Prof. Dr. Angela Kunoth
Max Glembock, Laslo Hunhold

Matlab Präsenzaufgaben — WS 2017/2018

Übungsblatt 2

Ausgabe: 18.10.2017

Vektoren und Matrizen

Wir fassen Spaltenvektoren als einspaltige Matrizen und Zeilenvektoren als einzeilige Matrizen auf. Somit können wir uns auf die Betrachtung von Matrizen beschränken.
Matrizen können in Matlab auf mehrere Arten erzeugt werden. Bei vielen Matrizen-Typen ist es möglich, diese direkt über integrierte Matlab-Funktion zu generieren. Die Nullmatrix, die Einheitsmatrix und die Einsmatrix können über die Funktionen `zeros`, `eye` und `ones` erzeugt werden. Die Nullmatrix hat dabei nur Nullen als Einträge und die Einsmatrix entsprechend nur Einsen. Alle drei Funktionen haben die gleiche Syntax. Zum Beispiel erzeugt `zeros(m,n)` oder `zeros([m,n])` eine $m \times n$ Nullmatrix, während `zeros(n)` eine $n \times n$ Nullmatrix erzeugt. Die Dimensionen der Matrizen müssen entsprechend ihrer späteren Verwendung passend gewählt werden, damit die mathematischen Operationen wohldefiniert sind.
Mit `rand` erzeugt man Matrizen mit (Pseudo-)Zufallszahlen zwischen Null und Eins als Einträgen. Die Syntax ist die gleiche wie bei `zeros`. Ohne Argument gibt die Funktion eine einzelne (Pseudo-)Zufallszahl zurück. Die Pseudoeigenschaft rührt daher, dass Matlab die Zufallszahlen deterministisch bestimmt.
Mit der Funktion `diag` können Diagonalmatrizen angelegt werden. Für einen Vektor `x` erzeugt `diag(x)` eine Diagonalmatrix mit den Einträgen von `x` auf der Diagonalen.

Die explizite Erzeugung von Matrizen bzw. Vektoren erfolgt in Matlab über die Klammernotation. So kann eine 3×3 Matrix mit den ersten neun natürlichen Zahlen mit folgendem Befehl erzeugt werden:

```
Telefonmatrix = [1 2 3; 4 5 6; 7 8 9]
```

Das Ende einer Zeile (bis auf die letzte) wird über das Semikolon angegeben. Innerhalb einer Zeile können einzelne Elemente sowohl über ein Leerzeichen als auch über ein Komma getrennt werden.

Häufig sehr praktisch ist das Erzeugen von Matrizen durch Angabe von Blöcken, anstatt der einzelnen Elemente. Sei bspw. `B = [1 2; 3 4]`, dann ergibt `C = [B zeros(2); ones(2), eye(2)]` die Matrix

$$C = \begin{pmatrix} 1 & 2 & 0 & 0 \\ 3 & 4 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix}.$$

Blockdiagonalmatrizen können noch einfacher und direkt über die Funktion `blkdiag` erzeugt werden. So ergibt `A = blkdiag(B, ones(2))` die Matrix

$$A = \begin{pmatrix} 1 & 2 & 0 & 0 \\ 3 & 4 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}.$$

Die direkte Indizierung von Matrixeinträgen erfolgt in Matlab durch runde Klammern und startet bei 1. Bei Matrizen steht der erste Index für die Zeilen-, der zweite für die Spaltennummer des Elements. Für die obige Matrix A gibt Matlab für `A(2,1)` den Eintrag 3, für `A(3,3)` den Eintrag 1 und für `A(1,4)` den Eintrag `0` wieder. Innerhalb von Matrizen kann auch auf einzelne Teilmatrizen zugegriffen werden. Dafür ersetzt man einfach beim Indizieren bzw. beim Zugriff die Zahlen durch Vektoren.

Wichtig: Es gibt in Matlab (im Unterschied zu C und Java) nur positive Indizes. So wird etwa der Befehl `A(0,0)` mit einer Fehlermeldung bestraft.

Aufgabe 6 (Zufallsmatrizen und Blockmatrizen)

a) Erzeugen Sie eine 4×4 Zufallsmatrix und speichern Sie ihre Diagonaleinträge in den Variablen a, b, c und d. Erstellen Sie anschließend eine Diagonalmatrix mit den Einträgen a, b, c und d.
b) Erstellen Sie eine 8×8 Blockmatrix, die zweimal die erzeugte Diagonalmatrix auf der Diagonalen und sonst nur Nulleinträge enthält.
c) Lassen Sie sich die obere linke und die untere rechte 2×2 Matrix der Blockmatrix ausgeben.

