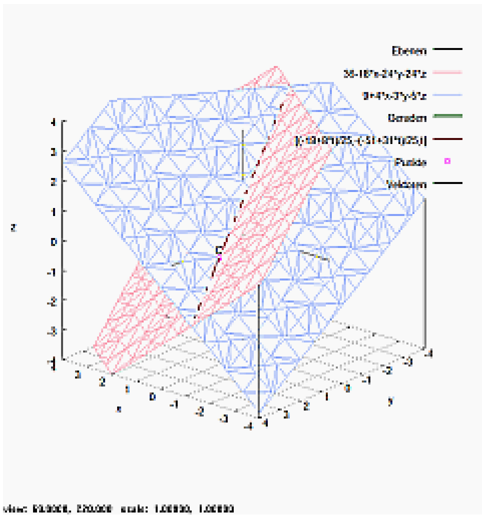
wxMaxima

angeom.mac

Analytische Geometrie mit Maxima CAS

Hans W. Hofmann
hw@lemitec.de



Computer Algebra System

Funktionen des Paketes angeom.mac

Maxima-Befehle und -Funktionen

Beachten Sie dass MAXIMA zwischen Groß- und kleinschreibung unterscheidet!
Zuweisung an eine Variable.
Einen einfachen Ausdruck oder Wert speichern Sie mit dem
Zuweisungsoperator :
und funktionale Ausdrücke über := Funktionen haben ein Argument wie z.B. f(x)

```
(%i2) Wert:5.5; f(x):=2*x+1;
```

(%o2) 5.5

(%o3) $f(x):=2x+1$

```
(%i4) f(1); Wert*f(-1);
```

(%o4) 3

(%o5) -5.5

Wie Sie feststellen, werden Maxima-Eingaben mit einer Zeilennummer versehen.
Eine Zeilenangabe beginnt mit einem Prozentzeichen % gefolgt von
%i1 - erste Eingabe wie i Input und schliesst durch ein Semikolon ab.
Ausgaben von Maxima beginnen ebenfalls mit einem %-Zeichen
%o1 - erste Ausgabe wie o Output

```
(%i6) %o2-1; %o2-5$
```

(%o6) 4.5

Über die Zeilennummer kann auf Ausgaben zugegriffen werden. Hier ziehe 1 von der zweiten Ausgabezeile ab! Den zweiten Ausdruck schliesse ich mit dem Dollarzeichen ab.
$ verhindert die Ausgabe - es wird KEINE Ausgabezeile erzeugt - der Ausdruck, wird aber sehr wohl berechnet!
Das %-Zeichen selber bezieht sich immer auf die letzte Ausgabe, hier also auf das Ergebnis des letzten Ausgabeausdrucks: %o2-5$

```
(%i8) %;
```

(%o8) 0.5

Sie können bei einem CAS wie Maxima auch Variablen in einem Ausdruck verwenden,

```
(%i9) Kosten:a*a-a/2+1;
```

(%o9) $a^2-\frac{a}{2}+1$

und diesen Ausdruck für ein bestimmtes a auswerten: Die Maxima-Funktion dafür
ev(Ausdruck, Argument=Wert); auswerten
Wenn Sie also die Kosten für a=sqrt(2) (Wurzel aus 2) ermitteln wollen

```
(%i10) ev(Kosten,a=sqrt(2));
```

(%o10) $3-\frac{\sqrt{2}}{2}$

So weit möglich lässt **Maxima** ein symbolisches Rechnen zu und hält Brüche oder Wurzeln in einem Ausdruck fest - bis Sie gezielt die numerische Auflösung anstossen:
Über den Menü-Befehl Numerisch oder per maxima-Anweisung .., numer

```
(%i11) ev(Kosten,a=sqrt(2)), numer;
```

(%o11) 2.292893218813453

oder wenn Sie aus dem Ausdruck Kosten eine Gleichung machen
Kosten = 1
und nach a auflösen, also a berechnen wollen. Sie lösen die Gleichung mit der maxima-Funktion
solve(Gleichung, Argument); lösen
und weisen das Ergebnis einer Variable K0 zu:

```
(%i12) K0:solve(Kosten=1,a);
```

(%o12) $[a=\frac{1}{2}, a=0]$

Dieses Ergebnis wollen Sie weiter verarbeiten und in die folg. Gleichung einsetzen:

```
(%i13) Ertrag:8*a^3+3*a+3/2;
```

(%o13) $8a^3+3a+\frac{3}{2}$

Da wäre zuerst zu klären, wie das Ergebnis, das als Liste
[arg1,arg2,...,argn]
ausgegeben wurde, zu behandeln ist. Die Liste K0 hat zwei Werte, die über entsprechende Indizes angesprochen werden:

```
(%i14) K0[1]; K0[2]$
```

(%o14) $a=\frac{1}{2}$

Nun ist klar, wie über die bereits bekannte Funktion ev() die Werte übernommen werden können:

```
(%i16) ev(Ertrag,K0[1]);
```

(%o16) 4

Das Ansprechen von Ergbnisausdrücken wie a=1/2 können Sie auch über die angeom-Funtion 'aus' vornehmen (angeom.mac üssen Sie als Paket laden, maxima-Befehl load() oder Datei-Menü)

```
(%i17) a aus K0;
```

(%o17) $\frac{1}{2}$

Anmerkung: Lösen von Wurzelgleichungen
erfolgt über ein gesondertes Paket, das bei manchen Versionen erst geladen werden muss

```
(%i1) /* nur bei manchen Versionen notwendig */
      load(topoly_solver);
```

(%o2) C:/Programme/Maxima-5.23.2/share/maxima/5.23.2/share/contrib/topoly_solver.mac

```
(%i2) f(a):=16*sqrt(4*a^2+16)-16*a+1280;
      to_poly_solve(f(a)=1344,a);
```

(%o2) $f(a) := 16\sqrt{4a^2+16} - 16a + 1280$

(%o3) $[[a=\frac{8}{3}],[a=0]]$

Maxima vereinfacht algebraische Terme automatisch. Daneben gibt es die Möglichkeit die Vereinfachung oder das Faktorisieren sowie Ausmultiplizieren von Termen zu veranlassen:
Terme vereinfachen faktorisieren ausmultiplizieren
ratsimp(Term); vereinfachen zusammenfassen
factor(Term); faktorisieren
expand(Term); ausmultiplizieren

```
(%i18) ratsimp((3*a-2)*z+(2-a)*y+2*x-3*(3*a-2)-2*(2-a)-2);
```

(%o18) $(3a-2)z+(2-a)y+2x-7a$

```
(%i19) factor(a^2-b^2);
```

(%o19) $-(b-a)(b+a)$

```
(%i20) expand((3*a-2)*z+(2-a)*y+2*x-7*a);
```

(%o20) $3az-2z-ay+2y+2x-7a$

Versionshinweise

!   !

Sie dürfen keine Wertzuweisungen für Laufparameter oder Achsenvariablen vornehmen! In Geraden- oder Ebenengleichungen dürfen die Laufparameter KEINE Wertzuweisung mitbringen/haben.
Bestimmte Variablennamen unterliegen einer besonderen Verwendung in angeom:
a b c d e f - Scharparameter in Geraden/Ebenen
g h i j k l m n o p q r s t u v w – Laufvariablen in Parameterformen
x y z – Achsenvariable
O - Liste von Punkten zum Plotten

!   !

- Die Datei angeom.mac ist kompatibel ab Version **5.23**
- Bis Version **5.22** verwenden Sie angeom522.mac!
- Befehlszeilenvervollständigung: Befehlssilbe+Strg-k
  schni+Strg-k = Liste aller mit Silbe „schni“ beginnenden Fuktionen
- Befehlsreferenz: angeomReferenz.html

  Für schnelle Zugriffe auf häufig verwendete Befehlsstrukturen Sie können die Befehlsreferenzen per Doppelklick in die Zwischenablage zum Einfügen (Strg+v) übernehmen!

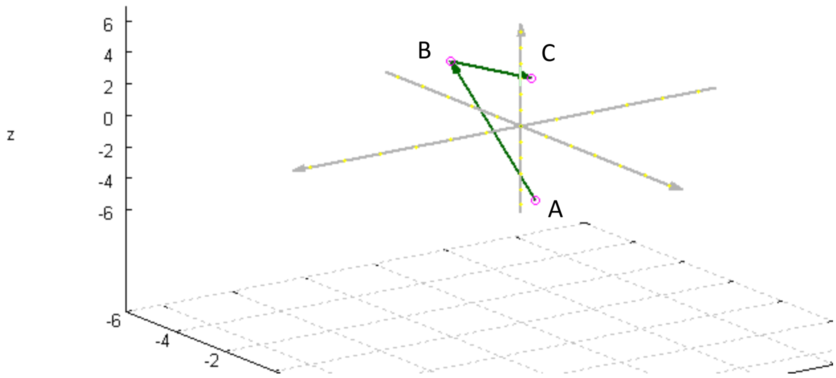
```
(%i1) load(angeom);
(%o1) ı/Maxima-5.14.0/share/maxima/5.14.0/share/vector/angeom.mac
```

Analytische Geometrie mit Maxima/angeom.mac

Punkte und Vektoren definieren sich als Liste:

```
(%i21) A:[1,2,-3]; B:[2,0,5]; C:[-1,-1,2]$
```

```
(%o21) [1,2,-3]
(%o22) [2,0,5]
```



/* $\vec{AB}$ = B-A ; $\vec{BC}$ = C-B; $\vec{AC}$ = $\vec{AB}$ + $\vec{BC}$;
O:[A,B,C]$ PlottVektor:[A,B-A]$ PlottVektor1:[B,C-B]$ PLOTT_S("","","","","","","",-5,6);
C: A+(B-A)+(C-B);

Funktion definieren: Berechnen des Spiegelpunktes über einen Punkt Zentrum

```
(%i22) spiegelPunkt(Zentrum,P):=2*Zentrum-P$
```

Listen sind das A und O in Maxima. Listen können wiederum Listen enthalten, also eine Liste aus den Punkten A und B.

```
(%i24) Punkte:[A,B];
```

```
(%o24) [[1,2,-3],[2,0,5]]
```

Mit den Listenelementen per Index angesprochen

```
(%i25) Punkte[1]; Punkte[1][1]; Punkte[1][2]; Punkte[2][1];
```

```
(%o25) [1,2,-3]
(%o26) 1
(%o27) -3
(%o28) 2
```

Als Laufparameter für Gerade und Ebenen sind die kleinen Buchstaben g bis w reserviert. x,y,z stellen die Achsenvariablen dar. Als Parameter für eine Geraden- oder Ebenenschar stehen die kleinen Buchstaben a bis f zur Verfügung.

