerkung A.55. Algorithmus A.54 lässt sich in dieser Form *ohne jede Änderung* auf einen *beliebigen Vektorraum V mit Skalarprodukt* $(\cdot, \cdot)$ übertragen.

Dass die von Algorithmus A.54 erzeugten Vektoren tatsächlich die gewünschten Eigenschaften erfüllen, lässt sich einfach verifizieren. Die Orthogonalität (A.8.3) etwa folgt direkt aus der Konstruktion von $\widehat{\underline{q}}_\ell$. Numerisch ist Algorithmus A.54 problematisch; das wird in Abschnitt **??** noch näher diskutiert.

Als eine Anwendung des Gram-Schmidt-Algorithmus können wir jede Orthonormalbasis eines Teilraums des $\mathbb{C}^n$ zu einer unitären Matrix ergänzen.

Lemma A.56. *Sei $r < m$. Falls $\boldsymbol{V}_1 = (\underline{v}_1, \ldots, \underline{v}_r) \in \mathbb{C}^{m \times r}$ orthonormale Spalten hat, d.h. $\underline{v}_i^{\mathsf{H}}\, \underline{v}_j = \delta_{ij}$ mit dem Kronecker-Symbol $\delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$, dann existiert $\boldsymbol{V}_2 := (\underline{v}_{r+1}, \ldots, \underline{v}_m) \in \mathbb{C}^{m \times (m-r)}$ derart, dass die Matrix*

$$\boldsymbol{V} = (\boldsymbol{V}_1, \boldsymbol{V}_2)$$

unitär ist, d.h. $\boldsymbol{V}^{\mathsf{H}} \boldsymbol{V} = \boldsymbol{I}_n$ und $\mathrm{im}(\boldsymbol{V}_1)^\perp = \mathrm{im}(\boldsymbol{V}_2)$.

Beweis. Aus dem Basisergänzungsatz folgt die Existenz von Vektoren $\underline{x}_{r+1}, \ldots, \underline{x}_m \in \mathbb{C}^m$, so dass

$$\{\underline{v}_1, \ldots, \underline{v}_r, \underline{x}_{r+1}, \ldots, \underline{x}_m\}$$

eine Basis des $\mathbb{C}^m$ bilden. Der Gram-Schmidt-Algorithmus reproduziert in den ersten r Schritten die Vektoren $\underline{v}_1, \ldots, \underline{v}_r$ (Beweis Übung). In den letzten $m - r$ Schritten werden Vektoren $\underline{v}_{r+1}, \ldots, \underline{v}_m$ erzeugt, die wegen (A.8.2) und (A.8.3) die geforderten Eigenschaften erfüllen. □

A.9 Singulärwertzerlegung (SVD)

Die **SVD** (engl. "Singular Value Decomposition") einer (nicht notwendig quadratischen!) Matrix $\boldsymbol{A}$ ist die mit Abstand wichtigste Matrixzerlegung in Anwendungen der (numerischen) linearen Algebra. Dementsprechend firmiert die SVD – je nach Anwendung – unter weiteren Bezeichnungen und Akronymen, z.B.: PCA – Principal Component Analysis, LSI – Latent Semantic Indexing, POD - Proper Orthogonal Decomposition.

Satz A.57 (Singulärwertzerlegung)**.** *Für $m, n \in \mathbb{N}$ sei $\boldsymbol{A} \in \mathbb{C}^{m\times n}$, $r = \operatorname{rang}(\boldsymbol{A}) \le \min\{m, n\}$. Dann existieren reelle Werte $\sigma_1 \ge \sigma_2 \ge \cdots \ge \sigma_r > 0$ sowie unitäre Matrizen $\boldsymbol{U} \in \mathbb{C}^{m\times m}$ und $\boldsymbol{V} \in \mathbb{C}^{n\times n}$ derart, dass*

$$\boldsymbol{A} = \boldsymbol{U}\,\boldsymbol{\Sigma}\,\boldsymbol{V}^{\mathsf{H}}, \tag{A.9.1}$$

wobei

$$\boldsymbol{\Sigma} = \begin{pmatrix} \sigma_1 & & & \\ & \ddots & & \boldsymbol{0} \\ & & \sigma_r & \\ & \boldsymbol{0} & & \boldsymbol{0} \end{pmatrix} \in \mathbb{R}^{m\times n}\,.$$

