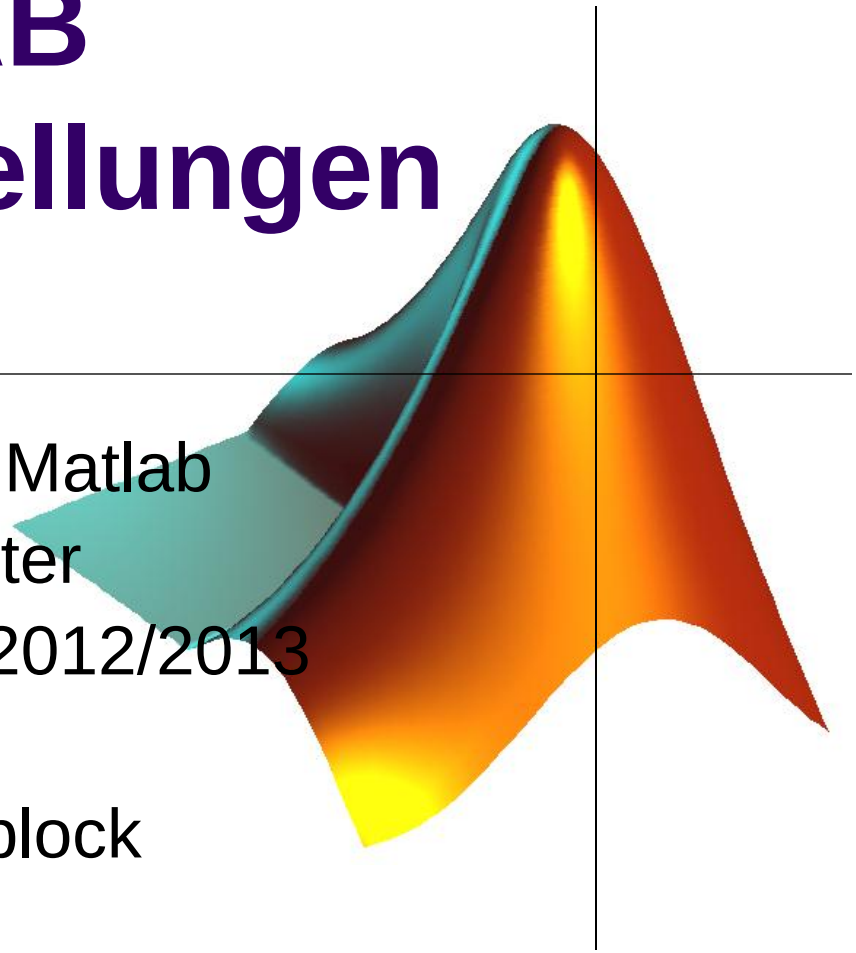
MATLAB
Flächendarstellungen



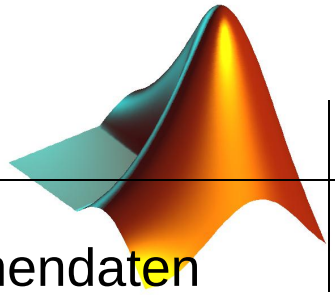
Einführung in Matlab

3. Semester

Wintersemester 2012/2013

3. Themenblock

Allgemeines

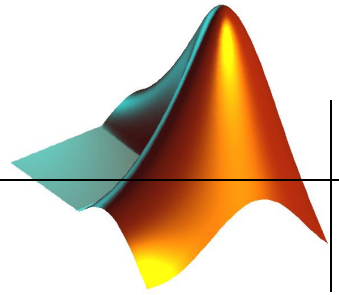


- Flächendarstellungen: graphische Darstellung von Flächendaten

- Flächendaten: Jeder Punkt eines 2D-Koordinatenrasters (*x, y*) besitzt ein Attribut in Abhängigkeit von *x* und *y*

- Funktion: $z = f(x, y)$ oder $(u, v) = f(x, y)$
- Attribute:
  - 1D-Attribute:
    - Höhe: u.a. für Gelände- oder Oberflächenmodelle
    - Temperatur: Kartierung von Wärmequellen
    - Grauwert: digitale Bilder
  - 2D-Attribute:
    - Geschwindigkeitsvektor: z.B. für Deformationsdarstellung
    - Steigungen: z.B. für Abflussmodelle in einem DGM

1D: Höhendarstellung

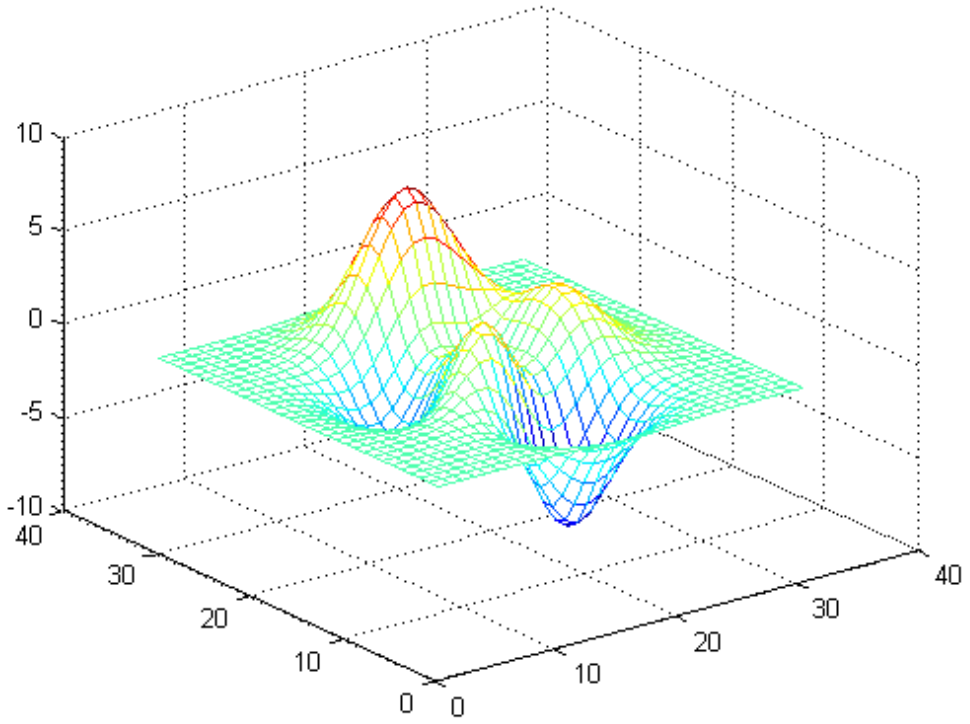


3D-Drahtgitter:

