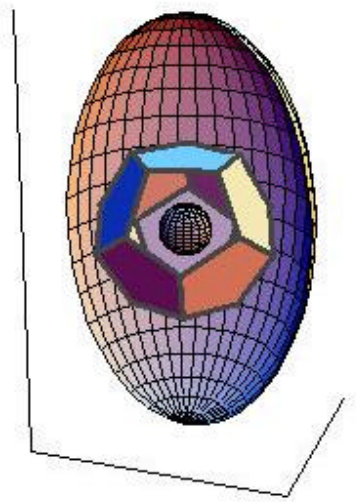
Einführung in Octave / Matlab

von

Rolf Wirz



Print-Ausgabe

Version 1.3.1 vom 23.09.2007 (mit Updates vom 21. 10 2009/ 17. 1. 2010)

Matlab_Octave00.rtf erstellt mit MS-Word

Adresse des Autors: Hochschule für Architektur, Bau und Holz
HSB
Pestalozzistrasse 20, CH-3400 Burgdorf
Tel. +41 (0)34 426 42 30

INHALTSVERZEICHNIS

Hinweis: *„Matlab_Octave001.RTF“* ist ein File-Name. Dieses File kann im Verzeichnis, in dem diese kurze Einführung gefunden worden ist, zu Copy-Paste-Zwecken beim Arbeiten mit Matlab heruntergeladen werden.

Einführung in Octace / Matlab

von Rolf Wirz

Herkunft von Matlab, Geschichte und daraus folgende Situation bei der Darstellung von Zahlen (Matlab_Octave001.RTF)

Nachfolgend werden einige erwähnenswerte Fakten stichwortartig wiedergegeben:

- Octave ist ein Clone von Matlab (Nachahmung mit Einschränkungen). Octave ist „Public domaine Software“ (gratis). Download siehe im folgenden Kapitel.
- Matlab: Numerikprogramm, rechnet mit Approximationen, vgl. http://de.wikipedia.org/wiki/MATLAB, http://en.wikipedia.org/wiki/MATLAB
- MATLAB: Abkürzung für "MATrix LABoratory". MATLAB wurde in den späten 1970-Jahre erfunden durch Cleve Moler, Chef des Computer Science Department der University von New Mexico. MATLAB wurde damals entworfen um den Studenten Zugang zu LINPACK und EISPACK zu ermöglichen, ohne dass zuerst Fortran gelernt werden musste. Das Paket hatte sich schnell verbreitet. Der Ingenieur Jack Little lernte es während einem Besuch 1983 an der Stanford University 1983 kennen und erkannte sein wirtschaftliches Potential. Zusammen mit Steve Bangert wurde das Packet in C umgeschrieben (JACKPAC) und 1984 die Firma MathWorks gegründet....
- Matlab war ursprünglich eine Fortranroutinen-Sammlung. Das sieht man dem Programm heute noch an (single und double-Precision).
- Traditionell wurde es oft eingeführt bei Ingenieuren.
- In der Technik: Praktisch findet man meistens nur Approximationen von Massen (Messungen)
- Datentypen: Integer => 2 Byte, Bereich -32768 bis 32767
- Real: 8 Bytes 16 Halbbytes, Ziffern Platz in Halbbyte ==> 15 Zeichen
- Problem: Binär werden abbrechende Dezimalbrüche oft periodisch ==> Näherungen, Fehler (1 ungefähr auf 10^{16} genau)
- Problem bei Summation: Grössere Zahlen zuerst ===> Man beginnt so mit dem grösserem Fehler. Einfluss auf die genaueren Stellen bei den kleineren Zahlen: Zählen nur wenig mit, Fehler hier also grösser.
- Besser mit kleineren Summanden zu summieren beginnen.....
- Problem der Operationsabfolge bei Berechnungen: Ausdrücke sind oft so umformbar, dass genauere oder ungenauere Resultate entstehen (die Fehlergrösse ist durch den Weg beeinflussbar).
- Bsp.: Rekursives Berechnen von abgebrochenen Reihen ist oft besser....

(***Matlab_Octave001.RTF***: File im Download-Menü mit dem Inhalt des Kapitels für copy-paste.)

Etwas herumspielen mit Octave / MATLAB (Matlab_Octave002.RTF):

Downloads aus dem Internet

Octave für Windows 95/98/NT/2000, Ghostscript

- **Octave für Windows 95/98/NT/2000**
- **Installation von Octave unter Windows 95/98 und Windows NT_1**
- **Installation von Octave unter Windows 95/98 und Windows NT_2**
- **Online Version of the Manual**
- **Ghostscript 5.50 und Ghostview 2.7, Adobe Acrobat Reader 4.0, Octave für Windows 9x/NT/2000**

Gnuplot (für Octave)

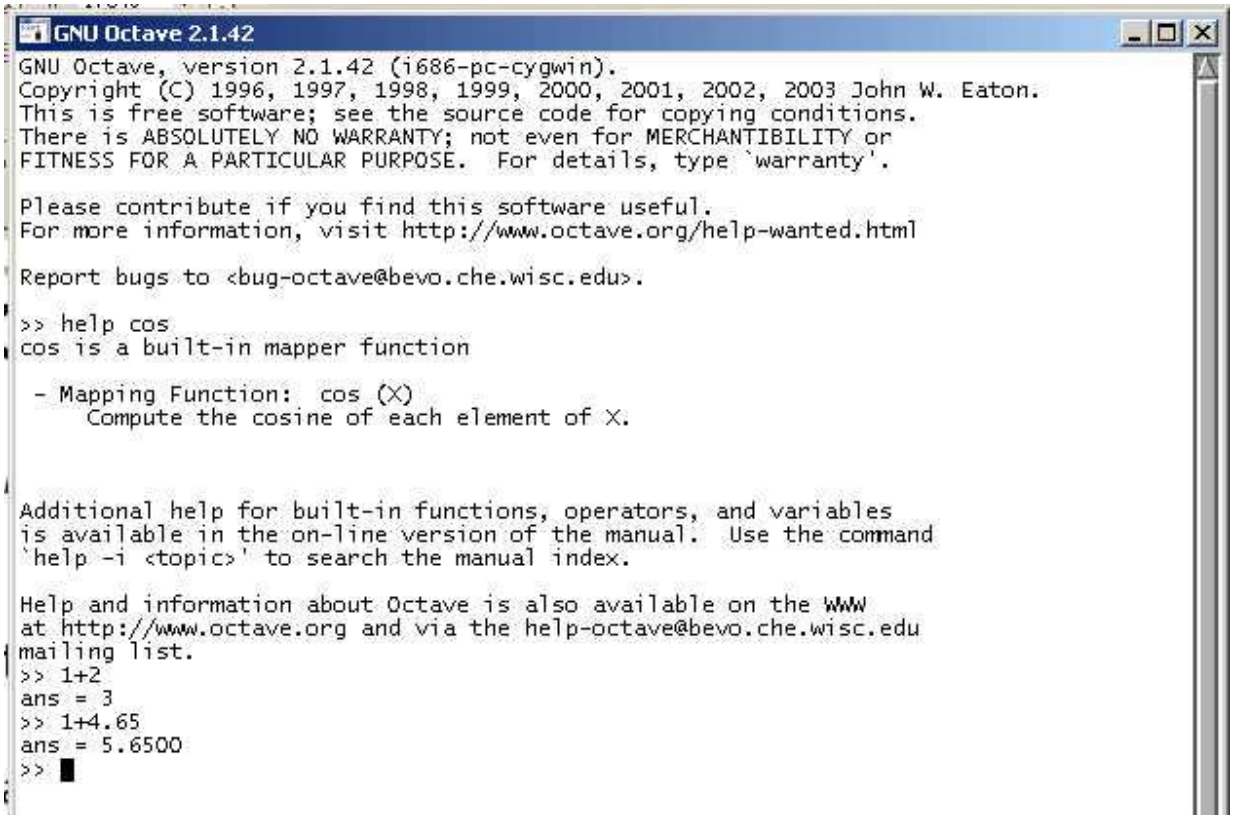
- **GNUPLOT Kurzanleitung**
- **Gnuplot Grundkurs**
- **GnuPlot Online Manual**
- **gnuplot homepage**
- ***Online Version of the octave-manual**:*
  - http://www.network-theory.co.uk/docs/octave/index.html

Octave starten

Starten:
Programm wählen in der Programmsammlung....
(man muss wissen, wo auf dem jeweiligen Computer ein Startbefehl existiert, bei Windows wird ein Icon erzeugt, z.B. unter Start – Programme – oder auf dem Desktop)

Beenden:
exit *ENTER:* ***Beendet*** *das Programm.*

Help, simple Arithmetik



Die Eingabe: >> help cos *ENTER erzeugt*

cos is a built-in mapper function

- Mapping Function: cos (X)
  Compute the cosine of each element of X.

Additional help for built-in functions, operators, and variables
is available in the on-line version of the manual. Use the command
`help -i <topic>' to search the manual index.

Help and information about Octave is also available on the WWW
at http://www.octave.org and via the help-octave@bevo.che.wisc.edu
mailing list.

helpdesk *ENTER: Funktioniert nicht in Octave, ebenso "*doc*"*
Jedoch funktioniert help

Eingaben erzeugen jeweils eine Ausgabe ans (answer ein Speicherplatzinhalt)

Achtung Problem bei Octave: Copy-paste funktioniert im Gegensatz zu Matlab nicht! Dafür kann man mit den **Pfeiltasten** frühere Eingaben zurückscrollen und abändern.

Simple Arithmetik:

```
>> 1+2
ans = 3

>> 1+4.65
ans = 5.650

>> 2*3
ans = 6

>> 2/3
ans = 0.66667

>> 2^3
ans = 8

>> 2^(1/2)
ans = 1.4142

>> sqrt(2)
ans = 1.4142

>> ans
ans = 1.4142
```

Arbeiten mit Funktionen:

```
>> cos(pi)
ans = -1

>> cos(0.5)
ans = 0.87758

>> cos(pi)
ans = -1

>> cos(0.5)
ans = 0.87758
```

Spiele etwas mit MATLAB

Scrolle mit den Pfeiltasten. Probiere dann die folgenden Befehle aus, wenn es sonst nicht geht mit help*:*

cos, acos, cosh, acosh
sin, asin, sinh, asinh
tefunc, dst, idst

Wie man aus einigen Beispielen sehen kann, ist der Befehlsumfang von Octave kleiner als derjenige von MATLAB.

Arbeiten mit Variablen und Befehlsketten

```
>> x=1.8; X=1e-4; x*X
ans = 0.00018000
```

Mittelwert:

```
>> a=2.65; b=1.3; c=5.5; m=(a+b+c)/3
m = 3.1500
```

Ausgabe auf später aufsparen:

```
>> u=1-1/2+1/3-1/4+1/5-1/6+1/7;
>> u
u = 0.75952
```

Speichern und laden

***Das Speichern und Laden** illustrieren wir an folgendem Beispiel:*

Wir starten Octave neu.
Eingabe einer Variablen und speichern:

```
>> x=1
x = 1
>> save('C://work_loeschen/myFile')
```

Achtung auf den Pfad! In einem Computernetzwerk muss das verwendete Laufwerk nicht existieren.
Verwende dann ein anderes, wenn möglich eigenes Laufwerk.

