Technische Universität Chemnitz
Fakultät für Mathematik
Professur Mathematik in Industrie und Technik

Sommersemester 2009
Prof. Dr. Peter Benner
Dipl.-Math. Jens Saak
Dipl.-Ing. Martin Bernauer

Numerische Mathematik– 1. Übung

A) Wiederholung

Aufgabe 1

Erklären Sie die Begriffe

a) Vektorraum,

b) lineare Unabhängigkeit, Basis,

c) Orthogonalität.

Aufgabe 2

a) Gegeben $A = \begin{bmatrix} 4 & 16 & 32 \\ 0 & 4 & 8 \\ 0 & 0 & 4 \end{bmatrix}$, sowie $b = \begin{bmatrix} 53 \\ 13 \\ 5 \end{bmatrix}$. Lösen Sie $Ax = b$ durch Rückwärtseinsetzen.

b) Gegeben $A = \begin{bmatrix} 4 & 1 & -2 \\ 0 & 4 & 8 \\ 1 & 0 & 5 \end{bmatrix}$, sowie $b = \begin{bmatrix} -3 \\ 4 \\ 5 \end{bmatrix}$. Lösen Sie $Ax = b$ durch Gauß-Elimination.
Zusatz: Geben Sie eine LR Zerlegung von A an.

Aufgabe 3

Sind die Vektoren $u = [1; 6; 3\,]^T$, $v = [1; 2; 3]^T$, $w = [-2; -1; -6]^T$ linear unabhängig?

Aufgabe 4

Prüfen Sie $u = [2; 1; -1; 0]^T$, $v = [0; -3; 2; -4]^T$, $w = [1; -2; 4; 1]^T$ auf lineare Unabhängigkeit und ergänzen sie durch eine Auswahl aus $x = [1; 0; -3; -5]^T$, $y = [2; -1; 6; 5]^T$, $z = [-1; 1; 0; 5]^T$ zu einer Basis des $\mathbb{R}^4$.

Aufgabe 5

Geben sie die allgemeine Form eines Vektors $x \in \mathbb{R}^3$ an, der orthogonal zu $y = [3; -2; 5]^T$ ist.

Aufgabe 6

Überführen sie $u = [5; 0; 0]^T$, $v = [0; 1; -\sqrt{3}]^T$, $w = [6; -2; 8]^T$ in eine Orthonormal-Basis des $\mathbb{R}^3$.

B) Einführung in `Matlab`

Interpretieren Sie die Bildschirm-Ausgaben nach Eingabe der folgenden MATLAB-Anweisungen.
Achten Sie auf unterschiedliche Eingabemöglichkeiten (Bedeutung von `, ; :`) und deren Auswirkungen aufs Ergebnis. — Notieren Sie sich ggf. die Bedeutung einzelner Anweisungen.
(Tip: mit Kursortasten ↑, ↓ kann man vorherige Eingabezeilen *„zurückholen“* und ändern):

1. **Eingabe, Verwendung von Variablen, Matrizen und Vektoren**

```
≫ pwd
≫ cd matlab
≫ 3*4
≫ pi/2
≫ A = [1 2 3 4 5 6 7 8 9]
≫ A = [1 2 3; 4 5 6; 7 8 9]
≫ M = magic(6)
≫ B = [4, 5, 6.1], x=[ -1.3  sqrt(3)  (1+2+3)^3 ]
≫ E = ones(3), F = zeros(3)
≫ e = ones(1,3), f = zeros(3,1)
≫ G = 3*diag(e) - 2*E
≫ E * G, E .* G, p1 = e*e', p2 = e'*e
≫ x(8) = -x(2); y=x';
```

2. **Ausgabe von Ergebnissen in unterschiedlichem Format**

```
≫ display(3*4)                                  Standardausgabeformat mit Variablenname
≫ disp(3*4)                                     nur Werte ohne Variablenname
≫ disp('      A         B'), disp(rand(4,2))
≫ fprintf('Die Zahl pi lautet: %20.16f...\n',pi)        mit Textformatierung
≫ pause; disp(1); pause(5); disp(2); pause(0)
```

3. **Verschiedene Zahlendarstellungen**

```
≫ x, y
≫ format long
≫ x
≫ format long e
≫ x
≫ format +
≫ x
≫ format short
≫ omega = sqrt(-3)
≫ z1 = 10^20
≫ z2 = 1e20
≫ s=1/0, t=1/s, u=t*s
```