- Befehl mesh
  - Verbindet je zwei benachbarte Gitterpunkte
  - Farbkodierung in Abhängigkeit vom Wert

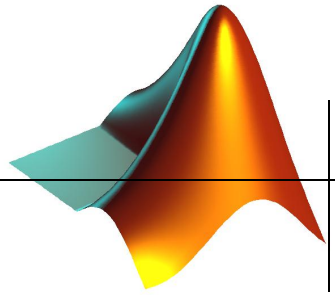
```
Z = peaks(31);

mesh(Z);
```



NB: peaks(31) erzeugt Matrix *Z* der Größe 31x31 mit Beispielwerten

1D: Höhendarstellung

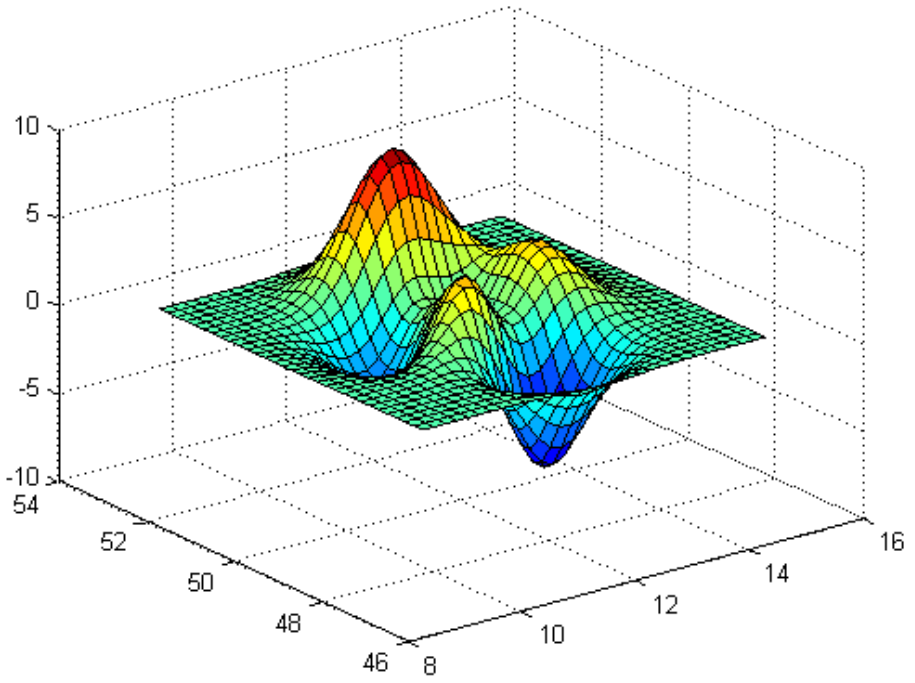


3D-Fläche:

- Befehl surf
  - Fläche innerhalb von 4 Punkten erhält farbliche Kodierung

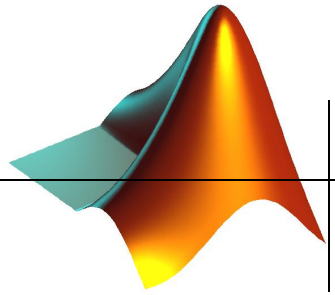
```
surf(Z);
```

```
x = 9:0.2:15;
y = 47:0.2:53;
surf(x,y,Z);
```



NB: die Längen der Vektoren *x* und *y* müssen den Dimensionen von *Z* entsprechen!

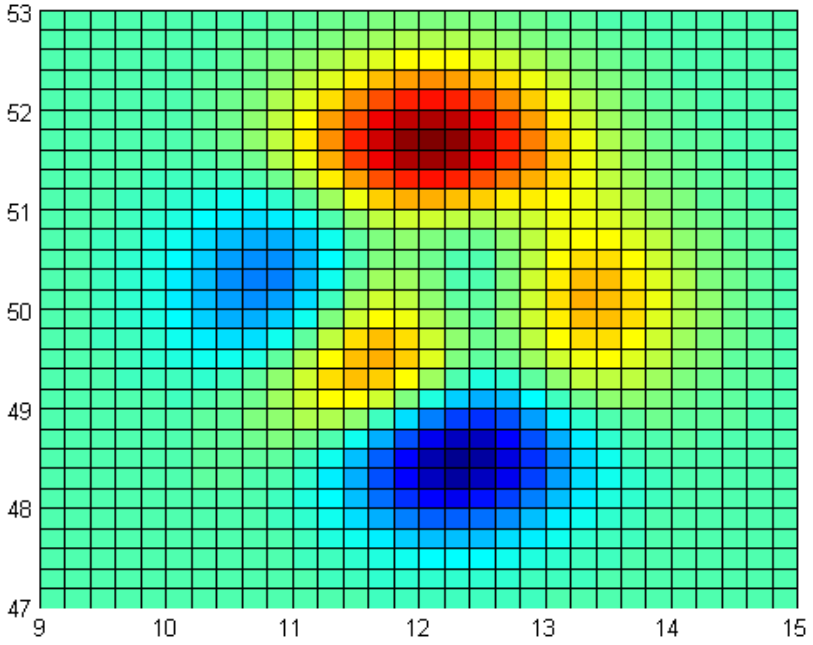
1D: Höhendarstellung



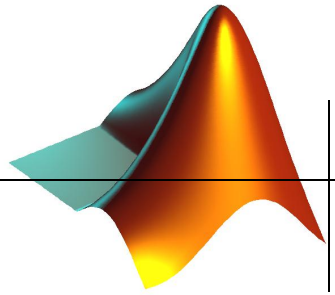
Schachbrettdarstellung:

- Befehl pcolor
  - Pseudo-Farbdarstellung
  - Wie surf, nur zweidimensional

```
pcolor(x,y,Z);
```



1D: Höhendarstellung

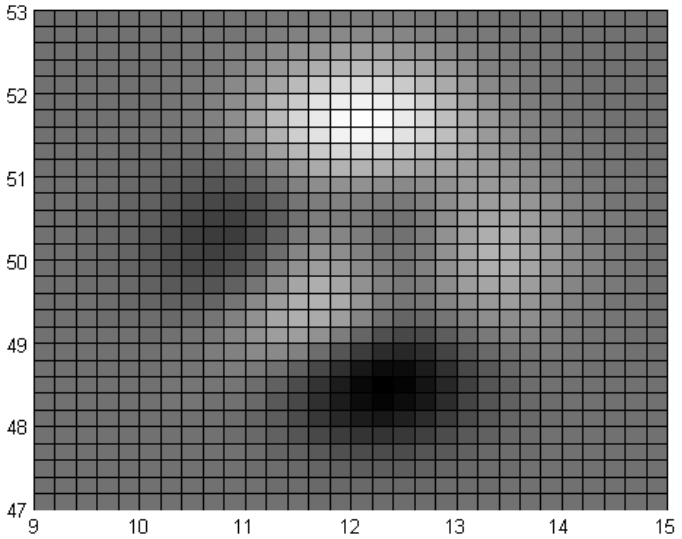


Farbanpassungen (1):

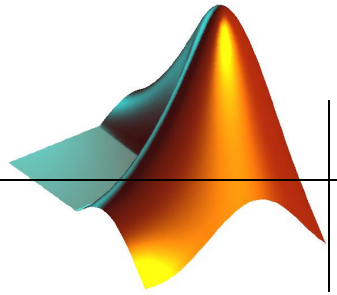
- colormap: Veränderung der Farbskala
  - jet (default): von blau nach rot
  - gray: Grauwertskala, ideal für GW-Bilder
  - hsv: Farbkreis, Minimum hat gleichen Farbwert wie Maximum (z.B. für Darstellung von Winkelwerten)
  - weitere Farbskalen: hot, cool, copper, pink, bone, u.a. (siehe help hsv)

```
pcolor(x,y,Z);

colormap gray;
```

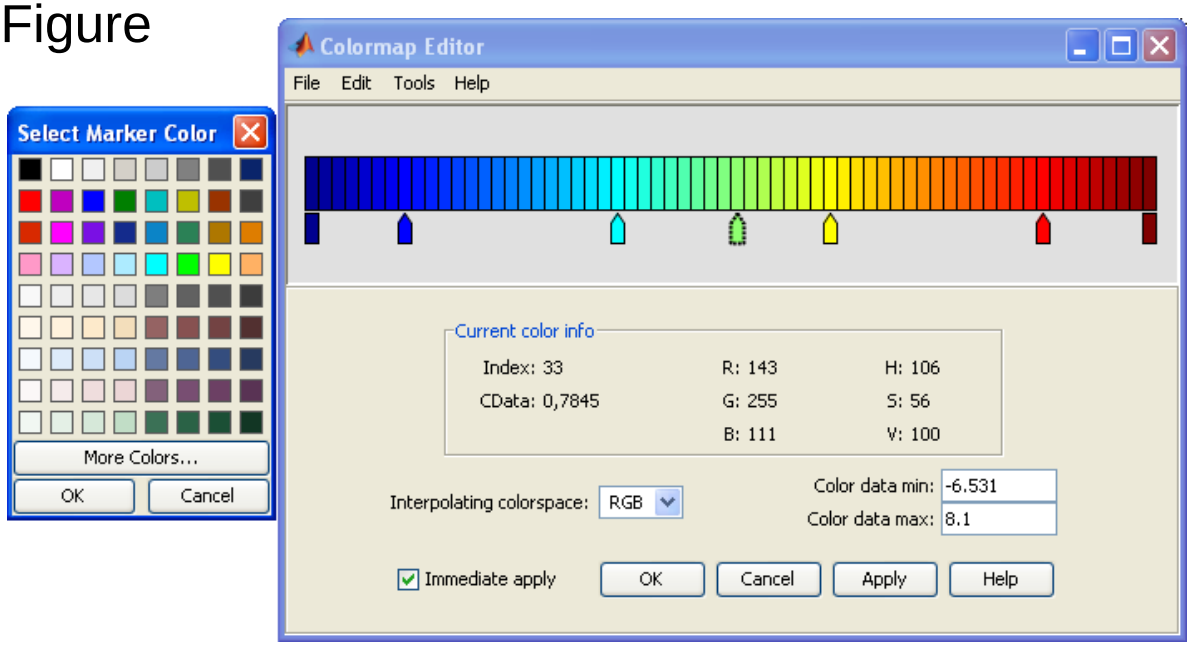


1D: Höhendarstellung

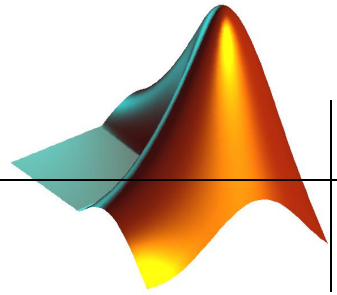


Farbanpassungen (2):

