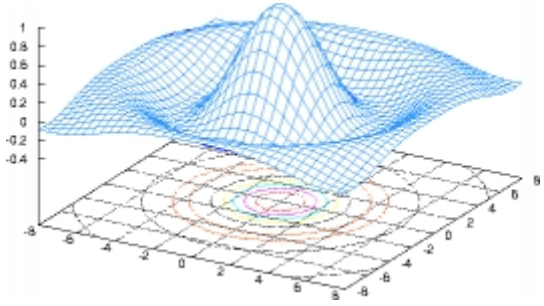
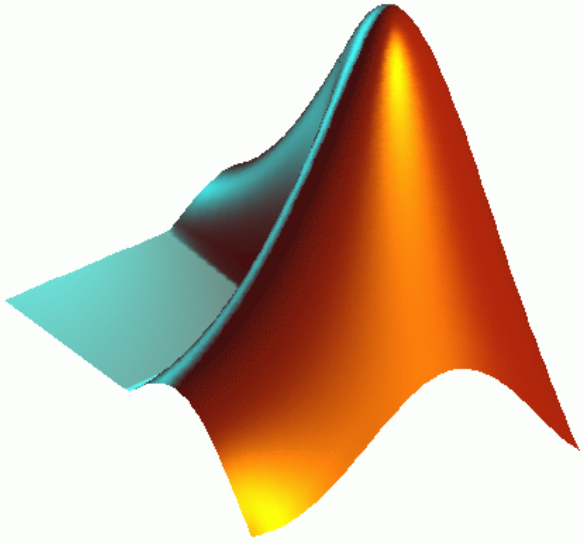
TU Clausthal

Matlab – Eine Einführung (contd.)

Barbara Hammer, Alexander Hasenfuß,
TU Clausthal





Matrizen IV

- ... and they lived long and happily ever after.
- In a kingdom far far away, the King decided that the time has come to find a husband for his princess daughter. The King wanted to find a worthy lad for his princess, so he promised to give his daughter away to the first young (or old) man who would solve the puzzle that has stumped the best of his court mathematicians for years. The puzzle is very simple: in a palace, there are 25 rooms arranged in a square -5 rows of rooms with 5 rooms in each row. In every room there is a light switch which not only switches on/off the light in that room, but also switches the lights in the adjacent rooms - the room to the right, to the left, the room above and the room below. Initially, all of the lights are turned off. The goal is to turn the lights on in every room of the palace.

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

Matrizen IV

- Zustand pro Zelle in {0,1}
- Anfangszustand = s0 = $[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]^t$
- Endzustand = s1 = $[1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1]^t$
- binäre Addition: 0 + 0 = 1 + 1 = 0, 0 + 1 = 1 + 0 = 1
  - kommutativ
  - assoziativ
- Schalter in Raum i betätigen entspricht der Addition eines Vektors:
  - R1: (1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
  - R2: (1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
  - R3: (0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
  - R4: (0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
  - ...

Matrizen IV

Kommutativität, Assoziativität → die Reihenfolge der Additionen ist egal!

Wir suchen also Anzahlen x1, ..., xn der Betätigung der einzelnen Schalter, so dass

$$x1*R1 + x2*R2 + \ldots + x25*R25 + s0 = s1$$

bzw.

$$x1*R1 + x2*R2 + \ldots + x25*R25 = s1$$

bzw.

$$R^tX + s1 = 0$$

bzw.

$$(R,s1)^t(X,1) = 0$$

mit der Operation '+' wie oben definiert.

Matrizen IV

‚normale' lineare Gleichungssysteme lösen:

$(R,s1)^t(X,1) = 0$

→ Gauss Verfahren macht daraus eine einfache Matrix, wo man das Ergebnis ablesen kann: $(Rred,s1red)^t(X,1) = 0$

etwa:

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{pmatrix} \rightarrow \begin{pmatrix} 1 & 0 & -1 & -2 \\ 0 & 1 & 2 & 3 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

d.h.: x1 = x3 +2, x2 = 2 x3 – 3, x3 kann frei gewählt werden

Matlab: rref(Matrix)

→ *Beispiel7.m*

Matrizen IV

lineare Gleichungssysteme modulo 2 lösen?

Mathematik sagt: dasselbe modulo 2 rechnen

→ rref aus
(C:\Programme\MATLAB\R2007a)\toolbox\matlab\matfunc\rref.m
kopieren und abändern!

Matrizen IV

→*rrefmod2.m*

```
function A = rrefmod2(A)

A = mod (A, 2);
 [m,n] = size(A);

i = 1; j = 1;
while (i <= m) & (j <= n)
  % Find value and index of largest element in the remainder of column j.
  [p,k] = max(abs(A(i:m,j))); k = k+i-1;
  % Swap i-th and k-th rows.
  A([i k],j:n) = A([k i],j:n);
  % Subtract multiples of the pivot row from all the other rows.
  for k = [1:i-1 i+1:m]
    A(k,j:n) = mod(A(k,j:n) - A(k,j)*A(i,j:n), 2);
  end
  i = i + 1;
  j = j + 1;
end
```

Matrizen IV

- Eingabe von R:

R ist

A	I	0	0	0
I	A	I	0	0
0	I	A	I	0
0	0	I	A	I
0	0	0	I	A

mit A =

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

I = Identität, 0 = Nullmatrix

Matrizen IV

→ *Beispiel8.m*

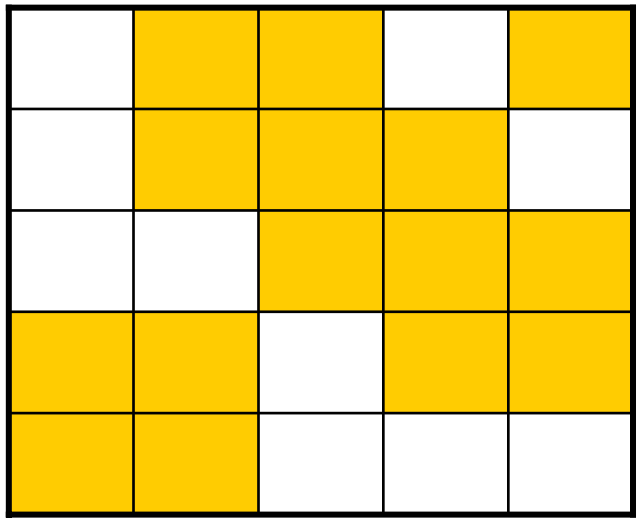
```
A = [1 1 0 0 0; 1 1 1 0 0; 0 1 1 1 0; 0 0 1 1 1; 0 0 0 1 1];
N = zeros(5,5);
I = diag(ones(1,5));
R = [A I N N N; I A I N N; N I A I N; N N I A I; N N N I A];
sf = ones(25,1);

Rext = [R sf]

rrefmod2(Rext)
```

Matrizen IV

- Lösung:

  Zwei freie Parameter (die letzten zwei Räume), für die anderen erhält man die Abhängigkeit als L*(x24,x25,1) mit Matrix L

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

→ 

etwa x24=x25=0, dann muß man da das Licht anmachen!

