Kapitel 4

Operatoren für Matrizen - Lineare Algebra

Als Matrizen bezeichnet man eine rechteckige Anordnung von Zahlen (oder Variablen). Im Unterschied zu den Feldern (Arrays), die exakt das gleiche Aussehen haben, werden hier Matrizen als Konstrukte der linearen Algebra aufgefasst, für die natürlich andere Regeln in Bezug auf für Multiplikation und Division gelten.

Eine Matrix $\mathbf{A}$ kann geschrieben werden als

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & a_{24} & \dots & a_{2n} \\ a_{31} & a_{32} & a_{33} & a_{34} & \dots & a_{3n} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & a_{m3} & a_{m4} & \dots & a_{mn} \end{bmatrix} = [a_{jk}] \, . \tag{4.1}$$

Dies ist eine $m \times n$ Matrix mit m Zeilen (rows) and n Spalten (columns). Die einzelnen Elemente werden in der Mathematik mit Hilfe von Indizes a_{jk} bezeichnet, in MATLAB lautet die Schreibweise `A(j,k)`. Die Matrix $\mathbf{A}$ ist 2-dimensional, es gibt jedoch keine Beschränkung in der Anzahl der Dimensionen. Die beiden MATLAB Befehle `ndims` und `size` geben die jeweilige Dimension der Matrix und die Größe in jeder Dimension. Einige Befehle und die zugrundeliegenden Konzepte (z.B.: Transponieren) sind aber nur für 2-dim Matrizen definiert.

Spezielle zweidimensionale Matrizen sind:

Spaltenvektor, column vector: Matrix mit nur einer Spalte,

$$\mathbf{a} = \begin{bmatrix} a_{11} \\ a_{21} \\ a_{31} \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = [a_j] \, . \tag{4.2}$$

Die Eingabe in MATLAB erfolgt mit `a=[1;2;3]` oder `a=[1,2,3].'`. Die Anzahl der Dimensionen ist 2, der Befehl `size` liefert `[3 1]`.

Zeilenvektor, row vector: Matrix mit nur einer Zeile,

$$\mathbf{b} = \begin{bmatrix} b_{11} & b_{12} & b_{13} \end{bmatrix} = \begin{bmatrix} b_1 & b_2 & b_3 \end{bmatrix} = [b_j] \ . \tag{4.3}$$

Die Eingabe in MATLAB erfolgt mit `b=[1,2,3]`. Die Anzahl der Dimensionen ist 2, der Befehl `size` liefert `[1 3]`.

Skalar, scalar: Matrize reduziert auf eine einzige Zahl,

$$s = \mathbf{s} = s_{11} = s_1 = [s_j] \ . \tag{4.4}$$

Die Anzahl der Dimensionen ist auch hier 2, der Befehl `size` liefert `[1 1]`.

4.1 Transponieren einer Matrix

Es erweist sich als praktisch, die Transponierte einer Matrix zu definieren. Die transponierte Matrix $\mathbf{A}^T$ einer $m \times n$ Matrix $\mathbf{A} = [a_{jk}]$ ist eine $n \times m$ Matrix, wobei die Zeilen in Spalten und die Spalten in Zeilen verwandelt werden,

$$\mathbf{A}^T = [a_{kj}] \ . \tag{4.5}$$

Mit Hilfe der transponierten Matrix können zwei Klassen von reellen quadratischen Matrizen definiert werden:

Symmetrische Matrix: Für eine symmetrische Matrix gilt

$$\mathbf{A}^T = \mathbf{A} \ . \tag{4.6}$$

Schiefsymmetrische Matrix: Für eine schiefsymmetrische Matrix gilt

$$\mathbf{A}^T = -\mathbf{A} \ . \tag{4.7}$$

Zerlegung: Jede quadratische Matrix $(n \times n)$ lässt sich in eine Summe aus einer symmetrischen und einer schiefsymmetrischen Matrix zerlegen,

$$\mathbf{A} = \mathbf{S} + \mathbf{U} \ , \tag{4.8}$$

$$\mathbf{S} = \frac{1}{2}\left(\mathbf{A} + \mathbf{A}^T\right) \ , \tag{4.9}$$

$$\mathbf{U} = \frac{1}{2}\left(\mathbf{A} - \mathbf{A}^T\right) \ . \tag{4.10}$$

In MATLAB steht zum Transponieren der Operator `.'` oder der Befehl `transpose` zur Verfügung.