Falls $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, so gilt (A.9.1) *mit $\boldsymbol{U} \in \mathbb{R}^{m\times m}$ und $\boldsymbol{V} \in \mathbb{R}^{n\times n}$ orthogonal.*

Beweis. Für $\boldsymbol{A} = \boldsymbol{0}$ gilt der Satz formal mit $r = 0$, $\boldsymbol{U} = \boldsymbol{I}_m, \boldsymbol{V} = \boldsymbol{I}_n$. O.b.d.A. sei also $r \ge 1$. Dann ist $\boldsymbol{A} \ne \boldsymbol{0}$. Wir erinnern an die Definition der 2-Norm von $\boldsymbol{A}$:

$$\|\boldsymbol{A}\|_2 = \max_{\|\underline{x}\|_2=1} \|\boldsymbol{A}\underline{x}\|_2.$$

[Wegen der Stetigkeit der Norm und der Kompaktheit der Einheitskugel wird das Maximum angenommen.] Also existiert ein Vektor $\underline{v}_1 \in \mathbb{C}^n$ mit $\|\underline{v}_1\|_2 = 1$, so dass $\|\boldsymbol{A}\underline{v}_1\|_2 = \|\boldsymbol{A}\|_2$. Da $\sigma_1 := \|\boldsymbol{A}\|_2 \ne 0$ folgt mit $\underline{u}_1 := \boldsymbol{A}\underline{v}_1/\sigma_1$ die Beziehung

$$\underline{u}_1^{\mathsf{H}}\boldsymbol{A}\underline{v}_1 = \sigma_1.$$

Nach Lemma A.56 existieren $\boldsymbol{U}_2 \in \mathbb{C}^{m\times(m-1)}$, $\boldsymbol{V}_2 \in \mathbb{C}^{n\times(n-1)}$ derart, dass die Matrizen

$$\boldsymbol{U} = \big(\underline{u}_1, \boldsymbol{U}_2\big) \in \mathbb{C}^{m\times m}, \quad \boldsymbol{V} = \big(\underline{v}_1, \boldsymbol{V}_2\big) \in \mathbb{C}^{n\times n}$$

unitär sind. Weiterhin gilt

$$\begin{aligned} \boldsymbol{U}^{\mathsf{H}}\boldsymbol{A}\,\boldsymbol{V} &= \big(\underline{u}_1, \boldsymbol{U}_2\big)^{\mathsf{H}}\boldsymbol{A}\big(\underline{v}_1, \boldsymbol{V}_2\big) \\ &= \left(\begin{array}{c|c} \underline{u}_1^{\mathsf{H}}\boldsymbol{A}\underline{v}_1 & \underline{u}_1^{\mathsf{H}}\boldsymbol{A}\boldsymbol{V}_2 \\ \hline \boldsymbol{U}_2^{\mathsf{H}}\boldsymbol{A}\underline{v}_1 & \boldsymbol{U}_2^{\mathsf{H}}\boldsymbol{A}\boldsymbol{V}_2 \end{array}\right) =: \left(\begin{array}{c|c} \sigma_1 & \underline{w}^{\mathsf{H}} \\ \hline \underline{0} & \boldsymbol{B} \end{array}\right) =: \boldsymbol{A}_1\,. \end{aligned}$$

Da (nachrechnen!)