Jetzt beenden wir beenden und starten das Programm neu. Die Variable ist dann gelöscht.

```
>> load('C://work_loeschen/myFile2')
```

Nun fragen können wir die neu geladene Variable ab:

```
>> x
x = 1
```

Achtung: Keine Sessions abspeichern, in der schon Ladebefehle drin sind. Bei neuen Laden wird dann sofort wieder ein Ladebefehl aktiviert, was in einem Loop enden könnte.

clear *löscht den Speicher,* clear x *die Variable* x*:*

```
>> x=1
x = 1
>> clear x
>> x
error: `x' undefined near line 4 column 1
>>
>> x=1; y=2
y = 2
>> clear
>> y
error: `y' undefined near line 6 column 1

>> save('C://work_loeschen/myFile')
```

Etwas spielen mit Plots und Funktionen (Beispiele)

Wir definieren eine Funktion, den Definitionsbereich und ploten dies.
Wandle die Grössen etwas ab und schaue, was passiert! (Plot ➔ in neuem Fenster!)

```
>> a=2; r=0.5;x=0:0.1:10;y=x ./(1+a.*x+r.*x.^2);plot(x,y);
```
http://rowicus.ch/Wir/Matlab_Octave/out1.jpg

```
>> x=0:0.1:pi;y=sin(x);plot(x,y)
```
Output siehe http://rowicus.ch/Wir/Matlab_Octave/out2.jpg

Workspace abfragen, nachdem wieder Variablen definiert worden sind:

```
>> whos

*** local user variables:

prot  type                 rows   cols  name
====  ====                   ====   ====  ====
 rwd  scalar                   1      1  X
 rwd  scalar                   1      1  a
 rwd  scalar                   1      1  b
 rwd  scalar                   1      1  c
 rwd  scalar                   1      1  m
 rwd  scalar                   1      1  u
```

rwd scalar 1 1 x

Grundkenntnisse: Arithmetik, Funktionen, Formate, Loops, Fehler, löschen,.... (Matlab_Octave003.RTF)

Zahlenformate

Probiere, die folgenden Sequenzen am Computer aus.

>> 1/3 *(Am Ende einer solchen Zeile immer ENTER drücken!)*
ans = 0.33333 *(Genauigkeit in der Regel 8 Bytes oder Zeichen inkl. +/- etc., 8 Bit pro Byte)*

>> realmax *(Grösste darstellbare Zahl)*
realmax = 1.7977e+308

>> b=realmax;

>> c=b+100000000
c = 1.7977e+308

>> c=b+1e+308
c = Inf *(Unendlich, da nicht grösser darstellbar)*

>> format long *(In Fortran IV double precision)*

Schlaufen

>> x=0; for i=1:10000 x=x+1/i^2;end *(Schlaufenprogrammierung)*

>> x
x = 1.64483407184806

>> x=0; for i=10000:-1:1 x=x+1/i^2; end

>> x
x = 1.64483407184806

>> format short *(In Fortran IV single precision)*

>> x=0; for i=1:10000 x=x+1/i^2; end

>> x

x = 1.64

>> x=0; for i=10000:-1:1 x=x+1/i^2; end

>> x
x = 1.64

>> format long

Arithmetische Operationen, Wurzel, Fehler, kleinste Zahl

>> a=sqrt(987600) *(Quadratwurzel)*
a = 993.780659904388

>> b=sqrt(987599) *(Quadratwurzel)*

b = 993.780156775129

>> a-b
ans = 0.0005031292585045

>> 1/(a+b) *(x+y)(x-y) = x^2-y^2, x-y = (x^2-y^2)/(x+y)*
ans = 0.0005031292585404

>> c=a-b
c = 0.0005031292585045

>> d=1/(a+b)
d = 0.0005031292585404

>> c-d
ans = -3.59550713854850e-14 *(Fehler!)*

>> beep *(Es ertönt ein Beep-Ton)*

>> realmin *(Kleinste darstellbare Zahl)*
realmin = 2.22507385850720e-308

Clear, beep, who

>> clear d *(d löschen)*

>> beep *(Es ertönt ein Beep-Ton)*

>> who *(Situation, Variablen anzeigen)*

*** currently compiled functions:

beep puts

*** local user variables:

a b c i x

Einige Funktionen und Konstanten: exp, log, e, floor, round, rem, sign

>> exp(e) *(e hoch e)*
ans = 15.1542622414793

>> exp(1) *(e hoch 1)*
ans = 2.71828182845905

>> log(1.4) *(Logarithmus naturalis von 1.4)*
ans = 0.336472236621213

>> log10(1.4) *(10-er Logarithmus von 1.4)*
ans = 0.146128035678238

>> log10(10) *(10-er Logarithmus von 10)*
ans = 1

>> log(e) *(Logarithmus naturalis von e)*
ans = 1

>> e *(Eulersche Zahl e)*
e = 2.71828182845905

>>
>> floor(pi) *(Nächst kleinere ganze Zahl)*

ans = 3

>> round(pi+0.5) *(Rundung)*
ans = 4

>> rem(5,6) *(Divisionsrest)*

ans = 5

>> rem(70,12)
ans = 10

```
>> sign(-4)          (Signum-Funktion)
ans = -1
```

Vektoren, Skalarprodukt, Vektorprodukt

```
>> u=[3,4,6]; v=[1,2,3]; dot(u,v)   (Vektoren und Skalarprodukt)
ans = 29

>> cross(u,v)       (Vektorprodukt)
ans =
              0              -3               2
```

Imaginäre Einheit, real, imag, conj

```
>> j                (Imaginäre Einheit, i ist vorher als Index missbraucht worden)
j = 0 + 1i

>> i
i = 1

>> j^2
ans = -1

>> real(j+1)              (Realanteil)
ans = 1

>> g=j+2; conj(g)              (Konjugiert komplexe Zahl)
ans = 2 - 1i

>> imag(g)              (Imaginäranteil)
ans = 1

>> g=(3+j); imag(g)
ans = 1

>> g=1+3j; imag(g)
ans = 3
```

Komposition von Vektoren, Kenngrössen (Matlab_Octave004.RTF)

(Vektoren sind hier Arrays resp. Listen.)

Sequenzen mit Vektoren: Elementextraktion, Transponierte, Vektorenerzeugung

Probiere die folgenden Sequenzen mit Vektoren am Computer aus:

```
>> a=[3.4 2.9 6 -4.76 8]
                (Eingabe eines Vektors mit den angegebenen Komponenten)
a =

   3.40000   2.90000   6.00000  -4.76000   8.00000

>> a(4)             (4. Element des Vektors)
ans = -4.7600

>> a(3)             (2. Element des Vektors)

ans = 6

>> b=[3.4; 2.9; 6; -4.76; 8]             (Eingabe eines Spaltenvektors mit geg. Komp. )
b =

   3.40000
   2.90000
   6.00000
  -4.76000
   8.00000

>> a1=a'             (a wird transponiert)
a1 =

   3.40000
   2.90000
   6.00000
  -4.76000
   8.00000

>> c=0:7             (Zeilenvektor mit den Elementen von 0 bis 7, Schrittweite 1)
c =
  0  1  2  3  4  5  6  7

>> x=1:0.1:3             (Schrittweite 0.1 )
x =
 Columns 1 through 8:

  1.00000  1.10000  1.20000  1.30000  1.40000  1.50000  1.60000  1.70000
```

Columns 9 through 16:

1.80000 1.90000 2.00000 2.10000 2.20000 2.30000 2.40000 2.50000

Columns 17 through 21:

2.60000 2.70000 2.80000 2.90000 3.00000

>> y=linspace(0,1.5*pi) *(Zeilenvektor, 100 Komponenten von 0 bis 1.5 pi)*
y =

Columns 1 through 8:

0.00000 0.04760 0.09520 0.14280 0.19040 0.23800 0.28560 0.33320

Columns 9 through 16:

0.38080 0.42840 0.47600 0.52360 0.57120 0.61880 0.66640 0.71400

Columns 17 through 24:

0.76160 0.80920 0.85680 0.90440 0.95200 0.99960 1.04720 1.09480

Columns 25 through 32:

1.14240 1.19000 1.23760 1.28520 1.33280 1.38040 1.42800 1.47560

Columns 33 through 40:

1.52320 1.57080 1.61840 1.66600 1.71360 1.76120 1.80880 1.85640

Columns 41 through 48:

1.90400 1.95160 1.99920 2.04680 2.09440 2.14199 2.18959 2.23719

Columns 49 through 56:

2.28479 2.33239 2.37999 2.42759 2.47519 2.52279 2.57039 2.61799

Columns 57 through 64:

2.66559 2.71319 2.76079 2.80839 2.85599 2.90359 2.95119 2.99879

Columns 65 through 72:

3.04639 3.09399 3.14159 3.18919 3.23679 3.28439 3.33199 3.37959

Columns 73 through 80:

3.42719 3.47479 3.52239 3.56999 3.61759 3.66519 3.71279 3.76039

Columns 81 through 88:

3.80799 3.85559 3.90319 3.95079 3.99839 4.04599 4.09359 4.14119

Columns 89 through 96:

4.18879 4.23639 4.28399 4.33159 4.37919 4.42679 4.47439 4.52199

Columns 97 through 100:

4.56959 4.61719 4.66479 4.71239

>> y1=linspace(1,7,25) *(25 Elemente zwische 1 und 7)*

y1 =

Columns 1 through 8:

1.00000 1.25000 1.50000 1.75000 2.00000 2.25000 2.50000 2.75000

Columns 9 through 16:

3.00000 3.25000 3.50000 3.75000 4.00000 4.25000 4.50000 4.75000

Columns 17 through 24:

5.00000 5.25000 5.50000 5.75000 6.00000 6.25000 6.50000 6.75000

Column 25:

7.00000

>> w=[c 10:12] *(w besteht aus c und den Zahlen von 10 bis 12)*

w =

Columns 1 through 8:

0 1 2 3 4 5 6 7

Columns 9 through 11:

10 11 12

>> u=w(1:2:8) *(Komponenten von w mit Index 1 bis 8, Indexschrittweite 2)*

u =

```
     0     2     4     6

>> beta=w(3:-1:1)          (Komponenten von w mit Index 3 bis 1, Indexschrittweite -1)
beta =
     2     1     0

>> w([1 1 7])             (Komponenten von w mit Index 1, 1 und 7)
ans =

     0     0     6

>> zeros(1,6)             (6 mal die 0)
ans =

     0     0     0     0     0     0

>> ones(1,4)              (4 mal die 1)
ans =

     1     1     1     1

>> ones(6,1)              (Die 1 in 6 Zeilen und 1 Spalte)
ans =

     1
     1
     1
     1
     1
     1
```

Vektoroperationen, diskrete Plots (Matlab_Octave005.RTF:)

length, size, sum

```
>> u=[0 2 4 10 11]        (Grundvektor)
u =

     0     2     4    10    11

>> length(u)         (Anzahl Komponenten)
ans = 5

>> size(u)
```

ans =

1 5 *(eine Zeile, 5 Spalten)*

>> size(u.')
ans =

5 1 *(5 Zeilen, 1 Spalte)*

>> norm(u)
ans = 15.524 *(Vektorlänge)*

>> sum(u) *(Komponentensumme)*
ans = 27

Addition einer Konstante, Addition von Vektoren, Operationen komponentenweise ausführen

