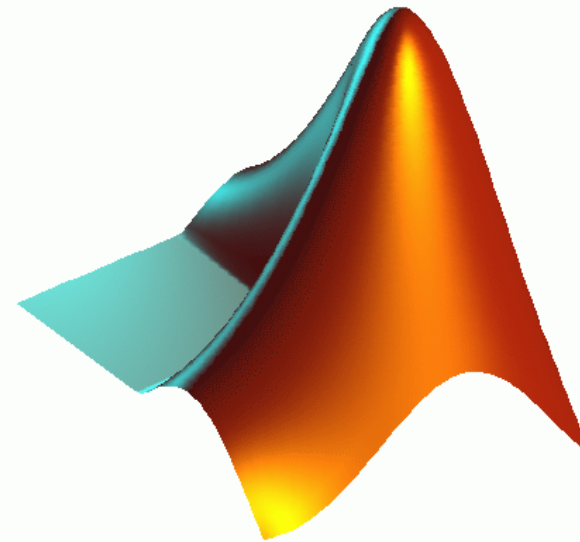
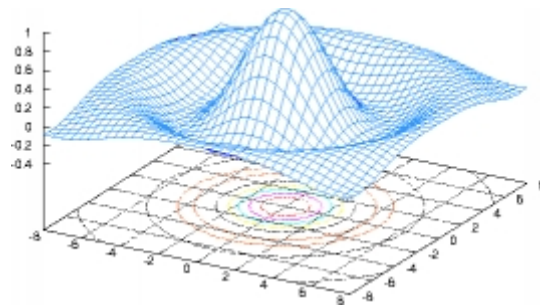




Matlab – Eine Einführung

Barbara Hammer, Alexander Hasenfuß,
TU Clausthal





Was ist Matlab / Octave?

- Matlab ist ein Produkt von MathWorks (www.mathworks.com) für
 - Studierende, Akademiker, kommerzielle Nutzer (gestaffelte Preise, Studierendenzulassung um die 100 €, sonst deutlich teurer)
 - an der Uni z.B. über das RZ oder im IFI Pool verfügbar (Server gebunden)
- Octave ist eine unter den Bedingungen der GNU General Public Licence frei verteilte Software (www.gnu.org/software/octave)
 - etwas weniger komfortabel (nur Kommandozeile), aber in großen Teilen mit Matlab kompatibel
 - historisch aus der Begleitsoftware zu einem Lehrbuch über Chemical reactor design (Rawlings/Ekerdt) entstanden
 - nützlich, etwa wenn man zu Hause ohne Serververbindung arbeiten muß
- online Einführung:
 - matlab: homepages.fh-regensburg.de/~wah39067/Matlab/MTut/
 - octave: www.gnu.org/software/octave/doc/interpreter/



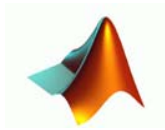
Was ist Matlab / Octave?

- Kern: extrem mächtige Anwendungssprache für mathematische Berechnungen (Lösen von Gleichungssystemen, Nullstellensuche, ...) und mathematische Visualisierung (Funktionsplot, , ...)
- durch anwendungsspezifische Toolboxen und Programme erweiterbar
- Anwendungsgebiete:
 - Wissenschaftliches Rechnen (viele Akademiker publizieren Algorithmen in der Forschung in Matlab)
 - automatisierte Visualisierung von Versuchsdaten
 - Kontrolle und Design von Systemen, Signalverarbeitung (DLR, VW, tcelectronic, Toyota Racing, Boeing, NASA, KURSK Bergungsteam, US airforce, ..., ingenieurwissenschaftliche Studiengänge an der TUC)
 - Hardwareentwurf (technische Informatik)
 - Bioinformatik, Finanzmathematik, Maschinelles Lernen, Numerik
 - ...