Ein weiterer wichtiger Operator ist der Doppelpunkt-Operator. Mit seiner Hilfe können spezielle Zeilenvektoren erzeugt werden, die z.B. bei der Indizierung in `for`-Schleifen oder beim Plotten von Funktionen verwendet werden. Dabei wird ausgehend von einer Zahl solange eine Einheit addiert und in dem Vektor gespeichert, bis ein vorgegebenes Ende erreicht oder überschritten wurde. Die allgemeine Syntax ist:

`<Start>:<Ende>` oder `<Start>:<Increment>:<Ende>`

Dem Doppelpunkt-Operator verwandt ist die Funktion `linspace`, die als Eingabe neben Start und Ende die Anzahl der zu erzeugenden Punkte verlangt (anstatt des Abstands). `linspace(a,b,n)` erzeugt `n` Punkte gleichen Abstands zwischen `a` und `b`. Der Standardwert für `n` ist 100.
Zur Wiedergabe von Blockmatrizen wird häufig auch der Doppelpunkt-Operator verwendet. Die Teilmatrix einer Matrix `A` bestehend aus der Schnittmenge der Zeilen `p` bis `q` und den Spalten `r` bis `s` wird mit `A(p:q,r:s)` zurückgegeben. Für die oben definierte Matrix A gibt daher `A(2:3,1:4)` die Matrix A' mit

$$A' = \begin{pmatrix} 3 & 4 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

wieder. Ein Sonderfall ist ein einzelner Doppelpunkt, dieser wählt sämtliche Zeilen oder Spalten aus. `A(:,j)` bezeichnet also die `j`-te Spalte, und `A(i,:)` die `i`-te Zeile von `A`. Eine alternative Schreibweise zu `A(2:3,1:4)` ist daher `A(2:3,:)`.
Das Schlüsselwort `end` steht für den letzten Index in der angegeben Dimension; `A(end,:)` bezeichnet also die letzte Zeile und `A(:,end)` die letzte Spalte von `A`. Folglich ist `A(2:3,1:end)` ebenfalls eine alternative Schreibweise zu `A(2:3,1:4)`.

Aufgabe 7 (Doppelpunkt-Operator und Wiedergabe von Untermatrizen)
Erzeugen Sie eine 5×5 Zufallsmatrix und speichern Sie diese in der Variablen `A`. Lassen Sie sich die erste Zeile sowie die letzte Spalte ausgeben. Lassen sie sich außerdem die innere 3×3 Matrix ausgeben und nutzen Sie den Doppelpunkt-Operator.

Rechnen mit Matrizen

Matrizen kann man in Matlab wie gewohnt mit + addieren und mit - subtrahieren. Damit keine Fehlermeldung ausgegeben wird, ist es wichtig, dass beide Matrizen die gleichen Dimensionen besitzen. Matlab interpretiert den Multiplikationsoperator * als Matrixprodukt oder als Multiplikation mit einem Skalar. Bei ersterem muss die Anzahl der Spalten des ersten Arguments gleich der Anzahl der Zeilen des zweiten Argumentes sein. Daneben gibt es noch den **elementweisen** Multiplikationsoperator `.*`. Diesen kann man ebenfalls sowohl mit Skalaren als auch mit Matrizen gleicher Größe verwenden. Man mache sich jedoch klar, dass `[1 2; 3 4]*[1 2; 3 4]` das Matrixprodukt `[7 10; 15 22]` ausgibt, wogegen `[1 2; 3 4].*[1 2; 3 4]` das elementweise Matrixprodukt `[1 4; 9 16]` berechnet. Bei einem elementweisen Matrixprodukt werden in Matlab also die Elemente beider Faktoren an den Positionen `(i,j)` miteinander multipliziert.
Das Potenzieren ^ wird ebenfalls im Sinne des Matrixproduktes interpretiert. Analog zur Multiplikation gibt es die „gepunktete“ bzw. elementweise Version `.^` für Skalare und gleichgroße Matrizen.
Die Division gibt es für Matrizen in zwei Ausführungen: den Slash-Operator `/` und den Backslash-Operator \.
Beide entsprechen dem (ggf. approximativen) Lösen eines linearen Gleichungssystems. Für eine invertierbare Matrix A und eine Matrix B steht `B/A` für BA^{-1} und `A\B` für $A^{-1}B$. Natürlich gibt es auch wieder das elementweise Dividieren `./` für Skalare und gleichgroße Matrizen. Dieses sollte eingesetzt werden, wenn bspw. ein Vektor durch einen Skalar geteilt wird.
Um eine Matrix zu transponieren, stellt Matlab den Apostroph-Operator `'` zur Verfügung. `A'` steht also für A^T.
Die Inverse einer regulären Matrix `A` berechnet man (ggf. approximativ) mit `inv(A)`. Bei nicht quadratischen oder singulären Matrizen wird eine Fehlermeldung zurück gegeben.