$$\left\|\boldsymbol{A}_1\begin{pmatrix}\sigma_1\\ \underline{w}\end{pmatrix}\right\|_2^2 = \left\|\begin{array}{c}\sigma_1^2 + \underline{w}^{\mathsf{H}}\underline{w} \\ \boldsymbol{B}\,\underline{w}\end{array}\right\|_2^2 = (\sigma_1^2 + \underline{w}^{\mathsf{H}}\underline{w})^2 + \|\boldsymbol{B}\,\underline{w}\|_2^2 \ge (\sigma_1^2 + \underline{w}^{\mathsf{H}}\underline{w})^2\,,$$

gilt auf der einen Seite

$$\|\boldsymbol{A}_1\|_2^2 = \sup_{\underline{0}\neq\underline{x}\in\mathbb{C}^n} \frac{\|\boldsymbol{A}_1\,\underline{x}\|_2^2}{\|\underline{x}\|_2^2} \geq \frac{\left\|\boldsymbol{A}_1 \begin{pmatrix}\sigma_1\\ \underline{w}\end{pmatrix}\right\|_2^2}{\left\|\begin{pmatrix}\sigma_1\\ \underline{w}\end{pmatrix}\right\|_2^2} \geq \frac{(\sigma_1^2 + \underline{w}^{\mathsf{H}}\underline{w})^2}{\sigma_1^2 + \underline{w}^{\mathsf{H}}\underline{w}} = \sigma_1^2 + \underline{w}^{\mathsf{H}}\underline{w}\,. \tag{A.9.2}$$

Wegen der Invarianz der 2-Norm unter unitären Transformationen (siehe Proposition A.50) gilt auf der anderen Seite $\sigma_1 = \|\boldsymbol{A}\|_2 = \|\boldsymbol{U}^{\mathsf{H}}\boldsymbol{A}\boldsymbol{V}\|_2 = \|\boldsymbol{A}_1\|_2$. Zusammen mit (A.9.2) folgt

$$\sigma_1^2 = \|\boldsymbol{A}_1\|_2^2 \geq \sigma_1^2 + \underline{w}^{\mathsf{H}}\underline{w},$$

also $\|\underline{w}\|_2^2 = \underline{w}^{\mathsf{H}}\underline{w} = 0$ und damit $\underline{w} = \underline{0}$. Somit ergibt sich

$$\boldsymbol{A}_1 = \left(\begin{array}{c|c} \sigma_1 & \underline{0}^{\mathsf{T}} \\ \hline \underline{0} & \boldsymbol{B} \end{array}\right).$$

Per Induktion (ähnlich wie im Beweis von Satz A.63) ist damit die Behauptung bewiesen. □

Übung A.58. Im zweiten Schritt der Induktion im Beweis von Satz A.57 ist, gemäss der Definition von σ_1, $\sigma_2 := \|\boldsymbol{B}\|_2$. Zeige, dass $\sigma_2 \leq \sigma_1$.

Man kann zeigen, dass die Werte $\sigma_1, \ldots, \sigma_r$ in der SVD eindeutig bestimmt sind. Gilt ausserdem $\sigma_1 > \sigma_2 > \cdots > \sigma_r$, sind sogar die Vektoren $\underline{u}_1, \ldots, \underline{u}_r$, $\underline{v}_1, \ldots, \underline{v}_r$ bis auf skalare Vielfache vom Betrag 1 eindeutig bestimmt.

Definition A.59. Betrachte eine SVD (A.9.1) von $\boldsymbol{A} \in \mathbb{C}^{m\times n}$ und partitioniere

$$\boldsymbol{U} = (\underline{u}_1, \ldots, \underline{u}_r, \underline{u}_{r+1}, \ldots, \underline{u}_m), \quad \boldsymbol{V} = (\underline{v}_1, \ldots, \underline{v}_r, \underline{v}_{r+1}, \ldots, \underline{v}_n). \tag{A.9.3}$$

Dann heissen

$\sigma_1, \ldots, \sigma_r$	die **Singulärwerte**,
$\underline{u}_1, \ldots, \underline{u}_r$	die **linken Singulärvektoren**,
$\underline{v}_1, \ldots, \underline{v}_r$	die **rechten Singulärvektoren**

von $\boldsymbol{A}$.