4.2 Addition und Subtraktion von Matrizen

Diese beiden Operationen existieren nur "elementweise", d.h. es gibt keinen Unterschied zwischen Matrix- und Array-Operationen

$$\mathbf{C} = \mathbf{A} \pm \mathbf{B} \implies [c_{jk}] = [a_{jk}] \pm [b_{jk}] \ , \tag{4.11}$$

und daher ist die Defintion von `.+` und `.-` Operatoren nicht notwendig.

Voraussetzung: Gleiche Dimension und gleiche Größe von $\mathbf{A}$ und $\mathbf{B}$.

Ausnahme: A oder B ist ein Skalar, dann wird die skalare Größe zu allen Elementen addiert (oder von allen subtrahiert).

Beispiele: Mit $\mathbf{A} = [1\ 2\ 3]$ und $\mathbf{B} = [4\ 5\ 6]$ ergibt sich,

$$\mathbf{C} = \mathbf{A} + \mathbf{B} = \begin{bmatrix} 5\ 7\ 9 \end{bmatrix} \ , \tag{4.12}$$

$$\mathbf{D} = \mathbf{A} + 1 = \begin{bmatrix} 2\ 3\ 4 \end{bmatrix} \ . \tag{4.13}$$

Fehler: Die Rechnung

$$\mathbf{C} = \mathbf{A} + \mathbf{B}^T = \begin{bmatrix} 1\ 2\ 3 \end{bmatrix} + \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix} = \text{Error} \tag{4.14}$$

ergibt natürlich eine Fehlermitteilung.

4.3 Skalar Multiplikation

Das Produkt einer $m \times n$ Matrix $\mathbf{A} = [a_{jk}]$ und eines Skalars c (Zahl c) wird geschrieben als $c\mathbf{A}$ und ergibt die $m \times n$ Matrix $c\mathbf{A} = [ca_{jk}]$.

4.4 Matrix Multiplikation

Dies ist eine Multiplikation im Sinne der linearen Algebra. Das Produkt $\mathbf{C} = \mathbf{AB}$ einer $m \times n$ Matrix $\mathbf{A} = [a_{jk}]$ und einer $r \times p$ Matrix $\mathbf{B} = [b_{jk}]$ ist nur dann definiert, wenn gilt $n = r$. Die Multiplikation ergibt eine $m \times p$ Matrix $\mathbf{C} = [c_{jk}]$, deren Elemente gegeben sind als,

$$c_{jk} = \sum_{l=1}^{n} a_{jl} b_{lk} \ . \tag{4.15}$$

In MATLAB steht für die Matrizenmultiplikation der Befehl `C=mtimes(A,B)` oder die Operatorform `C=A*B` zur Verfügung, wobei nach den oben genannten Regeln die "inneren" Dimensionen übereinstimmen müssen, d.h., die Anzahl der Spalten von A muss mit der Anzahl der Zeilen von B übereinstimmen (Index l in 4.15).

Im Unterschied zur Multiplikation von Skalaren ist die Multiplikation von Matrizen nicht kommutativ, im Allgemeinen gilt daher

$$\mathbf{AB} \neq \mathbf{BA} \; . \tag{4.16}$$

Außerdem folgt aus $\mathbf{AB} = 0$ nicht notwendigerweise $\mathbf{A} = 0$ oder $\mathbf{B} = 0$ oder $\mathbf{BA} = 0$.