Getting started ...

matlab:

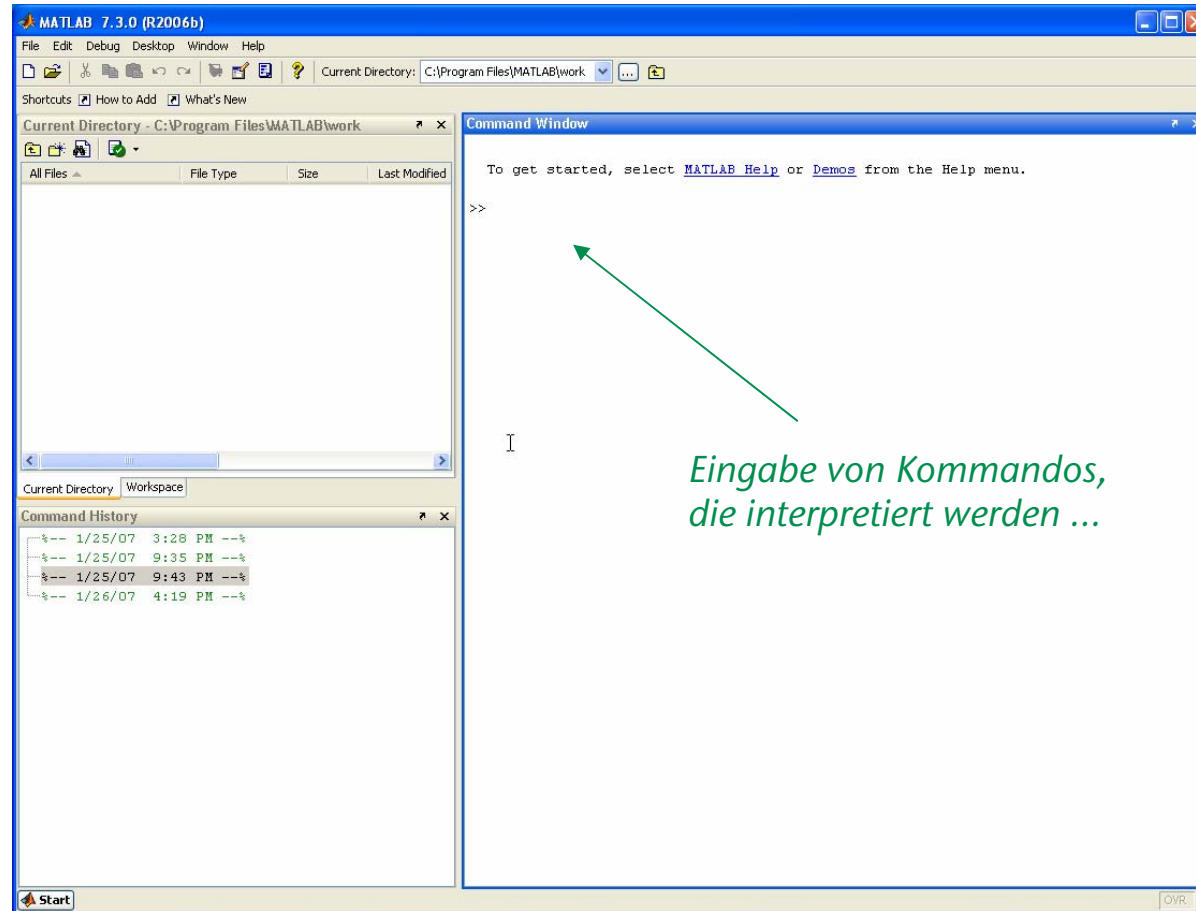


Variablen und Dateien

Historie

beenden

Hilfe



Eingabe von Kommandos,
die interpretiert werden ...

Getting started ...

- octave: Kommandozeile `octave` bzw `icon`



```
Hammer@bhammer ~
$ octave
GNU Octave, version 2.1.73 (i686-pc-cygwin).
Copyright (C) 2006 John W. Eaton.
This is free software; see the source code for copying conditions.
There is ABSOLUTELY NO WARRANTY; not even for MERCHANTABILITY or
FITNESS FOR A PARTICULAR PURPOSE. For details, type 'warranty'.

Additional information about Octave is available at http://www.octave.org.

Please contribute if you find this software useful.
For more information, visit http://www.octave.org/help-wanted.html

Report bugs to <bug@octave.org> (but first, please read
http://www.octave.org/bugs.html to learn how to write a helpful report).

octave:1> _
```

Eingabe von Kommandos,
die interpretiert werden...

Hilfe: `help -i`

Historie: Pfeile `↑ ↓`

Variablen: Eingabe des Namens

beenden: `exit`



Hello world!