- colormapeditor: manuelle Colormap-Anpassung in GUI
  - Direktwahl aller standard-colormaps
  - Einfügen/Löschen von Markern
  - Verschieben/Farbänderung von Markern
  - Live-Vorschau im Figure
  - Min/Max-Wahl



1D: Höhendarstellung



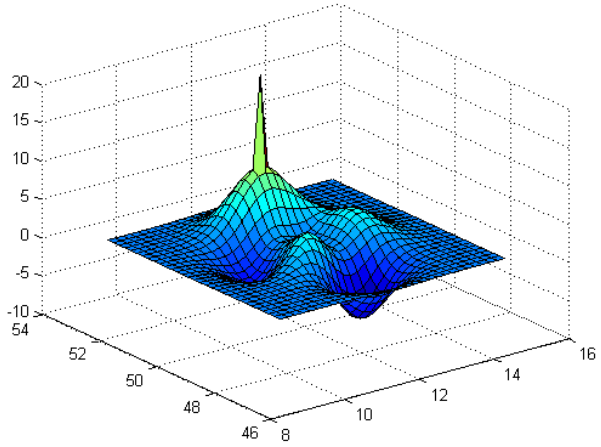
Farbanpassungen (3):

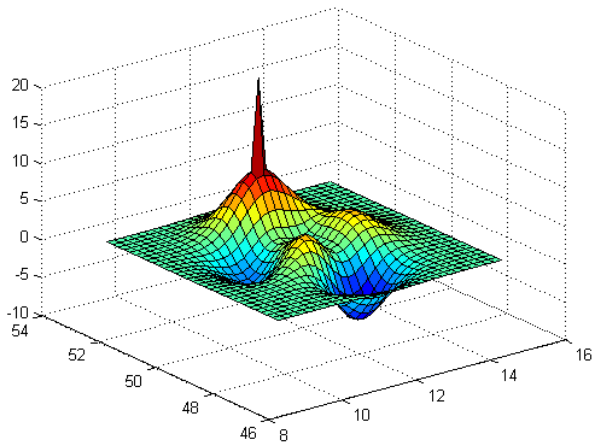
- caxis: manuelle Pseudocolor Skalierung
  - Häufiger Fall: Ausreißer in Daten
  - Zuweisung von **`caxis([cmin cmax])`** auf colormap-Maxima
  - Beispiel: **`Z = peaks(31);`** mit **`Z(24,16)=20;`** (vorher: 8.1)

```
surf(Z);
```

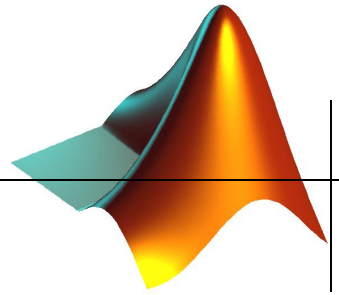
```
surf(Z);

caxis([-6.5 8.1])
```





1D: Höhendarstellung

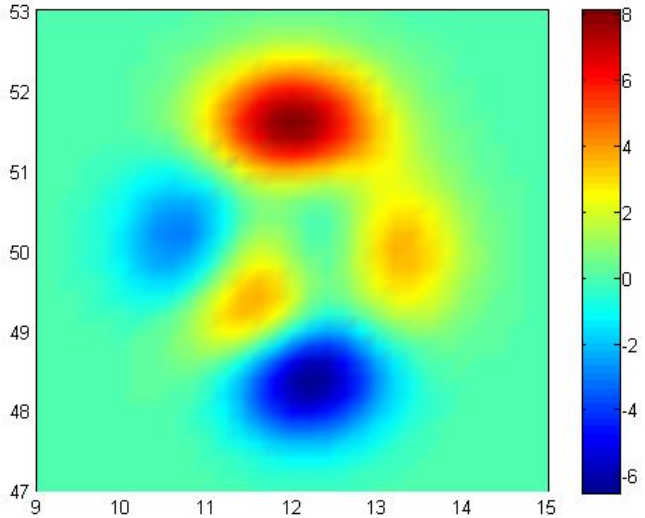


Farbanpassungen (4):

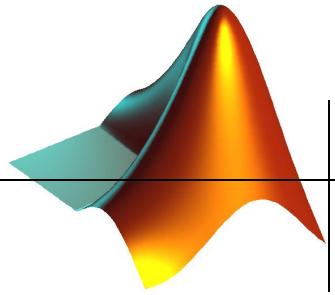
- shading: Anpassung der Farbdarstellung
  - faceted (default):
    - jede Linie bzw. Fläche hat eine Farbe
    - bei surf und pcolor werden die Gitterlinien schwarz angezeigt
  - flat: wie faceted, nur ohne Gitterlinien
  - interp: Farben von Linien und Flächen erhalten interpolierten Farbverlauf

```
pcolor(x,y,Z);
shading interp;
colorbar;
```

NB: colorbar zeichnet einen Balken mit der Farbskala



1D: Höhendarstellung

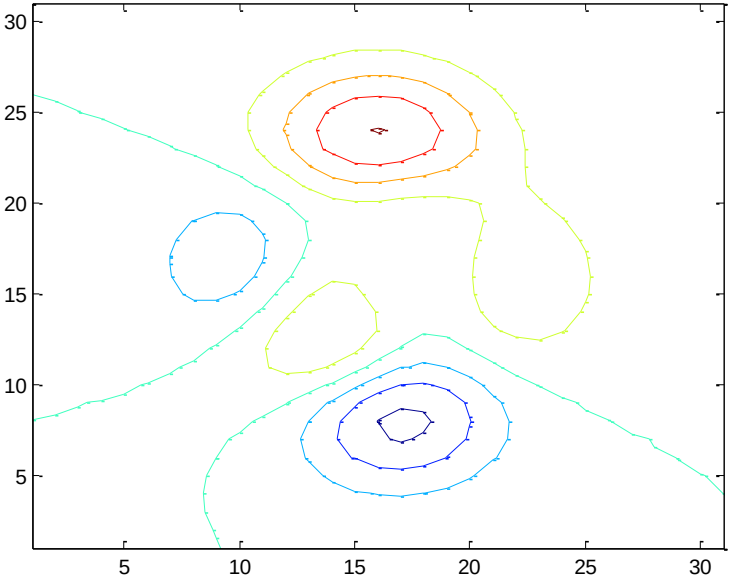


Darstellung von Höhenlinien (Isolinien) (1):