KEINE Wertzuweisungen für Laufparameter oder Achsenvariablen!

Geradengleichung show und °

Für eine Gerade weisen Sie einem Symbol wie z.B. gt eine Gleichung aus Ortsvektor und Richtungsvektor zu gt:o+t*rv.
Es ist eine gute Praxis den Laufparameter im Geraden-Symbol zu vermerken.

```
(%i28) gt:A+t*(B-A);
```

(%o28) $[1-2t, 2-2t, 3-t]$

Zum Rechnen in **Maxima** hat sich die Listenschreibweise bewährt.
Um eine besser lesbare Darstellungen zu erhalten verwenden Sie die
Funktion show() für Geraden und Ebenen oder die Vorsilbe ° für Vektoren:

```
(%i29) show(gt); °(B-A);
```

$$g: \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} + t * \begin{bmatrix} -2 \\ -2 \\ -1 \end{bmatrix}$$

(%o30) $\begin{bmatrix} -2 \\ -2 \\ -1 \end{bmatrix}$

Schnell-Eingabemöglichkeiten
Punkte A,B Kordinaten durch Leerzeichen getrennt
defPunkte("A 2 3 1 B -3 2 1/2");
Geraden mit Punkt A und Richtungvektor nv: defGeradeRV("g1 l A nv")
aus zwei Punkten A,B: defGeradeAB("g1 l A B")

```
(%i02) defPunkte("A 2 3 1 B -3 2 1/2");
```

(%o2) done

```
(%i03) defEbeneRV("E1 t s [0,-2,4] [1,0,-1] [2,1,-3]");
```

(%o3) $[2s+t, -2+s, 4-3s-t]$

```
(%i04) defGeradeRV("g1 l [1,-2,0] ERV1(E1)><ERV2(E1)");
```

(%o4) $[1-l, -2-l, -l]$

Ebenengleichungen
Parameterform: Beispiel einer Ebene
Um die Ausgabe der Definition in Listenform zu unterdrücken schließe ich mit dem Dollarzeichen$ ab und erweitere die Eingabezeile mit der show-Funktion.

```
(%i31) Euv:[-2,0,3]+u*[1,1,0]+v*[-2,1,1]$ show(Euv);
```

$$E: \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} -2 \\ 0 \\ 3 \end{bmatrix} + u * \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} + v * \begin{bmatrix} -2 \\ 1 \\ 1 \end{bmatrix}$$

Koordinatengleichung /Vektorgleichung mit dem Normalenvektor nv: [1,-1,3]:

```
(%i32) NFE:ratsimp(nv.([x,y,z]-Euv));
```

(%o32) $3z - y + x - 7$

Oder allgemein: Parameterform P + u*rv1 + v*rv2 mit nv skalarmultipliziert:
nv.([x,y,z]-P)=0

```
(%i34) show(%);
```

$$\begin{bmatrix} 1 \\ -1 \\ 3 \end{bmatrix} .\left(\begin{bmatrix} x \\ y \\ z \end{bmatrix} - \begin{bmatrix} 7 \\ 0 \\ 0 \end{bmatrix} \right) = 0$$

hesseNF(NFE); aus der Koordinatengleichung NFE bestimmen.
Für eine Auswertung einer Ebenen oder Geraden-Gleichung steht die Funktion **F(Gleichung,[werteliste der laufparameter])** oder die Maxima Funktion ev() zur Verfügung -> gt für t=0 oder Euv für v=1 und u =-1 berechnen:

```
(%i33) F(gt,[0]); F(Euv,[1,-1]); ev(gt,t=-1);
```

(%o33) [1 , 2 , 3]
(%o34) [1 , 0 , 2]
(%o35) [3 , 4 , 4]

Skalarprodukt
wird mit dem . Operator berechnet
[a1,a2,a3].[b1,b2,b3] = a3*b3+a2*b2+a1*b1

```
(%i36) [1,2,3].[1,0,1];
```

(%o36) 4

Vektor- oder Kreuzprodukt
mit dem >< Operator
[a1,a2,a3]><[b1,b2,b3] = [a2*b3-a3*b2,a3*b1-a1*b3,a1*b2-a2*b1]
nv ist z.B. der Normalenvektor der Ebene Euv

```
(%i37) nv:[1,1,0]><[-2,1,1]$ °%;
```

(%o38) $\begin{bmatrix} 1 \\ -1 \\ 3 \end{bmatrix}$

Die Flächenmaßzahl des Parallelogramms, das von den beiden Vektoren a und b aufgespannt wird: | a x b |.

Orts- und Richtungsvektoren

Um den Ortsvektor einer Geraden zu erhalten verwenden Sie die Funktion GOV() und für den Richtungsvektor GRV():

```
(%i39) print("Ortsvektor  ",°GOV(gt),"Richtungsvektor", °GRV(gt))$
```

Ortsvektor $\begin{bmatrix} 1 \\ 2 \\ -3 \end{bmatrix}$ *Richtungsvektor* $\begin{bmatrix} 1 \\ -2 \\ 8 \end{bmatrix}$

Für Ebenen (verkürzt in Listendarstellung):

```
(%i41) EOV(Euv); ERV1(Euv); ERV2(Euv);
```

(%o41) [- 2 , 0 , 3]
(%o42) [1 , 1 , 0]
(%o43) [- 2 , 1 , 1]

Schnell-Eingabemöglichkeiten

mit Punkt A und Richtungsvektoren av, bv: defEbeneRV("E1 t s A av bv")
aus drei Punkten A,B,C: defEbeneRV("E1 t s A B C")
in Koordinatenform aus
Normalvektor nv und Punkt A: defEbeneNF("NE1 nv A")
drei Punkten A,B,C: defEbeneNF("NE2 A B C")

PLOTT Funktionen

Zur einfachen 3D Darstellung verwenden Sie
PLOTT_S() bzw PLOTT_L() eigenes gnuplott-Fenster interaktiv drehbare Scene oder
PLOTT_wx() im Dokument eingebettes Fenster (Animationsfolge möglich)

Allgemeine Vorgaben:
O enthält eine Liste mit Punkten, die in der Plot-Scene dargestellt werden:

```
O:[[0,0,0],GOV(gt)]
```

markiert den Ursprung und den Ortsvektor der Geraden gt als Punkt.

```
S_Labels:["A","B","C","D","E","F","G","H"]
```

ist eine Liste von Labels, die die ersten 8 Punkte von O beschriften.
Sie ändern z.B: das Label A auf S1 mit S_Labels[1]:"S1". S_Labels muss immer 8 Einträge aufweisen und die Reihenfolge muss der in O entsprechen. Die Funktionen für Label Schnellvorlagen sind:

```
abc_Labels() -> S_Labels:["A","B","C","D","E","F","G","H"]
pqr_Labels() -> S_Labels:["P","Q","R","S","T","U","V","W"]
clear_Labels()
```

PlottVektor bzw. PlottVektor1..8 ist eine Vektorangabe die einen Vektor von A nach B darstellen kann:

```
PlottVektor:[A,B-A]
```

Die Festlegung PlottVektor:[[0,0,0],[0,0,0]] erzeugt keine Vektordarstellung in der Plot-Scene.
clearPlottVektors() setzt alle PlottVektor-Variablen zurück!

plottVektorS(vec)
paarweise Angabe von, bis in einer Liste vec
erzeugt eine Vektorkette vec = [[0,0,0],A,A,B,B,C] $\overrightarrow{OA}$ $\overrightarrow{AB}$ $\overrightarrow{BC}$
Vektoraddition vec = [[0,0,0],A,A,A+B,A+B,A+B+C]

```
(%i44) O:[[0,0,0],GOV(gt)]; PlottVektor:[EOV(Euv),nv];

(%o44) [[0,0,0],[1,2,3]]
(%o45) [[-2,0,3],[1,-1,3]]
```

Die PLOTT_ Funktion können Ebenen in Normalenform und Geraden darstellen:

```
PLOTT_L([Liste von max 6 Ebenen],[Liste mit max. 6 Geraden],von,bis)
PLOTT_S(NFE1,NFE2,NFE3,g1,g2,g3,von,bis)
PLOTT_D([Liste von max 6 Ebenen],[Liste mit max. 6 Geraden],von,bis)
PLOTT_wx(NFE1,NFE2,NFE3,g1,g2,g3,von,bis)
```

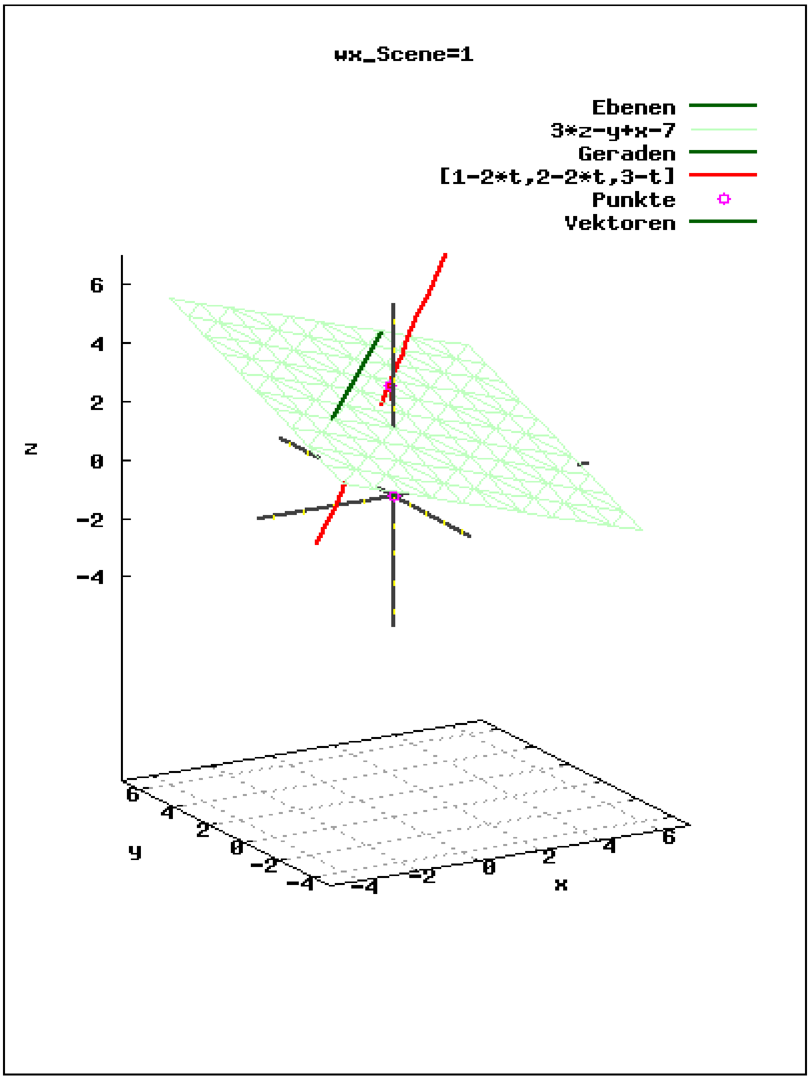
stellen Ebenen oder Geraden in einem Bereich von/bis dar.