- **Ausgabe:** `fprintf('hello world\n')`
- **Dateien:** m-file-Skripts (Extension `.m`)
 - enthalten eine Liste ausführbarer Befehle → *Beispiel1.m*
 - Kommentar: `% ...`
 - ausführen: Eingabe des Dateinamens auf der Kommandozeile
 - benutzt Variablen des aktuellen workspace
- **Variablen:**
 - Typen
 - Zahlen (z.B. `42`, `1.0e-10`), komplexe Zahl (z.B. `42 + 42i`)
 - String (z.B. `'helloworld'`) → *Beispiel2.m*
 - Vektor (z.B. `[1 2 3]`), Matrix (z.B. `[1 2 3 ; 4 5 6]`)
 - Struktur (Komponenten als `x.y`)
 - werden einfach zugewiesen, gegebenenfalls überschrieben, `ans` ist je der zuletzt berechnete Wert, `clear` macht den workspace leer
 - Matlab kann rechnen!



Hello world!

- **Beispiel** – Matrixoperationen, die Basis von Matlab:

```
% Definition von Vektoren und Matrizen
a = [1 2 3]    % Vektor
a = [1,2,3]
a = [1 : 3]    % Vektor, angegeben durch Grenzen
b = [1 2 3 ; 4 5 6] % Matrix
b = [1:3 ; 4:6]
c = [1:3:19]  % Schrittweite 3
d = ones(7,7) % Matrix mit Eintrag 1
e = rand(7,7) % Matrix mit gleichverteilten Zufallszahlen
f = diag(c)   % Diagonalmatrix mit Diagonale c
```

→ [Beispiel3.m](#)

```
% einfache Zugriffsoperationen
[rows columns] = size(a) % Dimensionen einer Matrix
a(1)    % Elemente
a(1,2)  % Indizierung: Zeile, Spalte
b(2,3)
b(5)    % Konvertierung Matrix -> Vektor durch Konkatenierung der Spalten
```



Hello world!

■ Beispiel – Matrixoperationen (contd.):

```
% Matrixoperationen
b = b' % Transponieren
b(5)
d + f
d/f
d*10
d./f
b(1,:) % Zeilen
b(:,1) % Spalten
d(1:3 , 2:4) % Teile einer Matrix
[d d] % Konkatenieren von Matrizen
b(:,2)=[] % zweite Spalte löschen
b == 1 % binäre Matrix mit elementweisem Test

% komplexere Funktionen
sum(e) % summiert die Elemente Spaltenweise
[sortiert index] = sort(e) % sortiert alle Elemente einer Spalte
[sortiertrows indexrows] = sortrows(e,2) % sortiert die Zeilen basierend auf Spalte 2
```




Hello world!

→ *Beispiel4.m*

■ Beispiel – ein erstes Programm:

% Berechne für 'helloworld' die Summe der Zahlen entsprechend den Buchstaben

```
a = 'helloworld'           % String = Vektor von Buchstaben
a = ['hello' 'world']      % Vektor von Strings wird als Vektor von Buchstaben abgespeichert
b = [ 'h' 'e' 'l' 'l' 'o' ; 'w' 'o' 'r' 'l' 'd' ] % Matrix von Einträgen
b = [ 'hello' ; 'world' ]  % Matrix von Strings derselben Länge
%Anmerkung: Strings unterschiedlicher Länge werden in Cell Arrays gespeichert

% Buchstaben in Zahlen umrechnen
alphabet = ['a' : 'z']
zahlen = [1 : 26]

variable = a               % hier kann man auch b einsetzen
```



Hello world!

■ Beispiel – ein erstes Programm (contd.):

```
[m n] = size(variable); % semicolon: Variable wird nicht ausgegeben  
[o p] = size(alphabet); % Achtung: Zeilen und Spalten werden angegeben!
```

```
erg = 0;  
for i = 1:m  
    for j = 1:n  
        for k = 1:p  
            if ( variable(i,j) == alphabet(k) )  
                erg = erg + zahlen(k);  
            end  
        end  
    end  
end  
end
```

```
fprintf('Die Summe gibt %d\n',erg)
```



Hello world!

→ *Beispiel5.m*

■ Beispiel – Ablaufstrukturen in Matlab:

```
erg=0;  
% for-Schleife: durchläuft den Index i wie angegeben eine feste Anzahl  
% Schritte
```

```
for i=1:10  
    fprintf('Durchlauf Nummer %d\n',i);  
    erg=erg+i;  
end;  
erg %enthält 1+2+3+...+10
```

```
erg=0;  
for i=1:2:10  
    fprintf('Durchlauf mit Variable i= %d\n',i);  
    erg=erg+i;  
end;  
erg %enthält 1+3+5+7+9
```



Hello world!