- Befehl contour: Plot von Höhenlinien
  - Linien gleichen Höhenwertes

```
contour(Z);
```

Automatische Erzeugung der Isolinien an ganzzahligen Höhenwerten

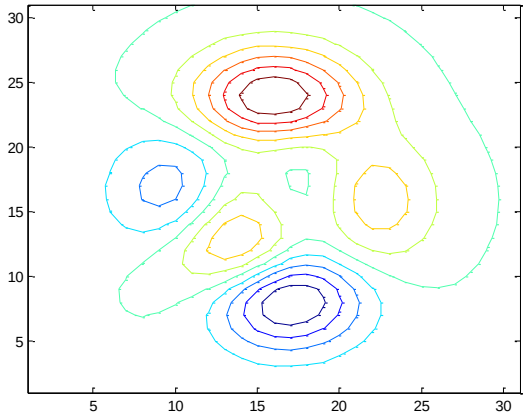


1D: Höhendarstellung

Darstellung von Höhenlinien (Isolinien) (2):

- Plot von 10 Höhenlinien

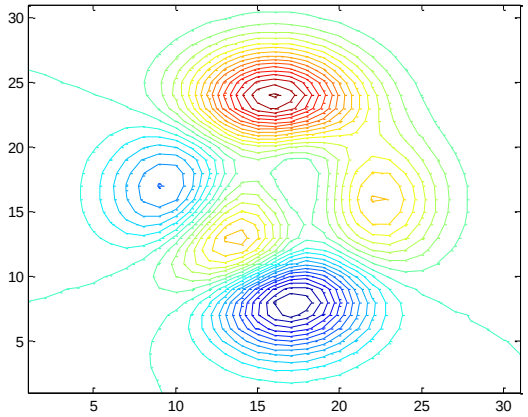
```
contour(Z,10);
```



→ automatische Aufteilung des Wertebereichs in 10 gleichabständige Höhenwerte

- Plot von Höhenlinien an gewünschten Höhenwerten

```
contour(Z,-6:0.5:8);
```



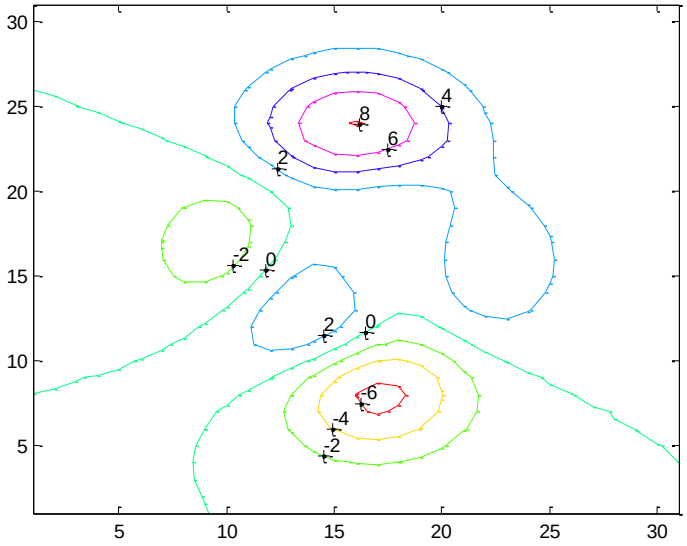
→ Plot von Höhenlinie an jedem Wert im Vektor

1D: Höhendarstellung

Darstellung von Höhenlinien (Isolinien) (3):

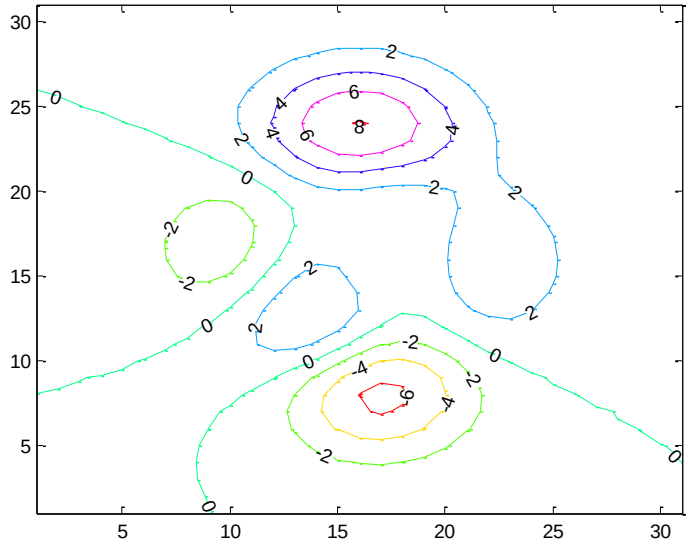
- Beschriftung von Höhenlinien

```
c = contour(Z,-6:2:8);
clabel(c);
```



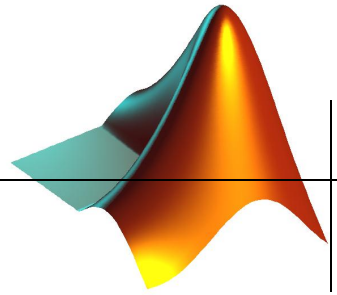
→ Kreuz mit Höhenwert

```
[c,h] = contour(Z,-6:2:8);
clabel(c,h);
```