Zum Auslassen von Plotargumenten bei PLOTT_S/PLOTT_wx für Ebenen oder Geraden verwenden Sie ersatzweise einfach die Koordinatenebenen x,y,z oder einen leeres Textargument "". PLOT_L/PLOT_D erhalten eine je eine Liste für max. 6 Ebenen/Geraden als Argument, ggf. eine leere Liste.
rangeVon(O), rangeBis(O) kann anstelle von,bis gestellt werden um die Plotgrenzen aus der Liste der Punkte O zu berechnen.

```
Beispiel
S_Enhanced:false$ S_Qvox:6$
O:[KM,SX,SY,SZ]$ PLOTT_L([NFEO,NFESz,NFESy,NFESx],[gk,gt,gr,gh,gl,go],-5,6);
S_Enhanced:true (farbige Oberfläche)
S_Qvox:Maschendichte der Oberflächen (Voreinstellung 12)
       je größer desto feiner und rechenintesiver
Die hier vorgestellten PLOTT-Funktionen sollen möglichst einfach anzuwenden
sein, für eine „ausgereiftere“ Darstellung lohnt es sich mit GNU-Plot zu
experimentieren.
```

```
(%i46) wx_Vrot:77$ wx_Hrot:330$
       PLOTT_wx(normalForm(Euv),"","",gt,"","",-4,6);
```

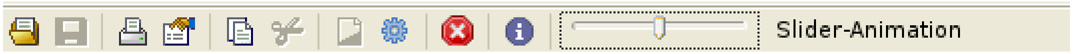
(%t48)



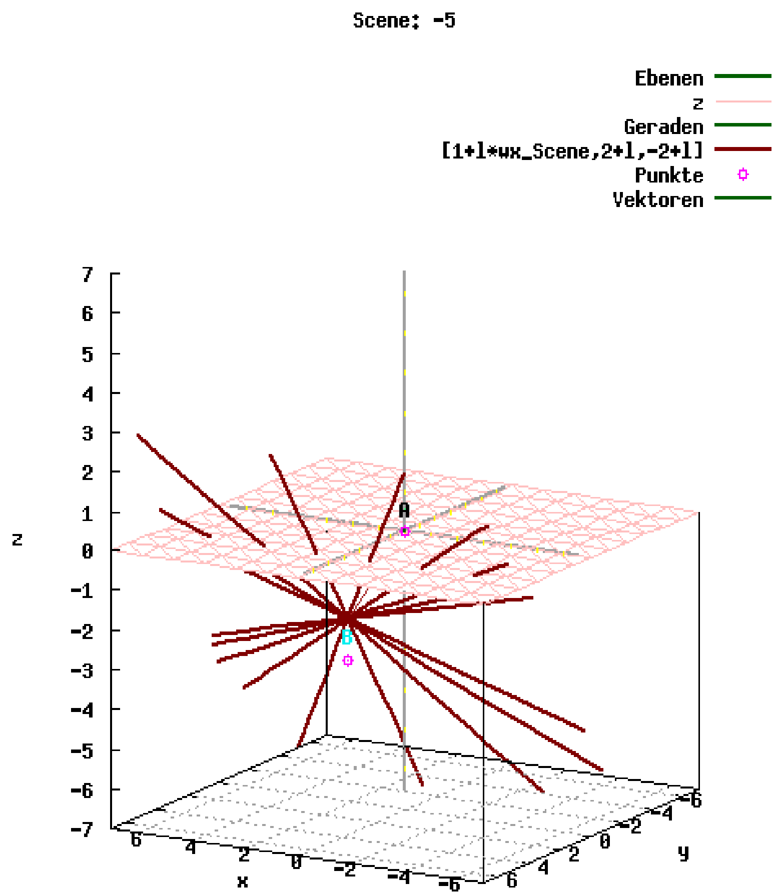

```
Spezielle PLOTT_wx Vorgaben betreffen
wx_Vrot:80$ vertikale Rotation der Scene
wx_Hrot:100$ horizontale Rotation der Scene
wx_Breite:400$ Breite der Plotscene in Px
wx_Hoehe:600$ Höhe der Plotscene in Px
wx_ScVon:1$ Animationsschritte Laufparameter wx_Scene von
wx_ScBis:1$ Animationsschritte Laufparameter wx_Scene bis
wx_AO:true$ Oberflächendarstellung (true, false = Oberflächen durchsichtig)

Es bietet sich an erst mit PLOTT_S/PLOTT_L die Darstellung zu testen, bevor
sie mit PLOTT_wx/PLOTT_D ins Dokument integriert wird!
```

```
(%i49) wx_ScVon:-5$ wx_ScBis:5$
wx_Vrot:102$ wx_Hrot:194$ PlottVektor:[[0,0,0],[0,0,0]]$
PLOTT_wx("","",z,[1,2,-2]+l*[wx_Scene,1,1],"","",-6,6);
```



(%o49)



In dieser Animation verändern Sie den Richtungsvektor der Geraden gl über die Variable wx_Scene von wx_ScVon bis wx_ScBis mit dem Schieberegeler in der Menüleiste von wxMaxima: wxScene nimmt die Werte von-5 bis 5 an und beeinflusst die x-Koordinate der Geraden gl.

S_Ranges:[];
Stellt eine Liste bereit, die die Plot-Bereiche von Ebenen einschränkt. Abweichend von den Plot-Argumenten von, bis können für jede Koordinatenachse individuelle Vorgaben gesetzt werden:

S_Ranges(von_x, bis_x, von_y, bis_y, von_z, bis_z)

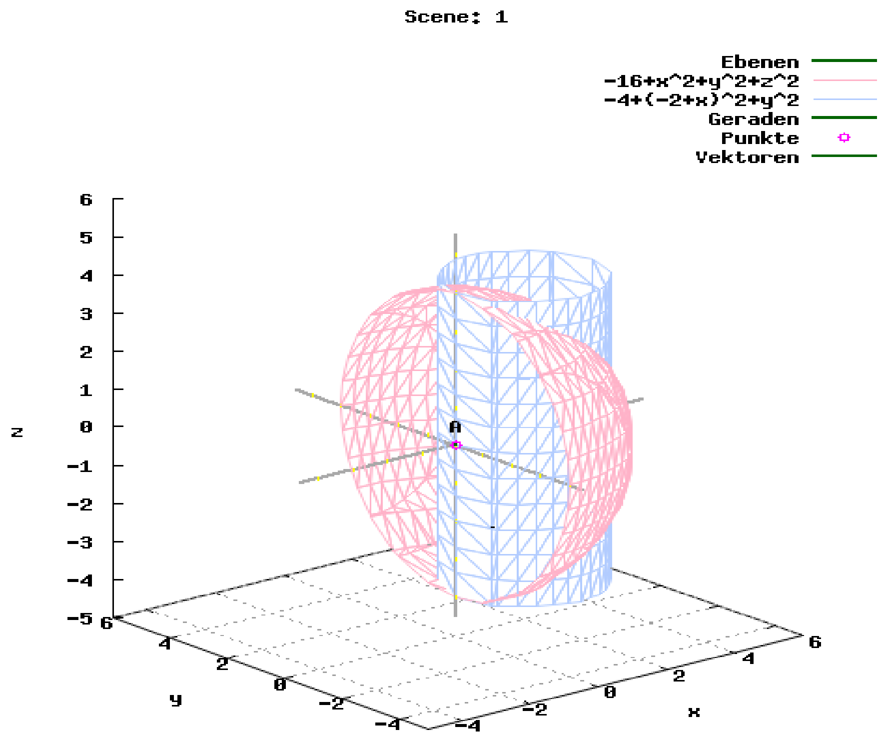
Beispiel für eine Halbkugel im Bereich 0 < x < 4 und
für einen Zylinder -4 < z < 4 :

```
(%i05) K:betragLaenge([x,y,z])^2-4^2;
       Z:(x-2)^2+y^2-4;
```

(%o6) $-16+x^2+y^2+z^2$

(%o7) $-4+(-2+x)^2+y^2$

```
(%i12) S_Ranges:[0,4,-6,6,-4,4]$
       wx_Vrot:71$ wx_Hrot:320$
       PLOTT_D([K,Z],[],-4,5)$
```