>> m=[43 0 -5]; n=[1 1 6 2]; *(Eingabe zweier Vektoren)*

>> m+7 *(Addiert zu jeder Komponente von m 7)*
ans =

50 7 2

>> 3*m *(Multipliziert jede Komponente von m mit 3)*
ans =

129 0 -15

>> m+n *(Erzeugt Error, da ungleich lange Vektoren nicht addiert werden können)*

error: operator +: nonconformant arguments (op1 is 1x3, op2 is 1x4)
error: evaluating binary operator `+' near line 28, column 2

>> m=[4 5 3 -5]; *(m neu eingeben)*

>> m+n *(Addition von zwei Vektoren)*
ans =

5 6 9 -3

>> m.*n *(Komponentenweise Multiplikation)*
ans =

```
        4        5       18       -10

>> m./n                    (Komponentenweise Division)
ans =

   4.00000   5.00000   0.50000  -2.50000

>> n.^(1.5)                 (Komponentenweise potenzieren)
ans =

  1.0e+01  *

  0.10000  0.10000  1.46969  0.28284
```

Sortieren, Maximum, Minimum, Mittelwert

```
>> sort(n)                  (n der Grösse nach sortieren)
ans =

       1       1       2       6

>> [min(m) max(m)]                (Kleinste und grösste Komponente)
ans =

      -5        5

>> [sum(m) mean(m)]                (Komponentensumme und Mittelwert)
ans =

  7.0       1.75000
  8.0
```

Elemente finden unter gesetzten Bedingungen, rechnen unter Bedingungen

```
>> w1=find(m>1)                (Liest die Komponenten grösser als 1 von m aus)
w1 =

       1       2       3

>> m<1                   (Setzt 1 falls die Bedingung wahr ist, sonst 0)
ans =

       0       0       0       1

>> m>1
ans =
```

```
     1     1     1     0
```

```
>> m(m>1)
```
(Liest die Komponenten von m aus, für die die Bedingung wahr ist)

```
ans =

     4     5     3
```

```
>> m(m<1)
ans = -5
```

```
>> cos(m)
```
(Berechnet in m elementweise den Cosinus)

```
ans =

  -0.65364   0.28366  -0.98999   0.28366
```

```
>> cos(cos(m))
ans =

  0.79387  0.96004  0.54870  0.96004
```

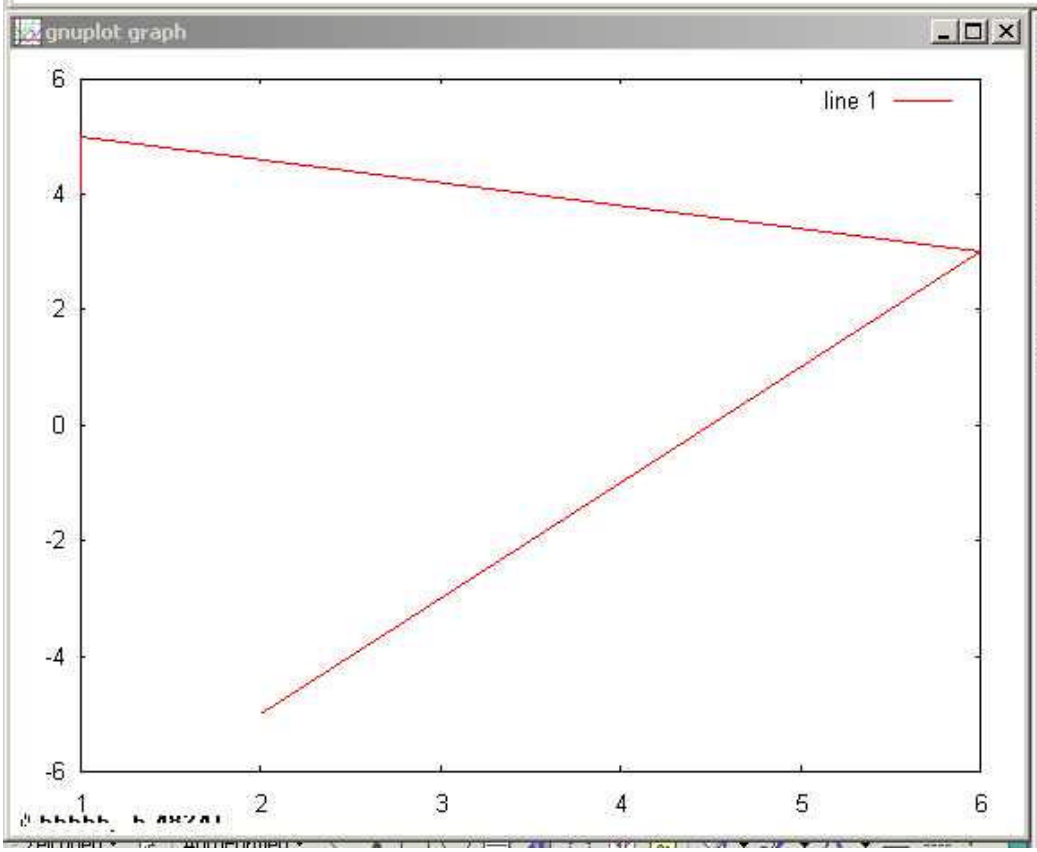
Plots

```
>> plot(m,n,'o')
```
(Plotet 'o' bei jeweils der x-Koord. aus m und der y-Koord. aus n)

```
>> plot(n,m)
```
(Plotet Geradenstücke zwischen den Punkten im letzten)

GNU Octave 2.1.42

```
   4.00000   5.00000   0.50000  -2.50000

>> m.^(1.5)
ans =

  1.0e+01  *

   0.80000 + 0.00000i   1.11803 + 0.00000i   0.51962 + 0.00000i  -0.00000 - 1.11803i

>> sort(n)
ans =

        1        1        2        6

>> cos(m)
ans =

  -0.65364   0.28366  -0.98999   0.28366

>> cos(cos(m))
ans =

  0.79387  0.96004  0.54870  0.96004

>> plot(m,n,'o')
>> plot(n,m)
```

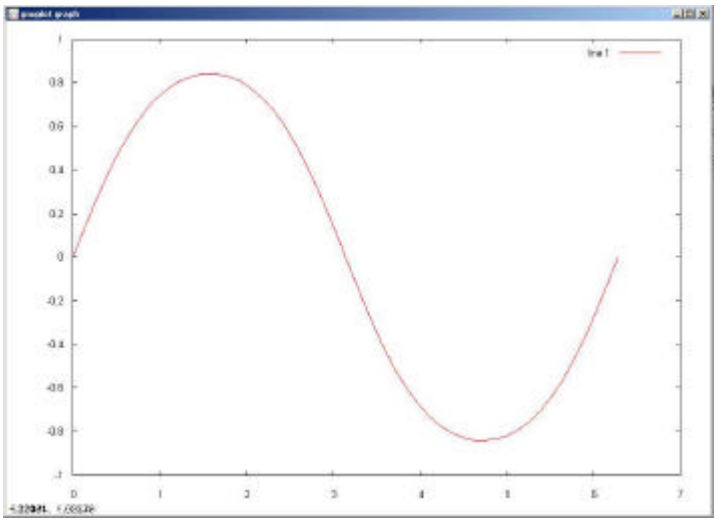



```
gnuplot
plot> set nologscale
                ^
      warning: depre
plot> set nopolar
                ^
      warning: depre
plot> pl 'C:/PROGRA~
plot> set nologscale
                ^
      warning: depre
plot> set nopolar
                ^
      warning: depre
plot> pl 'C:/PROGRA~
plot>
```

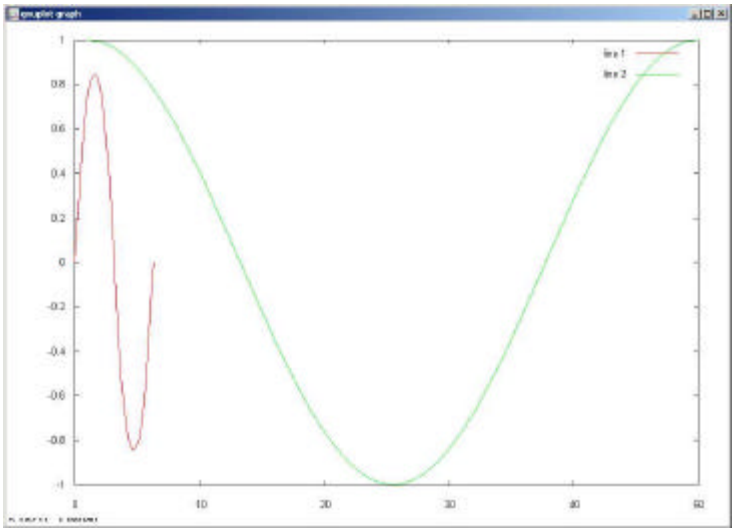
Plots (Matlab_Octave006.RTF)

Probiere die folgenden verschiedenartigen Plots aus!

```
>> x=linspace(0,2*pi,50); y=sin(sin(x)); plot(x,y)
>>                                    (Einteilung der x-Achse, y-Wert definieren, Plot)
```



```
>> plot(x,y,':',x,cos(x),'g')          (Zwei Funktionen in ein Diagramm ploten, Farbwahl)
```



Gitter, Plots übereinanderlegen, Labels, Unterfenster

Aufgabe: Probiere die nachfolgenden Befehle aus!

Achtung: Die Bereiche und Funktionen müssen zuerst definiert sein!

>> grid *(Im nächsten Plot Gitter aktivieren)*

>> plot(x,cos(x)) *(Plot)*

>> xlabel('Variable x') *(Im nächsten Plot x-Achse beschriften)*

>> z=sin(sin(x)).^2; *(Funktion definieren)*

>> hold on;plot(x,z,'r') *(Plot über den alten Plot legen, rot)*

>> hold off *(Übereinander legen abschalten)*

>> subplot(2,1,1)
(Graphikfenster mit Unterfenster in 2 Zeilen, 1 Spalte erzeugen, 1. Unterfenster aktivieren, Problem in Octave)

>> plot(x,y) *(Plot ins 1. Unterfenster)*

>> subplot(2,1,2)
(Graphikfenster mit Unterfenster in 2 Zeilen, 1 Spalte erzeugen, 2. Unterfenster aktivieren, Problem in Octave)

>> plot(x,z) *(Plot ins 2. Unterfenster)*

Graphikfenster löschen, leeres Fenster öffnen, Plot speichern

>> clf *(Graphikfenster löschen)*

>> figure
ans = 1
>> figure; *(Leeres Fenster öffnen)*

>> h = plot(x,sin(x.^2)); *(Grafik als Objekt in h speichern - Problem in Octave)*

error: value on right hand side of assignment is undefined
error: evaluating assignment expression near line 4, column 3

Ploteigenschaften anzeigen und ändern, Graphik drucken

>> get(h) *(Ploteigenschaften anzeigen - Problem in Octave)*

error: `get' undefined near line 4 column 1

>> set(h,'Color',[1 0 1],'LineWith',2) *(Graphikeigenschaften ändern - Problem in Octave)*

error: `h' undefined near line 4 column 5
error: evaluating argument list element number 1

>> help print *(Info Graphik ausdrucken resp. an LPT1 schicken - vgl. Skript TU München)*

print is the user-defined function from the file
/usr/share/octave/2.1.42/m/plot/print.m

Copyright (C) 1999 Daniel Heiserer
Copyright (C) 2001 Laurent Mazet

This program is free software; it is distributed in the hope that it
will be useful, but WITHOUT ANY WARRANTY; without even the implied
warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See
the GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this file; see the file COPYING. If not, write to the
Free Software Foundation, 59 Temple Place - Suite 330, Boston, MA
02111-1307, USA.