Grafik I

→ *Beispiel9.m*

- Es werde Licht.
- Die Illumination des Palastes durch den glücklichen Prinzen soll visualisiert werden.
- Darstellung von Matrizen/Bitmaps:
  - image(matrix)
  - Farben: colormap(..), vordefiniert etwa Jet (blau→rot), Gray (schwarz→weiss) , Lines (gemischt)
  - Achsen, Skalierung, Beschriftungen etc. können gesetzt werden

Grafik I

→ *Beispiel10.m*

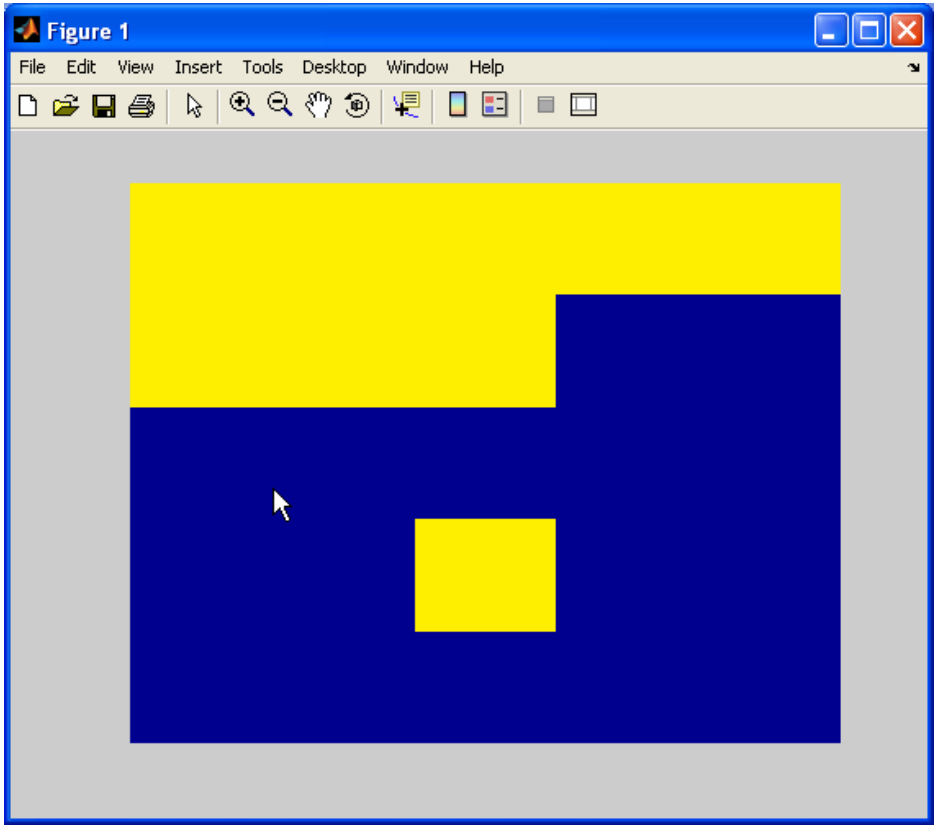
- Objekte erscheinen in Figures
  - figure, figure(h), h=figure erzeugt ein Fenster für die Grafik mit Referenz h
  - falls h schon existiert, wird durch diesen Befehl die Grafik die aktuelle
- Jedes Grafik-Objekt erscheint in der aktuellen Figure, es hat ebenfalls eine eindeutige Referenz
  - ein neues Objekt überschreibt ein altes per default
  - umschalten:
    - hold on → zufügen
    - hold off → überschreiben
  - durch die Referenzen kann man Objekteigenschaften erfragen und setzen (get, set)

Grafik I

→ *Beispiel11.m*

```
...
reduced = rrefmod2(Rext); solution = reduced(:,26);
im=zeros(5,5); state=zeros(25,1);
h = figure
image(40*im+1)
colormap(jet)
axis off
axis image
pause
 % Bilder nacheinander darstellen
for i=1:25
   if solution(i)==1
      state = state+R(:,i); state = mod(state,2);
      im = reshape(state,5,5)';
      image(40*im+1)
      colormap(jet)
      axis off
      axis image
      pause
   end
end
close(h)
```

