Freie Universität Berlin

Crashkurs: MATLAB (Teil II)

Übungsaufgaben

Im Rahmen des Mentorings (SoSe 2019)

10.04.2019

Funktionen, Kontrollstrukturen, Abbildungen

Hinweis. *Schreiben Sie jede der folgenden Aufgaben in eine eigene Skript-Datei und speichern Sie diese unter einem geeigneten Dateinamen in Ihrem bereits angelegten Arbeitsverzeichnis (bspw.* ***matlabGrundlagen****) ab.*

Aufgabe 0. (Wiederholung Theorie)

1. Nutzen Sie eine Informationsquelle Ihrer Wahl, um die folgenden Fragen zu beantworten.

   (a) Wie können Kopien von Matrizen widerholt werden?

   (b) Wie kann man eine Zahl auf- bzw. abrunden?

   (c) Mit welcher Funktion kann man die Diagonale einer Matrix $A \in \mathbb{R}^{n\times n}$ extrahieren, d.h. den Vektor $A_{11}, \ldots, A_{nn} \in \mathbb{R}^n$?

   Bemerkung. Für umfangreiche Ergebnisse lohnt es sich häufig die Suche in englischer Sprache durchzuführen.

Aufgabe 1. Es soll eine Funktion erstellt werden, die den Umfang und den Flächeninhalt eines Kreises berechnet.

Bemerkung. MATLAB besitzt neben bereits integrierten Funktionen auch integrierte Konstanten. Für die Kreiszahl π lautet die Konstante `pi`.

Aufgabe 2. Implementieren Sie die Funktion f, die für $x \in \mathbb{R}$ durch

$$f(x) \quad := \quad \begin{cases} 0 & falls \quad x < 0 \\ \frac{1}{2} \cdot x^2 & falls \quad 0 \leq x \leq 1 \\ x & falls \quad x > 1. \end{cases}$$

definiert ist. Wählen Sie einen kompatiblen Namen für Ihre Funktion und vergewissern Sie sich von der Richtigkeit Ihrer Implementierung, indem Sie `f` an geeigneter Stelle auswerten.

Aufgabe 3. Implementieren Sie die Betragsfunktion, die für $x \in \mathbb{R}$ durch

$$f(x) \quad := \quad \begin{cases} x & falls \quad x \geq 0 \\ -x & falls \quad x < 0. \end{cases}$$

definiert ist. Legen Sie dazu eine neue Funktion `myabs` an und testen Sie Ihre Implementierung, indem Sie die Ergebnisse von `myabs` für geeignete Eingabeparameter mit denen der Standardfunktion `abs` vergleichen.

Aufgabe 4. (for-Schleifen: Weitere Konstruktionen)
Lesen Sie die folgenden Code-Beispiele durch und versuchen Sie den Programmablauf nachzuvollziehen. Führen Sie die Code-Beispiele anschließend in einzelnen Skript-Dateien aus und vergleichen Sie Ihre Erwartungen mit den tatsächlichen Ergebnissen.

```
% Standardbeispiel: Iteriere ueber alle Zahlen von 1 bis 10
% und berechne das Quadrat der jeweiligen Ierierten.

for i = 1:10        % i wird ein Vektor mit den Zahlen 1 bis 10
  display(i^2);     % als Inkrement zugeordnet.
end
```

```
% Verschachtelte for-Schleifen.
for i = 1:3             %  Erst wird die aeussere Schleife,
  for j = 1:3           %  dann die innere Schleife durchlaufen.
    display(i*j);
  end
end
```

```
% Beispiel mit einem beliebigen Vektor.
c = 0;
for i = [7 4 6 2 8]
  c   = c+i;            % Addiere die aktuelle Ierierte auf c.
end


display('Die Summe aller Ierierten:');   % Ausgabe Endergebnis.
display(c);
```

```
% Elegantere Version:
v = [7 4 6 2 8]
c = 0;
for i = 1:length(v)
  c   = c+v(i);
end


% Berechne die Inverse eines jeden Eintrags.
v   = [5 3 8 4 0 9 2];
for vi = v

  % Falls vi == 0 ist, ist die Inverse nicht definiert.
  % Wir brechen die aktuelle Iteration ab und machen
  % beim naechsten vi weiter.

  if vi == 0
    display('vi = 0 kann nicht invertiert werden.');
    continue;
  end

  display(1/vi);
end
```