Beispiele: Multiplikation einer (2×3)-Matrix mit einer (3×2)-Matrix:

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix} = \begin{bmatrix} 14 & 32 \\ 32 & 77 \end{bmatrix} \; . \tag{4.17}$$

Multiplikation einer (3×2)-Matrix mit einer (2×3)-Matrix:

$$\begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix} \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} = \begin{bmatrix} 17 & 22 & 27 \\ 22 & 29 & 36 \\ 27 & 36 & 45 \end{bmatrix} \; . \tag{4.18}$$

Multiplikation einer (3×2)-Matrix mit einer (3×2)-Matrix:

$$\begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix} \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix} = \text{Error} \; . \tag{4.19}$$

Inneres Produkt zweier Vektoren:

$$\begin{bmatrix} 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix} = 32 \; . \tag{4.20}$$

Äußeres Produkt zweier Vektoren:

$$\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \begin{bmatrix} 4 & 5 & 6 \end{bmatrix} = \begin{bmatrix} 4 & 5 & 6 \\ 8 & 10 & 12 \\ 12 & 15 & 18 \end{bmatrix} \; . \tag{4.21}$$

Fehler:

$$\begin{bmatrix} 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} 4 & 5 & 6 \end{bmatrix} = \text{Error} \; . \tag{4.22}$$

Für das Transponieren von Produkten, $\mathbf{C} = \mathbf{AB}$, kann sich ganz leicht davon überzeugen, dass gilt:

$$\mathbf{C}^T = \mathbf{B}^T \mathbf{A}^T \tag{4.23}$$

$$\mathbf{C} = (\mathbf{B}^T \mathbf{A}^T)^T \tag{4.24}$$

$$(c\mathbf{A})^T = c\mathbf{A}^T \tag{4.25}$$

4.5 Inneres Produkt zweier Vektoren

Wenn $\mathbf{a}$ und $\mathbf{b}$ Spaltenvektoren der Länge n sind, dann ist $\mathbf{a}^T$ ein Zeilenvektor und das Produkt $\mathbf{a}^T\mathbf{b}$ ergibt die 1×1 Matrix (bzw. die Zahl), welche **inneres Produkt** von $\mathbf{a}$ und $\mathbf{b}$ genannt wird. Das innere Produkt wird mit $\mathbf{a} \cdot \mathbf{b}$ bezeichnet

$$\mathbf{a} \cdot \mathbf{b} = \mathbf{a}^T\mathbf{b} = \begin{bmatrix} a_1 & \dots & a_n \end{bmatrix} \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix} = \sum_{l=1}^{n} a_l b_l \ . \tag{4.26}$$

In MATLAB existiert dafür der Befehl `dot(a,b)`. Er berechnet das innere Produkt zweier Vektoren, und kümmert sich nicht um deren "Ausrichtung" als Zeilen- oder Spaltenvektor.

Das innere Produkt hat interessante Anwendungen in der Mechanik und der Geometrie.

4.6 Spezielle Matrizen

Für die Beschreibung der Matrix Division beschränken wir uns vorerst auf quadratische Matrizen ($n \times n$). Dafür benötigen wir zuerst die Definition der **Einheitsmatrix**

$$\mathbf{I} = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \end{bmatrix} \ . \tag{4.27}$$

Für die Einheitsmatrix gilt

$$\mathbf{AI} = \mathbf{IA} = \mathbf{A} \ . \tag{4.28}$$

In MATLAB steht für die Erzeugung der Befehl `eye` (zB. `eye(3)`) zur Verfügung.

Die **inverse Matrix** ist definiert durch

$$\mathbf{A}\mathbf{A}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I} \ . \tag{4.29}$$

Eine Matrix für die

$$\mathbf{A}^T = \mathbf{A}^{-1} \tag{4.30}$$

gilt, wird als **orthogonale** Matrix bezeichnet.

In MATLAB gibt es zur Berechnung der inversen Matrix den Befehl `inv(A)`.