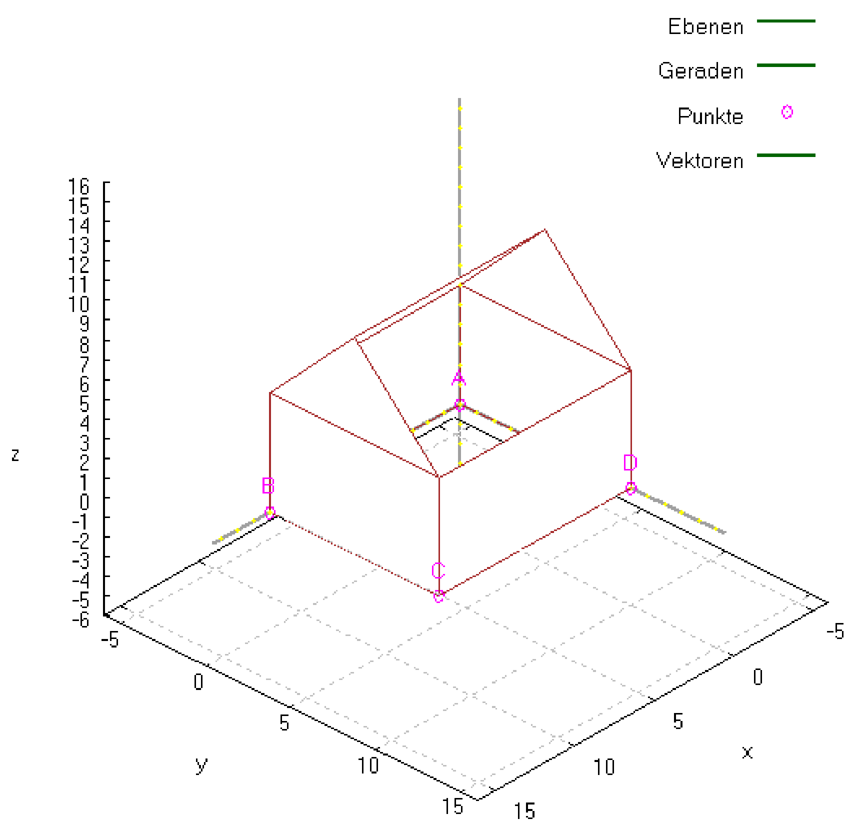
3-D Polygon (mesh), um Figuren zu zeichnen.
Dazu stehen die Variablen S_Figur, S_Figur1...S_Figur4 zur Verfügung.
mesh erhält mind. 2 Listen mit Punkten als Parameter, die per Linienzug verbunden werden sollen. Die erste Punkteliste wird räumlich mit den weiteren Listen verbunden. Für Flächen einfach zwei identische Punktlisten angeben.
Quader ABCD (Grundfläche), EFGH (Deckfläche) mit Dachfirst KL:

```
(%i13) A:[0,0,0]$ B:[12,0,0]$ C:[12,10,0]$ D:[0,10,0]$
       E:[0,0,6]$ F:[12,0,6]$ G:[12,10,6]$ H:[0,10,6]$
       K:[0,5,11] $ L:[12,5,11]$
```

```
(%i14) S_Figur:  mesh([A,B,C,D,A], [E,F,G,H,E],[K,L,G,H,K]) $
       draw3d(S_Figur)$
```

Direktversuch zur Kontrolle der Zeichnung und dann Übernahme in PLOTT

```
(%i14) O:[A,B,C,D]$ PLOTT_L([],[],-5,15)$
```

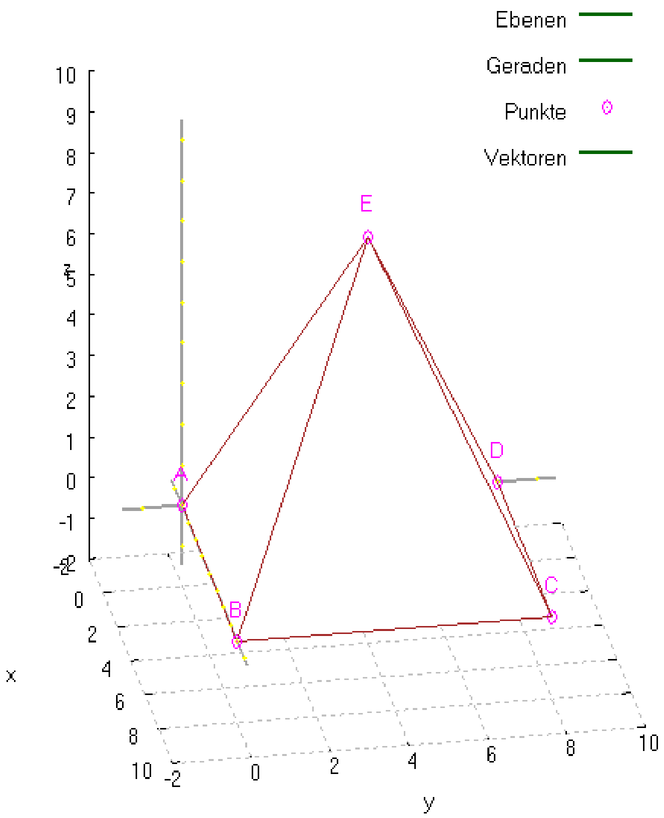


Pyramide ABCDE mit den Punkten

```
(%i15) A :[0,0,0]$ B :[8,0,0]$ C :[8,8,0]$ D :[0,8,0]$ E :[4,4,8]$
```

```
(%i16) E_Figur:mesh([A,B,C,D,A],[A,B,E],[E,B,C],[C,D,E],[E,A,D])$
       draw3d(%);
```

```
(%i16) O:[A,B,C,D,E]$ PLOTT_L([],[],-1,9)$
```



Vektoren und Punkte

schwerPunktABC(A,B,C);
winkelHalbierende(vec a,vec b); winkelHalbierende(B-A,C-A)
mittelSenkrechtenDreieck(A,B,C)

Bestimme den Schwerpunkt des Dreiecks aus den Punkten A,B,C, die winkelhalbierender Vektor zwischen den Vektoren a und b.
Die drei Mittelsenkrechten und Umkreismittelpunkt des Dreiecks ABC

untersucheLage3Punkte(A,B,C);

Zusammenfassung von Ergebnissen zur Untersuchung der Lage dreier Punkte im Raum.

(%i57) untersucheLage3Punkte(A,B,[0,-3,4]);

Es handel sich um ein Dreieck..

das nicht rechtwinkelig ist!

Winkel alpha ABC: 100.2634278919935

Winkel beta CAB: 45.11789288319442

Winkel gamma ACB: 34.61867922481201

Länge der Seiten:

AB = 3 / 3.0

BC = $\frac{14}{\sqrt{14}}$ / 3.741657386773942

CA = $\frac{9}{\sqrt{3}}$ / 5.196152422706633

*Ebene aufgespannt von ABC (A+r*AB+s*AC):*

$[-s-2r+1, -5s-2r+2, s-r+3]$

Normalform: $8z+3y-7x-23$

Abstand vom Ursprung: $\frac{23}{\sqrt{122}}$

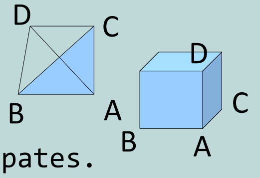
Schwerpunkt von A,B,C (return SPabc)

SPabc: $[0, -\frac{1}{3}, 3]$

Flächeninhalt: F= $\frac{61}{\sqrt{122}}$

(%o57) 5.522680508593631

spatProdukt(a,b,c);



V= (a><b).c = G*h
a1*b2*c3-a2*b1*c3-a1*b3*c2+a3*b1*c2+a2*b3*c1-a3*b2*c1

Beschreibt das Volumen des durch die Vektoren aufgespannten Spates.
V:spatProdukt(vec AB, vec AC, vec AD); Volumen Tetraeder Vt:V/6

```
(%i58) spatProdukt(B-A,C-A,D-A);
```

(%o58) 23

betragLaenge(p);
streckeLaenge(A,B)

Betrag oder Länge eines Vektors p. Länge der Strecke von A nach B.

```
(%i59) betragLaenge(B-A);
```

(%o59) 3

Geradenfunktionen

punktAufGerade(P,g);

Überprüft, ob ein Punkt P auf einer Geraden liegt
true oder false

```
(%i60) P:[0,-3,4]; punktAufGerade(P,gt);
```

(%o60) [0 , - 3 , 4]

(%o61) false

abstandPunktGeradeFp(P,g);

Berechnet den Abstand eines Punktes P von der Geraden g
und fällt das Lot auf die Gerade unter Angabe des Lotfusspunktes **Fp.**

```
(%i62) abstandPunktGeradeFp(P,gt); float(%);
```

Lotfusspunkt auf der Geraden (return Fp)

Fp: $[-\frac{13}{9}, -\frac{4}{9}, \frac{16}{9}]$

(%o62) $\frac{122}{3\sqrt{122}}$

(%o63) 3.681787005729087

spurPunkteGerade(g);

Berechnet die Spurpunkte der Geraden g,
also die Schnittpunkte mit den Koordinatenachsen x=0, y=0, z=0

```
(%i64) spurgt:spurPunkteGerade(gt);
```

Schnittpunkte mit den Koordinatenebenen:

(%o64) $[[0, 1, \frac{5}{2}], [-1, 0, 2], [-5, -4, 0]]$

```
(%i65) gs:[1,-2,-1]+s*[3,1,1]$ °gs;
```

(%o66) $\begin{bmatrix} 3\,s+1 \\ s-2 \\ s-1 \end{bmatrix}$

lageUntersuchungGeraden(g1,g2);

```
Überprüft die Lage der Geraden g1 und g2 im Raum und berechnet
ggf. den Abstand/Schnittpunkt:
Identische Geraden: true
Parallele Geraden: Abstand
Windschiefe Geraden: Abstand
Sich schneidende Geraden: Schnittpunkt
```

```
(%i67) lageUntersuchungGeraden(gt,gs);
```

Abstand der windschiefen Geraden: 2.82842712474619

(%o67) $\frac{4}{\sqrt{2}}$

```
(%i68) lageUntersuchungGeraden(gt,[6,7,11/2]+r*[1,1,1/2]);
```

Dependent equations eliminated: (3 2)

paralle Geraden...

Beide Geraden sind identisch!