→ Höhenwert auf der Linie

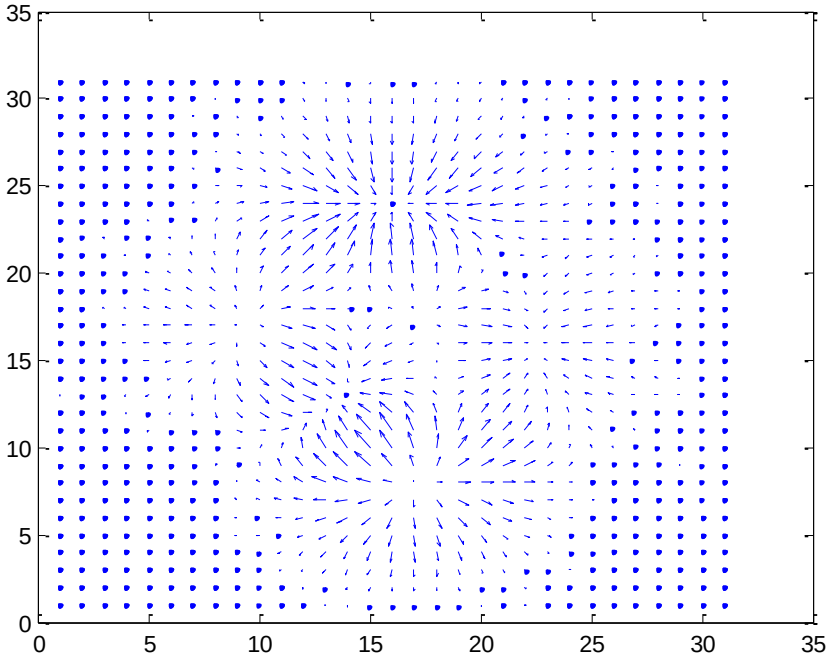
2D: Vektordarstellung



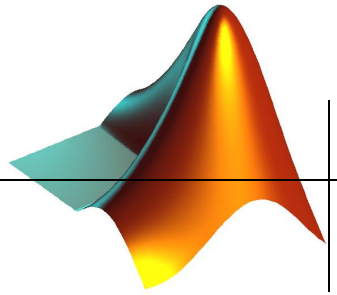
- Befehl quiver
- Darstellung von gerichteten Vektoren
- Ursprung des Vektors im Rasterpunkt

```
Z = peaks(31);
[gx,gy] = gradient(Z);
quiver(gx,gy);
```

NB: gradient erzeugt die partiellen Gradienten (Steigungen, Ableitungen) in x-Richtung (*gx*) und y-Richtung (*gy*) in jedem Gitterpunkt



2D: Vektordarstellung

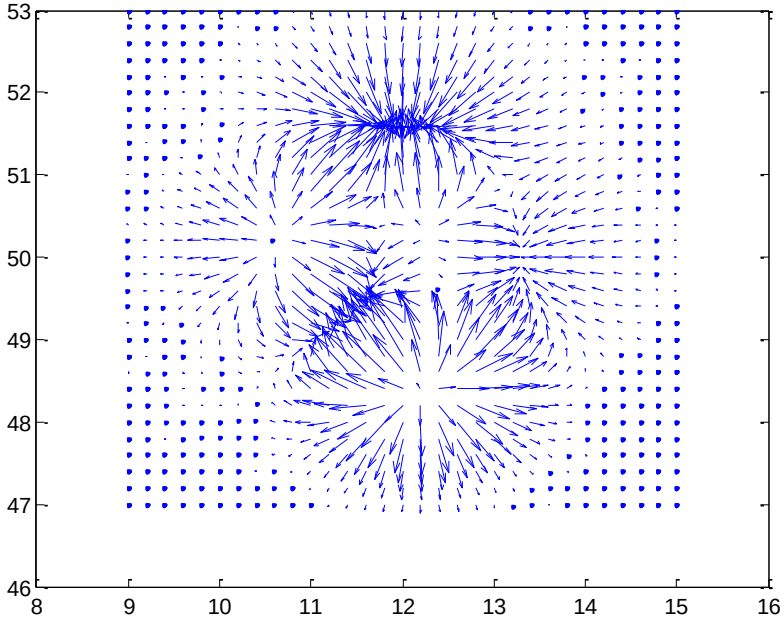


```
x = 9:0.2:15;
y = 47:0.2:53;
[gx,gy] = gradient(Z,0.2,0.2);
[xx,yy] = meshgrid(x,y);
quiver(xx,yy,gx,gy,3);
```

NB: gradient(Z, 0.2, 0.2) setzt für die Steigungsberechnung den Punktabstand in x- bzw. y-Richtung jeweils auf 0.2

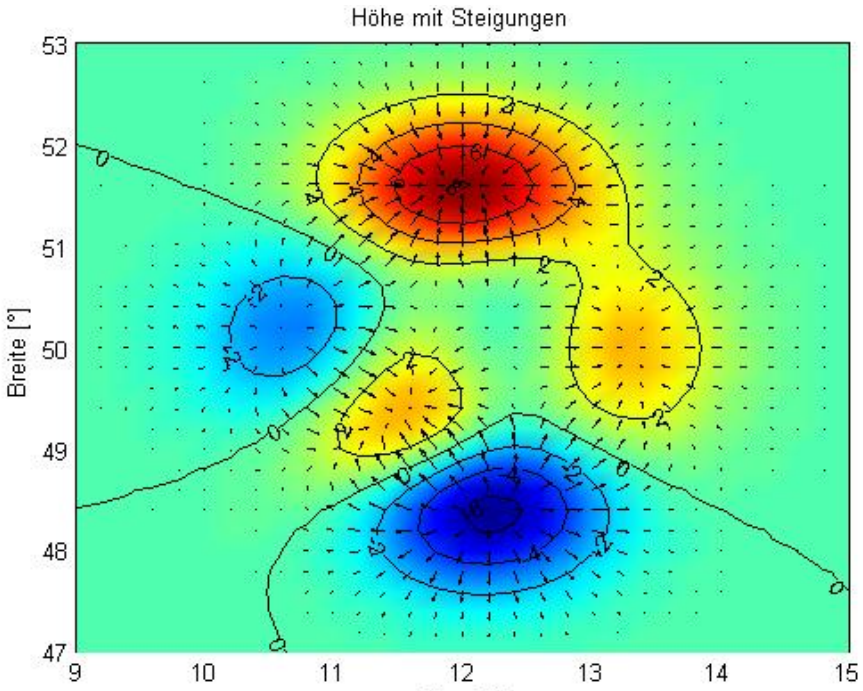
meshgrid erzeugt die Matrizen *xx* und *yy* mit den x- bzw. y- Koordinaten an den Gitterpunkten → praktisch für koordinatenabhängige Berechnungen