Im Gegensatz zur Schur-Form können bei reellem $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ auch die Transformationsmatrizen $\boldsymbol{U}, \boldsymbol{V}$ in der SVD reell (und damit orthogonal) gewählt werden.

Bemerkung A.60. Statt der vollen SVD von $\boldsymbol{A}$ betrachtet man oft die sogenannte **kompakte SVD** $\boldsymbol{A} = \boldsymbol{U}_0\boldsymbol{\Sigma}_0\boldsymbol{V}_0^{\mathsf{H}}$, die sich aus der SVD mit der Partitionierung (A.9.3) mittels

$$\begin{aligned} \boldsymbol{U}_0 &= (\underline{u}_1, \ldots, \underline{u}_r) \\ \boldsymbol{\Sigma}_0 &= \operatorname{diag}(\sigma_1, \ldots, \sigma_r) \\ \boldsymbol{V}_0 &= (\underline{v}_1, \ldots, \underline{v}_r) \end{aligned}$$

ergibt.

```python
# Singulaerwerte von A
s = np.linalg.svd(A, compute_uv=False)

# SVD von A
U, s, Vt = np.linalg.svd(A)

# kompakte SVD
U, s, Vt = np.linalg.svd(A, full_matrices=False)
```

Die SVD erlaubt es auf einfache Weise, viele theoretisch und praktisch relevante Aussagen über $\boldsymbol{A}$ zu treffen. So folgt z.B. sofort, dass $\boldsymbol{A}$ sich als Summe von r Matrizen vom Rang 1 schreiben lässt:

$$\boldsymbol{A} = \sigma_1 \underline{u}_1 \underline{v}_1^{\mathsf{H}} + \cdots + \sigma_r \underline{u}_r \underline{v}_r^{\mathsf{H}}.$$

Weiterhin können aus $\boldsymbol{U}, \boldsymbol{V}$ Orthonormalbasen für das Bild und den Kern von $\boldsymbol{A}$ bzw. $\boldsymbol{A}^{\mathsf{H}}$ gewonnen werden.

Proposition A.61. *Betrachte eine SVD von $\boldsymbol{A} \in \mathbb{C}^{m\times n}$ mit der Partitionierung (A.9.3). Dann gilt*

$$\begin{aligned}
\ker(\boldsymbol{A}) &= \operatorname{span}\{\underline{v}_{r+1}, \ldots, \underline{v}_n\}, \\
\operatorname{im}(\boldsymbol{A}) &= \operatorname{span}\{\underline{u}_1, \ldots, \underline{u}_r\}, \\
\ker(\boldsymbol{A}^{\mathsf{H}}) &= \operatorname{span}\{\underline{u}_{r+1}, \ldots, \underline{u}_m\}, \\
\operatorname{im}(\boldsymbol{A}^{\mathsf{H}}) &= \operatorname{span}\{\underline{v}_1, \ldots, \underline{v}_r\}.
\end{aligned}$$

Beweis. Die aus der SVD folgende Beziehungen $\boldsymbol{A}\boldsymbol{V} = \boldsymbol{U}\boldsymbol{\Sigma}$ und $\boldsymbol{A}^{\mathsf{H}}\boldsymbol{U} = \boldsymbol{V}\boldsymbol{\Sigma}^{\mathsf{H}}$ spaltenweise aufgeschrieben:

$$\boldsymbol{A}\,\underline{v}_i = \begin{cases} \sigma_i\,\underline{u}_i & 1 \le i \le r, \\ 0 & r+1 \le i \le n, \end{cases}$$

$$\boldsymbol{A}^{\mathsf{H}}\,\underline{u}_i = \begin{cases} \sigma_i\,\underline{v}_i & 1 \le i \le r, \\ 0 & r+1 \le i \le m. \end{cases}$$

