# Einführung in die Programmierung mit MATLAB

Abschlussaufgaben: Abgabe bis Freitag, 14.10.2016, 18 Uhr

---

- Die Abgabe erfolgt bis spätestens Freitag, 14.10.2016, 18:00 Uhr, an folgende E-Mail-Adresse: ulrich.hartleif@wwu.de. Abgaben, die später eintreffen, werden nicht berücksichtigt! Die Aufgaben sollen als E-Mail-Anhang verschickt werden.
- Die Abgabe erfolgt in folgender Form: Für jede Aufgabe wird ein eigenes M-File (siehe Aufgabenbeschreibungen) benutzt, welches mit dem gegebenen Beispiel die Aufgabe ausführt.
- Die Aufgaben auf diesem Blatt müssen **einzeln** bearbeitet und abgegeben werden!
- Der Code muss lauffähig sein und sollte vor Abgabe mit eigenen Beispielen getestet werden, um die richtige Funktionalität des Programms sicherzustellen.
- Der Code sollte gut kommentiert sein. Jede Funktion sollte einem Kommentar beginnen, der kurz und kanpp erläutert, was die Funktion macht. Innerhalb der Funktion sollten wichtige Befehle, Abfragen oder Schleifen kommentiert werden, wozu sie da sind und wie sie funktionieren. Besser zu viel kommentieren als zu wenig.
- Wenn in der Aufgabe etwas von angeben oder ausgeben gesagt wird, so ist eine Ausgabe im Command Window gemeint: Dies kann mit dem Befehl **disp** erreicht werden.
- Benutzen Sie die Online-Dokumentation, falls Sie die genaue Funktion mancher Befehle vergessen haben sollten, oder ziehen Sie das Praktikumsskript zurate.
- Es ist immer eine gute Idee, das Problem zuerst mit Stift und Papier zu verstehen, um sich dann eine Code-Implementierung zu überlegen.

Viel Erfolg!

## Aufgabe 1: Matrizen und Vektoren

Schreiben sie ein M-File *Aufgabe1.m*, welches folgendes Aufgaben in dieser Reihenfolge ausführt:

- Schließen Sie alle offenen Fenster, löschen Sie alle Variablen des Workspaces und entfernen Sie alle alten Befehle des Command Windows.
- Erstellen Sie 8 Spaltenvektoren mit jeweils 5 zufälligverteilten Zahlen zwischen 0 und 10 und erstellen Sie aus ihnen eine Matrix $M$, die die Vektoren als Zeilen enthält. **Tipp:** 'Kleben' Sie die Vektoren geschickt zusammen.
- Setzen alle Einträge der Matrix $M$, die kleiner als 5 sind, auf Null. Löschen Sie die dritte, vierte und fünfte Zeile der Matrix.

- Erstellen Sie eine Diagonalmatrix $D$, die die Diagonaleinträge von $M$ enthält.
- Addieren Sie zu $D$ die horizontal sowie vertikal gespiegelte Matrix von $M$. Transponieren Sie das Ergebnis.
- Geben Sie die Anzahl der Einträge von $D$ aus. **Tipp:** Nutzen Sie den Befehl **size**.
- Formen Sie $D$ in einen langen Zeilenvektor um und speichern ihn in der Variable $v$. **Tipp:** Nutzen Sie den Befehl **reshape** oder **:**.
- Geben Sie die Länge von $v$, das größte und kleinste Element und der Index, wo sie zu finden sind, sowie den Mittelwert und Summe von $v$ aus. Nutzen Sie zur Ausgabe einen vollständigen Satz und nutzen Sie den Befehl **num2str** um die Zahlen in den Satz zu bekommen.
- Erstellen Sie eine Matrix $N$, die die gleiche Größe wie $M$ besitzt, jedoch nur mit Nullen gefüllt ist.
- Verändern sie $N$ so, dass unten links eine 3x3 Block mit Einsen und oben rechts ein 2x2 Block mit einer Einheitsmatrix entsteht.
- Speichern Sie alle Variablen des Workspaces in eine Datei namens *MatrizenUndVektoren.mat* ab.