4.7 Matrix Division - Lineare Gleichungssysteme

Die Division von Matrizen kann man sich am besten mit Hilfe von linearen Gleichungssystemen vorstellen. Solche Gleichungssysteme können sehr elegant mit Hilfe von Matrizen formuliert werden. Bei bekanntem $\mathbf{A}$ und $\mathbf{b}$ kann eine Gleichung für $\mathbf{x}$ folgendermaßen geschrieben werden

$$\mathbf{A}\mathbf{x} = \mathbf{b} \; . \tag{4.31}$$

Im Allgemeinen ist die Koeffizientenmatrix $\mathbf{A} = [a_{jk}]$ die $m \times n$ Matrix

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix} \text{, und } \mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ \vdots \\ x_n \end{bmatrix} \text{ und } \mathbf{b} = \begin{bmatrix} b_1 \\ \vdots \\ b_m \end{bmatrix} \tag{4.32}$$

sind Spaltenvektoren. Man sieht, dass die Anzahl der Elemente von $\mathbf{x}$ gleich n und von $\mathbf{b}$ gleich m ist. Dadurch wird ein lineares System von m Gleichungen in n Unbekannten beschrieben:

$$\begin{array}{ccccccccc} a_{11}x_1 & + & a_{12}x_2 & + & \dots & + & a_{1n}x_n & = & b_1 \\ a_{21}x_1 & + & a_{22}x_2 & + & \dots & + & a_{2n}x_n & = & b_2 \\ \vdots & + & \vdots & + & \ddots & + & \vdots & = & \vdots \\ a_{m1}x_1 & + & a_{m2}x_2 & + & \dots & + & a_{mn}x_n & = & b_m \end{array} \tag{4.33}$$

Die a_{jk} sind gegebene Zahlen, die **Koeffizienten** des Systems genannt werden. die b_i sind ebenfalls gegebene Zahlen. Wenn alle b_i gleich Null sind, handelt es sich um ein **homogenes System**, wenn hingegen zumindest ein b_i ungleich Null ist, handelt es sich um ein **inhomogenes System**.

Die Lösung von 4.33 ist ein Vektor $\mathbf{x}$ der Länge n, der alle m Gleichungen erfüllt. Ist das System 4.33 homogen, hat es zumindest die **triviale** Lösung

$$x_1 = 0 \; , \; x_2 = 0 \; , \; \dots \; , \; x_n = 0 \; . \tag{4.34}$$

Ein System heißt

überbestimmt, wenn es mehr Gleichungen als Unbekannte hat ($m > n$);

bestimmt, wenn es gleich viel Gleichungen wie Unbekannte hat ($m = n$);

unterbestimmt, wenn es weniger Gleichungen als Unbekannte hat ($m < n$).

Ein unterbestimmtes System hat immer eine Lösungsschar, ein bestimmtes Gleichungssystem hat mindestens eine Lösung, und ein überbestimmtes System hat nur eventuell eine Lösung.

Um $\mathbf{x}$ zu berechnen, kann man nun formal 4.31 von Links mit $\mathbf{A}^{-1}$ multiplizieren

$$\mathbf{A}^{-1}\mathbf{A}\mathbf{x} = \mathbf{A}^{-1}\mathbf{b} \implies \mathbf{x} = \mathbf{A}^{-1}\mathbf{b} = \mathbf{A}\backslash\mathbf{b}. \tag{4.35}$$

Dafür steht in MATLAB der Befehl `x=A\b` bereit. Das Zeichen \ steht für die sogenannte "Matrix-Linksdivision". Es wird hier in der Numerik verwendet, in der mathematischen Beschreibung der linearen Algebra aber nicht.

Handelt es sich um ein **bestimmtes** Gleichungsystem, löst MATLAB das System mit Hilfe des Gaußschen Eliminierungsverfahrens.

Handelt es sich um eine **über-** oder **unterbestimmtes** Gleichungssystem, findet MATLAB die Lösung mit Hilfe des "Least Squares"-Verfahrens, dass später besprochen wird.