■ Beispiel – Strukturen in Matlab (contd.):

```
% if-Anweisung: springt je nachdem, was erfüllt ist, zum entsprechenden  
% Programmteil
```

```
erg=8;
```

```
if (floor(erg/2)==erg/2)  
    fprintf('%d ist gerade\n',erg);  
else  
    fprintf('%d ist ungerade\n',erg);  
end;
```



Hello world!

■ Beispiel – Strukturen in Matlab (contd.):

% while Schleife: durchläuft diese solange, bis die Bedingung nicht mehr erfüllt ist

```
i=0;
erg=0;
while i<10
    erg=erg+i;
    i=i+1;
end;
erg    %das hätte man auch durch for-Schleife machen können
```

```
erg=19;
i=0;
while ((erg>1)&&(erg<10000))
    if (floor(erg/2)==erg/2) erg = erg/2
    else erg = 3*erg +1
    end;
    i=i+1;
end;
i    %das dagegen ginge nicht mit einer for-Schleife
```



Hello world!

→ *buchstabensumme.m*

- **Beispiel** – ein erstes Programm (als m-Funktion):
 - gespeichert im file <name>.m, Aufruf <name>(..)
 - lokale Variablen, nargin = Anzahl Eingabeelemente

```
function [erg] = buchstabensumme(variable)
if (nargin<1) % genug Parameter?
    fprintf('keine Eingabe'); erg = 0; return
end
% Buchstaben in Zahlen umrechnen
alphabet = ['a' : 'z']; zahlen = [1 : 26];
[m n] = size(variable); [o p] = size(alphabet); erg = 0;
for i = 1:m
    for j = 1:n
        for k = 1:p
            if ( variable(i,j) == alphabet(k) )
                erg = erg + zahlen(k);
            end
        end
    end
end
end
```



Hello world!

→ *buchstabensummeopt.m*

- **Beispiel** – ein erstes Programm (als Matrix):

```
function [erg] = buchstabensummeopt(variable)

if (nargin<1)
    fprintf('keine Eingabe');
    erg = 0;
    return
end

% Buchstaben in Zahlen umrechnen
alphabet = ['a' : 'z'];
zahlen = [1 : 26];
erg = 0;
for k = 1:26
    erg = erg + sum(zahlen(k)* (variable == alphabet(k)));
end
erg = erg';
erg = sum(erg);
```



Hello world!

- Matrizen als Basiselemente
 - viele Operationen arbeiten direkt auf Matrizen
 - Matrixoperationen extrem schnell
- Variablen
 - direkt zuweisen
 - ein globaler workspace (clear, save(name), load(name) -> .mat files)
- Strukturelemente:
 - m-files mit Funktionen und lokalen Variablen
 - if <expression> then ... elseif ... end
 - for <variable> = <expression> ... end
 - while <expression> ... end
 - switch, break und return
- viele spezielle Funktionen ...



Hello world!

- **Beispiel** – Rekursion:

→ *fac.m*

```
function [erg] = fac(a)

if a==0
    fprintf('hello world!\n')
    erg=1;
else
    fprintf('hello world!\n')
    erg = a*fac(a-1);
end
```

- **Achtung:** um selbst definierte Funktionen zu finden
 - Pfad (file → set path → add folder → save)
 - oder current directory passend setzen



Matrizen I

Die Schneehöhe:

- Es schneit und schneit und schneit ganz gleichmäßig. Jede Stunde wird die Schneehöhe gemessen: 8 Uhr – 5 cm, 9 Uhr – 5.2 cm, 10 Uhr – 5.5 cm, 11 Uhr – 6 cm, 12 Uhr – 6.3 cm, 13 Uhr – 6.7 cm, 14 Uhr – 7.2 cm, 15 Uhr – 7.9 cm, 16 Uhr – 8.5 cm, 17 Uhr – 9 cm, 18 Uhr – 9.2 cm, 19 Uhr – 9.5 cm, 20 Uhr – 10 cm, 21 Uhr – 10.2 cm, 22 Uhr – 10.4 cm, 23 Uhr – 10.7 cm
- Wie viel Schnee muss voraussichtlich morgen früh um 8 geschippt werden?