Grafik I



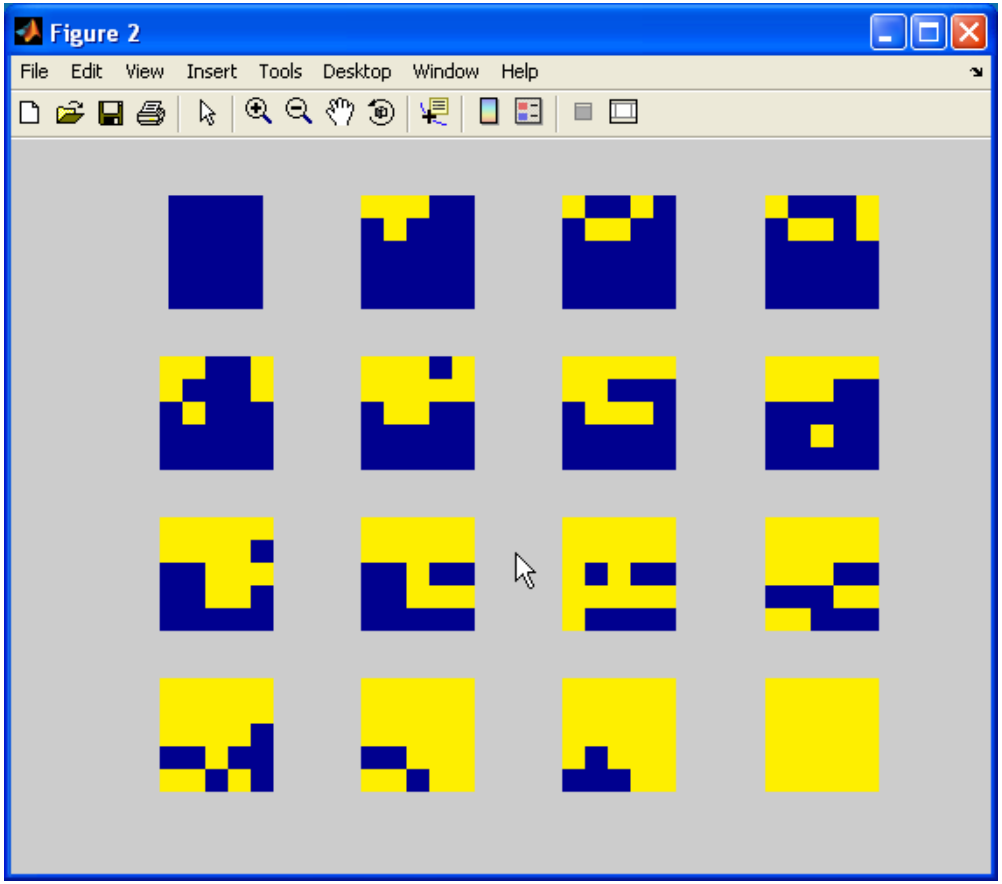
Grafik I

→ *Beispiel12.m*

- alles in eine Grafik → subplot(m,n,i) = die ite Grafik in insgesamt nxm Bereichen ist relevant, je Bereich gibt es eine eigene Referenz

```
...
num=sum(solution);
siz=ceil(sqrt(num+1));
subplot(siz,siz,1)
...
nummer=2;
for i=1:25
   if solution(i)==1
      ...
      subplot(siz,siz,nummer)
      ...
  end
end
...
```

Grafik I



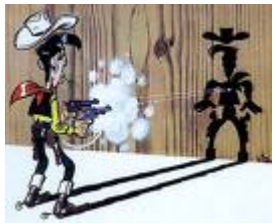
Grafik II

- ... 'schneller' Visualisieren
- function [n m] = Testmatrix(i) berechnet die Zeit zur Quadrierung einer ixi Matrix mit Einträgen eins direkt durch Matlab (n) bzw. durch explizite Programmierung in Schleifen (m)

  → *testmatrix.m*

- Vergleich der Werte für einige n:

n	intern	iterativ
0	0.0000	0.0000
10.0000	0.0001	0.0004
20.0000	0.0001	0.0026
30.0000	0.0002	0.0075
40.0000	0.0004	0.0168
50.0000	0.0006	0.0364
60.0000	0.0011	0.0968
70.0000	0.0018	0.0845
80.0000	0.0016	0.0745
90.0000	0.0013	0.0618
100.0000	0.0017	0.1135
110.0000	0.0024	0.1415
120.0000	0.0029	0.1729
130.0000	0.0036	0.2140
140.0000	0.0046	0.2847
150.0000	0.0055	0.3353
160.0000	0.0068	0.4233
170.0000	0.0080	0.4915
180.0000	0.0095	0.5949
190.0000	0.0112	0.6764
200.0000	0.0132	0.8306
210.0000	0.0557	0.9729
220.0000	0.0179	1.0713
230.0000	0.0204	1.1921
240.0000	0.0230	1.3893
250.0000	0.0483	1.5574



Grafik II

→ *Beispiel13.m*

```
n = 251;
erg = zeros((n-1)/10,3);

for i=1:10:n
   number = (i-1)/10+1;
   erg(number,1)= i-1;
   [erg(number,2),erg(number,3)]=testmatrix(i);
end

erg
```

Grafik II

→ *Beispiel13.m*

```
plot(erg(:,1),erg(:,2),'-o',erg(:,1),erg(:,3),'-x')
xlabel('Matrix dimension')
ylabel('Time fore computing the square')
% text(n,erg(number,2),'Matrix')
% text(n,erg(number,3),'Iterative')
h=legend('Matrix','Iterative')
set(h,'Location','Northwest')
title('Demonstration of the efficiency of implicit matrix operations')

print -deps2 'wow.eps'
```