Lautet das lineare Gleichungssystem hingegen

$$\mathbf{y}\mathbf{A} = \mathbf{c}\ , \tag{4.36}$$

wobei $\mathbf{y}$ und $\mathbf{c}$ jetzt Zeilenvektoren der Länge m bzw. n sind, muss man von rechts mit $\mathbf{A}^{-1}$ multiplizieren und erhält

$$\mathbf{y}\mathbf{A}\mathbf{A}^{-1} = \mathbf{c}\mathbf{A}^{-1} \implies \mathbf{y} = \mathbf{c}\mathbf{A}^{-1} = \mathbf{c}/\mathbf{A}\ . \tag{4.37}$$

Dafür steht in MATLAB der Befehl `y=c/A` bereit. Das Zeichen `/` steht dafür für "Matrix-Rechtsdivision". Mit Hilfe der Regeln für das Transponieren, kann 4.37 umgeformt werden in

$$\mathbf{y} = \left((\mathbf{A}^{-1})^T\mathbf{c}^T\right)^T = \left(\mathbf{A}^T\backslash\mathbf{c}^T\right)^T\ , \tag{4.38}$$

wobei dies die Form ist, die MATLAB intern verwendet (`y=(A.'\c.').'`).

Für die Skalare s und r würde gelten

$$\begin{array}{rcccl} s/r & = & sr^{-1} & = & s/r \\ r\backslash s & = & r^{-1}s & = & s/r, \end{array} \tag{4.39}$$

was in beiden Fällen das Gleiche ist, da die Multiplikation von Skalaren kommutativ ist. Dies gilt jedoch nicht für die Matrizenmultiplikation.

MATLAB hat darüber hinaus den Vorteil, dass lineare Gleichungssysteme simultan für verschiedene inhomogene Vektoren $\mathbf{b}$, die in einer Matrix $\mathbf{B}$ zusammengefasst

sind, gelöst werden können. Man kann 4.33 umschreiben als:

$$\begin{array}{ccccccccc}
a_{11}x_{11} & + & a_{12}x_{21} & + & \dots & + & a_{1n}x_{n1} & = & b_{11} \\
a_{21}x_{11} & + & a_{22}x_{21} & + & \dots & + & a_{2n}x_{n1} & = & b_{21} \\
\vdots & + & \vdots & + & \ddots & + & \vdots & = & \vdots \\
a_{m1}x_{11} & + & a_{m2}x_{21} & + & \dots & + & a_{mn}x_{n1} & = & b_{m1} \\
\hline
a_{11}x_{12} & + & a_{12}x_{22} & + & \dots & + & a_{1n}x_{n2} & = & b_{12} \\
a_{21}x_{12} & + & a_{22}x_{22} & + & \dots & + & a_{2n}x_{n2} & = & b_{22} \\
\vdots & + & \vdots & + & \ddots & + & \vdots & = & \vdots \\
a_{m1}x_{12} & + & a_{m2}x_{22} & + & \dots & + & a_{mn}x_{n2} & = & b_{m2} \\
\hline
\vdots & + & \vdots & + & \vdots & + & \vdots & = & \vdots \\
\hline
a_{11}x_{1p} & + & a_{12}x_{2p} & + & \dots & + & a_{1n}x_{np} & = & b_{1p} \\
a_{21}x_{1p} & + & a_{22}x_{2p} & + & \dots & + & a_{2n}x_{np} & = & b_{2p} \\
\vdots & + & \vdots & + & \ddots & + & \vdots & = & \vdots \\
a_{m1}x_{1p} & + & a_{m2}x_{2p} & + & \dots & + & a_{mn}x_{np} & = & b_{mp}
\end{array} \tag{4.40}$$

Dieses Gleichungssystem kann formal geschrieben werden als

$$\mathbf{AX} = \mathbf{B} \, , \tag{4.41}$$

wobei die $m \times p$ Inhomogenitätsmatrix $\mathbf{B}$ aus p nebeneinander angeordneten Spaltenvektoren $\mathbf{b}$ besteht. Auf die genau gleiche Weise liegen die Lösungen in den p Spaltenvektoren der $n \times p$ Matrix $\mathbf{X}$. Die Lösung erfolgt in MATLAB mit dem analogen Befehl `X=A\B`.