(%o68) true

schnittwinkelGeraden(g1,g2);

Berechnet den Schnittwinkel zwischen zwei Geraden g1 und g2

```
(%i69) schnittwinkelGeraden(gt,gs), numer;
```

(%o69) 154.7605981793211

```
Schnittwinkel von Ebenen über die Normalvektoren
schnittwinkelGeraden(l*nv1,t*nv2)
```

Ebenen und Geraden

lotFusspunkt();
lotFusspunkt(EG,P);

```
Berechnet den Fusspunkt des Punktes P auf die Gerade oder Ebene EG in
Parameter oder Koordinatenform.
```

```
(%i68) FP:lotFusspunkt([4,-1,4]+t*[1,-2,1],[-6,-5,3]);
```

(%o68) $[\frac{7}{2}, 0, \frac{7}{2}]$

(%i76) lotFusspunkt(4*x-y-4, [-6,-3,5]);

(%o76) $[-\frac{2}{17}, -\frac{76}{17}, 5]$

lotFusspunkt2Geraden(g1,g2);

Berechnet die Lotfusspunkte des kürzesten Abstandes 2er Geraden,
Die beiden Lotpunkte auf den Geraden in einer Liste:

(%i70) Fp:lotFusspunkt2Geraden(gt,gs);

(%o70) $[[-\frac{17}{3}, -\frac{14}{3}, -\frac{1}{3}], [-5, -4, -3]]$

und damit den Abstand der beiden Geraden:

(%i71) betragLaenge(Fp[2]-Fp[1]);

(%o71) $\frac{4}{\sqrt{2}}$

(%i72) PlottVektor:[Fp[1],Fp[2]-Fp[1]]; O:append([[0,0,0]],spurgt,Fp);

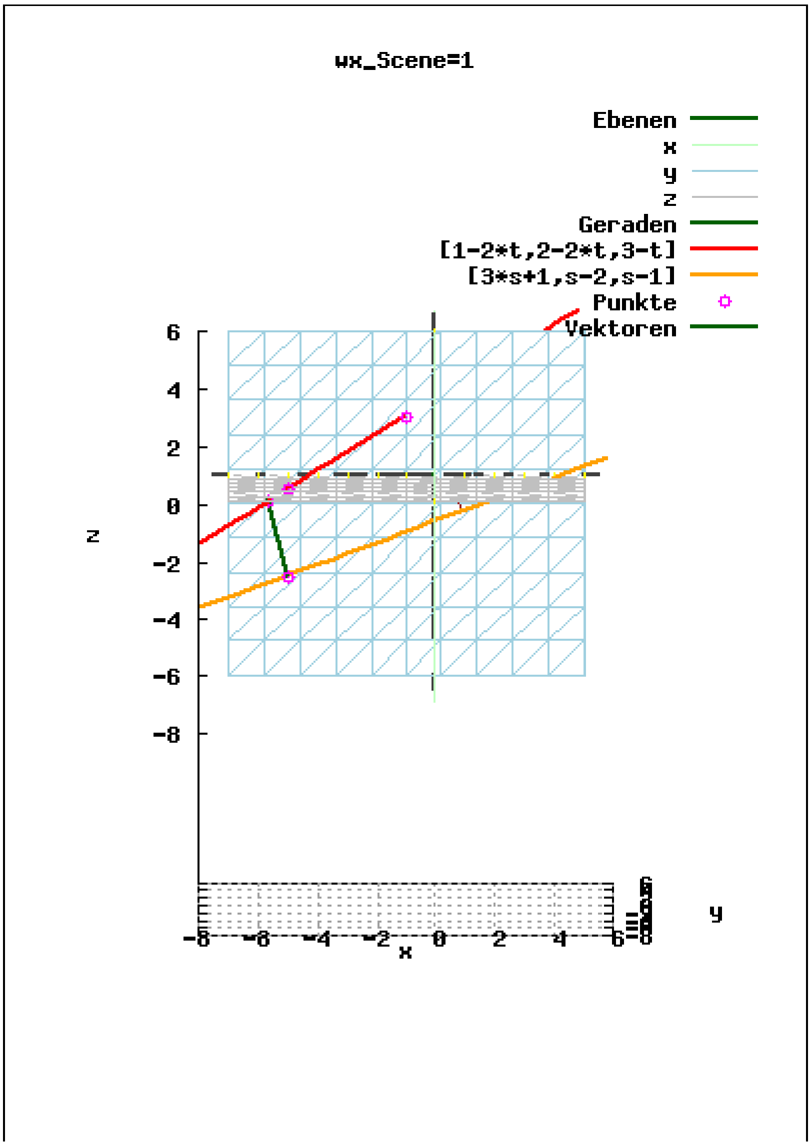
(%o72) $[[-\frac{17}{3}, -\frac{14}{3}, -\frac{1}{3}], [\frac{2}{3}, \frac{2}{3}, -\frac{8}{3}]]$

(%o73) $[[0, 0, 0], [0, 1, \frac{5}{2}], [-1, 0, 2], [-5, -4, 0], [-\frac{17}{3}, -\frac{14}{3}, -\frac{1}{3}], [-5, -4, -3]]$

```
PLOTT Scene mit Darstellung der Geraden gt, gs
Spurpunkte gt über O eingetragen und der PlottVektor verbindet
die LotFusspunkte des kürzesten Abstandes zwischen den Geraden
```

```
(%i74) wx_Vrot:85$ wx_Hrot:0$ PLOTT_wx(x,y,z,gt,gs,"",-7,5);
```

(%t76)



Für eine Normalform setzt F(NF,Punkt) die Koordinaten des Punktes in die Gleichung ein:
d.h der Punkt [1,2,3] liegt auf der Ebene 3*x-y-2*z+5 = 0

```
(%i77) F(3*x-y-2*z+5,[1,2,3]);
```

(%o77) 0

schnittEbeneGerade(E,g);
schnittNFEbeneGerade(NF,g);

Berechnet den Schnittpunkt von Ebene und Gerade. E steht für eine Parameterform und NF für eine in Normalform ax+by+cz+d=0 gegebene Ebene
[] oder S Schnittpunkt

```
(%i78) schnittEbeneGerade(Euv,gs);
```

(%o78) $[\frac{26}{5},-\frac{3}{5},\frac{2}{5}]$

```
(%i79) schnittEbeneGerade(Euv,EOV(Euv)+t*ERV1(Euv));
```

Dependent equations eliminated: (1)
(%o79) $[t-2,t,3]$

```
(%i80) schnittEbeneGerade(Euv,P+t*ERV1(Euv));
```

(%o80) [ ]

schnittwinkelEbeneGerade(E,g);

Berechnet den Schnittwinkel zwischen Gerade g und Ebene E (Parameterform)
0 (kein Schnittpunkt) oder Winkelangabe in Grad

```
(%i81) schnittwinkelEbeneGerade(Euv,gs);
```

(%o81) 27.03569178941229

Ebenen

normalForm(E);

Berechnet die NF Normalform, Koordinatendarstellung einer Ebene aus der Parameterform

```
(%i82) NFE1:normalForm(Euv);
```

(%o82) $3z-y+x-7$

normalVektorNF(NFE1);

aus der Normalform den Normalvektor ableiten.

paramForm(NF,s,t);

Berechet aus der Normalform einer Ebene NFE eine Paramterdarstellung mit den Laufparametervariablen s und t

(%i83) Est:paramForm(NFE1,s,t)$ show(Est);

$$E: \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 7 \\ 0 \\ 0 \end{bmatrix} + s * \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} + t * \begin{bmatrix} -3 \\ 0 \\ 1 \end{bmatrix}$$

(%o84)

spurPunkteEbene(NFE);

Berechnet die Spurpunkte, die Schnittpunkte mit den Koordinatenachsen, x=0, y=0, z=0 damit können dann z.B. die Spurgeraden angegeben werden:

(%i85) spurE1:spurPunkteEbene(NFE1);

Schnittpunkte mit den Koordinatenachsen:

(%o85) $[[7,0,0],[0,-7,0],[0,0,\frac{7}{3}]]$

(%i86) spurGxy:spurE1[1]+p*(spurE1[2]-spurE1[1]);

(%o86) $[7-7p,-7p,0]$

(%i87) spurGyz:spurE1[2]+u*(spurE1[3]-spurE1[2]);

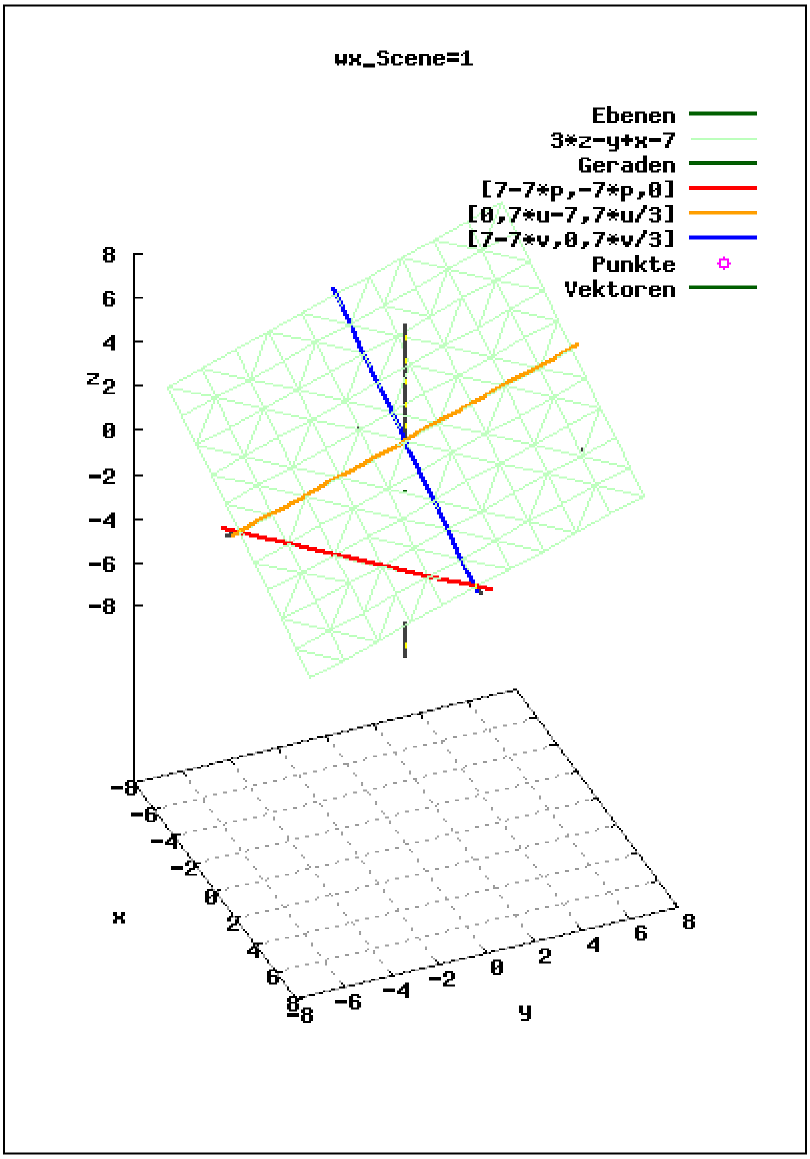
(%o87) $[0,7u-7,\frac{7u}{3}]$

(%i88) spurGxz:spurE1[1]+v*(spurE1[3]-spurE1[1]);