Daraus folgt z.B. sofort, dass $\operatorname{span}\{\underline{v}_{r+1}, \ldots, \underline{v}_n\} \subseteq \ker(\boldsymbol{A})$. Da aber die Dimension des Kerns $n-r$ ist, muss Gleichheit gelten. Die anderen Beziehungen folgen analog. □

Zwischen den Singulärwerten bzw. -vektoren von $\boldsymbol{A}$ und den Eigenwerten bzw. -vektoren von $\boldsymbol{A}^{\mathsf{H}}\boldsymbol{A}$ und $\boldsymbol{A}\boldsymbol{A}^{\mathsf{H}}$ gibt es enge Zusammenhänge. Aus $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathsf{H}}$ folgen die Spektralzerlegungen

$$\boldsymbol{U}^{\mathsf{H}}\boldsymbol{A}\boldsymbol{A}^{\mathsf{H}}\boldsymbol{U} = \boldsymbol{\Sigma}\boldsymbol{\Sigma}^{\mathsf{H}} = \operatorname{diag}(\sigma_1^2, \ldots, \sigma_r^2, 0, \ldots, 0) \in \mathbb{R}^{m\times m},$$

$$\boldsymbol{V}^{\mathsf{H}}\boldsymbol{A}^{\mathsf{H}}\boldsymbol{A}\boldsymbol{V} = \boldsymbol{\Sigma}^{\mathsf{H}}\boldsymbol{\Sigma} = \operatorname{diag}(\sigma_1^2, \ldots, \sigma_r^2, 0, \ldots, 0) \in \mathbb{R}^{n\times n}.$$

Wir erinnern (Satz A.57), dass $r \le \min\{m, n\}$ ist. Insbesondere ist $\lambda \neq 0$ genau dann ein Eigenwert von $\boldsymbol{A}^{\mathsf{H}}\boldsymbol{A}$ (oder $\boldsymbol{A}\boldsymbol{A}^{\mathsf{H}}$), wenn $\sqrt{\lambda}$ ein Singulärwert von $\boldsymbol{A}$ ist.

A.10 Jordan- und Schur-Normalform*

Die Transformation von Matrizen auf einfachere Gestalt ist ein entscheidendes Hilfsmittel in der Theorie und Numerik der linearen Algebra. In diesem Abschnitt rekapitulieren wir zwei Normalformen, die über Ähnlichkeitstransformation $\boldsymbol{X}^{-1}\boldsymbol{A}\boldsymbol{X}$ erreicht werden können. Aus Proposition A.25 wissen wir bereits, dass eine solche Transformation die Eigenwerte von $\boldsymbol{A}$ nicht verändert. Die folgende Normalform ist vor allem für die Theorie von Interesse.

Satz A.62 (Jordan'sche Normalform)**.** *Für jede Matrix $\boldsymbol{A} \in \mathbb{C}^{n\times n}$ existiert eine nichtsinguläre Matrix $\boldsymbol{X} \in \mathbb{C}^{n\times n}$ derart, dass*

$$\boldsymbol{X}^{-1}\,\boldsymbol{A}\,\boldsymbol{X} = \operatorname{diag}(\boldsymbol{J}_1, \ldots, \boldsymbol{J}_t)$$

die sogenannte **Jordan'sche Normalform** *von $\boldsymbol{A}$ ist, wobei die* **Jordan-Blöcke** *$\boldsymbol{J}_i$ gegeben sind durch*

$$\boldsymbol{J}_i = \begin{pmatrix} \lambda_i & 1 & 0 & \cdots & 0 \\ 0 & \lambda_i & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & \ddots & 1 \\ 0 & \cdots & \cdots & 0 & \lambda_i \end{pmatrix} \in \mathbb{C}^{n_i\times n_i}$$

mit $\sum_{i=1}^{t} n_i = n$ und $\Lambda(\boldsymbol{A}) = \{\lambda_1, \ldots, \lambda_t\}$.