quiver(xx,yy,gx,gy,3) plottet die Vektoren *(gx,gy)* an den Rasterpunkten *(xx,yy)* und skaliert den Vektor um Faktor 3

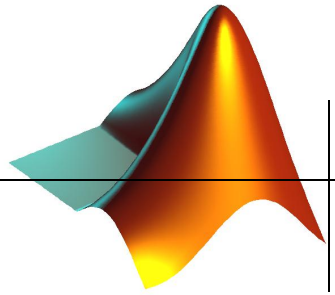


Kombination von Darstellungen

```
Z = peaks(31);
x = 9:0.2:15;
y = 47:0.2:53;
pcolor(x,y,Z);
shading interp;
xlabel('Länge [°]')
ylabel('Breite [°]')
title('Höhe [m]')
hold on;
[c,h] = contour(x,y,Z,-6:2:8,'k');
clabel(c,h);
[dx,dy] = gradient(Z,0.2,0.2);
[xx,yy] = meshgrid(x,y);
quiver(xx,yy,dx,dy,'k');
title('Höhe mit Steigungen')
```



Bilddarstellung

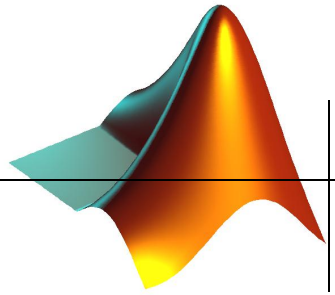


- Einlesen eines Bildes als Variable mit imread:

```
Im = imread('filename.ext');
```

- Größe von *Im*: *r* x *c* x *n*
  - *r*: Anzahl Zeilen
  - *c*: Anzahl Spalten
  - *n*: Anzahl Kanäle
    - 1 für GW-Bilder
    - 3 für Farbbilder, je 1 *rxc*-Matrix pro Farbkanal (RGB)
- Darstellung:
  - image
  - imagesc
    - Skaliert die Farbskala auf den Wertebereich des Bildes

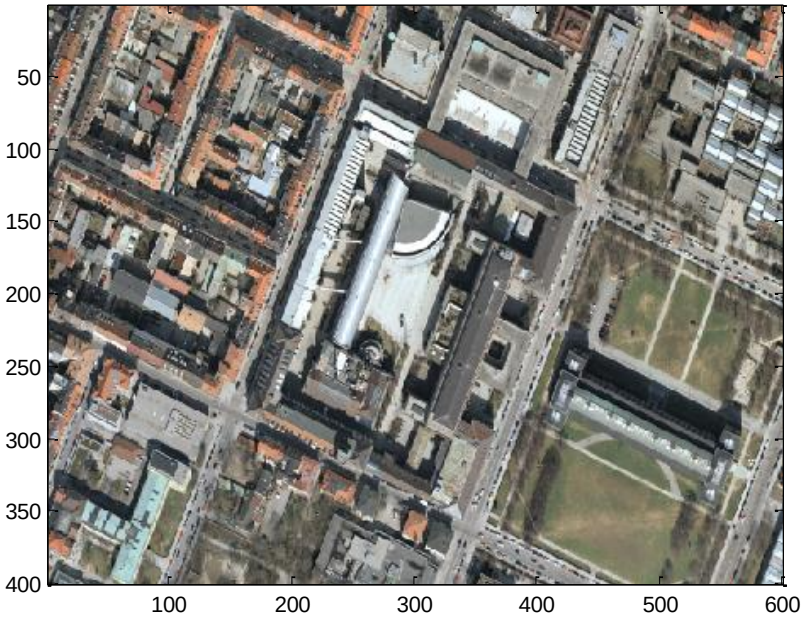
Bilddarstellung



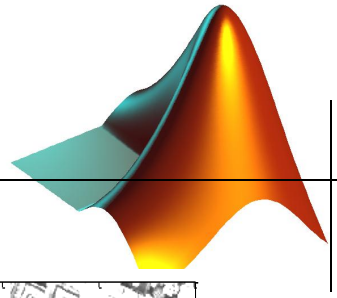
Beispiele:

- Farbbild

```
Im = imread('TUM_color.tif');
image(Im);
```

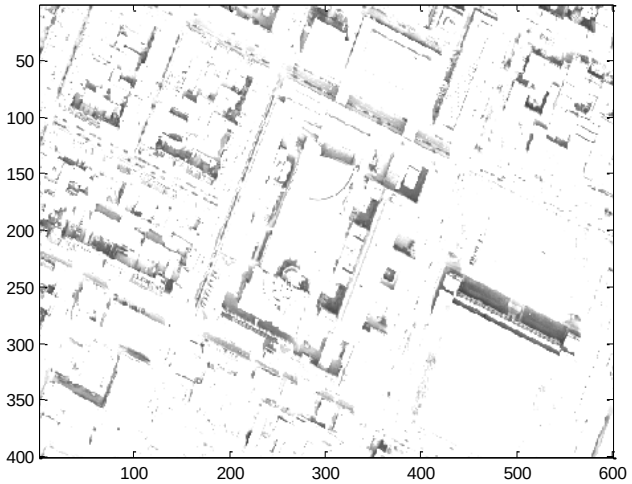


Bilddarstellung

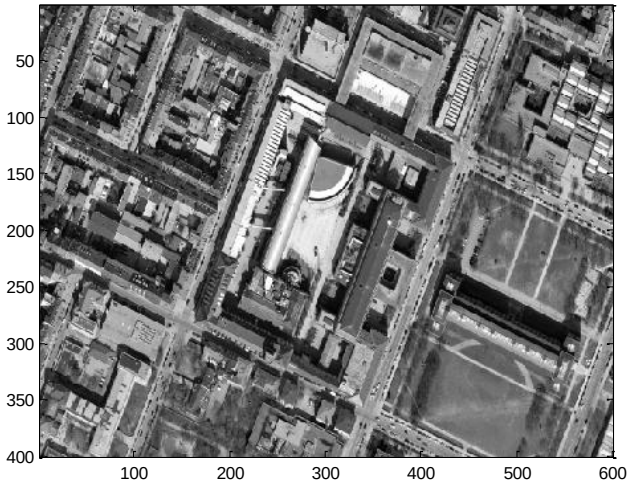


- GW-Bild

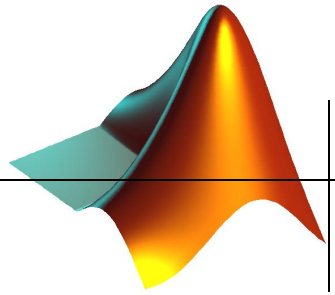
```
Im = imread('TUM_gray.tif');
image(Im);
colormap gray;
```




```
imagesc(Im);
colormap gray;
```



Bilddarstellung



- Achsen:
  - Bildkoordinatensystem:
    - In der Bildverarbeitung: wie Matrix-Index (Zeile, Spalte)
    - Zeile (row) von oben nach unten
    - Spalte (column) von links nach rechts
  - Modifizierte Achsenskalierung und –beschriftung:
    - Beschriftung der x-Achse (c-Achse) von links nach rechts
    - Beschriftung der y-Achse (r-Achse) von oben nach unten

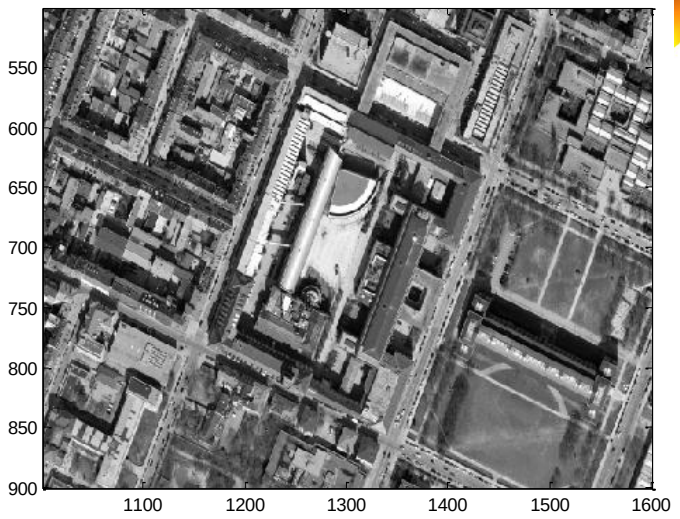
NB: Soll die y-Achse aufsteigende Werte von unten nach oben erhalten (z.B. zur Angabe von GK-Koordinaten), muss der Vektor *y* z.B. mit fliplr(y) umgekehrt und anschließend die y-Achse mit axis xy gekippt werden

Bilddarstellung

```
x = 1001:1000+size(Im,2);
y = 501:500+size(Im,1);
imagesc(x,y,Im);
```

NB: Länge von *x* entspricht der Anzahl der Spalten von *Im;*

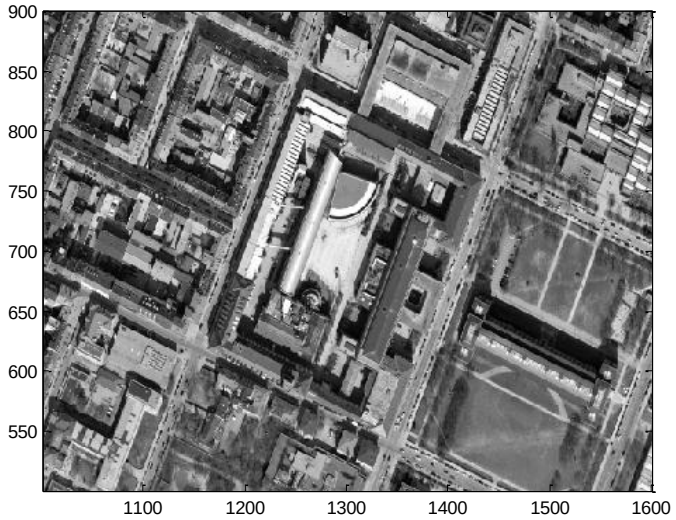
Länge von *y* entspricht der Anzahl der Zeilen von *Im*

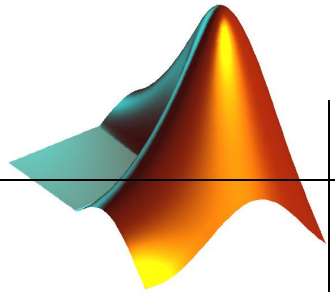



```
imagesc(x,fliplr(y),Im);
axis xy;
```

NB: Speichern von Bildern

imwrite(Im,‘dateiname.jpg‘,‘jpg‘)





Übung:

09:45 – 11:30 Uhr