(%o88) $[7-7v,0,\frac{7v}{3}]$

(%i89) O:[[0,0,0]]$ wx_Vrot:66$ wx_Hrot:67$
PLOTT_wx(NFE1,"","",spurGxy,spurGyz,spurGxz,-7,7);

(%t92)



schnitt2Ebenen(NF1,NF2, t);

Schnittgerade mit Laufvariable t zweier Ebenen in Normalform

```
(%i93) NFE1:2*x-y+z-5; NFE2:x+y+2;
```

(%o93) $z - y + 2\,x - 5$

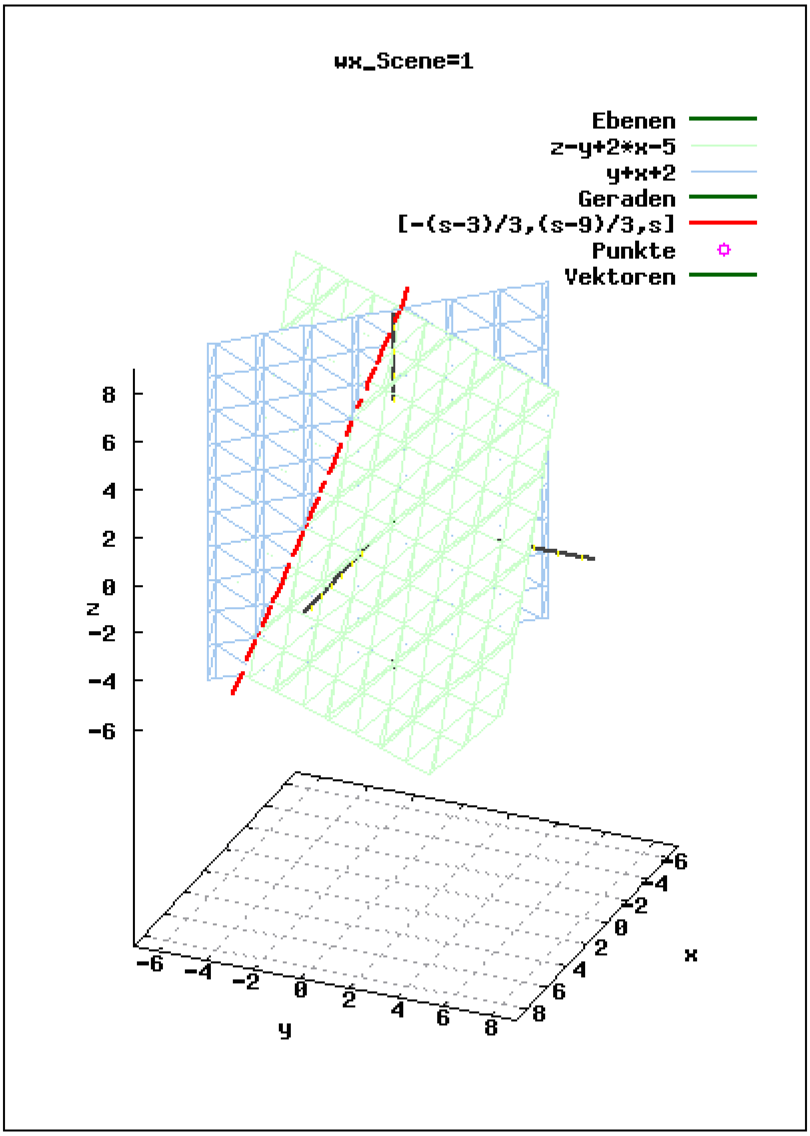
(%o94) $y + x + 2$

```
(%i95) gs:schnitt2Ebenen(NFE1,NFE2, s)$ show(gs);
```

$$g:\ \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 1 \\ -3 \\ 0 \end{bmatrix} + s * \begin{bmatrix} -\frac{1}{3} \\ \frac{1}{3} \\ 1 \end{bmatrix}$$

```
(%i97) wx_Vrot:72$ wx_Hrot:113$ PLOTT_wx(NFE1,NFE2,"",gs,"","",-6,8);
```

(%t99)



Schnittbeispiel zu parallelen Ebenen

```
(%i100) NFE2:-3*x+3/2*y-3/2*z+1;
```

(%o100) $-\frac{3\,z}{2}+\frac{3\,y}{2}-3\,x+1$

```
(%i101) schnitt2Ebenen(NFE1,NFE2, s);
```

(%o101) [ ]

abstandPunktEbene(NF1,P);

Abstand des Punktes P von der Ebene (Normalform) NF1

```
(%i102) abstandPunktEbene([3,-2,6].[x,y,z]-27,[2,-4,1]);
(%0102)  1
```

Anhang

Kugelgleichung

Koordinatendarstellung einer Kugel mit dem Radius r und dem Mittelpunkt $M:[m_1,m_2,m_3]$ (Übergabe an Plott als Ebene)

K: $(([x,y,z]-[m_1,m_2,m_3])^{\wedge\wedge}2 - r^2)[1][1]$
oder
KU: $(x-m_1)^2 + (y-m_2)^2 + (z-m_3)^2 - r^2$
und
xyzKugelMittelpunktForm(KA)
erzeuge die Koordinatengleichung KU aus KA: $x^2 + ax + y^2 + by + z^2 +cz + d$

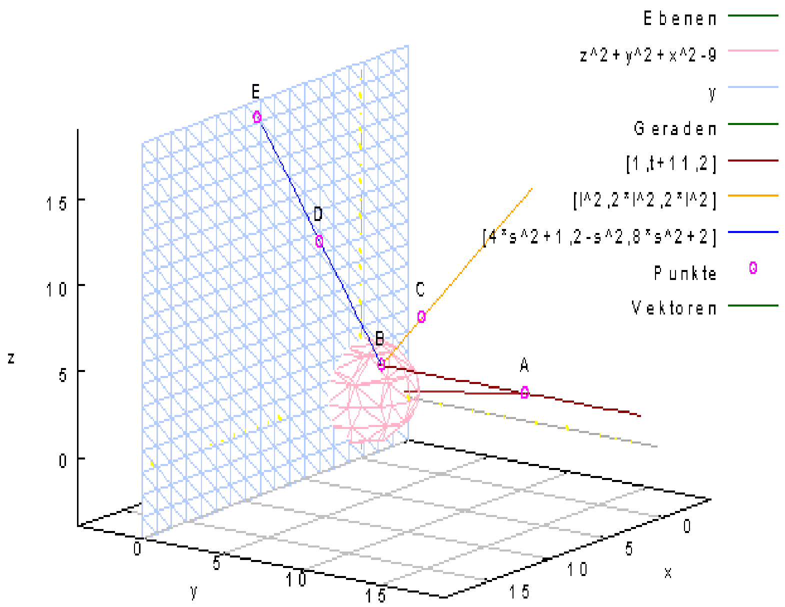
Parameterdarstellung einer Kugel mit dem Radius r um den Ursprung
(Polarkoordinaten - können mit angeom nicht verwendet werden)
$x = m + r\cdot[\sin(t)\cdot\cos(s), \sin(t)\cdot\sin(s), \cos(t)]$

```
plot3d([1,0,1] + 3*[sin(t)*cos(s),sin(t)*sin(s),cos(t)], [t,-5,5], [s,-5,5],
[gnuplot_pm3d,false])$
```

```
(%i2) A:[1,11,2]; Gt:A+t*[0,1,0];
      KO:([x,y,z]^^2-9)[1][1];
```

```
(%o2) [1,11,2]
(%o3) [1,t+11,2]
(%o4) z²+y²+x²-9
```

```
(%i20) O:[A,B,C,D,E]$ PLOTT_L([KO,y],[Gt,l^2*B,Gs2],-3,18);
```



SpiegelungGeradeanKugel.wxm

Kreisgleichung

Parameterdarstellung eines Kreises in einer Ebene E1

Kreis: X = M + r·u·cos(t) + r·v·sin(t)·

M(Mittelpunkt) r(Radius) u,v zueinander orthogonale Vektoren mit der Länge 1 aus der Ebene in der der Kreis liegt. (Übergabe an Plott als Gerade)

Beispiel:
Schnittkreis von Ebene und Kugel

```
(%i2) radius:3;
      M:[2,1,0];
      A:[3,2,1.5]; B:[4,1,1.5]; C:[2,4,-3/2];
```

```
(%i8) E1:A+r*(B-A)+s*(C-A)$ show(E1);
```

$$E: \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 3 \\ 2 \\ 1.5 \end{bmatrix} + r * \begin{bmatrix} 1 \\ -1 \\ 0.0 \end{bmatrix} + s * \begin{bmatrix} -1 \\ 2 \\ -3.0 \end{bmatrix}$$

```
(%i10) NFE1:normalForm(E1);
(%o10) 2*z+6*y+6*x-33
```

Kreismittelpunkt
Lot von Kugelmittelpunkt M auf die Schnittebene NFE1

```
(%i14) KM:lotFusspunkt(NFE1,M);
(%o14) [121/38,83/38,15/38]
(%i15) dM:streckeLaenge(M,KM);
```

(%o15) $15/(2*\sqrt{19})$

Kr Radius Kreis über Pythagoras

```
(%i16) Kr:sqrt(radius^2-dM^2), numer;
(%o16) 2.457534065727376
```

Zu uv Richtungsvektor konstruiere einen senkrechten RV vv aus E1

```
(%i19) uv:ERV1(E1); vv:ERV1(E1)+r*ERV2(E1);
(%o19) [1,-1,0]
(%o20) [1-r,2*r-1,-3.0*r]
```

Othogonal: Skalrprodukt uv.vv = 0

```
(%i21)  rv:solve(uv.vv,r);
(%o21) [r=2/3]
```