Wir bemerken zur Jordan'schen Normalform:

(i) Die Eigenwerte λ_i sind in den verschiedenen Jordanblöcken nicht notwendig verschieden. Die *Anzahl* der Jordanblöcke zu einem Eigenwert λ_i ist die **geometrische Vielfachheit** von λ_i und entspricht der Dimension von $\ker(\boldsymbol{A} - \lambda_i\boldsymbol{I}_n)$. Die Summe der Grössen n_i aller Jordanblöcke zu einem Eigenwert λ_i ist die **algebraische Vielfachheit** von λ_i und entspricht der Vielfachheit der Nullstelle λ_i des charakteristischen Polynoms $\det(\boldsymbol{A} - \lambda\boldsymbol{I}_n)$.

(ii) Eine Matrix heisst **diagonalisierbar**, wenn alle Jordanblöcke der Grösse 1 sind, also bei allen Eigenwerten die algebraischen und geometrischen Vielfachheiten identisch sind.

(iii) Aus verschiedenen Gründen lässt sich die Jordan'sche Normalform numerisch nur schwer oder gar nicht berechnen. Zum einen kann eine beliebig kleine Störung der Einträge von $\boldsymbol{A}$ (etwa durch Rundungsfehler, siehe Kap. 1) die Jordan-Normalform komplett verändern. Zum anderen kann die Transformationsmatrix $\boldsymbol{X}$ sehr schlecht konditioniert sein, also $\|\boldsymbol{X}\|_2\,\|\boldsymbol{X}^{-1}\|_2$ sehr gross sein (siehe Abschnitt **??** zum Begriff der Kondition einer Matrix).

In der Numerik verlangt man deshalb weniger als die Jordan'sche Normalform und schränkt die zulässigen Transformationen auf unitäre Matrizen ein. Dies wird damit "bezahlt", dass die transformierte Normalform der Matrix weniger Nulleinträge hat.

Satz A.63 (Schur-Form). *Sei $\boldsymbol{A} \in \mathbb{C}^{n\times n}$ und seien $\lambda_1, \ldots, \lambda_n \in \mathbb{C}$ die Eigenwerte von $\boldsymbol{A}$. Dann existiert eine unitäre Matrix $\boldsymbol{Q} \in \mathbb{C}^{n\times n}$, so dass*

$$\boldsymbol{Q}^{\mathsf{H}}\,\boldsymbol{A}\,\boldsymbol{Q} = \begin{pmatrix} \lambda_1 & & * \\ & \ddots & \\ 0 & & \lambda_n \end{pmatrix} =: \boldsymbol{T},$$

d.h., $\boldsymbol{T}$ ist rechte Dreiecksmatrix, mit den Eigenwerten von $\boldsymbol{A}$ auf der Diagonalen.

Beweis. Der Beweis erfolgt per Induktion über n. Für $n = 1$ ist die Aussage mit $\boldsymbol{Q} = 1$ offensichtlich erfüllt. Sei $n \geq 2$ und sei die Aussage für alle Matrizen $\boldsymbol{B} \in \mathbb{C}^{(n-1)\times(n-1)}$ erfüllt. Betrachte nun für $\boldsymbol{A} \in \mathbb{C}^{n\times n}$ den Eigenvektor $\underline{u}_1$ zu λ_1. O.B.d.A. können wir $\|\underline{u}_1\|_2 = 1$ annehmen. Wegen Lemma A.56 existieren Vektoren $\underline{u}_2, \ldots, \underline{u}_n \in \mathbb{C}^n$, so dass $\boldsymbol{Q}_0 = (\underline{u}_1, \ldots, \underline{u}_n)$ unitär ist. Wegen $\boldsymbol{A}\underline{u}_1 = \lambda_1 \underline{u}_1$ gilt

$$\boldsymbol{A}\boldsymbol{Q}_0 = \boldsymbol{Q}_0 \begin{pmatrix} \lambda_1 & * \\ \underline{0} & \boldsymbol{A}_1 \end{pmatrix}.$$