Additional help for built-in functions, operators, and variables
is available in the on-line version of the manual. Use the command
`help -i <topic>' to search the manual index.

Help and information about Octave is also available on the WWW
at http://www.octave.org and via the help-octave@bevo.che.wisc.edu
mailing list.

Vektoren, Matrizen, Gleichungssysteme (Matlab_Octave007.RTF)

Matrix eingeben, Element abfragen, Größe der Matrix abfragen

Matrix eingeben

>> A=[3 4;2 2;-1,6]

A =

3 4

```
     2     2
    -1     6
```

Element abfragen

```
 >> A(3,2)
ans = 6
>> A(3,:)
ans =

    -1     6
```

Größe der Matrix abfragen

```
 >> size(A)
ans =

    3     2
```

Elemente umordnen, als Vektor darstellen

Elemente umordnen

```
 >> B = reshape(A,2,3)
B =

     3    -1     2
     2     4     6
```

Als Vektor darstellen

```
  >> x=A(:)
x =

     3
     2
    -1
     4
     2
     6
```

Matrixprodukt, Elemente quadrieren

Matrixprodukt

```
 >> B*A
ans =

     5    22
     8    52

>> A*B
ans =

    17    13    30
    10     6    16
     9    25    34
```

Elemente quadrieren

```
 >> A.*A
ans =

     9    16
     4     4
     1    36

>> B.*B
ans =

     9     1     4
     4    16    36
```

Inverse, Transponierte, Gleichung mit der Inversen lösen

Neue Matrix und Vektor eingeben

```
 >> C=[1 2; 3 4]
C =

     1     2
     3     4

>> b=[6;8]
b =

     6
     8
```

Inverse

```
 >> inv(C)
ans =

  -2.00000   1.00000
   1.50000  -0.50000
```

Transponierte

```
>> G=[1 2; 3 4]
G =

       1       2
       3       4

>> G'
ans =

       1       3
       2       4
```

Gleichung mit der Inversen lösen

```
 >> inv(C)*b
ans =

  -4.00000
   5.00000
```

Gauss-Verfahren auf Matrix oder erweiterte Matrix anwenden

```
>> rref(A)
ans =

       1       0
       0       1
       0       0

>> rref(B)
ans =

       1       0       1
       0       1       1

>> E=[6 4 8 9;3 2 1 7;6 5 3 2]
```

```
E =

     6     4     8     9
     3     2     1     7
     6     5     3     2

>> rref(E)
ans =

  1.0e+01  *

   0.10000   0.00000   0.00000   1.00556
   0.00000   0.10000   0.00000  -1.11667
   0.00000   0.00000   0.10000  -0.08333
```

Diverse Gleichungslösungsmethoden

```
 >> inv(C)*b
ans =

  -4.00000
   5.00000

>> C\b
ans =

     -4
      5
```

Was passiert hier?

```
  >> A/A
ans =

   0.71345   0.45029   0.04094
   0.45029   0.29240  -0.06433
   0.04094  -0.06433   0.99415

>> D=A/A
D =

   0.71345   0.45029   0.04094
   0.45029   0.29240  -0.06433
   0.04094  -0.06433   0.99415
```

```
>> D*A
ans =

   3.00000   4.00000
   2.00000   2.00000
  -1.00000   6.00000

>> A./A
ans =

      1      1
      1      1
      1      1
```

Manipulation von Vektoren, Beispiel einer Funktion, Aufsummieren von Teillängen (Matlab_Octave008.RTF)

Vektoren manipulieren: Verlängern, Elemente auswählen, Elemente anders einfügen u.s.w.

```
>> u1=[1 2 3 4 5]
u1 =

      1      2      3      4      5

>> size(u1)
ans =

      1      5

>> size(u1,2)
ans = 5

>> u1(5)
ans = 5

>> u1=[3,6,7,9,3]
u1 =

      3      6      7      9      3

>> u1(size(u1,2))
ans = 3
```

```
>> u1(1)
ans = 3

>> u1(2)
ans = 6
>>
```

Beispiel einer eigenen Funktion (in Octave)

Funktion definieren, die eine Serie von Flächenprodukten rechnet. Die Koordinaten verschiedener Punkte in der Ebene sind durch zwei Vektoren gegeben

```
>> function z=flaechePolygon(x,y)
z=dot([0 x],[y y(1)])-dot([x x(1)],[0 y])
endfunction
```

Vektoren mit x- und y-Koordinaten definieren

```
>> x=[1 2 3 4 5]
x =

   1   2   3   4   5

>> y=[6 7 8 9 0]
y =

   6   7   8   9   0
```

Funktion anwenden

```
>> flaechePolygon(x,y)
z = -30
ans = -30
```

x-Vektor anders definieren

```
>> x=[1 3 4 7 8]
x =

   1   3   4   7   8
```

Funktion anwenden

```
>> flaechePolygon(x,y)
z = -59
ans = -59
```

Eigene Funktionen definieren in Matlab

Klicke im Matlab oben links auf File ➔ New ➔ M-file, dann öffnet sich ein Editor. Eingabe:

```
function z = f(y);  z = y^2 ;
```

Speichere dann die Datei unter dem Namen „f.m“. Die Datei muss wie oben erwähnt denselben Namen haben wie die Funktion. Dann kann man die Funktion unter dem Namen f in Matlab aufrufen, zum Beispiel:

f und g kann man dann wie vordefinierte Matlab-Funktionen benutzen :

```
>> f(5)

ans =

  25

>> [f(0) f(1) f(2) f(3) f(4) f(5)]

ans =

    0     1     4     9    16    25
```

Oder ein Beispiel mit 2 Variablen: Editor aufrufen, Speichere Datei unter dem Namen „g.m“ speichern:

```
function z1 = g(y1,y2)
        z1 = y1^2-exp(y2);
```

Dann in Matlab aufrufen, zum Beispiel:

```
>> g(2,3)

ans =

  -16.0855
```

Aufsummieren von Teillängen (Quadratwurzeln), die aus zwei Vektoren gewonnen werden (alle x- und y-Werte von Punkten der Ebene als zwei Vektoren)

Funktion diff und Operationen auf dieser Funktion studieren

```
>> diff(x)
ans =

      2       1       3       1

>> diff(x).^2
ans =

      4       1       9       1

>> diff(y).^2
ans =

      1       1       1      81

>> diff(x).^2+diff(y).^2
ans =

      5       2      10      82


>> sqrt(diff(x).^2+diff(y).^2)
ans =

  2.23607  1.41421  3.16228  9.05539
```

Funktion sum studieren und damit eine Summen von Teillängen nach Pythagoras berechnen

```
>> sum([1 2 3])
ans = 6

>> sum(sqrt(diff(x).^2+diff(y).^2))
ans = 15.868
```

Spezielle Matrizen, Determinante (Matlab_Octave009.RTF)

Untersuchung diverser Matrix-Typen

Untersuche, wie die folgenden Matrizen aufgebaut sind

```
>> eye(4)

ans =

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

>> hadamard(4)

error: `hadamard' undefined near line 2 column 1

>> help hadamard

help: sorry, `hadamard' is not documented

>> hilb(3)

ans =

  1.00000  0.50000  0.33333
  0.50000  0.33333  0.25000
  0.33333  0.25000  0.20000

>> hilb(4)

ans =

  1.00000  0.50000  0.33333  0.25000
  0.50000  0.33333  0.25000  0.20000
  0.33333  0.25000  0.20000  0.16667
  0.25000  0.20000  0.16667  0.14286

>> magic(4)

ans =

     16      2      3     13
      5     11     10      8
      9      7      6     12
      4     14     15      1

>> ones(4,3)

ans =
```

```
     1     1     1
     1     1     1
     1     1     1
     1     1     1
```

```
>> pascal(4)

ans =

     1     1     1     1
     1     2     3     4
     1     3     6    10
     1     4    10    20
```

```
>> rand(4,5)

ans =

  0.52325  0.41896  0.37745  0.62157  0.31168
  0.48056  0.34537  0.28893  0.43059  0.16517
  0.79917  0.60014  0.89480  0.53227  0.44299
  0.21291  0.08396  0.02582  0.88215  0.00202
```

```
>> vander(4)

ans = 1
```

```
>> vander(6)

ans = 1
```

```
>> vander([1 2 3 4])



ans =

     1     1     1     1
     8     4     2     1
    27     9     3     1
    64    16     4     1
```

Determinante

```
>> v=vander([1 2 3 4])
v =
```

```
     1     1     1     1
     8     4     2     1
    27     9     3     1
    64    16     4     1

>> det(v)
ans = 12.000
>> v1=vander([1 2])
v1 =

     1     1
     2     1

>> det(v1)
ans = -1
```

Differenzieren und Integrieren (Matlab_Octave010.RTF)

(Exkurs)

Problematik

Um symbolisch differenzieren zu können, ist der Befehl „syms“ notwendig. Dieser funktioniert jedoch in Octave nicht:

```
>> syms x;diff(x*cos(x)+2,x,1)
parse error:

>>> syms x;diff(x*cos(x)+2,x,1)
       ^

>> syms x;
parse error:

>>> syms x;
       ^
```

Hingegen kann man numerisch integrieren oder differenzieren:

Integration (Beispiel)

```
>> quad('sin',0,pi)

ans = 2
```

Differentiation (Beispiel, Berechnung eines Differenzenquotienten)

```
>> h=10^-6

h =  1.0000e-06

>> (sin(pi/2+h)-sin(pi/2))/h

ans =  -5.0004e-07

>>
```

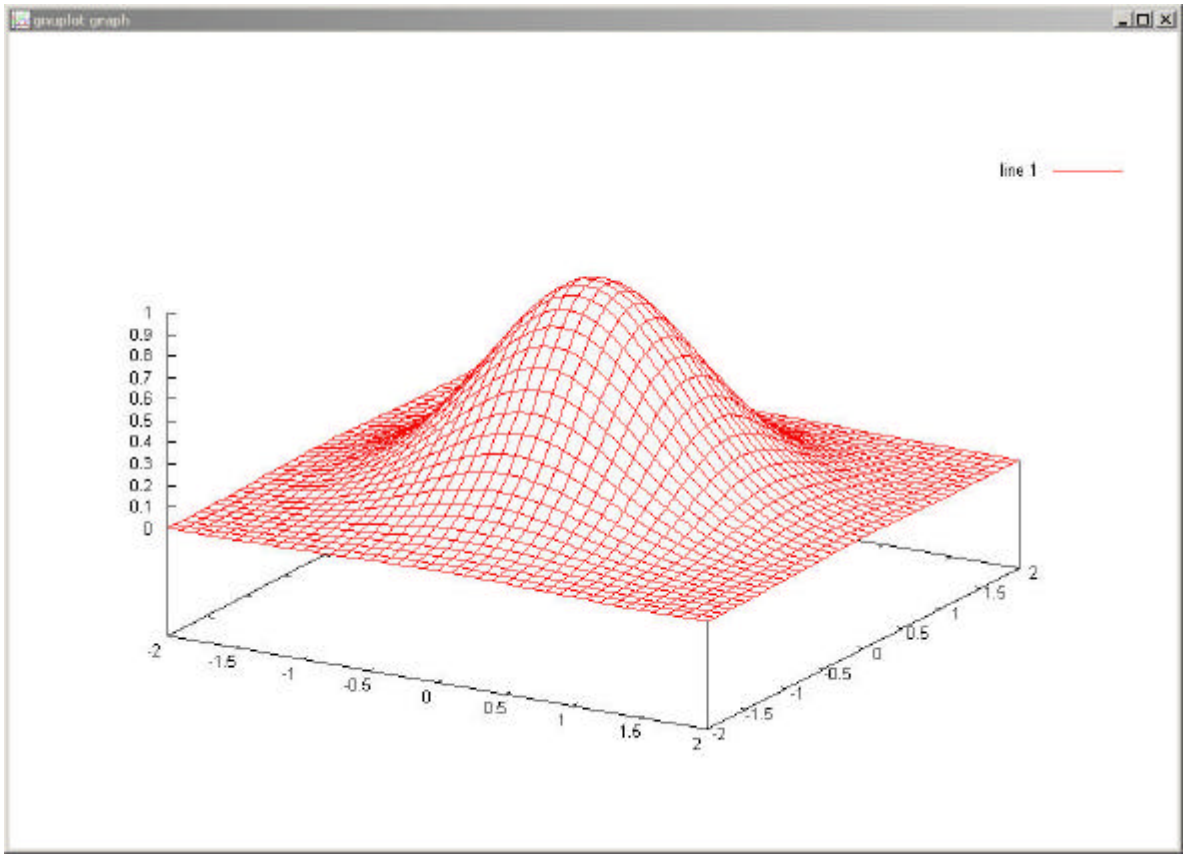
3D-Graphiken (Matlab_Octave011.RTF)