Aufgabe 5. Implementieren Sie

$$\sum_{k=0}^{n} k^2 = 0 + 1 + 4 + 9 + \dots$$

als Funktion. Benennen Sie die neu angelegte Funktion mit `mySum`. Für $n = 5$ sollte der Wert 55 ausgegeben werden.

Aufgabe 6. Implementieren Sie eine Funktion `alternatingSum`, die zu einem Vektor $v \in \mathbb{R}^n$ die alternierende Summe

$$-v_1 + v_2 - v_3 + \dots$$

berechnet. Prüfen Sie Ihre Implementierung mit geeigneten Beispielen. Für $v = (-2, 5, 10, 0)$ sollte der Rückgabewert beispielsweise -3 sein.

Aufgabe 7. Definieren Sie eine Funktion `myprod`, die zu einem Eingabevektor $v \in \mathbb{R}^n$ das Produkt seiner Einträge zurückgibt, d.h.

$$v_1 \cdot v_2 \cdot v_3 \cdots v_n.$$

Prüfen Sie die Richtigkeit Ihrer Implementierung, indem Sie die Ergebnisse von `myprod` für geeignete Eingabeparameter mit denen der Standardfunktion `prod` vergleichen. Für $v = (15, -6, -1, 8)$ sollte der Wert 720 herauskommen.

Bemerkung. Die Standardfunktion `prod()` ist eine kürzere Schreibweise zur Berechnung des Produkts der Einträge eines Vektors $v \in \mathbb{R}^n$, d.h. $v_1 \cdot v_2 \cdot v_3 \cdots v_n$.

Aufgabe 8. Definieren Sie ein Funktionen-Handle, das der mathematischen Abbildung

$$f(x, y) = x^2 + y^2$$

entspricht. Stellen Sie dabei sicher, dass ihre Funktion auch für Matrizen sinnvoll ist. D.h. für $X, Y \in \mathbb{R}^{m \times n}$ soll $f(X, Y) = Z \in \mathbb{R}^{m \times n}$ gelten. Testen Sie ihr Funktionen-Handle mit folgenden Matrizen:

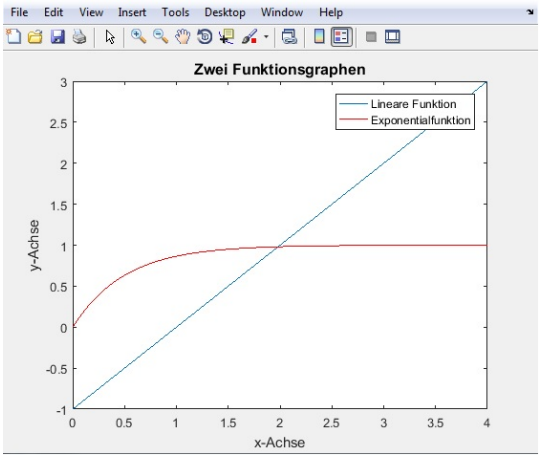
$$B = \begin{pmatrix} 1 & 0 \\ \frac{1}{\sqrt{2}} & 1 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & 1 \\ \frac{1}{\sqrt{2}} & 1 \end{pmatrix}, \quad f(X, Y) = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix}.$$

Aufgabe 9. (Plotten)

1. Die Graphen der Gerade $y = x - 1$ und der Exponentialfunktion $z = 1 - e^{-2x}$ sollen beide im Intervall $I = [0, 4]$ in einem Fenster geplottet werden. Darüber hinaus soll der Plot angemessen beschriftet sein (Titel, Achsenbeschriftung, Legende, ...) und die beiden Graphen farblich voneinander unterschieden werden können.