Wichtig: Möchten Sie in Matlab ein lineares Gleichungssystem lösen, so ist es **nicht** empfehlenswert, erst die Inverse über `inv` zu berechnen. Dieses Vorgehen benötigt wesentlich mehr Rechenoperationen als ein direkter Gebrauch des Backslash-Operators. Grund dafür ist, dass in Matlab bereits spezielle numerische Verfahren zur Lösung von LGS implementiert sind (siehe AlgoMath-Vorlesung).

Aufgabe 8 (Rechnen mit Matrizen)
Betrachten Sie die Matrix A und den Vektor b, die wie folgt definiert sind:

$$A = \begin{pmatrix} 4 & 3 & -2 \\ 5 & 1 & 7 \\ -1 & -3 & 8 \end{pmatrix} \quad \text{und} \quad b = \begin{pmatrix} 5 \\ 28 \\ 16 \end{pmatrix}.$$

a) Berechnen Sie das Quadrat von A (d.h. `A^2`), das elementweise Quadrat von A (d.h. `A.^2`) sowie das elementweise Matrixquadrat von A (d.h. `A.^A`). Was sind die Unterschiede?
b) Berechnen Sie die euklidische Norm von b. (Es gilt $\|b\|_2 := \langle b, b\rangle_2^{1/2} = (b^T b)^{1/2}$.)
c) Bestimmen Sie die Inverse von A.
d) Lösen Sie das Gleichungssystem $Ax = b$ mit Hilfe des Backslash-Operators.

Matlab verfügt über unzählige Funktionen, die Vektoren und Matrizen als Eingabe erwarten. Grundsätzlich verändert eine Funktion in Matlab kein Eingabeargument. Man kann diese Funktionen grob in die Kategorien *Matrixfunktionen*, *Vektorfunktionen* und *skalare Funktionen* unterteilen. Matrixfunktionen sind z.B.:

- `length`: Gibt die größere der beiden Dimensionen einer Matrix zurück (sinnvoll bei Vektoren).
- `size`: Gibt die Dimensionen einer Matrix zurück.
- `det`: Gibt die Determinante einer quadratischen Matrix zurück.
- `abs`: Gibt eine Matrix mit den Absolutbeträgen der Eingabematrix zurück.

Skalare Funktionen sind solche, die komponentenweise wirken. Beispiele sind `sin`, `cos`, `exp` und `factorial` (Fakultät). Übergibt man diesen Funktionen eine Matrix, hat die Rückgabe die gleichen Dimensionen wie das Eingabeargument und die Funktion wurde auf jeden Eintrag des Feldes angewandt. Bspw. gibt `sin([pi/2 0; 0 pi/2])` die Matrix `[1 0; 0 1]` zurück.

Vektorwertige Funktionen operieren auf Vektoren und geben einen Skalar oder einen Vektor zurück. Beispiele sind:

- `max` / `min`: Gibt das Maximum / Minimum eines Vektors zurück.
- `sum` / `prod`: Gibt die Summe / das Produkt aller Einträge eines Vektors zurück.
- `diff`: Gibt einen Vektor mit der Differenz der jeweils aufeinander folgenden Elemente des Eingabevektors wieder.
- `cumsum`: Gibt einen Vektor mit der kumulierten Summe der jeweils aufeinander folgenden Elemente des Eingabevektors wieder.

Matlab bietet noch viele weitere Funktionen. Zwei übersichtliche Listen kann man sich mit den Befehlen `help matfun` und `help elmat` ausgeben lassen.

Aufgabe 9 (Matrizen- und Vektorfunktionen)
Betrachten Sie die Matrix B, die wie folgt definiert ist:

$$B = \begin{pmatrix} 7 & 2 & -1 & 3 \\ -2 & 0 & 5 & 4 \\ 0 & -5 & 6 & -2 \end{pmatrix}.$$

a) Initialisieren Sie die Matrix B in Matlab und geben Sie die Kommandos `length(B)` und `size(B)` ein. Was ist der Unterschied zwischen diesen Befehlen?
b) Berechnen Sie die Determinante der 3×3-Untermatrix von B, die aus den ersten drei Spalten von B besteht.
c) Lassen Sie sich das größte Element der Matrix B ausgeben.