Matrizen I

Die Schneehöhe:

- Vektor der x-Werte: [8,9,10,...,23]
- Vektor der y-Werte:
[5,5.2,5.5,6,6.3,6.7,7.2,7.9,8.5,9,9.2,9.5,10,10.2,10.4,10.7]
- wir nehmen ein lineares Modell an: $y = a + b \cdot x$ (→ Ausgleichsgerade durch die Meßwerte)
- a und b müssen aus den Daten bestimmt werden
- dann ist die Schneehöhe morgen um 8 Uhr der Wert $a + b \cdot (8 + 24)$



Matrizen I

Die Schneehöhe:

- a und b müssen aus den Daten bestimmt werden
- Methode der kleinsten Quadrate: minimiere

$$\begin{aligned} \sum_i ((a + b \cdot x_i) - y_i)^2 &= \left\| \begin{pmatrix} 1 & x_1 \\ \dots & \dots \\ 1 & x_n \end{pmatrix} \cdot \begin{pmatrix} a \\ b \end{pmatrix} - \begin{pmatrix} y_1 \\ \dots \\ y_n \end{pmatrix} \right\|^2 \\ &= \left\| X^t A - Y \right\|^2 \end{aligned}$$

- Ableiten und gleich Null setzen \rightarrow

$$X^t(X^t A - Y) = 0 \Rightarrow A = (X^t X)^{-1} X^t Y$$

- sogenannte Pseudoinverse von X



Matrizen I

→ *schnee.m*

Die Schneehöhe:

```
function [hoehe] = schnee(wann)
```

```
%Messwerte
```

```
X = [8:23];
```

```
Y = [5,5.2,5.5,6,6.3,6.7,7.2,7.9,8.5,9,9.2,9.5,10,10.2,10.4,10.7];
```

```
X = [ones(1,16);X];
```

```
X=X';
```

```
Y=Y';
```

```
%Parameter durch Pseudoinverse bestimmen
```

```
A = (X'*X)^-1*X'*Y
```

```
%alternativ ist das eingebaut
```

```
A = pinv(X)*Y
```

```
hoehe = A(1) + A(2)*wann;
```