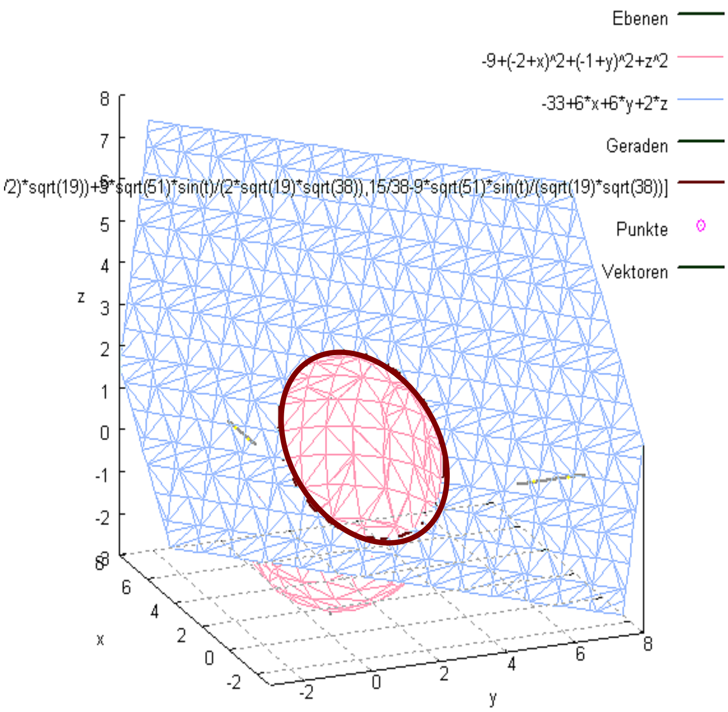
```
(%i22) vv:ev(vv, rv); (uv.vv);

(%o22) [1/3,1/3,-2]
(%o23) 0.0

(%i24) Kt:KM+Kr*uv/betragLaenge(uv)*cos(t)+Kr*vv/betragLaenge(vv)*sin(t);

(%o24) [0.3987*sin(t)+1.2288*sqrt(2)*cos(t)+121/38,
        0.3987*sin(t)-1.2288*sqrt(2)*cos(t)+83/38,
        15/38-2.3920*sin(t)]

(%i25) S_Qvox:16$ PLOTT_L([KU,NFE1],[Kt],-2,7);
```



view: 107.000, 112.000 scale: 1.00000, 1.00000

SchnittkreisEbeneKugel.wxm

```
LageEbeneKugel.wxm

E Ebene, M Mittelpunkt Kugel, r Radius Kugel
Rückgabe
Km: Schnittkreismittelpunkt (Tangentenpunkt)
Kr: Schnittkreisradius
return(Kreisgleichung)
```

```
lageUntersuchungEbeneKugel(E,M,r):=block(
[_u_,_v_,l,m,o],
if not listp(E) then E:paramForm(E,m,o),
Km:lotFusspunkt(E,M),
Kr:streckeLaenge(M,Km),
if r-Kr < 0 then(
    print("Ebene schneidet die Kugel nicht..."),
    none
)else(
    if r - Kr = 0 then(
        print("Ebene ist Tangente im Punkt Km: "),
        return(Km)
    )else(
        Kr:sqrt(r^2- Kr^2),
        print("Ebene schneidet die Kugel im Kreis"),
        print("Mittelpunkt Km:  ",Km," Radius Kr: ",Kr),
        _u_:ERV1(E), _v_:ERV2(E),
        if _u_._v_ # 0 then(
            _rv_:solve(_u_.(_u_+l*_v_),l),
            _v_:ev(_u_+l*_v_, _rv_)
        ),
        Km+Kr*_u_/betragLaenge(_u_)*cos(t)+Kr*_v_/betragLaenge(_v_)*sin(t)
))
)$
```

Ab Version 0.9 können auch Flächen in Parameterform angegeben werden.
Ebenengleichungen können nun auch in Parameterdarstellung geplotet werden!

Beispielaufgabe
Sei ga die Achse eines Zylinders vom Radius Rd. Zur Parameter-Darstellung des Zylinders erzeuge eine zur Geraden ga senkrechte Ebene E1.
Für einen Kreis aus dem Zylindermantel benötige ich ein senkrecht stehendes Vektorenpaar aus der Ebene E1:

```
(%i1) ga:[0,-2,-1]+m*[2,-2,1]$ Rd:6$
```

```
(%i4) NE1:GRV(ga).([x,y,z]-GOV(ga)), ratsimp;
      E1:paramForm(NE1,r,s)$ show(E1);
```

(%o4) $-3+2x-2y+z$

$$E: \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \frac{3}{2} \\ 0 \\ 0 \end{bmatrix} + r * \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} + s * \begin{bmatrix} -\frac{1}{2} \\ 0 \\ 1 \end{bmatrix}$$

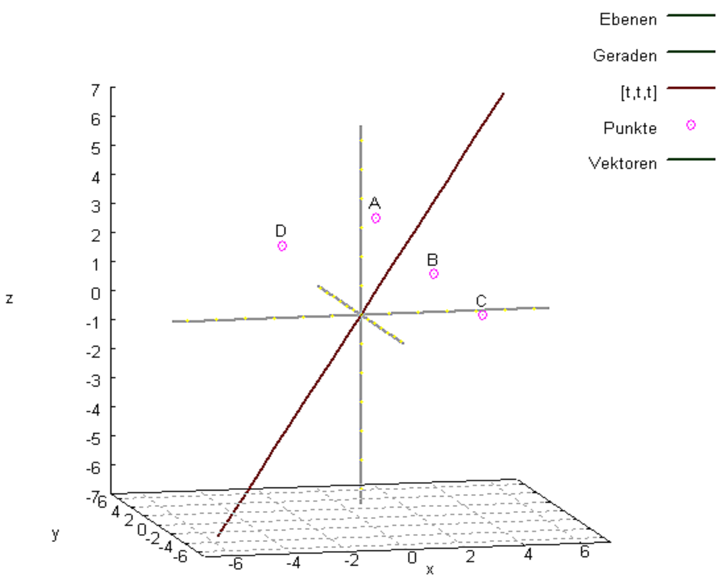
ERV1 senkrecht nv:

```
(%i7) ERV1(E1).
( ERV1(E1)+t*ERV2(E1) );
      nv:solve(%,t);
      nv:ERV1(E1)+t*ERV2(E1),
%;
```

(%o7) $2-\frac{t}{2}$

(%o8) $[t=4]$

(%o9) $[-1,1,4]$



Kreis K mit Mittelpunkt Ortsvektor ga Radius Rd in der Ebene E1

```
(%i11) K:GOV(ga)+Rd*ERV1(E1)/
betragLaenge(ERV1(E1))*cos(t)
+
Rd*nv/betragLaenge(nv)*sin(t)
$ °K
```

$$(\%o11) \begin{bmatrix} 3\sqrt{2}\cos(t)-\sqrt{2}\sin(t) \\ -2+3\sqrt{2}\cos(t)+\sqrt{2}\sin(t) \\ -1+2^{5/2}\sin(t) \end{bmatrix}$$

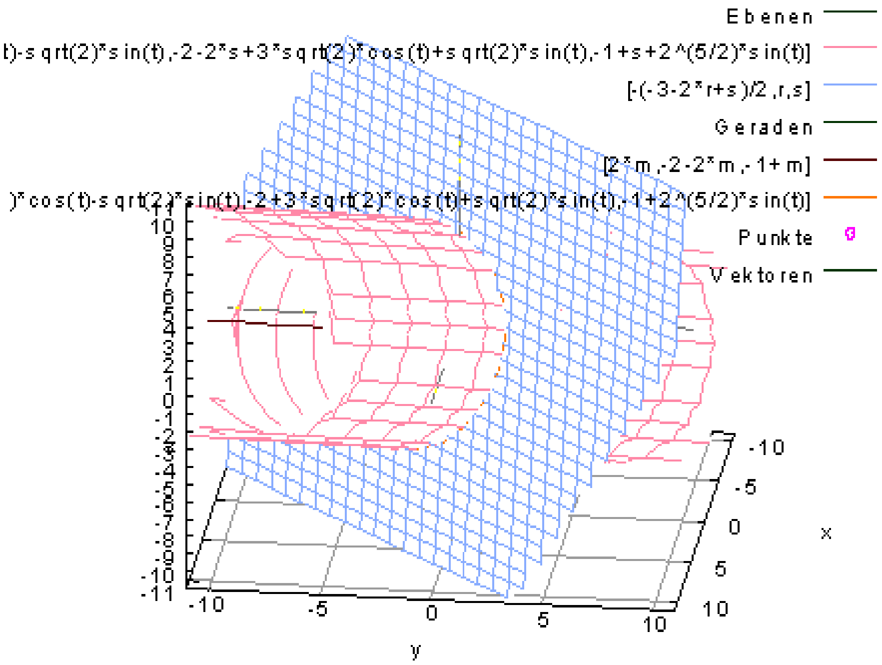
view: 81.0000, 347.000 scale: 1.00000, 1.00000

Kreis K mit Richtungsvektor der Geraden ga zu einem Zylinder ausdehnen:

```
(%i16) Zm:K+s*GRV(ga);
```

(%o16) $[2s+3\sqrt{2}\cos(t)-\sqrt{2}\sin(t), -2-2s+3\sqrt{2}\cos(t)+\sqrt{2}\sin(t), -1+s+2^{5/2}\sin(t)]$

```
(%i17) PLOTT_L([Zm,E1],[ga,K],-10,10)$
```




```
Parametergleichung Zylinder...
```

Abbildungen

Betrachte Abbildungen ϕ: R^3 -> R^3 durch eine Matrix beschrieben
ϕ: T.x = x'
T wird durch die Bilder der Basisvektoren beschrieben.

