```
> restart:
  with(LinearAlgebra):
  with(VectorCalculus):
  with(plots):
  with(plottools):
  interface(displayprecision=5):
```

Plots

```
> PV := proc(v,col,wid)
  PlotVector(v, color=col, width=.03*wid, head_width=.1*wid, border=
  false, scaling=constrained);
  end proc:
  PT := proc(v,lab)
  local tf,n;
  tf := 0.1;
  n := sqrt(v[1]^2+v[2]^2+v[3]^2):
  if n<>0 then
  textplot3d([v[1]*(1+tf/n),v[2]*(1+tf/n),v[3]*(1+tf/n),lab])
  else
  textplot3d([tf,tf,tf,lab])
  fi;
  end proc:
  ee[1] := Vector([1,0,0]):
  ee[2] := Vector([0,1,0]):
  ee[3] := Vector([0,0,1]):
  E := IdentityMatrix(3):
  PC := proc(A,N,col,wid)
  local i,v,e,tf;
  for i from 1 to 3 do
  v[i] := MatrixVectorMultiply(A,ee[i]);
  od;
  tf := 1.1;
  seq([PV(v[i],col[i],wid),PT(v[i],N*e[i])][],i=1..3);
  end proc:
```

Spektralsatz: Diagonalisierung einer symmetrischen reellen 3x3-Matrix und Bestimmung der Hauptachsen:
A=UJU^(-1)=UJU^T mit Diagonalmatrix J und orthogonaler Matrix U

```
> Spektral := proc(A)
  local d,J,U,Q,i;
  d,U := evalf(Eigenvectors(A));
  Q := convert(GramSchmidt(
            [seq(Transpose(Transpose(U)[i]),i=1..3)]
            ,normalized=true),Matrix);
  # Mache Q eine Drehmatrix
  if Determinant(Q)<0 then
  for i from 1 to 3 do
  Q[i,3] := - Q[i,3]
  od fi;
  J := Matrix(1..3,1..3);
  for i from 1 to 3 do J[i,i] := d[i] od;
  # print(J,Q);
  # Teste Korrektheit
  # MatrixMatrixMultiply(Transpose(Q),MatrixMatrixMultiply(A,Q));
  [J,Q];
  end proc:
```

```
> A := Matrix([ [2,1,1], [1,2,1], [1,1,6] ]);
  Spektral(A);
```

$$A := \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 6 \end{bmatrix}$$

$$\left[\begin{bmatrix} 6.56155 & 0 & 0 \\ 0 & 2.43845 & 0 \\ 0 & 0 & 1.00000 \end{bmatrix}, \begin{bmatrix} 0.26096 & -0.65719 & 0.70711 \\ 0.26096 & -0.65719 & -0.70711 \\ 0.92941 & 0.36905 & 9.63055\,10^{-11} \end{bmatrix}\right] \quad \textbf{(1)}$$

Visualisierung Spektralsatz:

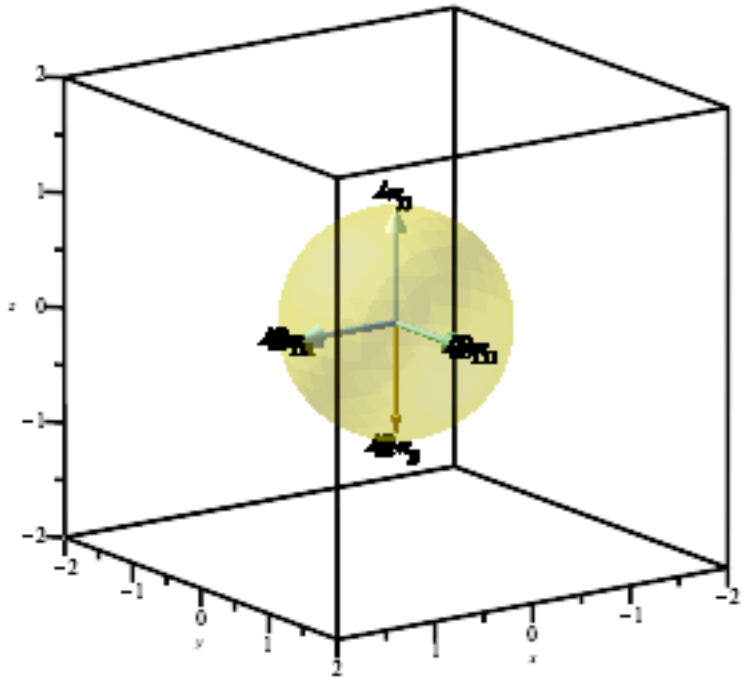
```
> SpektralVis := proc(A1);
  local A,JQ,J,Q,AQ,J1,Q1,AQ1,E,P,F,Quadrik;
  JQ := Spektral(A1);
  print(JQ[]);
  J1 := JQ[1];
  Q1 := JQ[2];
  AQ1 := MatrixMatrixMultiply(A1,Q1);
  P[1] := PC(IdentityMatrix(3),1,[navy,"DarkCyan","DarkTurquoise"],1)
  ;
  P[2] := PC(A1,'A',["CornflowerBlue",aquamarine,turquoise],2);
  P[3] := PC(Q1,'Q',[orange,coral,"Goldenrod"],1);
  P[4] := PC(AQ1,'AQ',["LightCoral","NavajoWhite","LightGoldenrod"],
  1);
  P[0] := P[1],P[2],P[3],P[4];
  Quadrik := implicitplot3d(
                  MatrixVectorMultiply(Transpose(Vector([x,y,z])),
                                       MatrixVectorMultiply(A1,Vector
  ([x,y,z])))
                  - 1,
                  x=-2..2, y=-2..2, z=-2..2,
                  transparency=.7,color=yellow,style=surface,
  numpoints=10000);
  F := proc(t) display(P[t],Quadrik) end proc:
```

```
  animate(F,[t],t=[0,1,2,3,4]);
  end proc:
> A := IdentityMatrix(3);
  SpektralVis(A);
```