Beispiel 1:

Die Befehle werden hier einzeln abgearbeitet, um bei Octave nicht nach jedem Error den kompletten Code neu eingeben zu müssen.

```
>> x=-2:0.1:2;
>>[x1,y1] = meshgrid(x,x);
>>z=exp(-x1.^2-y1.^2);
>>mesh(x,x,z);
```

Output :



Beispiel 2:

```
r1=0:1/18:1;r2=ones(1,18)-r1(2:19);r=[r1 r2];
>> t=0:pi/9:4*pi;
>> [R,T]=meshgrid(r,t);
>> X=abs(R).*cos(T);Y=abs(R).*sin(T);
>> h=-1:1/18:1;
>> [H,T]=meshgrid(h,t);
>> Z=sign(H).*sqrt(1-R.^2);
>> surface(X,Y,Z),axis equal,view(3)
```

Funktioniert nicht bei Octave

Kontrolliere den Code Zeile für Zeile und überlege, was passiert:!

```
>> r1=0:1/8:1;
>> r1
r1 =

 Columns 1 through 9:
```

```
  0.00000  0.12500  0.25000  0.37500  0.50000  0.62500  0.75000  0.87500  1.00000


>> r21=ones
r21 = 1
>> r21=ones(1,8)
r21 =

       1       1       1       1       1       1       1       1

>> r11=r1(2:9)
r11 =

  0.12500  0.25000  0.37500  0.50000  0.62500  0.75000  0.87500  1.00000

>> r2=r21-r11
r2 =

  0.87500  0.75000  0.62500  0.50000  0.37500  0.25000  0.12500  0.00000

>> r=[r1 r2]
r =

 Columns 1 through 10:

  0.00000  0.12500  0.25000  0.37500  0.50000  0.62500  0.75000  0.87500  1.00000
0.87500

 Columns 11 through 17:

  0.75000  0.62500  0.50000  0.37500  0.25000  0.12500  0.00000

>> t=0:pi/9:2*pi
t =

 Columns 1 through 10:

  0.00000  0.34907  0.69813  1.04720  1.39626  1.74533  2.09440  2.44346  2.79253
3.14159


 Columns 11 through 19:

  3.49066  3.83972  4.18879  4.53786  4.88692  5.23599  5.58505  5.93412  6.28319

>> [R T]=meshgrid(r,t)
R =
```

Columns 1 through 10:

```
  0.00000  0.12500  0.25000  0.37500  0.50000  0.62500  0.75000  0.87500  1.00000
0.87500
  0.00000  0.12500  0.25000  0.37500  0.50000  0.62500  0.75000  0.87500  1.00000
0.87500
  0.00000  0.12500  0.25000  0.37500  0.50000  0.62500  0.75000  0.87500  1.00000
0.87500
  0.00000  0.12500  0.25000  0.37500  0.50000  0.62500  0.75000  0.87500  1.00000
0.87500
  0.00000  0.12500  0.25000  0.37500  0.50000  0.62500  0.75000  0.87500  1.00000
0.87500
  0.00000  0.12500  0.25000  0.37500  0.50000  0.62500  0.75000  0.87500  1.00000
0.87500
  0.00000  0.12500  0.25000  0.37500  0.50000  0.62500  0.75000  0.87500  1.00000
0.87500
  0.00000  0.12500  0.25000  0.37500  0.50000  0.62500  0.75000  0.87500  1.00000
0.87500
  0.00000  0.12500  0.25000  0.37500  0.50000  0.62500  0.75000  0.87500  1.00000
0.87500
  0.00000  0.12500  0.25000  0.37500  0.50000  0.62500  0.75000  0.87500  1.00000
0.87500
  0.00000  0.12500  0.25000  0.37500  0.50000  0.62500  0.75000  0.87500  1.00000
0.87500
  0.00000  0.12500  0.25000  0.37500  0.50000  0.62500  0.75000  0.87500  1.00000
0.87500
  0.00000  0.12500  0.25000  0.37500  0.50000  0.62500  0.75000  0.87500  1.00000
0.87500
  0.00000  0.12500  0.25000  0.37500  0.50000  0.62500  0.75000  0.87500  1.00000
0.87500
  0.00000  0.12500  0.25000  0.37500  0.50000  0.62500  0.75000  0.87500  1.00000
0.87500
  0.00000  0.12500  0.25000  0.37500  0.50000  0.62500  0.75000  0.87500  1.00000
0.87500
  0.00000  0.12500  0.25000  0.37500  0.50000  0.62500  0.75000  0.87500  1.00000
0.87500
  0.00000  0.12500  0.25000  0.37500  0.50000  0.62500  0.75000  0.87500  1.00000
0.87500
  0.00000  0.12500  0.25000  0.37500  0.50000  0.62500  0.75000  0.87500  1.00000
0.87500
```

Columns 11 through 17:

```
  0.75000  0.62500  0.50000  0.37500  0.25000  0.12500  0.00000
  0.75000  0.62500  0.50000  0.37500  0.25000  0.12500  0.00000
  0.75000  0.62500  0.50000  0.37500  0.25000  0.12500  0.00000
  0.75000  0.62500  0.50000  0.37500  0.25000  0.12500  0.00000
```

```
   0.75000   0.62500   0.50000   0.37500   0.25000   0.12500   0.00000
   0.75000   0.62500   0.50000   0.37500   0.25000   0.12500   0.00000
   0.75000   0.62500   0.50000   0.37500   0.25000   0.12500   0.00000
   0.75000   0.62500   0.50000   0.37500   0.25000   0.12500   0.00000
   0.75000   0.62500   0.50000   0.37500   0.25000   0.12500   0.00000
   0.75000   0.62500   0.50000   0.37500   0.25000   0.12500   0.00000
   0.75000   0.62500   0.50000   0.37500   0.25000   0.12500   0.00000
   0.75000   0.62500   0.50000   0.37500   0.25000   0.12500   0.00000
   0.75000   0.62500   0.50000   0.37500   0.25000   0.12500   0.00000
   0.75000   0.62500   0.50000   0.37500   0.25000   0.12500   0.00000
   0.75000   0.62500   0.50000   0.37500   0.25000   0.12500   0.00000
   0.75000   0.62500   0.50000   0.37500   0.25000   0.12500   0.00000
   0.75000   0.62500   0.50000   0.37500   0.25000   0.12500   0.00000
   0.75000   0.62500   0.50000   0.37500   0.25000   0.12500   0.00000
   0.75000   0.62500   0.50000   0.37500   0.25000   0.12500   0.00000

T =

 Columns 1 through 10:

   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000
0.00000
   0.34907   0.34907   0.34907   0.34907   0.34907   0.34907   0.34907   0.34907   0.34907
0.34907
   0.69813   0.69813   0.69813   0.69813   0.69813   0.69813   0.69813   0.69813   0.69813
0.69813
   1.04720   1.04720   1.04720   1.04720   1.04720   1.04720   1.04720   1.04720   1.04720
1.04720
   1.39626   1.39626   1.39626   1.39626   1.39626   1.39626   1.39626   1.39626   1.39626
1.39626
   1.74533   1.74533   1.74533   1.74533   1.74533   1.74533   1.74533   1.74533   1.74533
1.74533
   2.09440   2.09440   2.09440   2.09440   2.09440   2.09440   2.09440   2.09440   2.09440
2.09440
   2.44346   2.44346   2.44346   2.44346   2.44346   2.44346   2.44346   2.44346   2.44346
2.44346
   2.79253   2.79253   2.79253   2.79253   2.79253   2.79253   2.79253   2.79253   2.79253
2.79253
   3.14159   3.14159   3.14159   3.14159   3.14159   3.14159   3.14159   3.14159   3.14159
3.14159
   3.49066   3.49066   3.49066   3.49066   3.49066   3.49066   3.49066   3.49066   3.49066
3.49066
   3.83972   3.83972   3.83972   3.83972   3.83972   3.83972   3.83972   3.83972   3.83972
3.83972
   4.18879   4.18879   4.18879   4.18879   4.18879   4.18879   4.18879   4.18879   4.18879
4.18879
   4.53786   4.53786   4.53786   4.53786   4.53786   4.53786   4.53786   4.53786   4.53786
4.53786
```

```
  4.88692  4.88692  4.88692  4.88692  4.88692  4.88692  4.88692  4.88692  4.88692
4.88692
  5.23599  5.23599  5.23599  5.23599  5.23599  5.23599  5.23599  5.23599  5.23599
5.23599
  5.58505  5.58505  5.58505  5.58505  5.58505  5.58505  5.58505  5.58505  5.58505
5.58505
  5.93412  5.93412  5.93412  5.93412  5.93412  5.93412  5.93412  5.93412  5.93412
5.93412
  6.28319  6.28319  6.28319  6.28319  6.28319  6.28319  6.28319  6.28319  6.28319
6.28319

 Columns 11 through 17:

  0.00000  0.00000  0.00000  0.00000  0.00000  0.00000  0.00000
  0.34907  0.34907  0.34907  0.34907  0.34907  0.34907  0.34907
  0.69813  0.69813  0.69813  0.69813  0.69813  0.69813  0.69813
  1.04720  1.04720  1.04720  1.04720  1.04720  1.04720  1.04720
  1.39626  1.39626  1.39626  1.39626  1.39626  1.39626  1.39626
  1.74533  1.74533  1.74533  1.74533  1.74533  1.74533  1.74533
  2.09440  2.09440  2.09440  2.09440  2.09440  2.09440  2.09440
  2.44346  2.44346  2.44346  2.44346  2.44346  2.44346  2.44346
  2.79253  2.79253  2.79253  2.79253  2.79253  2.79253  2.79253
  3.14159  3.14159  3.14159  3.14159  3.14159  3.14159  3.14159
  3.49066  3.49066  3.49066  3.49066  3.49066  3.49066  3.49066
  3.83972  3.83972  3.83972  3.83972  3.83972  3.83972  3.83972
  4.18879  4.18879  4.18879  4.18879  4.18879  4.18879  4.18879
  4.53786  4.53786  4.53786  4.53786  4.53786  4.53786  4.53786
  4.88692  4.88692  4.88692  4.88692  4.88692  4.88692  4.88692
  5.23599  5.23599  5.23599  5.23599  5.23599  5.23599  5.23599
  5.58505  5.58505  5.58505  5.58505  5.58505  5.58505  5.58505
  5.93412  5.93412  5.93412  5.93412  5.93412  5.93412  5.93412
  6.28319  6.28319  6.28319  6.28319  6.28319  6.28319  6.28319


>> X=abs(R).*cos(T)
X =

 Columns 1 through 9:

   0.00000   0.12500   0.25000   0.37500   0.50000   0.62500   0.75000   0.87500
1.00000
   0.00000   0.11746   0.23492   0.35238   0.46985   0.58731   0.70477   0.82223
0.93969
   0.00000   0.09576   0.19151   0.28727   0.38302   0.47878   0.57453   0.67029
0.76604
   0.00000   0.06250   0.12500   0.18750   0.25000   0.31250   0.37500   0.43750
0.50000
```

```
   0.00000   0.02171   0.04341   0.06512   0.08682   0.10853   0.13024   0.15194
0.17365
  -0.00000  -0.02171  -0.04341  -0.06512  -0.08682  -0.10853  -0.13024  -0.15194  -
0.17365
  -0.00000  -0.06250  -0.12500  -0.18750  -0.25000  -0.31250  -0.37500  -0.43750  -
0.50000
  -0.00000  -0.09576  -0.19151  -0.28727  -0.38302  -0.47878  -0.57453  -0.67029  -
0.76604
  -0.00000  -0.11746  -0.23492  -0.35238  -0.46985  -0.58731  -0.70477  -0.82223  -
0.93969
  -0.00000  -0.12500  -0.25000  -0.37500  -0.50000  -0.62500  -0.75000  -0.87500  -
1.00000
  -0.00000  -0.11746  -0.23492  -0.35238  -0.46985  -0.58731  -0.70477  -0.82223  -
0.93969
  -0.00000  -0.09576  -0.19151  -0.28727  -0.38302  -0.47878  -0.57453  -0.67029  -
0.76604
  -0.00000  -0.06250  -0.12500  -0.18750  -0.25000  -0.31250  -0.37500  -0.43750  -
0.50000
  -0.00000  -0.02171  -0.04341  -0.06512  -0.08682  -0.10853  -0.13024  -0.15194  -
0.17365
   0.00000   0.02171   0.04341   0.06512   0.08682   0.10853   0.13024   0.15194
0.17365
   0.00000   0.06250   0.12500   0.18750   0.25000   0.31250   0.37500   0.43750
0.50000
   0.00000   0.09576   0.19151   0.28727   0.38302   0.47878   0.57453   0.67029
0.76604
   0.00000   0.11746   0.23492   0.35238   0.46985   0.58731   0.70477   0.82223
0.93969
   0.00000   0.12500   0.25000   0.37500   0.50000   0.62500   0.75000   0.87500
1.00000
```