Spielgelung an Geraden

```
(%i1) spiegelungAnGerade(G):=block(
      to:-2*lotFusspunkt(G,[0,0,0]),
      transpose(matrix(
      2*lotFusspunkt(t*GRV(G),[1,0,0])-[1,0,0],
      2*lotFusspunkt(t*GRV(G),[0,1,0])-[0,1,0],
      2*lotFusspunkt(t*GRV(G),[0,0,1])-[0,0,1]
      )))$

(%i2) Gt:t*[1,1,1] ;
      Se:spiegelungAnGerade(Gt);
      determinant(Se);
```

(%o81) [t,t,t]

(%o82) $\begin{bmatrix} -\frac{1}{3} & \frac{2}{3} & \frac{2}{3} \\ \frac{2}{3} & -\frac{1}{3} & \frac{2}{3} \\ \frac{2}{3} & \frac{2}{3} & -\frac{1}{3} \end{bmatrix}$

(%o83) 1

```
(%i2) A:[1,2,3];
      B:m2l(Se.A)-to;
      C:[4,-1,0];
      D:m2l(Se.C)-to;
      O:[A,B,C,D]$
      PLOTT_L([],[Gt],-6,6)$
```

(%o3) [1,2,3]
(%o4) [-1/3,-10/3,-5/3]
(%o5) [4,-1,0]
(%o6) [-8/3,-5/3,8/3]

Spiegelung an Ebenen

```
(%i2) spiegelungAnEbene(NFE):=block(
      to:2*F(hesseNF(NFE),[0,0,0])*normalVektorNF(hesseNF(NFE)),
      transpose(matrix(
      2*lotFusspunkt(NFE-F(NFE,[0,0,0]),[1,0,0])-[1,0,0],
      2*lotFusspunkt(NFE-F(NFE,[0,0,0]),[0,1,0])-[0,1,0],
      2*lotFusspunkt(NFE-F(NFE,[0,0,0]),[0,0,1])-[0,0,1]
      )))$
```

```
(%i3) ES: x+3-2*y-z-3;
      Se:spiegelungAnEbene(ES); to;
      determinant(Se);
```

(%o94) $-3+x-2\,y-z$

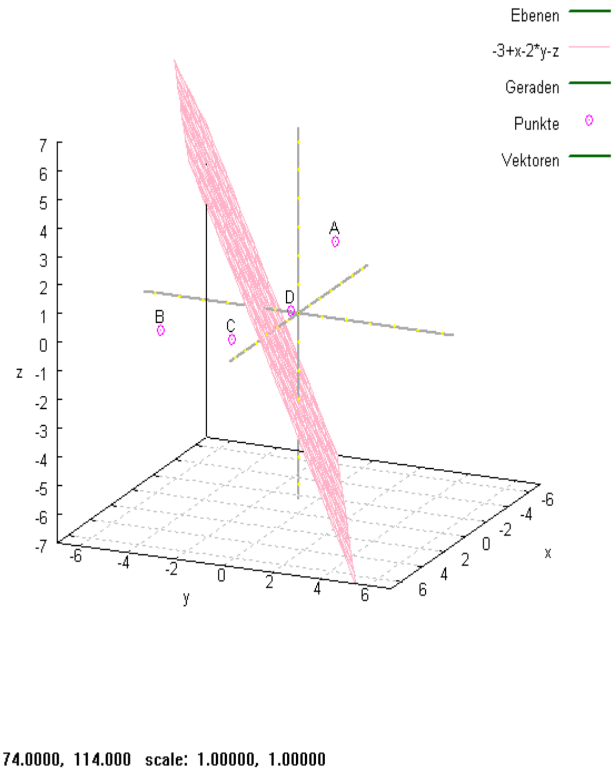
(%o95) $\begin{bmatrix} \frac{2}{3} & \frac{2}{3} & \frac{1}{3} \\ \frac{2}{3} & -\frac{1}{3} & -\frac{2}{3} \\ \frac{1}{3} & -\frac{2}{3} & \frac{2}{3} \end{bmatrix}$

(%o96) [-1,2,1]

(%o97) -1

```
(%i4) A:[1,2,3]; B:m2l(Se.A)-to;
      C:[4,-1,0]; D:m2l(Se.C)-to;
```

(%o98) [1,2,3]
(%o99) [4,-4,0]
(%o100) [4,-1,0]
(%o101) [3,1,1]



Drehungen um die KO-Achsen

```
(%i5) Dx(a):=matrix([1,0,0],[0,cos(a),-sin(a)],[0,sin(a),cos(a)]);
      Dy(a):=matrix([cos(a),0,sin(a)],[0,1,0],[-sin(a),0,cos(a)]);
      Dz(a):=matrix([cos(a),-sin(a),0],[sin(a),cos(a),0],[0,0,1]);
```

Gauss-Algorithmus

gaussLinGLS(M,[x,y,z]);

löst das lineare GLS Ax=b mit der Matrix M=[A,b]und kommentiert die Lösungsschritte.

```
(%i2) A:matrix(
       [ 1, 2, 3, 0, 0],
       [ 0, 1, 2, 3, 0],
       [ 0, 0, 1, 2, 0],
       [ 2, 3, 4, a, 1]
       )$

(%i3) gaussLinGLS(A,[x,y,z,u]);
```

Gaussalgorithmus:

$$\begin{bmatrix} 1 & 2 & 3 & 0 & 0 \\ 0 & 1 & 2 & 3 & 0 \\ 0 & 0 & 1 & 2 & 0 \\ 2 & 3 & 4 & a & 1 \end{bmatrix}$$

*A[2]=1*A[2]+0*A[1]*

*A[3]=1*A[3]+0*A[1]*

*A[4]=1*A[4]-2*A[1]*

$$\begin{bmatrix} 1 & 2 & 3 & 0 & 0 \\ 0 & 1 & 2 & 3 & 0 \\ 0 & 0 & 1 & 2 & 0 \\ 0 & -1 & -2 & a & 1 \end{bmatrix}$$

*A[3]=1*A[3]+0*A[2]*

*A[4]=1*A[4]+1*A[2]*

$$\begin{bmatrix} 1 & 2 & 3 & 0 & 0 \\ 0 & 1 & 2 & 3 & 0 \\ 0 & 0 & 1 & 2 & 0 \\ 0 & 0 & 0 & a+3 & 1 \end{bmatrix}$$

*A[4]=1*A[4]+0*A[3]*

$$\begin{bmatrix} 1 & 2 & 3 & 0 & 0 \\ 0 & 1 & 2 & 3 & 0 \\ 0 & 0 & 1 & 2 & 0 \\ 0 & 0 & 0 & a+3 & 1 \end{bmatrix}$$

A[4]=A[4]/A[4,4]

*A[1]=0*A[4]-1*A[1]*

*A[2]=3*A[4]-1*A[2]*

*A[3]=2*A[4]-1*A[3]*

$$\begin{bmatrix} -1 & -2 & -3 & 0 & 0 \\ 0 & -1 & -2 & 0 & \frac{3}{a+3} \\ 0 & 0 & -1 & 0 & \frac{2}{a+3} \\ 0 & 0 & 0 & 1 & \frac{1}{a+3} \end{bmatrix}$$

A[3]=A[3]/A[3,3]

*A[1]=-3*A[3]-1*A[1]*

*A[2]=-2*A[3]-1*A[2]*

$$\begin{bmatrix} 1 & 2 & 0 & 0 & \frac{6}{a+3} \\ 0 & 1 & 0 & 0 & \frac{1}{a+3} \\ 0 & 0 & 1 & 0 & -\frac{2}{a+3} \\ 0 & 0 & 0 & 1 & \frac{1}{a+3} \end{bmatrix}$$

A[2]=A[2]/A[2,2]

*A[1]=2*A[2]-1*A[1]*

$$\begin{bmatrix} -1 & 0 & 0 & 0 & -\frac{4}{a+3} \\ 0 & 1 & 0 & 0 & \frac{1}{a+3} \\ 0 & 0 & 1 & 0 & -\frac{2}{a+3} \\ 0 & 0 & 0 & 1 & \frac{1}{a+3} \end{bmatrix}$$

A[1]=A[1]/A[1,1]

(%o3) $\left[x = \frac{4}{a+3}, y = \frac{1}{a+3}, z = -\frac{2}{a+3}, u = \frac{1}{a+3} \right]$

Durch Mischen (Zusammenschmelzen) soll eine Legierung L für stabile Leichtmetallbauteile aus vorrätigen Legierungen L1..L5 hergestellt werden. Untersuchen Sie, ob die gewünschte Legierung L aus den vorrätigen Legierungen hergestellt werden kann
Die Legierungen L1*x + L2*y + L3*z + L4*u + L5*s = L sind in der nachfolgenden Matrix zusammengefaßt.
Spalten: L1, L2, L3,L4 L5,L
Zeilen: Aluminium, Zink, Kupfer, Magnesium

(%i19) matrix([90,89,94,93,92,92],[5,5,2,4,5,4],[3,3,1,2,1,2],[2,3,3,1,2,2]);

(%o19) $$\begin{bmatrix} 90 & 89 & 94 & 93 & 92 & 92 \\ 5 & 5 & 2 & 4 & 5 & 4 \\ 3 & 3 & 1 & 2 & 1 & 2 \\ 2 & 3 & 3 & 1 & 2 & 2 \end{bmatrix}$$

Unterbestimmtes GLS: Betrachte s (5.te Spalte) als Parameter (fast beliebig) und stelle um Spalte(6) – s * Spalte(5) :

(%i17) A:addcol(submatrix(A,5,6),col(A,6)-col(A,5)*s);

(%o17) $$\begin{bmatrix} 90 & 89 & 94 & 93 & 92-92s \\ 5 & 5 & 2 & 4 & 4-5s \\ 3 & 3 & 1 & 2 & 2-s \\ 2 & 3 & 3 & 1 & 2-2s \end{bmatrix}$$

und löse das GLS:

(%i20) gaussLinGLS(A,[x,y,z,u])$

(%o20) L:[x=11*s-1,y=1-8*s,z=2*s,u=1-6*s]

(%i21) ev(L,s=0.1);

(%o21) [x=0.1,y=0.2,z=0.2,u=0.4]

Bestimmung der Inversen
Gauss anwenden auf die um die Einheitsmatrix erweitere Matrix A,E

(%i1) A:matrix([1,2,1,1,0,0],[2,5,1,0,1,0],[1,4,a,0,0,1]);

(%i2) gaussLinGLS(A,[x,y,z]);

$$\begin{bmatrix} 1 & 0 & 0 & \frac{5a-4}{a+1} & \frac{4-2a}{a+1} & -\frac{3}{a+1} \\ 0 & 1 & 0 & \frac{1-2a}{a+1} & \frac{a-1}{a+1} & \frac{1}{a+1} \\ 0 & 0 & 1 & \frac{3}{a+1} & -\frac{2}{a+1} & \frac{1}{a+1} \end{bmatrix}$$