$$A := \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1.00000 & 0 & 0 \\ 0 & 1.00000 & 0 \\ 0 & 0 & 1.00000 \end{bmatrix}, \begin{bmatrix} 0. & 1.00000 & -0. \\ 1.00000 & 0. & -0. \\ 0. & 0. & -1.00000 \end{bmatrix}$$

$t = 0.$

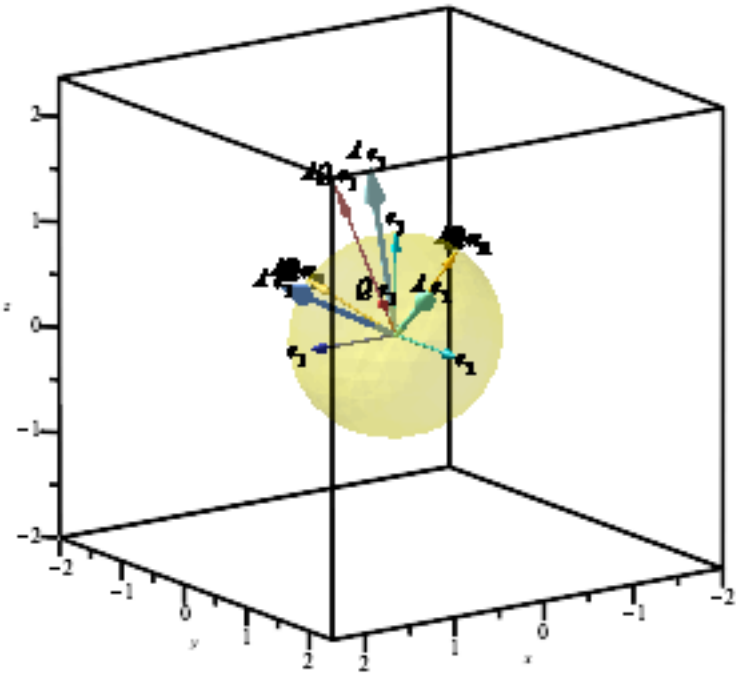


```
> A := Matrix([ [2,1,1], [1,2,1], [1,1,2]]);
  SpektralVis(A);
```

$$A := \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}$$

$$\begin{bmatrix} 4.00000 & 0 & 0 \\ 0 & 1.00000 & 0 \\ 0 & 0 & 1.00000 \end{bmatrix}, \begin{bmatrix} 0.57735 & -0.70711 & 0.40825 \\ 0.57735 & 0. & -0.81650 \\ 0.57735 & 0.70711 & 0.40825 \end{bmatrix}$$