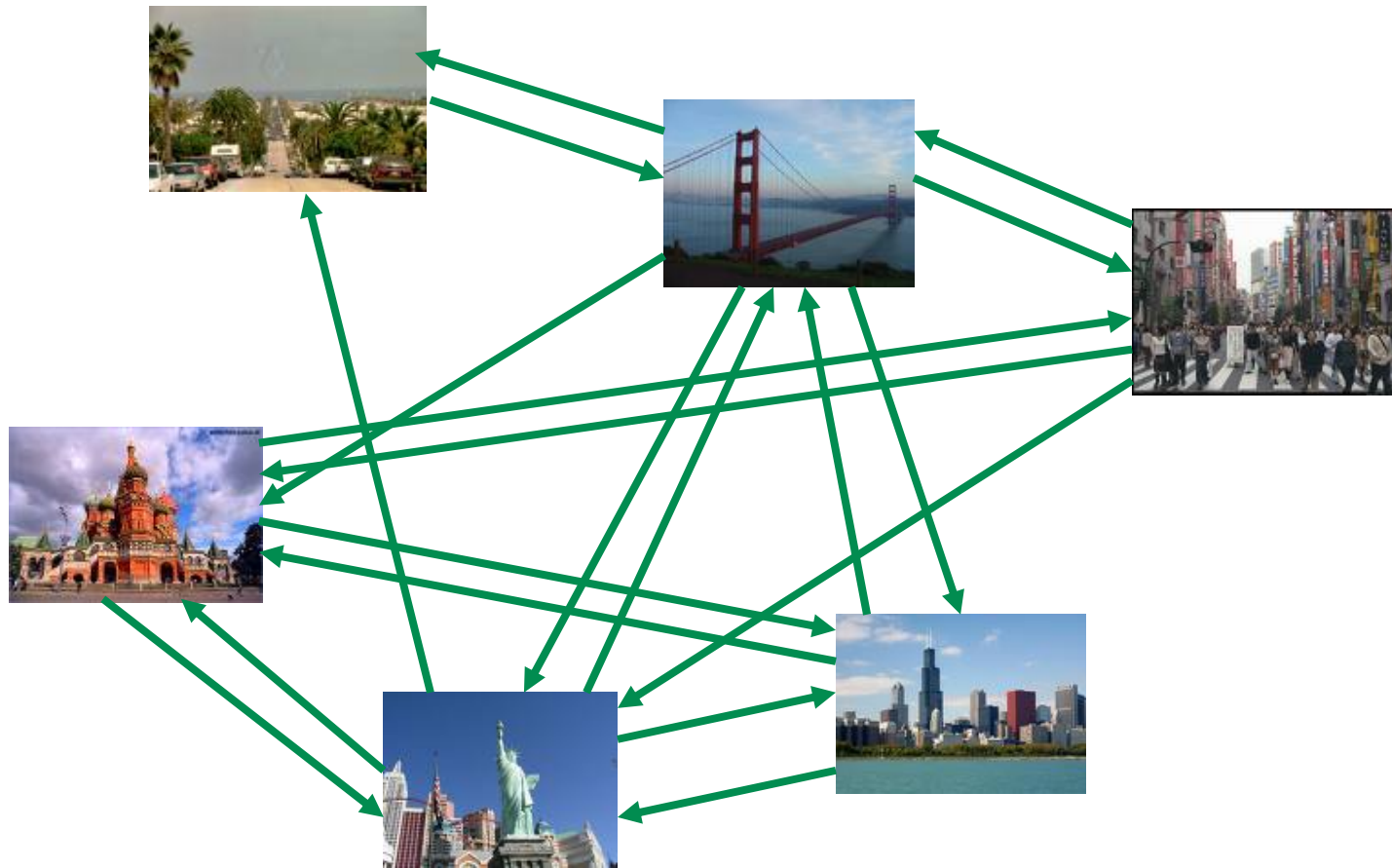
Matrizen II

- Around the world in 80 days:
- .. ist heute kein Problem mehr, eher die Vielzahl der schnellen Flugverbindungen (wenn auch nicht gerade von Clausthal aus ...). Angenommen, es gibt Direktflüge von
 - San Diego nach San Francisco
 - San Francisco überallhin
 - Chicago nach San Francisco, New York, Moskau
 - New York nach San Diego, San Francisco, Chicago, Moskau
 - Moskau nach Chicago, New York, Tokyo
 - Tokyo nach San Francisco, New York, Moskau
- Wieviele Wege mit maximal 80 Zwischenstopps gibt es?





Matrizen II





Matrizen II

- Adjazenzmatrix A_{ij} : gibt es einen Weg von i nach j ?
- Indizes 1: San Diego, 2: San Francisco, 3: Chicago, 4: New York, 5: Moskau, 6: Tokyo

0	1	0	0	0	0
1	0	1	1	1	1
0	1	0	1	1	0
1	1	1	0	1	0
0	0	1	1	0	1
0	1	0	1	1	0

- $(A^2)_{ij} = A_{i1}A_{1j} + \dots + A_{in}A_{nj}$ Anzahl der Wege mit 1 Stopover
- A^n = Anzahl der Wege mit genau $n-1$ Stopovers
- $A^1 + \dots + A^n$ = Anzahl der Wege mit maximal $n-1$ Stopovers



Matrizen II

→ *Beispiel6.m*

% Anzahl Verbindungen mit maximal 80 Zwischenstops

```
n = 80;
```

```
A = [  
0 1 0 0 0 0;  
1 0 1 1 1 1;  
0 1 0 1 1 0;  
1 1 1 0 1 0;  
0 0 1 1 0 1;  
0 1 0 1 1 0];
```

```
B = A;
```

```
An = A;
```

```
for i=2:n+1
```

```
    An = An * A;
```

```
    B = B + An;
```

```
end
```

```
An
```

```
B
```



Matrizen III

- Geheimbotschaften:

8 5 12 12 15 27 23 15 18 12 4

- 'hello world' kodiert mit der Nummer des Buchstabens im Alphabet

```
function [erg] = encrypt(text)
if (nargin<1)
    fprintf('keine Eingabe');
    erg = [];
    return
end
% Buchstaben in Zahlen umrechnen
alphabet = ['a' : 'z', ' ']
zahlen = [1 : 27]
[m n] = size(text);
erg = zeros(1,n);
for k = 1:27
    erg = erg + zahlen(k)* (text == alphabet(k));
end
```

→ *encrypt.m*



Matrizen III

- ziemlich leicht zu entschlüsseln

137 93 42 -153 239 155 75 -238 82 52 132 -56

- jeder Block wird zusätzlich mit einer Matrix M multipliziert
- $M = (3\ 1\ 4\ 5; 2\ 1\ 1\ 5; 8\ -2\ 3\ -4; -2\ -1\ -2\ -9)$