Grafik II

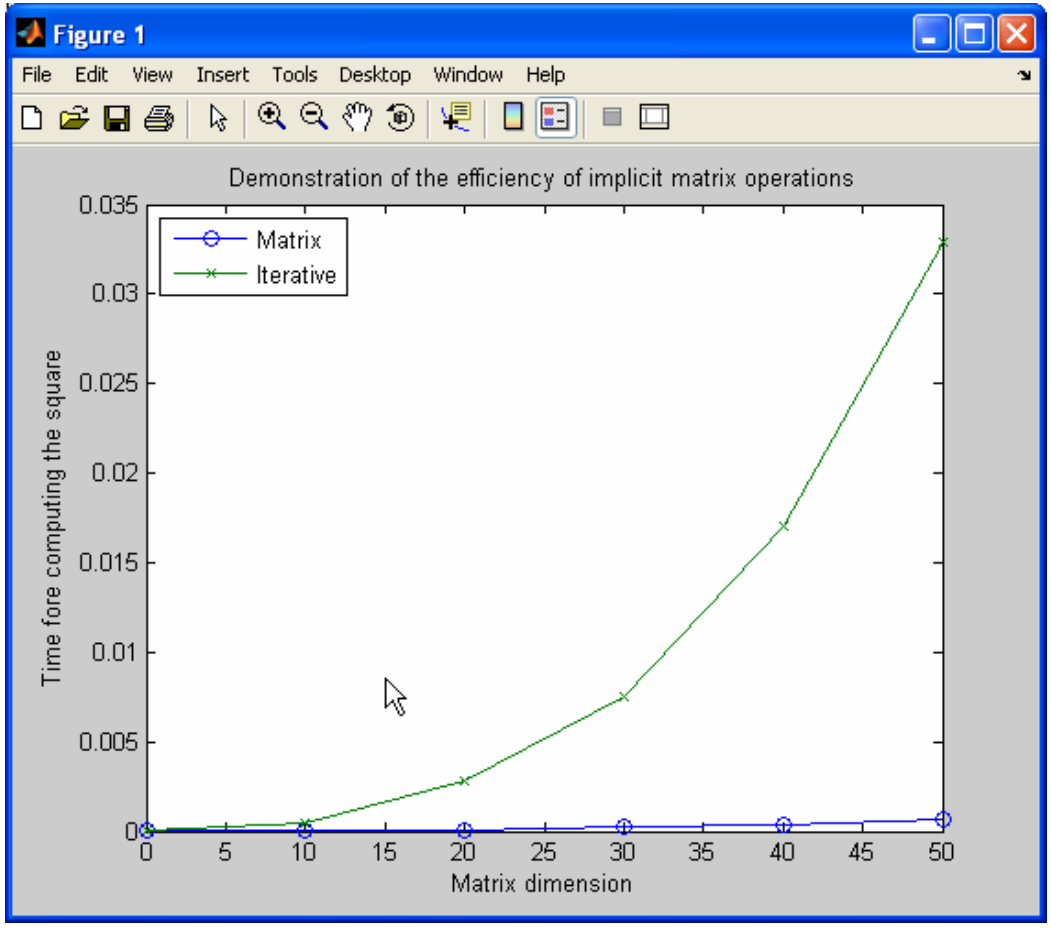


Figure 1
File Edit View Insert Tools Desktop Window Help
Demonstration of the efficiency of implicit matrix operations
Matrix
Iterative
Time fore computing the square
Matrix dimension
0.035
0.03
0.025
0.02
0.015
0.01
0.005
0
0 5 10 15 20 25 30 35 40 45 50

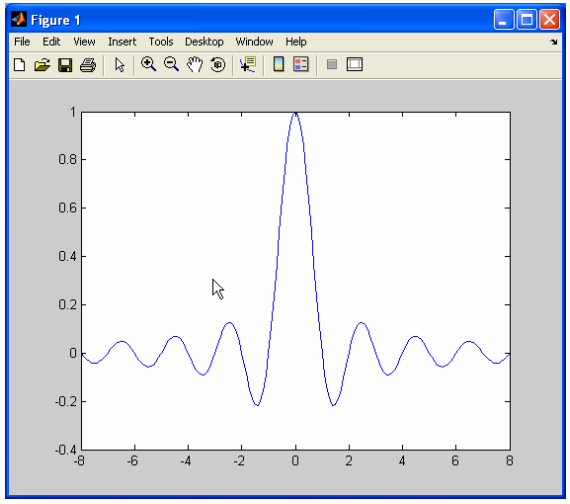
Grafik II

- Funktionen plotten:
  - plot(x-Werte,y-Werte)
  - mehrere Funktionen:
    - weitere Paare von Vektoren (Linienfarbe wird durchpermutiert)
    - hold on (Linienfarbe wird nicht durchpermutiert)
    - hold all (Liniefarbe wird durchpermutiert)
- Stil:
  - Linienstil und –farbe: z.B. 'r:+' → rot, gepunktet, Punkte als Kreuz
  - weitere Eigenschaften und Werte: z.B. 'LineWidth',2 → dickere Striche
- Format und Annotation:
  - xlabel(..), ylabel(..), title(..)
  - text(wohinx,wohiny, 'text')
  - legend('text','text',...)
- Drucken: print –dwohin name z.B. print –deps2 'tollegrafik.eps'

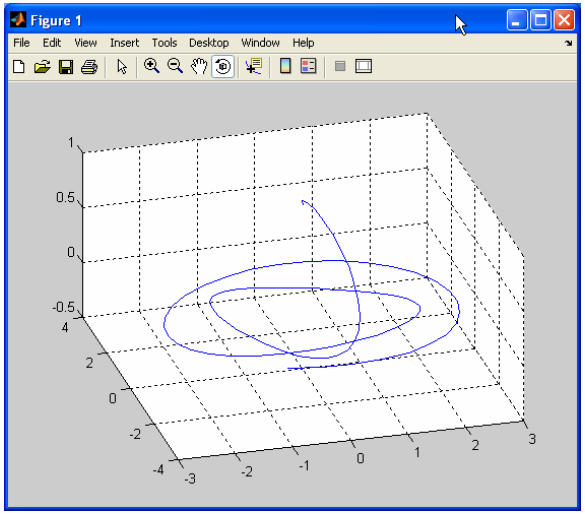
→ *Beispiel14.m*

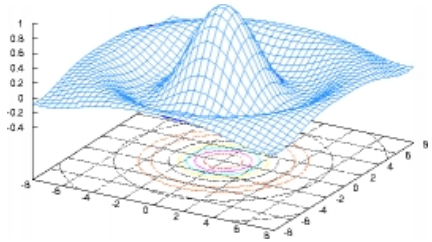
Grafik III

- x= [-8:0.1:8]; y=sinc(t); plot(x,y) →



- d=[0:0.1:8]; phi=[0:0.2:16];
  x=sqrt(d).*sin(phi); y=sqrt(d).*cos(phi);
  z=sinc(d); plot3(x,y,z) grid →





Grafik III

- Die dritte Dimension
- Plots von 3-D Funktionen:
  - plot3(xvektor,yvektor,zvektor,..) → Plot der Punktesequenz in 3D
  - contour(xmatrix,ymatrix,zmatrix) → 2D Contourplot
  - contour3(xmatrix,ymatrix,zmatrix) → Contourplot mit Höhen
  - mesh(xmatrix, ymatrix, zmatrix) → Drahtmodell der Punktfläche, die Koordinaten definieren die Schnittpunkte
  - meshc → Drahtmodell und Contourplot
  - surf(xmatrix,ymatrix,zmatrix) → Oberflächenplot der Punkte
  - surfc → Oberflächenplot und Contourplot
- Erzeugen von Gittern:
  - meshgrid(xvector,yvector) → Matrizen entsprechend der Rasterung des Raums

Grafik III

- entspricht der Funktion $(x,y,\mathrm{sinc}(x^2+y^2))$
- Rasterung des x/y-Raums:

  [u v] = meshgrid([1:5],[3:5]) →

1	2	3	4	5
1	2	3	4	5
1	2	3	4	5

3	3	3	3	3
4	4	4	4	4
5	5	5	5	5

- $z = \mathrm{sinc}(u.^2.+v.^2)$ → Matrix mit Einträgen $\mathrm{sinc}(u(i,j)^2+v(i,j)^2)$

1.0e-015 *

-0.0390	-0.0480	-0.0063	-0.0390	0.0275
-0.0275	-0.0390	-0.0063	-0.0390	0.1217
0.0480	-0.0000	0.0275	0.1217	0.0063

- darzustellen: Punktetupel (u,v,z)