Columns 10 through 17:

```
   0.87500   0.75000   0.62500   0.50000   0.37500   0.25000   0.12500   0.00000
   0.82223   0.70477   0.58731   0.46985   0.35238   0.23492   0.11746   0.00000
   0.67029   0.57453   0.47878   0.38302   0.28727   0.19151   0.09576   0.00000
   0.43750   0.37500   0.31250   0.25000   0.18750   0.12500   0.06250   0.00000
   0.15194   0.13024   0.10853   0.08682   0.06512   0.04341   0.02171   0.00000
  -0.15194  -0.13024  -0.10853  -0.08682  -0.06512  -0.04341  -0.02171  -0.00000
  -0.43750  -0.37500  -0.31250  -0.25000  -0.18750  -0.12500  -0.06250  -0.00000
  -0.67029  -0.57453  -0.47878  -0.38302  -0.28727  -0.19151  -0.09576  -0.00000
  -0.82223  -0.70477  -0.58731  -0.46985  -0.35238  -0.23492  -0.11746  -0.00000
  -0.87500  -0.75000  -0.62500  -0.50000  -0.37500  -0.25000  -0.12500  -0.00000
  -0.82223  -0.70477  -0.58731  -0.46985  -0.35238  -0.23492  -0.11746  -0.00000
  -0.67029  -0.57453  -0.47878  -0.38302  -0.28727  -0.19151  -0.09576  -0.00000
  -0.43750  -0.37500  -0.31250  -0.25000  -0.18750  -0.12500  -0.06250  -0.00000
  -0.15194  -0.13024  -0.10853  -0.08682  -0.06512  -0.04341  -0.02171  -0.00000
   0.15194   0.13024   0.10853   0.08682   0.06512   0.04341   0.02171   0.00000
```

```
   0.43750   0.37500   0.31250   0.25000   0.18750   0.12500   0.06250   0.00000
   0.67029   0.57453   0.47878   0.38302   0.28727   0.19151   0.09576   0.00000
   0.82223   0.70477   0.58731   0.46985   0.35238   0.23492   0.11746   0.00000
   0.87500   0.75000   0.62500   0.50000   0.37500   0.25000   0.12500   0.00000

>>
>> Y=abs(R).*sin(T)
Y =

 Columns 1 through 9:

   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000
0.00000
   0.00000   0.04275   0.08551   0.12826   0.17101   0.21376   0.25652   0.29927
0.34202
   0.00000   0.08035   0.16070   0.24105   0.32139   0.40174   0.48209   0.56244
0.64279
   0.00000   0.10825   0.21651   0.32476   0.43301   0.54127   0.64952   0.75777
0.86603
   0.00000   0.12310   0.24620   0.36930   0.49240   0.61550   0.73861   0.86171
0.98481
   0.00000   0.12310   0.24620   0.36930   0.49240   0.61550   0.73861   0.86171
0.98481
   0.00000   0.10825   0.21651   0.32476   0.43301   0.54127   0.64952   0.75777
0.86603
   0.00000   0.08035   0.16070   0.24105   0.32139   0.40174   0.48209   0.56244
0.64279
   0.00000   0.04275   0.08551   0.12826   0.17101   0.21376   0.25652   0.29927
0.34202
   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000
0.00000
  -0.00000  -0.04275  -0.08551  -0.12826  -0.17101  -0.21376  -0.25652  -0.29927  -
0.34202
  -0.00000  -0.08035  -0.16070  -0.24105  -0.32139  -0.40174  -0.48209  -0.56244  -
0.64279
  -0.00000  -0.10825  -0.21651  -0.32476  -0.43301  -0.54127  -0.64952  -0.75777  -
0.86603
  -0.00000  -0.12310  -0.24620  -0.36930  -0.49240  -0.61550  -0.73861  -0.86171  -
0.98481
  -0.00000  -0.12310  -0.24620  -0.36930  -0.49240  -0.61550  -0.73861  -0.86171  -
0.98481
  -0.00000  -0.10825  -0.21651  -0.32476  -0.43301  -0.54127  -0.64952  -0.75777  -
0.86603
  -0.00000  -0.08035  -0.16070  -0.24105  -0.32139  -0.40174  -0.48209  -0.56244  -
0.64279
  -0.00000  -0.04275  -0.08551  -0.12826  -0.17101  -0.21376  -0.25652  -0.29927  -
0.34202
```

-0.00000 -0.00000 -0.00000 -0.00000 -0.00000 -0.00000 -0.00000 -0.00000 -0.00000

Columns 10 through 17:

0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
0.29927	0.25652	0.21376	0.17101	0.12826	0.08551	0.04275	0.00000
0.56244	0.48209	0.40174	0.32139	0.24105	0.16070	0.08035	0.00000
0.75777	0.64952	0.54127	0.43301	0.32476	0.21651	0.10825	0.00000
0.86171	0.73861	0.61550	0.49240	0.36930	0.24620	0.12310	0.00000
0.86171	0.73861	0.61550	0.49240	0.36930	0.24620	0.12310	0.00000
0.75777	0.64952	0.54127	0.43301	0.32476	0.21651	0.10825	0.00000
0.56244	0.48209	0.40174	0.32139	0.24105	0.16070	0.08035	0.00000
0.29927	0.25652	0.21376	0.17101	0.12826	0.08551	0.04275	0.00000
0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
-0.29927	-0.25652	-0.21376	-0.17101	-0.12826	-0.08551	-0.04275	-0.00000
-0.56244	-0.48209	-0.40174	-0.32139	-0.24105	-0.16070	-0.08035	-0.00000
-0.75777	-0.64952	-0.54127	-0.43301	-0.32476	-0.21651	-0.10825	-0.00000
-0.86171	-0.73861	-0.61550	-0.49240	-0.36930	-0.24620	-0.12310	-0.00000
-0.86171	-0.73861	-0.61550	-0.49240	-0.36930	-0.24620	-0.12310	-0.00000
-0.75777	-0.64952	-0.54127	-0.43301	-0.32476	-0.21651	-0.10825	-0.00000
-0.56244	-0.48209	-0.40174	-0.32139	-0.24105	-0.16070	-0.08035	-0.00000
-0.29927	-0.25652	-0.21376	-0.17101	-0.12826	-0.08551	-0.04275	-0.00000
-0.00000	-0.00000	-0.00000	-0.00000	-0.00000	-0.00000	-0.00000	-0.00000

```
>> h=-1:1/9:1
h =
```

Columns 1 through 9:

-1.00000 -0.88889 -0.77778 -0.66667 -0.55556 -0.44444 -0.33333 -0.22222 -0.11111

Columns 10 through 18:

-0.00000 0.11111 0.22222 0.33333 0.44444 0.55556 0.66667 0.77778 0.88889

Column 19:

1.00000

```
>> [H T]=meshgrid(h,t)
H =
```

Columns 1 through 9:

-1.00000 -0.88889 -0.77778 -0.66667 -0.55556 -0.44444 -0.33333 -0.22222 -
0.11111
-1.00000 -0.88889 -0.77778 -0.66667 -0.55556 -0.44444 -0.33333 -0.22222 -
0.11111
-1.00000 -0.88889 -0.77778 -0.66667 -0.55556 -0.44444 -0.33333 -0.22222 -
0.11111
-1.00000 -0.88889 -0.77778 -0.66667 -0.55556 -0.44444 -0.33333 -0.22222 -
0.11111
-1.00000 -0.88889 -0.77778 -0.66667 -0.55556 -0.44444 -0.33333 -0.22222 -
0.11111
-1.00000 -0.88889 -0.77778 -0.66667 -0.55556 -0.44444 -0.33333 -0.22222 -
0.11111
-1.00000 -0.88889 -0.77778 -0.66667 -0.55556 -0.44444 -0.33333 -0.22222 -
0.11111
-1.00000 -0.88889 -0.77778 -0.66667 -0.55556 -0.44444 -0.33333 -0.22222 -
0.11111
-1.00000 -0.88889 -0.77778 -0.66667 -0.55556 -0.44444 -0.33333 -0.22222 -
0.11111
-1.00000 -0.88889 -0.77778 -0.66667 -0.55556 -0.44444 -0.33333 -0.22222 -
0.11111
-1.00000 -0.88889 -0.77778 -0.66667 -0.55556 -0.44444 -0.33333 -0.22222 -
0.11111
-1.00000 -0.88889 -0.77778 -0.66667 -0.55556 -0.44444 -0.33333 -0.22222 -
0.11111
-1.00000 -0.88889 -0.77778 -0.66667 -0.55556 -0.44444 -0.33333 -0.22222 -
0.11111
-1.00000 -0.88889 -0.77778 -0.66667 -0.55556 -0.44444 -0.33333 -0.22222 -
0.11111
-1.00000 -0.88889 -0.77778 -0.66667 -0.55556 -0.44444 -0.33333 -0.22222 -
0.11111
-1.00000 -0.88889 -0.77778 -0.66667 -0.55556 -0.44444 -0.33333 -0.22222 -
0.11111
-1.00000 -0.88889 -0.77778 -0.66667 -0.55556 -0.44444 -0.33333 -0.22222 -
0.11111
-1.00000 -0.88889 -0.77778 -0.66667 -0.55556 -0.44444 -0.33333 -0.22222 -
0.11111
-1.00000 -0.88889 -0.77778 -0.66667 -0.55556 -0.44444 -0.33333 -0.22222 -
0.11111