- hello world

→ 8 5 12 12 15 27 23 15 18 12 4

→ 8 5 12 12 15 27 23 15 18 12 4 0

→ $M \cdot (8\ 5\ 12\ 12)^t \ M \cdot (15\ 27\ 23\ 12)^t \ M \cdot (18\ 12\ 4\ 0)^t$

→ 137 93 42 -153 239 155 75 -238 82 52 132 -56



Matrizen III

→ *encryptmatrix.m*

```
function [erg] = encryptmatrix(text,matrix)

if (nargin<1)
    fprintf('keine Eingabe');  erg = []; return
end

if (nargin<2 || isempty(matrix) || sum(size(matrix) ~=size(matrix')) || det(matrix)==0 )
    matrix = [3 1 4 5; 2 1 1 5; 8 -2 3 -4; -2 -1 -2 -9];
    fprintf('encryption matrix is '); matrix fprintf('\n');
end

alphabet = ['a' : 'z', ' ', 'A' : 'Z', '0' : '9', '.', ',', ';', '-'];
zahlen = [1 : 67];

[m n] = size(text);
erg = zeros(1,n);

for k = 1:67
    erg = erg + zahlen(k)* (text == alphabet(k));
end
```



Matrizen III

```
[k l] = size(matrix);
```

```
if (mod(n,l) ~= 0)
```

```
    erg = [erg zeros(1,l-mod(n,k))];
```

```
    [m n] = size(erg);
```

```
end
```

```
for i=1:l:n-l+1
```

```
    erg(i:i+l-1) = matrix * erg(i:i+l-1)';
```

```
end
```



→juleverne aufrufen
→encryptmatrix(text)

Matrizen III

```

182 149 163 -214 272 198 96 -324 399 369
-149 -630 308 239 78 -361 128 78 183 -99
263 126 107 -201 64 40 59 -49 291 173
137 -266 141 53 119 -84 369 274 13 -473
123 65 7 -104 221 151 100 -249 281 206
30 -334 43 33 -9 -56 201 104 149 -151
173 148 -82 -264 122 83 84 -111 327 228
15 -335 212 123 75 -210 160 65 179 -96
141 83 133 -100 187 153 55 -230 342 217
494 -269 202 152 -49 -247 236 174 0 -300
195 113 -1 -192 408 215 64 -387 529 297
368 -469 233 163 307 -220 227 165 -13 -292
177 123 227 -158 239 185 131 -282 414 214
119 -387 424 334 -95 -581 796 566 282 -882
297 210 166 -338 529 296 382 -468 586 444
255 -655 121 93 -57 -150 374 281 -13 -488
463 253 161 -426 240 157 196 -223 169 139
-15 -226 337 241 37 -404 62 49 -36 -73
456 247 161 -420 223 188 51 -301 218 147
21 -244 148 116 -43 -181 181 130 158 -174
355 271 -95 -482 146 84 72 -138 204 158
-36 -280 61 42 13 -63 281 212 132 -334
737 517 249 -804 384 215 543 -270 203 173
97 -250 177 108 208 -142 255 210 96 -324
223 188 51 -301 230 160 51 -259 255 183
153 -278 55 31 77 -40 173 156 -101 -269
106 54 125 -75 244 182 -47 -309 259 163
109 -270 362 273 -45 -480 171 121 1 -208
246 143 92 -221 460 244 251 -417 233 168
105 -267 177 89 89 -148 143 81 23 -120