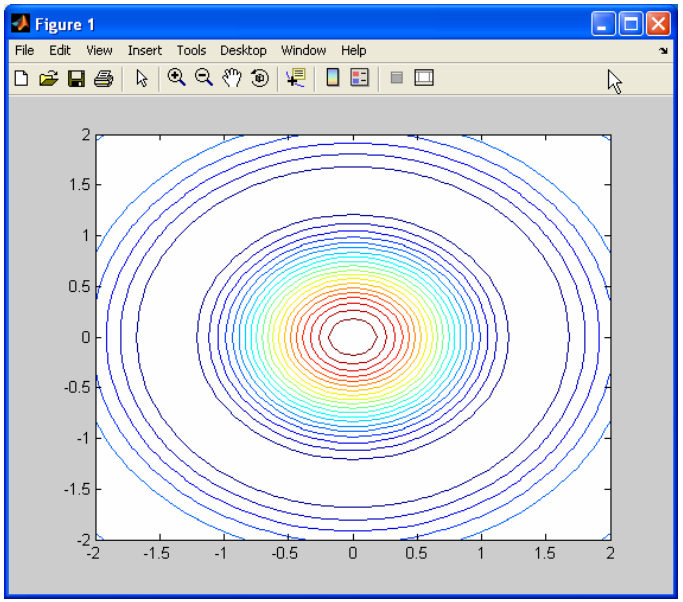
Grafik III

→ *Beispiel15.m*

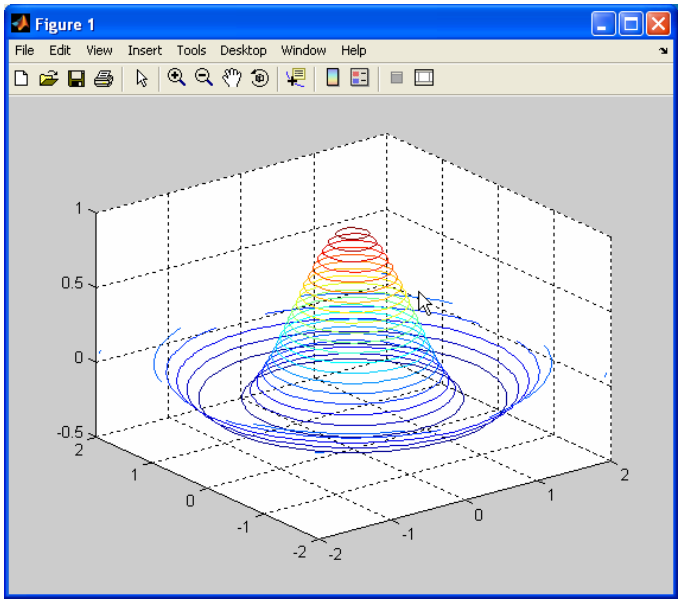
```
m     = 2;                    % Bereich zum Plotten
x     = -m:m/20:m;             % Raster der x- und y-Werte erzeugen
y     = x;
[X, Y] = meshgrid (x, y);
R     = sqrt (X.^2 + Y.^2);       % Spaltfunktion des Radius
Z     = sinc (R);
```