Columns 10 through 18:

-0.00000   0.11111   0.22222   0.33333   0.44444   0.55556   0.66667   0.77778
0.88889
-0.00000   0.11111   0.22222   0.33333   0.44444   0.55556   0.66667   0.77778
0.88889
-0.00000   0.11111   0.22222   0.33333   0.44444   0.55556   0.66667   0.77778
0.88889

-0.00000 0.11111 0.22222 0.33333 0.44444 0.55556 0.66667 0.77778
0.88889
-0.00000 0.11111 0.22222 0.33333 0.44444 0.55556 0.66667 0.77778
0.88889
-0.00000 0.11111 0.22222 0.33333 0.44444 0.55556 0.66667 0.77778
0.88889
-0.00000 0.11111 0.22222 0.33333 0.44444 0.55556 0.66667 0.77778
0.88889
-0.00000 0.11111 0.22222 0.33333 0.44444 0.55556 0.66667 0.77778
0.88889
-0.00000 0.11111 0.22222 0.33333 0.44444 0.55556 0.66667 0.77778
0.88889
-0.00000 0.11111 0.22222 0.33333 0.44444 0.55556 0.66667 0.77778
0.88889
-0.00000 0.11111 0.22222 0.33333 0.44444 0.55556 0.66667 0.77778
0.88889
-0.00000 0.11111 0.22222 0.33333 0.44444 0.55556 0.66667 0.77778
0.88889
-0.00000 0.11111 0.22222 0.33333 0.44444 0.55556 0.66667 0.77778
0.88889
-0.00000 0.11111 0.22222 0.33333 0.44444 0.55556 0.66667 0.77778
0.88889
-0.00000 0.11111 0.22222 0.33333 0.44444 0.55556 0.66667 0.77778
0.88889
-0.00000 0.11111 0.22222 0.33333 0.44444 0.55556 0.66667 0.77778
0.88889
-0.00000 0.11111 0.22222 0.33333 0.44444 0.55556 0.66667 0.77778
0.88889
-0.00000 0.11111 0.22222 0.33333 0.44444 0.55556 0.66667 0.77778
0.88889
-0.00000 0.11111 0.22222 0.33333 0.44444 0.55556 0.66667 0.77778
0.88889

Column 19:

1.00000
1.00000
1.00000
1.00000
1.00000
1.00000
1.00000
1.00000
1.00000
1.00000
1.00000
1.00000
1.00000

1.00000
1.00000
1.00000
1.00000
1.00000
1.00000

T =

Columns 1 through 10:

0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
0.34907 0.34907 0.34907 0.34907 0.34907 0.34907 0.34907 0.34907 0.34907 0.34907
0.69813 0.69813 0.69813 0.69813 0.69813 0.69813 0.69813 0.69813 0.69813 0.69813
1.04720 1.04720 1.04720 1.04720 1.04720 1.04720 1.04720 1.04720 1.04720 1.04720
1.39626 1.39626 1.39626 1.39626 1.39626 1.39626 1.39626 1.39626 1.39626 1.39626
1.74533 1.74533 1.74533 1.74533 1.74533 1.74533 1.74533 1.74533 1.74533 1.74533
2.09440 2.09440 2.09440 2.09440 2.09440 2.09440 2.09440 2.09440 2.09440 2.09440
2.44346 2.44346 2.44346 2.44346 2.44346 2.44346 2.44346 2.44346 2.44346 2.44346
2.79253 2.79253 2.79253 2.79253 2.79253 2.79253 2.79253 2.79253 2.79253 2.79253
3.14159 3.14159 3.14159 3.14159 3.14159 3.14159 3.14159 3.14159 3.14159 3.14159
3.49066 3.49066 3.49066 3.49066 3.49066 3.49066 3.49066 3.49066 3.49066 3.49066
3.83972 3.83972 3.83972 3.83972 3.83972 3.83972 3.83972 3.83972 3.83972 3.83972
4.18879 4.18879 4.18879 4.18879 4.18879 4.18879 4.18879 4.18879 4.18879 4.18879
4.53786 4.53786 4.53786 4.53786 4.53786 4.53786 4.53786 4.53786 4.53786 4.53786
4.88692 4.88692 4.88692 4.88692 4.88692 4.88692 4.88692 4.88692 4.88692 4.88692
5.23599 5.23599 5.23599 5.23599 5.23599 5.23599 5.23599 5.23599 5.23599 5.23599
5.58505 5.58505 5.58505 5.58505 5.58505 5.58505 5.58505 5.58505 5.58505 5.58505
5.93412 5.93412 5.93412 5.93412 5.93412 5.93412 5.93412 5.93412 5.93412 5.93412

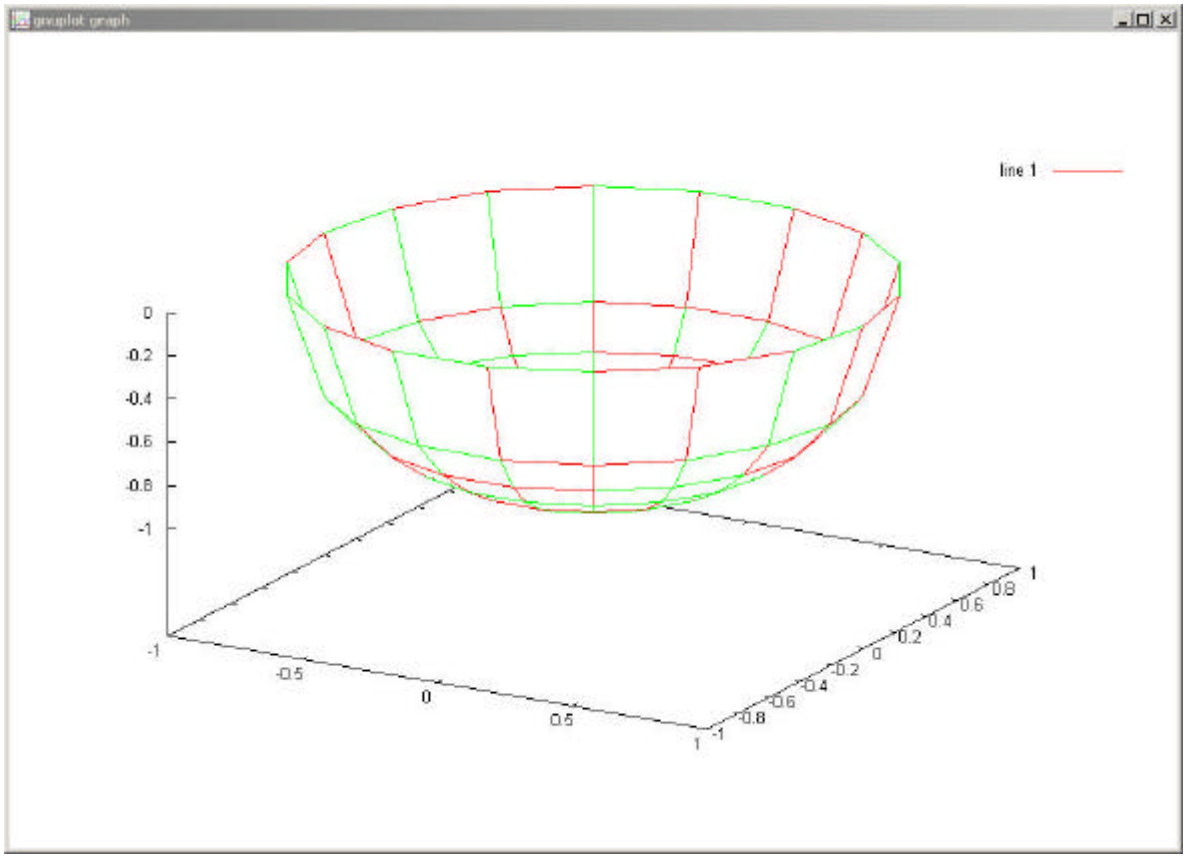
```
  6.28319  6.28319  6.28319  6.28319  6.28319  6.28319  6.28319  6.28319  6.28319
6.28319

 Columns 11 through 19:

  0.00000  0.00000  0.00000  0.00000  0.00000  0.00000  0.00000  0.00000  0.00000
  0.34907  0.34907  0.34907  0.34907  0.34907  0.34907  0.34907  0.34907  0.34907
  0.69813  0.69813  0.69813  0.69813  0.69813  0.69813  0.69813  0.69813  0.69813
  1.04720  1.04720  1.04720  1.04720  1.04720  1.04720  1.04720  1.04720  1.04720
  1.39626  1.39626  1.39626  1.39626  1.39626  1.39626  1.39626  1.39626  1.39626
  1.74533  1.74533  1.74533  1.74533  1.74533  1.74533  1.74533  1.74533  1.74533
  2.09440  2.09440  2.09440  2.09440  2.09440  2.09440  2.09440  2.09440  2.09440
  2.44346  2.44346  2.44346  2.44346  2.44346  2.44346  2.44346  2.44346  2.44346
  2.79253  2.79253  2.79253  2.79253  2.79253  2.79253  2.79253  2.79253  2.79253
  3.14159  3.14159  3.14159  3.14159  3.14159  3.14159  3.14159  3.14159  3.14159
  3.49066  3.49066  3.49066  3.49066  3.49066  3.49066  3.49066  3.49066  3.49066
  3.83972  3.83972  3.83972  3.83972  3.83972  3.83972  3.83972  3.83972  3.83972
  4.18879  4.18879  4.18879  4.18879  4.18879  4.18879  4.18879  4.18879  4.18879
  4.53786  4.53786  4.53786  4.53786  4.53786  4.53786  4.53786  4.53786  4.53786
  4.88692  4.88692  4.88692  4.88692  4.88692  4.88692  4.88692  4.88692  4.88692
  5.23599  5.23599  5.23599  5.23599  5.23599  5.23599  5.23599  5.23599  5.23599
  5.58505  5.58505  5.58505  5.58505  5.58505  5.58505  5.58505  5.58505  5.58505
  5.93412  5.93412  5.93412  5.93412  5.93412  5.93412  5.93412  5.93412  5.93412
  6.28319  6.28319  6.28319  6.28319  6.28319  6.28319  6.28319  6.28319  6.28319

Z=sign(H).*sqrt(1-R.^2);   funktioniert nicht in Oktave

>> Z=sign(H)*sqrt(1-R.^2)
Z =

 Columns 1 through 9:

  -1.00000  -0.99216  -0.96825  -0.92702  -0.86603  -0.78062  -0.66144  -0.48412
0.00000
  -1.00000  -0.99216  -0.96825  -0.92702  -0.86603  -0.78062  -0.66144  -0.48412
0.00000
  -1.00000  -0.99216  -0.96825  -0.92702  -0.86603  -0.78062  -0.66144  -0.48412
0.00000
  -1.00000  -0.99216  -0.96825  -0.92702  -0.86603  -0.78062  -0.66144  -0.48412
0.00000
  -1.00000  -0.99216  -0.96825  -0.92702  -0.86603  -0.78062  -0.66144  -0.48412
0.00000
  -1.00000  -0.99216  -0.96825  -0.92702  -0.86603  -0.78062  -0.66144  -0.48412
0.00000
  -1.00000  -0.99216  -0.96825  -0.92702  -0.86603  -0.78062  -0.66144  -0.48412
0.00000
  -1.00000  -0.99216  -0.96825  -0.92702  -0.86603  -0.78062  -0.66144  -0.48412
0.00000
```

```
  -1.00000  -0.99216  -0.96825  -0.92702  -0.86603  -0.78062  -0.66144  -0.48412
0.00000
  -1.00000  -0.99216  -0.96825  -0.92702  -0.86603  -0.78062  -0.66144  -0.48412
0.00000
  -1.00000  -0.99216  -0.96825  -0.92702  -0.86603  -0.78062  -0.66144  -0.48412
0.00000
  -1.00000  -0.99216  -0.96825  -0.92702  -0.86603  -0.78062  -0.66144  -0.48412
0.00000
  -1.00000  -0.99216  -0.96825  -0.92702  -0.86603  -0.78062  -0.66144  -0.48412
0.00000
  -1.00000  -0.99216  -0.96825  -0.92702  -0.86603  -0.78062  -0.66144  -0.48412
0.00000
  -1.00000  -0.99216  -0.96825  -0.92702  -0.86603  -0.78062  -0.66144  -0.48412
0.00000
  -1.00000  -0.99216  -0.96825  -0.92702  -0.86603  -0.78062  -0.66144  -0.48412
0.00000
  -1.00000  -0.99216  -0.96825  -0.92702  -0.86603  -0.78062  -0.66144  -0.48412
0.00000
  -1.00000  -0.99216  -0.96825  -0.92702  -0.86603  -0.78062  -0.66144  -0.48412
0.00000
  -1.00000  -0.99216  -0.96825  -0.92702  -0.86603  -0.78062  -0.66144  -0.48412
0.00000

 Columns 10 through 17:

  -0.48412  -0.66144  -0.78062  -0.86603  -0.92702  -0.96825  -0.99216  -1.00000
  -0.48412  -0.66144  -0.78062  -0.86603  -0.92702  -0.96825  -0.99216  -1.00000
  -0.48412  -0.66144  -0.78062  -0.86603  -0.92702  -0.96825  -0.99216  -1.00000
  -0.48412  -0.66144  -0.78062  -0.86603  -0.92702  -0.96825  -0.99216  -1.00000
  -0.48412  -0.66144  -0.78062  -0.86603  -0.92702  -0.96825  -0.99216  -1.00000
  -0.48412  -0.66144  -0.78062  -0.86603  -0.92702  -0.96825  -0.99216  -1.00000
  -0.48412  -0.66144  -0.78062  -0.86603  -0.92702  -0.96825  -0.99216  -1.00000
  -0.48412  -0.66144  -0.78062  -0.86603  -0.92702  -0.96825  -0.99216  -1.00000
  -0.48412  -0.66144  -0.78062  -0.86603  -0.92702  -0.96825  -0.99216  -1.00000
  -0.48412  -0.66144  -0.78062  -0.86603  -0.92702  -0.96825  -0.99216  -1.00000
  -0.48412  -0.66144  -0.78062  -0.86603  -0.92702  -0.96825  -0.99216  -1.00000
  -0.48412  -0.66144  -0.78062  -0.86603  -0.92702  -0.96825  -0.99216  -1.00000
  -0.48412  -0.66144  -0.78062  -0.86603  -0.92702  -0.96825  -0.99216  -1.00000
  -0.48412  -0.66144  -0.78062  -0.86603  -0.92702  -0.96825  -0.99216  -1.00000
  -0.48412  -0.66144  -0.78062  -0.86603  -0.92702  -0.96825  -0.99216  -1.00000
  -0.48412  -0.66144  -0.78062  -0.86603  -0.92702  -0.96825  -0.99216  -1.00000
  -0.48412  -0.66144  -0.78062  -0.86603  -0.92702  -0.96825  -0.99216  -1.00000
  -0.48412  -0.66144  -0.78062  -0.86603  -0.92702  -0.96825  -0.99216  -1.00000
  -0.48412  -0.66144  -0.78062  -0.86603  -0.92702  -0.96825  -0.99216  -1.00000

>>
surface(X,Y,Z),axis equal,view(3)
parse error:
```