```

```

8 48 13 -69 245 182 158 -286 239 181
-93 -308 188 87 207 -118 137 88 157 -113
262 175 194 -275 167 140 71 -223 204 174
-43 -289 128 67 8 -107 173 132 39 -201
221 126 57 -157 159 130 17 -191 91 43
93 -60 86 57 147 -64 152 92 70 -128
203 140 95 -211 355 339 -251 -604 165 132
160 -194 306 204 117 -323 226 179 -35 -300
225 153 58 -253 264 171 75 -290 185 103
7 -174 219 136 95 -215 362 282 385 -383
178 127 -18 -223 138 87 196 -99 144 100
110 -128 203 173 97 -250 270 162 231 -253
188 146 47 -231 180 160 -42 -272 231 176
36 -298 259 163 109 -270 252 163 43 -282
131 80 104 -108 416 223 567 -286 134 83
104 -115 149 119 145 -156 159 95 6 -147
171 89 17 -148 139 80 20 -118 124 66
173 -87 192 168 -10 -280 259 176 176 -254
472 257 151 -432 191 109 1 -184 192 130
15 -229 138 87 196 -99 164 120 94 -164
141 98 102 -126 230 148 -11 -267 275 193
-65 -336 223 153 153 -224 429 222 156 -396
242 180 -74 -308 252 163 43 -282 135 53
47 -84 108 66 223 -75 98 65 135 -86
392 267 454 -359 190 138 -53 -239 292 202
13 -345 223 153 153 -224 354 220 503 -275
169 115 69 -164 244 177 -8 -305 158 80
124 -122 287 218 120 -340 150 89 -18 -141
147 75 131 -98 279 184 85 -311 183 124
16 -218 240 186 117 -275 223 152 17 -253
235 154 196 -232 169 104 180 -146 164 101
206 -129 192 128 512 -128

```



Matrizen III

- Decodieren
- mithilfe der Inversen von M. Die Matrix M muss dazu invertierbar sein!

$$137 \ 93 \ 42 \ -153 \ 239 \ 155 \ 75 \ -238 \ 82 \ 52 \ 132 \ -56$$
$$= M(8 \ 5 \ 12 \ 12)^t \ M(15 \ 27 \ 23 \ 15)^t \ M(18 \ 12 \ 4 \ 0)^t$$

$$\rightarrow M^{-1}*(137 \ 93 \ 42 \ -153)^t \ M^{-1}*(239 \ 155 \ 75 \ -238)^t \ M^{-1}*(82 \ 52 \ 132 \ -56)^t$$
$$= M^{-1}M(8 \ 5 \ 12 \ 12)^t \ M^{-1}M(15 \ 27 \ 23 \ 15)^t \ M^{-1}M(18 \ 12 \ 4 \ 0)^t$$

$$\rightarrow 8 \ 5 \ 12 \ 12 \ 15 \ 27 \ 23 \ 15 \ 18 \ 12 \ 4 \ 0$$

$$\rightarrow 8 \ 5 \ 12 \ 12 \ 15 \ 27 \ 23 \ 15 \ 18 \ 12 \ 4$$

$$\rightarrow \text{hello world}$$



Matrizen III

→ *decryptmatrix.m*

```
function [erg] = decryptmatrix(text,matrix)
```

```
    erg=[];  
    if (nargin<1)    fprintf('keine Eingabe');    return  
    end
```

```
    if (nargin<2 || isempty(matrix) || sum(size(matrix) ~= size(matrix')) || det(matrix)==0 )  
        matrix = [3 1 4 5; 2 1 1 5; 8 -2 3 -4; -2 -1 -2 -9]^-1;  
        fprintf('decryption matrix is '); matrix  
    end
```

```
    [m n] = size(text);  
    [k l] = size(matrix);  
    if mod(n,l)~=0  
        fprintf('Keine Codierung mit der Matrix möglich, falsche Länge\n');    return  
    end
```

```
    for i=1:l:n-l+1  
        text(i:i+l-1) = matrix * text(i:i+l-1)';  
    end
```




Matrizen III

```
alphabet = ['a' : 'z', ' ', 'A' : 'Z', '0' : '9', '.', ',', ';', '-'];  
[m k] = size(alphabet);  
zahlen = [1 : k];
```

```
ende = n;  
epsilon=exp(-10);  
while (ende>0 && abs(text(ende))<epsilon) %Achtung: numerische Gleichheit  
    text = text(1:ende-1);  
    ende = ende-1;  
end
```

```
erg = 'a'; %Matlab zwingen, Buchstaben auszugeben
```

```
for i=1:ende  
    for j=1:k  
        if (abs(text(i)-zahlen(j))<epsilon)  
            erg(i)=alphabet(j);  
        end  
    end  
end  
end
```



Matrizen III



Chapter one.

In which Phileas Fogg and Passepartout accept each other, the one as master, the other as man.

Mr. Phileas Fogg lived, in 1872, at No. 7, Saville Row, Burlington Gardens, the house in which Sheridan died in 1814. He was one of the most noticeable members of the Reform Club, though he seemed always to avoid attracting attention; an enigmatical personage, about whom little was known, except that he was a polished man of the world. People said that he resembled Byron - at least that his head was Byronic; but he was a bearded, tranquil Byron, who might live on a thousand years without growing old.