Grafik III

- contour(X,Y,Z,20) →

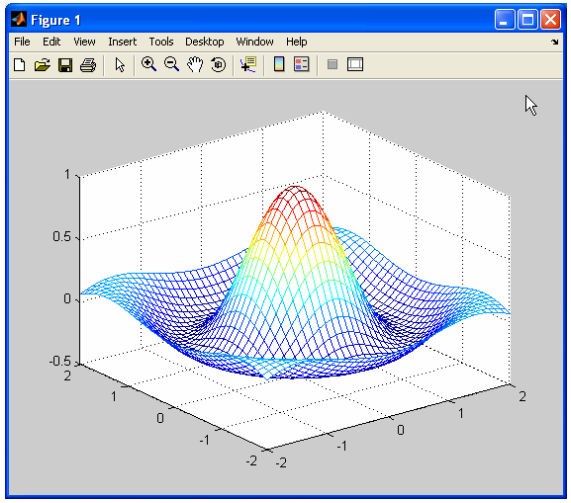


- contour3(X,Y,Z,20) →

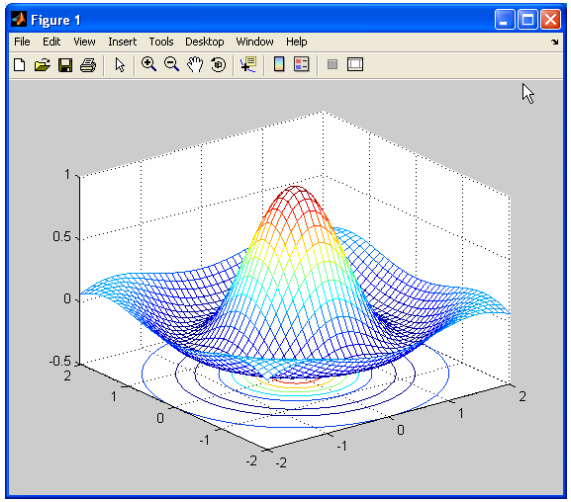


Grafik III

- mesh(X,Y,Z) →

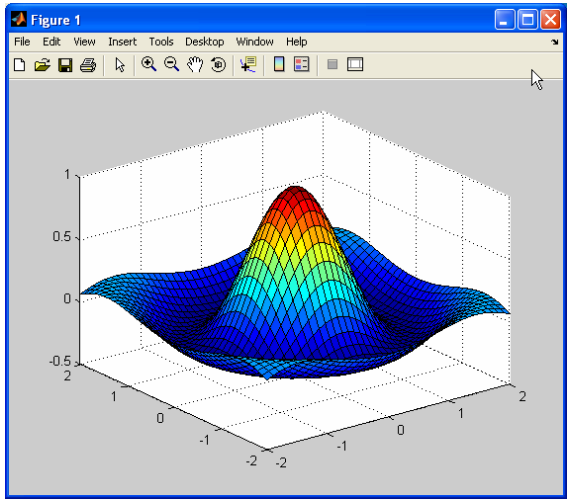


- meshc(X,Y,Z) →

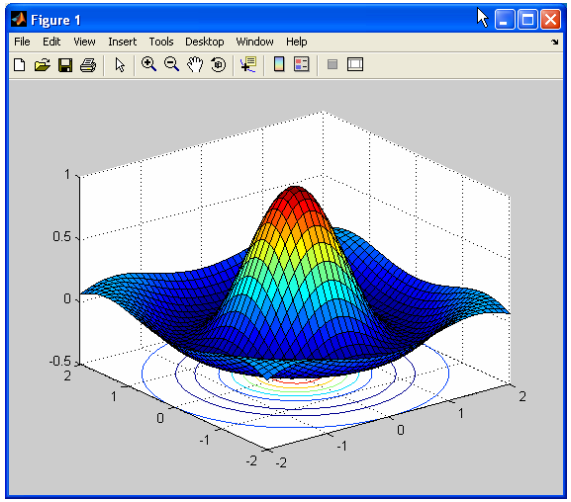


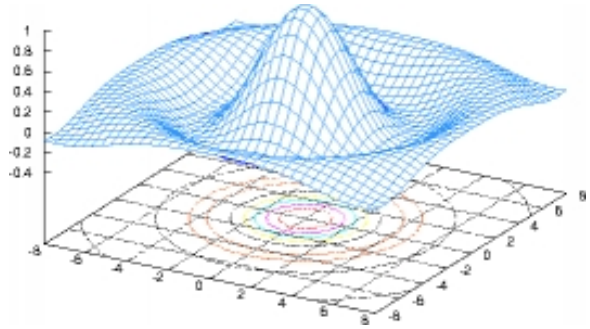
Grafik III

- surf(X,Y,Z) →



- surfc(X,Y,Z) →



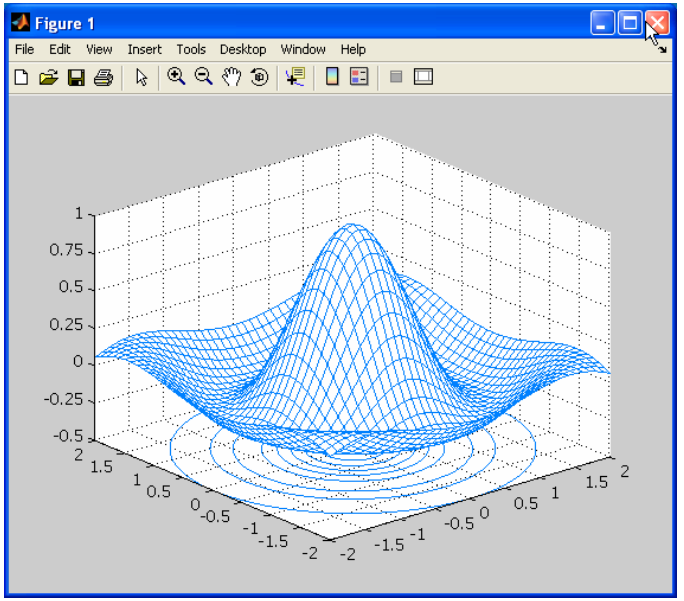


Grafik III

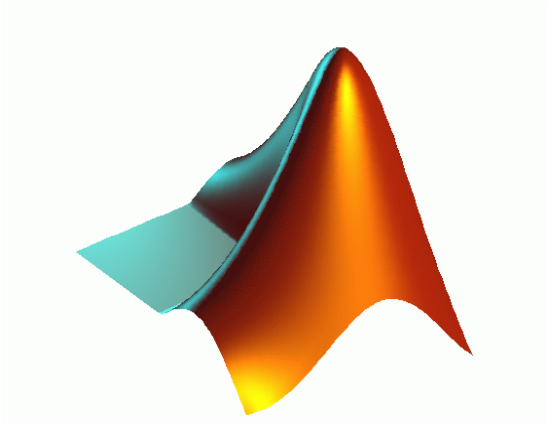
- Eigenschaften ...

```
m     = 2;                  % Bereich zum Plotten
x     = -m:m/20:m;              % Raster der x- und y-Werte erzeugen
y     = x;
[X, Y] = meshgrid (x, y);
R     = sqrt (X.^2 + Y.^2);
Z     = sinc (R);

h  = figure (1);
h1 = meshc(X, Y, Z);                % Plotten

set(gca,'XTick',[-2:0.5:2])   % Achseneinteilung
set(gca,'YTick',[-2:0.5:2])
set(gca,'ZTick',[-2:0.25:2])
set(gca,'Fontname','Tahoma') %Schriftart

set(h1,'EdgeColor',[0.0,0.5,1.0])
view(-40,30)
```

→ *octavestyle.m*



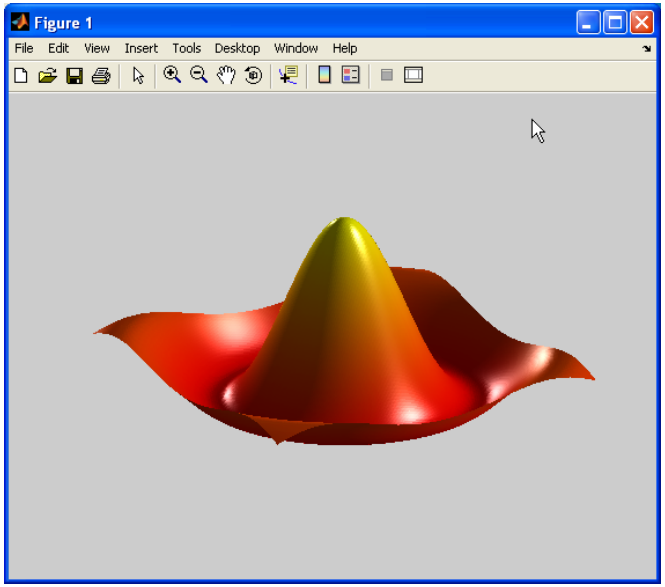
Grafik III



- Eigenschaften ...

```
m     = 2;                  % Bereich zum Plotten
x     = -m:m/100:m;              % Raster der x- und y-Werte erzeugen
y     = x;
[X, Y] = meshgrid (x, y);
R     = sqrt (X.^2 + Y.^2);       % Spaltfunktion des Radius
Z     = sinc (R);
figure (1);
surf(X, Y, Z);                % Plotten
axis off;
view (-30, 30);
colormap ('autumn');              % Licht und Farbe
shading interp;
material shiny;
light ('position', [2.5 -2.5 0.5]);  % 1. Lichtquelle
light ('position', [0 1.4 0]);      % 2. Lichtquelle
```