### Aufgabe 2: Matrix-Vektor-Multiplikation

Schreiben Sie eine Funktion *MatrixVektor*, die als Eingabeparameter eine Matrix und einen Vektor erhält und das Matrix-Vektor-Produkt berechnet und das Ergebnis zurückgibt. Sie dürfen **nicht** die Matrix-Vektor-Multiplikation ***** oder ein Skalarprodukt benutzen! Sie sollen alles über for-Schleifen lösen. Gehen Sie folgendermaßen vor:

- Prüfen Sie, ob Sie den Vektor von links oder von rechts mit der Matrix multiplizieren können. Falls keine Matrix-Vektor-Multiplikation möglich ist, brechen Sie das Programm mit dem Befehl **error** und einem sinnvollen Text ab.
- Implementieren Sie die Matrix-Vektor-Multiplikation von links und von rechts mit Hilfe der **for**-Schleife(n) und speichern Sie das Ergebnis in einem Vektor der korrekten Dimension ab.
- Geben Sie das Ergebnis als Rückgabewert zurück.

Erstellen Sie ein M-File *Aufgabe2.m* und erstellen Sie dort folgende Matrix

$$M = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$$

sowie die Vektoren $a = (-4, 7)$, $b = (-15, 5, 17)^T$ und $c = (-2, 7, -11)$. Testen Sie Ihre Funktion mit $M$ und jeweils allen drei Vektoren in der Reihenfolge $a$, $b$ und zuletzt $c$ und geben Sie die Ergebnisse aus.

### Aufgabe 3: Numerische Berechnung der Kubikwurzel

Zur Berechnung der Kubikwurzel einer beliebigen Zahl $a$ sei folgendes Iterationsverfahren gegeben:

$$x_{n+1} = \frac{2x_n + \frac{a}{x_n^2}}{3},$$

mit einen Startwert $x_0 = a$.

Schreiben Sie eine Funktion *Kubikwurzel*, die folgende Parameter erhält: die Zahl $a$ und eine Toleranz $\epsilon$. Die Funktion soll folgende Parameter zurückgeben: die berechnete Kubikwurzel sowie die Zahl der benötigten Iterationen. Verwenden Sie **keine** Matlabfunktion zure Berechnung der Wurzel!

Gehen Sie folgendermaßen vor:

- Initialiseren Sie den Counter und die Approximation der Kubikwurzel.
- Behandeln Sie den Fall $a = 0$ gesondert.
- Nutzen Sie eine while-Schleife mit Bedingung $|x_{n+1} - x_n| >= \epsilon$. Sie benötigen also eine Variable, in der Sie das alte Ergebnis speichern. Initialisieren Sie diese mit 0.
- Berechnen Sie innerhalb der Schleife die neue Approximation gemäß obiger Formel. Vergessen Sie nicht, den Counter zu erhöhen.

Schreiben Sie nun ein M-File *Aufgabe3.m* und testen Sie Ihr Programm mit folgenden Zahlen: -125, 0, 1000, 16 und -2. Geben Sie das Ergebnis sowie die benötigten Iterationen aus.

### Aufgabe 4: Grafische Ausgaben

Schreiben Sie ein M-File *Aufgabe4.m*, welches folgendes tut:

- Lassen Sie sich alle offenen Fenster schließen und alle Variablen des Workspaces löschen.
- Betrachten Sie die Funktionen $\sinh(x)$ und $\cosh(x)$ auf dem Intervall $[-2, 2]$. Diskretisieren Sie das Intervall ausreichend und werten Sie die Funktionen aus und plotten Sie sie mit **hold** in eine Zeichenfläche. Verwenden Sie unterschiedliche Farben und unterschiedliche Linientypen. Beschriften Sie die Achsen und vergeben Sie einen Titel. Erstellen Sie eine Legende und platzieren Sie sie so, dass sie keine Kurve verdeckt.
- Markieren Sie das Minimum von cosh mit einer Raute. **Tipp:** Plotten Sie einen weiteren Plot geschickt in die gleiche Zeichenfläche. Sie sollen das Minimum berechnen und **nicht** händisch angeben.
- Lassen Sie sich den Plot automatisch als .fig, .png und .pdf abspeichern.

Betrachten wir nun folgende Funktion:

$$f(x, y) = -(5 - x)^2 y^3 - 19 + 5xy^2.$$

- Schließen Sie alle offenen Fenster und löschen Sie alle Variablen.
- Definieren sie $f$ als anonyme Funktion, die auch mit Matrixeingaben umgehen kann. Diskretisieren sie $[-7, -3] \times [-5, 5]$ ausreichend und werten Sie $f$ an den Stellen aus. **Tipp:** Nutzen Sie **meshgrid**.
- Plotten Sie mit **subplot** die unterschiedlichen Varianten von 3D-Plots, die Sie kennen, in eine figure. Beschriften Sie alles ausreichend.