   Bemerkung. In MATLAB wird die Exponentialfunktion e^x durch die Funktion `exp()` dargestellt.

   Lösung:



2. Es soll ein 2×2 Subplot erstellt werden. Das erste Feld soll den Plot aus Teilaufgabe 9.1. enthalten. Die restlichen Felder sollen durch die Plots der folgenden drei Funktionen über dem Intervall $I = [0, 10]$ mit der Schrittweite 0.1 gefüllt werden. Die einzelnen Subplots sollen dabei sowohl einen Titel als auch eine Achsenbeschriftung besitzen.

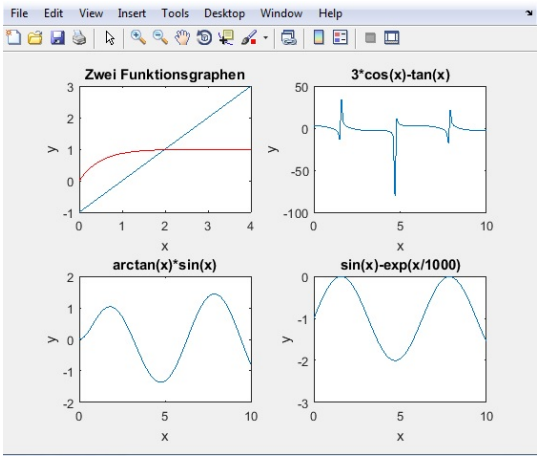
   $f_2(x) = 3 \cdot cos(x) - tan(x)$

   $f_3(x) = arctan(x) \cdot sin(x)$

   $f_4(x) = sin(x) - e^{(\frac{x}{1000})}$

   Bemerkung. In MATLAB wird der Arcustangens arctan durch die Funktion `atan()` dargestellt.

Lösung:



Zusatzaufgaben (Zum Knobeln)

Aufgabe 10. Schreiben Sie eine Funktion `evensum`, die zu einem Vektor $v \in \mathbb{R}^n$, die Summe aller geraden Einträge von v zurückgibt. Verwenden Sie dabei nicht die Funktion `sum`. Testen Sie Ihre Implementierung an einigen geeigneten Beispielen. Für $v = (5, -3, 2, 4)$ sollten Sie etwa das Ergebnis 6 erhalten.

Bemerkung. Die Benutzung der in MATLAB integrierten Modulofunktion `mod()` kann beim Lösen der Aufgabe von Vorteil sein. Des Weiteren erfordert die Aufgabe das Einfügen einer Bedingung.

Aufgabe 11. Schreiben Sie eine Funktion `sumfun`, die zu einer Matrix $A \in \mathbb{R}^{n \times m}$ den Wert

$$\sum_{i=1}^{m} \sum_{j=1}^{n} i \cdot j \cdot A_{ij}$$

berechnet. Testen Sie wie gewohnt Ihre Implementierung mittels geeigneter Testbeispiele auf Richtigkeit. Für `A = vander(1:3)` sollten Sie beispielsweise den Rückgabewert 82 erhalten.

Bemerkung. Angedacht ist, dass Sie diese Aufgabe mit Hilfe verschachtelter for-Schleifen lösen. Sie können das Problem alternativ aber auch mit einem recht kurzen „Einzeiler“ lösen.

Quellen

- `http://numerik.mi.fu-berlin.de/wiki/WS_2017/CoMaI_Dokumente/MATLAB_MiniTutorials.pdf`
- `http://www-m2.ma.tum.de/foswiki/pub/M2/Allgemeines/MatlabSoSe2012/blatt1.pdf`
- `http://people.inf.ethz.ch/arbenz/MatlabKurs/matlabintro.pdf`