→ *matlabstyle.m*

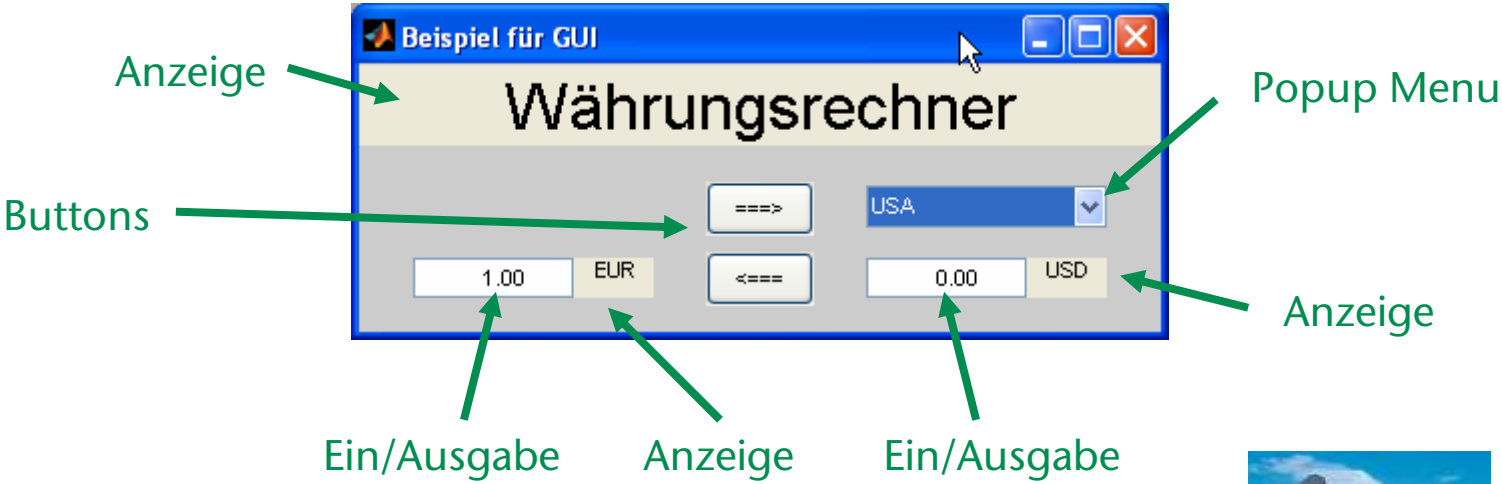


Add on ...

... das folgende wird nur erzählt, wenn noch Zeit ist (eher nicht), und ist nicht prüfungsrelevant (aber relevant, wenn Sie eine Oberfläche mit Matlab konfigurieren wollen)

Grafik IV

- Mach mal Urlaub!
- ... etwa in den USA, China, Australien, Norwegen. Matlab rechnet die Währung um.



Grafik IV

- Grafical user interfaces (guis) sind Bestandteil von figures
- zentrale vordefinierte Funktion für Bestandteile des guis: uicontrol
- Parameter von uicontrol:
  - Parent → Referenz der zugehörigen figure
  - typische Eigenschaften wie etwa Fontsize, Units, ...
  - Position → [linke untere Eckex/y, Breite, Höhe]
  - String → zu erscheinender Text (man kann auch Bilder einbetten)
  - Style → Typ des zu definierenden Objekts
- vordefinierte Typen sind etwa
  - text
  - pushbutton
  - popupmenu
  - edit
  - ...

Grafik IV

- Programmieren von Aktionen durch die Assoziation des Parameters 'Callback' mit {@<funktionsname>}
- <funktionsname> ist eine in der globalen Funktion definierte Unterfunktion (die Zugriff auf alle Variblen der globalen Funktion hat)
- Übergabeparameter sind
  - die Referenz des Objekts, für die der Callback aufgerufen wurde
  - eventdata ... für spätere Matlab-Versionen ☺
  - Struktur mit allen weiteren Referenzen der figure

Grafik IV

→ *guibeispiel.m*

- Bsp:

```
uicontrol('Parent',h0,...                                   % Referenz figure
          'Units','points',...                              % Maßeinheit
          'Position',[190 40 90 15], ...                    % Größe Fenster
          'String',['first choice'; 'second choice'],...    % Auswahlmöglichkeiten
          'Callback', {@popupmenu_callback}, ...            % Aktion
          'Style','popupmenu', ...                          % Typ
          'Value',1);                                       % Startwert

function popupmenu_Callback(source, eventdata, handles)
val = get(source,'Value');
switch val
case 1   % do something
case 2   % do something
end
end
```

Grafik IV

- Relevante Eigenschaften der Typen...
- text: keine Aktion
- pushbutton: wird aufgerufen, falls gedrückt
- popupmenu: Value in {1,2,...},
  - String ist hier eine Matrix und definiert je Zeile eine Auswahlmöglichkeit
- edit: String,
  - umwandeln in eine Zahl: str2double
  - Test, ob gültige Zahl: isnan