Aus der Induktionsvoraussetzung folgt die Existenz einer unitären Matrix $\boldsymbol{Q}_1 \in \mathbb{C}^{(n-1)\times(n-1)}$, so dass $\boldsymbol{T}_1 = \boldsymbol{Q}_1^{\mathsf{H}}\boldsymbol{A}_1\boldsymbol{Q}_1$ rechte Dreiecksmatrix ist, mit $\lambda_2, \ldots, \lambda_n$ auf der Diagonalen. Mit

$$\boldsymbol{Q} = \boldsymbol{Q}_0 \begin{pmatrix} 1 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{Q}_1 \end{pmatrix}$$

folgt die Behauptung. □

Bemerkung A.64. Da die Eigenwerte einer reellen Matrix komplex sein können, gibt es für $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ i.a. *keine reelle orthogonale* Matrix $\boldsymbol{Q}$, so dass $\boldsymbol{Q}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{Q}$ in rechter Dreiecksform ist. MATLAB gibt für reelles $\boldsymbol{A}$ die *reelle Schur-Form* zurück, bei der $\boldsymbol{T}$ nur fast rechte Dreiecksform hat, mit 2×2-Blöcken auf der Diagonalen.

```
from scipy.linalg import schur

# (reelle) Schur-Form von A
T, Q = schur(A)

# komplexe Schur-Form von A (selbst wenn A reell ist)
T, Q = schur(A, output='complex')
```

Korollar A.65 (Spektralzerlegung). *Ist $\boldsymbol{A} \in \mathbb{C}^{n\times n}$ eine* normale Matrix*, dann gibt es eine unitäre Matrix $\boldsymbol{Q}$, so dass $\boldsymbol{T} = \boldsymbol{Q}^{\mathsf{H}}\boldsymbol{A}\boldsymbol{Q}$* diagonal *ist.*

Beweis. Aus der Normalität von $\boldsymbol{A}$ folgt, dass die Schur-Form $\boldsymbol{T}$ von $\boldsymbol{A}$ die Beziehung $\boldsymbol{T}\boldsymbol{T}^{\mathsf{H}} = \boldsymbol{T}^{\mathsf{H}}\boldsymbol{T}$ erfüllt. Mit der Partitionierung $\boldsymbol{T} = \begin{pmatrix} \lambda_1 & \underline{w}^{\mathsf{H}} \\ \underline{0} & \boldsymbol{T}_1 \end{pmatrix}$ folgt daraus $|\lambda_1|^2 + \|\underline{w}\|_2^2 = |\lambda_1|^2$ und damit $\underline{w} = \underline{0}$. Wiederholtes Anwenden dieses Arguments auf $\boldsymbol{T}_1$ impliziert die Diagonalität von $\boldsymbol{T}$. □

Die Umkehrung von Korollar A.65 gilt ebenfalls: jede unitär diagonalisierbare Matrix ist normal.

Literaturverzeichnis

[1] R. Brent and P. Zimmermann. *Modern Computer Arithmetic*. Cambridge University Press, 2010.

[2] P. D. Düben, H. McNamara, and T. N. Palmer. The use of imprecise processing to improve accuracy in weather & climate prediction. *J. Comput. Phys.*, 271:2–18, 2014.

[3] G. H. Golub and C. F. Van Loan. *Matrix computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, fourth edition, 2013.

[4] N. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM, 2007. Second Edition.

[5] S. Leyffer, S. Wild, M. Fagan, M. Snir, K. Palem, K. Yoshii, and H. Finkel. Doing Moore with Less – leapfrogging Moore's Law with inexactness for supercomputing. Technical report, ArXiv:1610.02606v2, 2016.

[6] J. H. Wilkinson. *Rundungsfehler*. Springer, 1969.