4. **Verwendung des Doppelpunktes (Zähl-Intervalle, Indexbereiche)**

```
≫ n=10
≫ 1:n
≫ 1:2:n
≫ 2:2:n
≫ linspace(0,10,6), cos(linspace(0,pi,5))
≫ y = x(3:5)
≫ y(4:6)=x(1:3)
≫ r = [10 11 12]
```

```
≫ q = [13:16]', p = [13;14;15;16]
≫ B = [ A; r ]
≫ C = [ B p ]
≫ C = C(:,4:-1:1)
≫ A = C(1:2,:)
≫ size(B)
≫ A = C(:,1:3)
≫ [n,m] = size(A)
≫ v = A(:), w=reshape(A,1,m*n)
≫ x = pi*(0:1/2:2) oder: x=linspace(0,2*pi,5)
≫ s = sin(x), c = cos(x)
```

5. **Eigene Funktionen, Schleifen, Verzweigungen**

```
≫ edit potmat.m
```
ein sog. „M-file“ anlegen (auch mit belieb. Editor)

Inhalt der Datei `potmat.m`:

```
function A=potmat(x,n)
% potmat(x,n) - erzeugt m-kreuz-n Matrix deren Spalten aus
%               elementweisen Potenzen des Vektors x sind
%     A= [1 x x.^2 ... x.^(n-1)]
%
if ((n<1) || (size(x,2)~=1))
  error('n>=1 ist erforderlich, x muss Spaltenvektor sein');
else
  m=length(x);
  A=zeros(m,n);
  for i=1:n
    A(:,i)=x.^(i-1);
  end
  return;
end
```

Nutzung der Funktion `potmat`:

```
≫ help potmat
```
zeigt den selbst eingebauten Hilfetext an

```
≫ C=potmat([0;1;2;3],3)
```

6. **Rechnen mit Matrizen**

```
≫ v=logspace(-1, 1, 3)'
≫ D=diag(v)
```
Diagonalmatrix

```
≫ A=C+D;
```
! Dimensionen passen nicht

```
≫ B=C(1:3,:)
```
! Dimensionen passen nicht

```
≫ A=B+D,
≫ E=eig(A)
```
Eigenwerte von C

```
≫ [V,D]=eig(A)
```
Eigenvektoren $V_1, \ldots, V_n$ und Diagonalmatrix der EW

```
≫ diag(D)
≫ diag(E)
≫ W=A*V-V*D
```
sollte bei exakter Rechnung Null sein

```
≫ max(abs(W))
≫ max(max(abs(W)))
≫ c=rand(3,1)
```
Zufallsvektor

```
≫ b=A*c
≫ % Loese lineares Gleichungssystem Ax=b :
≫ x=A^(-1)*b, oder:  x=inv(A)*b,        aber besser:
≫ x=A\b    oder dasselbe:    x=(b'/A')'
≫ x-c                                          sollte bei exakter Rechnung Null sein
```

7. **Grafische Darstellungen**

```
≫ x=linspace(0,2*pi); c=cos(x); s=sin(x);
≫ plot(c)
≫ plot(x,c)
≫ subplot(2,1,1),plot(x,c),title('cos')
≫ subplot(2,1,2),plot(x,s),title('sin')
≫ subplot(1,1,1),plot(c,s)
≫ axis equal
≫ figure(2)
≫ plot(c), hold on, plot(s), hold off
≫ plot(x,c,x,s)
≫ plot(x,c,'o',x,s,':')
≫ Y = [ c; s ]; plot(x,Y)
≫ title('Funktionen')
≫ xlabel('x-Achse')
≫ ylabel('f(x)')
≫ grid
≫ x=-8:0.5:8; y=x';
≫ X=ones(size(y))*x; Y=y*ones(size(x));
≫ R=sqrt(X.^2+Y.^2)+eps;
≫ Z=sin(R)./R;
≫ mesh(Z);
≫ view(45,45)                          3D-Ansicht (Winkel in Kugelkoordinaten)
≫ view(0,90)
≫ view(45,90)
≫ view(0,0)
≫ view(3)                              (=Standardwerte für 3D-plot: (-37.5,30))
```

8. **Hilfsfunktionen**

```
≫ diary                    Ein-/Ausschalten der Protokollierung (in eine Datei)
≫ who                                          Liste aller Variablen
≫ whos                                         Variablenliste mit Speicherbedarf
≫ save temp                                    alle Variablen retten als 'temp.mat'
≫ clear; whos                                  alle Variablen löschen
≫ load temp A; whos                            nur Matrix A wieder einlesen
≫ load temp                                    alle geretteten Variablen wiederherstellen
≫ who
≫ save tmpmat A                                nur die Matrix A speichern
≫ save a.dat A -ascii                          dasselbe, aber lesbar
≫ dir
≫ clear A; who
≫ load a.dat; who                              Matrix A heißt jetzt a
```

Hilfe allgemein und zu einzelnen Kommandos:

```
≫ help                                                    auch: doc
```

```
≫ help [
≫ help punct
≫ help inv
≫ doc det
≫ help diag
≫ lookfor root
≫ ...
```

9. **Mitgelieferte Demos**

```
≫ intro
≫ demo
```