Grafik IV

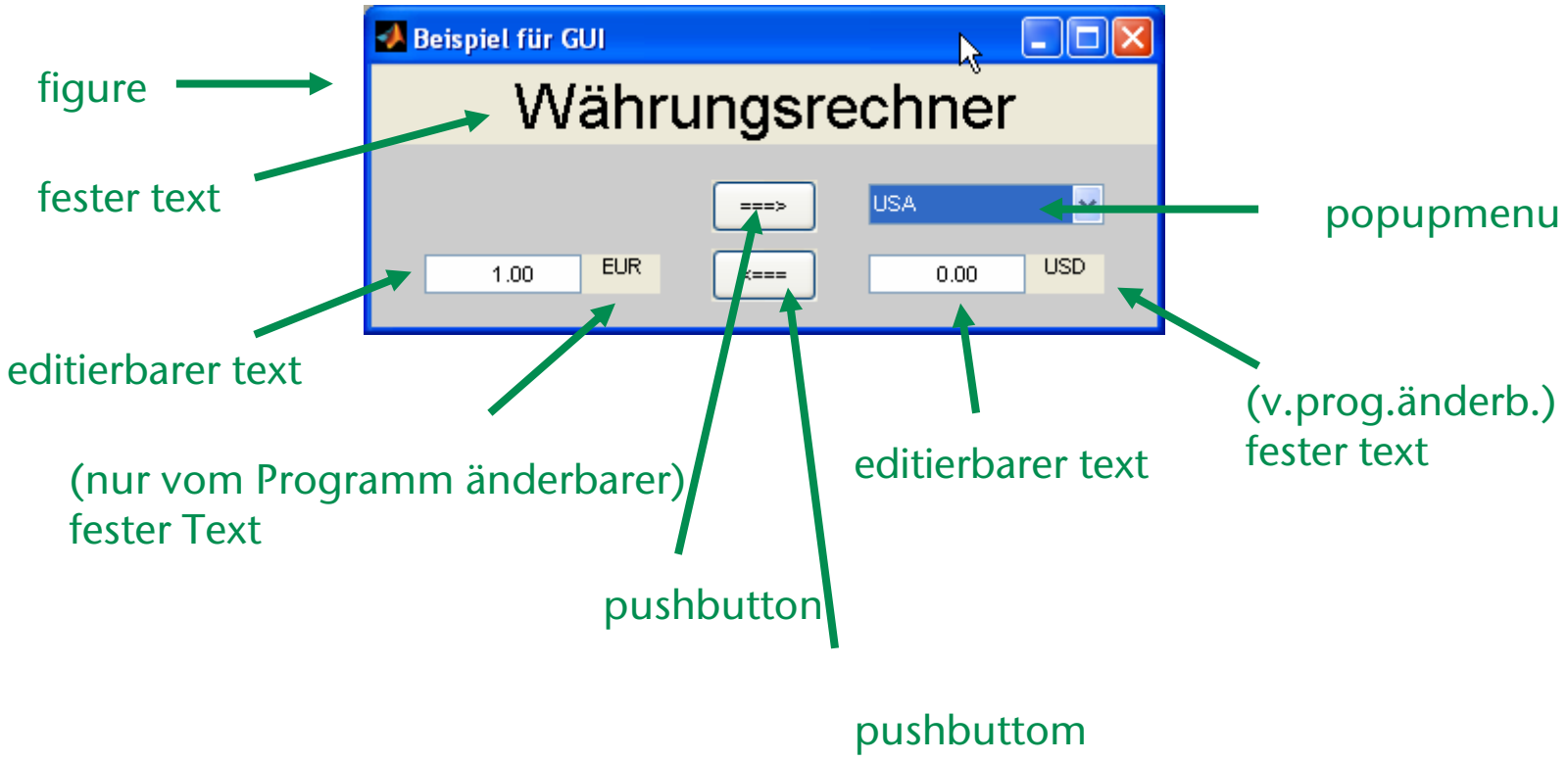


figure
fester text
editierbarer text
(nur vom Programm änderbarer)
fester Text
pushbutton
pushbuttom
editierbarer text
(v.prog.änderb.)
fester text
popupmenu
Beispiel für GUI
Währungsrechner
===>
<===
USA
1.00
EUR
0.00
USD

Grafik IV

figure

fester text

→ *trygui.m*

Grafik IV

```
function trygui

h0 = figure('Units','points', ...              % Fenster-Einstellungen
   'Position',[100 100 300 100], ...
   'NumberTitle','off', ...                 % Kein "Figure" in Titelzeile
   'Name','Beispiel für GUI', ...             % Titelzeile des Fensters
   'MenuBar','none');                        % Kein Menü

hueberschrift = uicontrol('Parent',h0, ...       % Textfeld
   'Units','points', ...
   'Position',[0 70 300 30], ...
   'FontSize',24, ...
   'String','Währungsrechner', ...
   'Style','text');
```

Grafik IV

popupmenu

Grafik IV

```
% Länderwahl
Faktor   = 1.2967;                    % Vorbelegung
Waehrung = 'USD';                      % Vorbelegung

hlaenderwahl = uicontrol('Parent',h0, ...       % Popup-Menü
   'Units','points', ...
   'Position',[190 40 90 15], ...
   'BackgroundColor',[1 1 1], ...
   'String',['USA      ';'China     ';'Australien';'Norwegen  '], ...
   'Callback', {@popupmenu_callback}, ...
   'Style','popupmenu', ...
   'Value',1);                         % Vorbelegung: USA

function popupmenu_callback(source,eventdata,handles)
      Land = get (source, 'Value');
      switch Land case 1, Faktor = 1.2967; Waehrung = 'USD';
                  case 2, Faktor = 10.057; Waehrung = 'CNY';
                  case 3, Faktor = 1.6739; Waehrung = 'AUD';
                  case 4, Faktor = 8.1237; Waehrung = 'NOK';
      end;
      set (hwaehrungland, 'String', Waehrung);
end
```

Grafik IV

editierbarer text

(nur vom Programm änderbarer)
fester Text

Grafik IV

```
% Wert in Euro

hwerteuro = uicontrol('Parent',h0, ...          % Eingabefeld
   'Units','points', ...
   'Position',[20 13 60 15], ...
   'BackgroundColor',[1 1 1], ...
   'String','    1', ...
   'Style','edit');

hwaehrungeuro = uicontrol('Parent',h0, ...      % Textfeld
   'Units','points', ...
   'Position',[80 13 30 15], ...
   'String','EUR', ...
   'Style','text');
```

Grafik IV

editierbarer text

(v.prog.änderb.)
fester text

Grafik IV

```
% Wert in Landeswährung

hwertland = uicontrol('Parent',h0, ...          % Eingabefeld
   'Units','points', ...
   'Position',[190 13 60 15], ...
   'BackgroundColor',[1 1 1], ...
   'String','    0.00', ...                    % Vorbelegung
   'Style','edit');

hwaehrungland = uicontrol('Parent',h0, ...      % Textfeld (Ausgabe)
   'Units','points', ...
   'Position',[250 13 30 15], ...
   'String','USD', ...                          % Vorbelegung
   'Style','text');
```

Grafik IV

pushbutton

Grafik IV

```
% Umrechnung

uicontrol('Parent',h0, ...
   'Units','points', ...
   'Position',[130 36 40 20], ...
   'Callback', {@pushbutton1_callback}, ...
   'Style','pushbutton', ...
   'String','===>');

function pushbutton1_callback(source,eventdata,handles)
   Wert_Euro = str2double(get(hwerteuro,'String')); ...
   Wert_Land = Wert_Euro * Faktor; ...
   set (hwertland, 'String', Wert_Land);, ...
   set (hwaehrungland, 'String', Waehrung);
end
```

Grafik IV

pushbuttom

Grafik IV

```
uicontrol('Parent',h0, ...
   'Units','points', ...
   'Position',[130 10 40 20], ...
   'Callback', {@pushbutton2_callback}, ...
   'Style','pushbutton', ...
   'String','<===');

function pushbutton2_callback(source,eventdata,handles)
   Wert_Land = str2double(get(hwertland,'String'));, ...
   Wert_Euro = Wert_Land / Faktor; ...
   set (hwerteuro,'String', Wert_Euro); ...
   set (hwaehrungland, 'String', Waehrung);
end

end
```

- Grafisches Tool zur Unterstützung der Programmierung von Oberflächen in Matlab: GUIDE ...

Resume

- ... auf in den Urlaub!!!