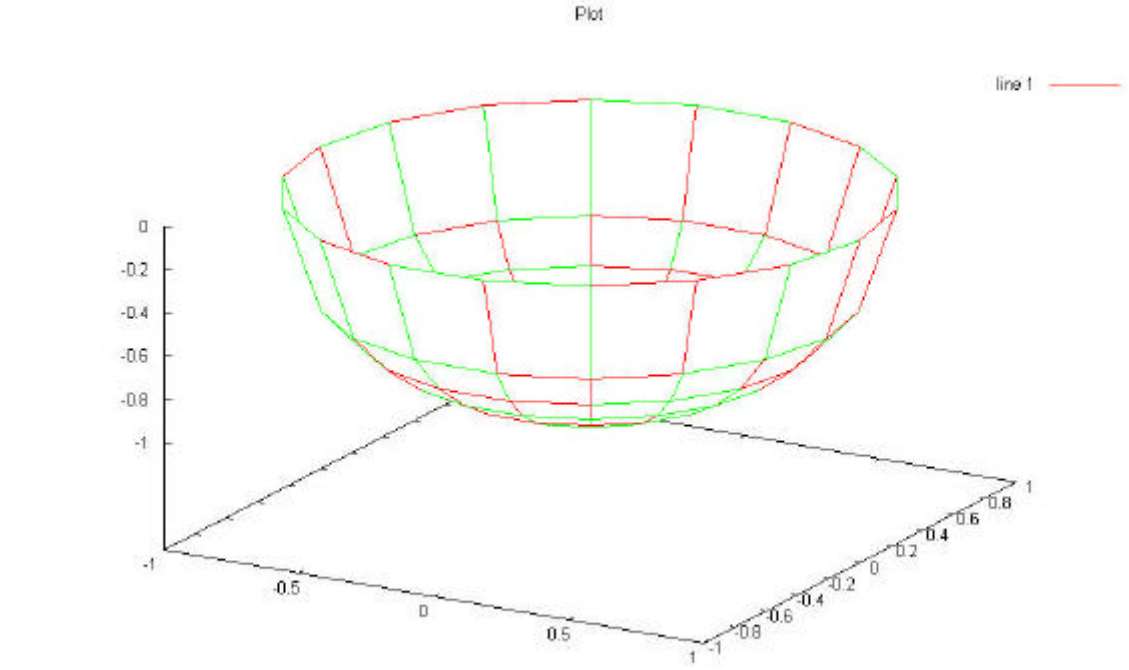
Funktioniert nicht

Folgender Code funktioniert:

>> mesh(X,Y,Z)



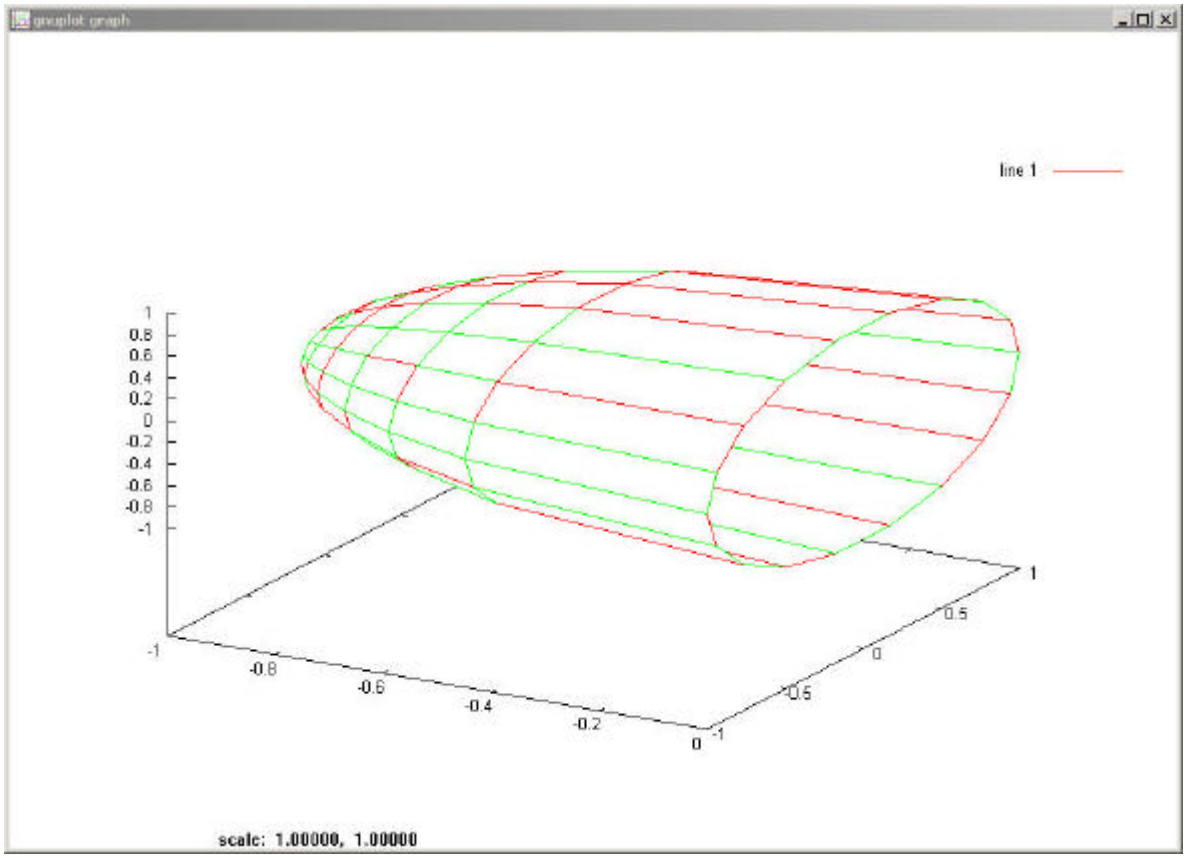
>> title('Plot')
>> replot



view: 60.0000, 30.0000 scale: 1.00000, 1.00000

Beispiel 3:

```
r1=0:1/8:1;
r21=ones(1,8);
r2=r21-r1(2:9);
r=[r1 r2];
 t=pi:pi/9:3*pi;
[R,T]=meshgrid(r,t);
X=abs(R).*cos(T);Y=abs(R).*sin(T);
h=-1:1/9:1;
[H,T]=meshgrid(h,t);
Z=sign(H).*sqrt(1-R.^2);
mesh(Z,X,Y)
```



Wie weiter?

Wird bei Gelegenheit fortgesetzt.
Inzwischen: Literatur konsultieren, selbst ausproblieren….

Literatur

Eine gute Anleitung für Octave (oft in Form gelöster Übungen) findet man unter:

http://homepages.uni-tuebingen.de/juergen.schweizer/Octave.html