$t = 0.$

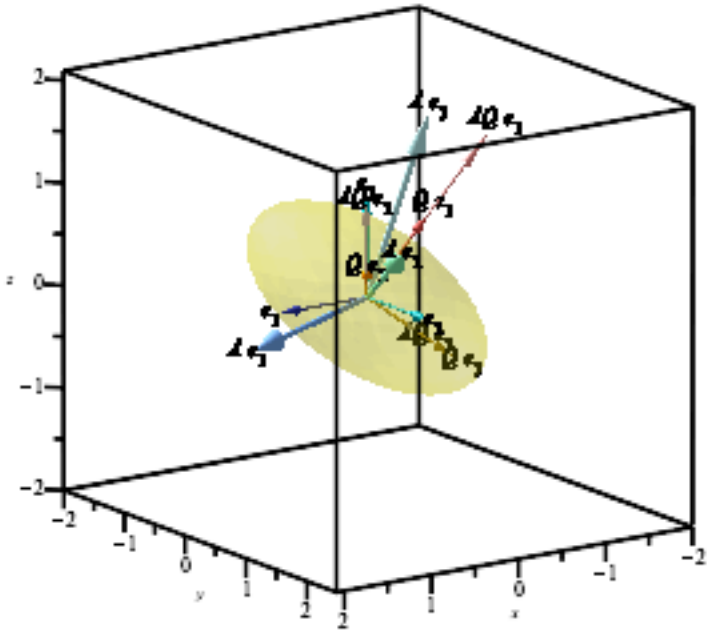


```
> A := Matrix([ [2,1,0], [1,2,1], [0,1,2]]);
  SpektralVis(A);
```

$$A := \begin{bmatrix} 2 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 2 \end{bmatrix}$$

$$\begin{bmatrix} 2.00000 & 0 & 0 \\ 0 & 3.41421 & 0 \\ 0 & 0 & 0.58579 \end{bmatrix}, \begin{bmatrix} -0.70711 & 0.50000 & -0.50000 \\ 0. & 0.70711 & 0.70711 \\ 0.70711 & 0.50000 & -0.50000 \end{bmatrix}$$

$t = 0.$



```
> A := Matrix([ [-2,1,0], [1,2,1], [0,1,2]]);
  SpektralVis(A);
```

$$A := \begin{bmatrix} -2 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 2 \end{bmatrix}$$

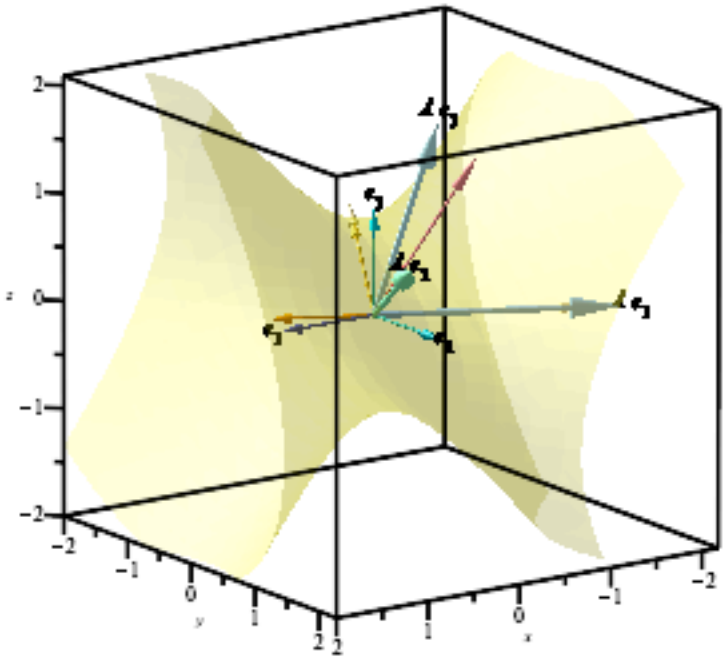
$$\begin{bmatrix} 3.10278 - 4.00000\,10^{-10}\,\mathrm{I} & 0 & 0 \\ 0 & -2.24914 + 0.\,\mathrm{I} & 0 \\ 0 & 0 & 1.14637 + 0.\,\mathrm{I} \end{bmatrix},$$

$$\left[\left[0.14367 + 1.09578\,10^{-10}\,\mathrm{I},\ 0.96877 - 2.14505\,10^{-12}\,\mathrm{I},\ -0.20209 - 4.90108\,10^{-11}\,\mathrm{I}\right],\right.$$

$$\left[0.73310 + 5.01683\,10^{-10}\,\mathrm{I},\ -0.24136 - 1.09457\,10^{-11}\,\mathrm{I},\ -0.63586 - 2.50091\,10^{-10}\,\mathrm{I}\right],$$

$$\left.\left[0.66478,\ 0.05680 - 9.92558\,10^{-12}\,\mathrm{I},\ 0.74488 - 2.26783\,10^{-10}\,\mathrm{I}\right]\right]$$

$t = 0.$

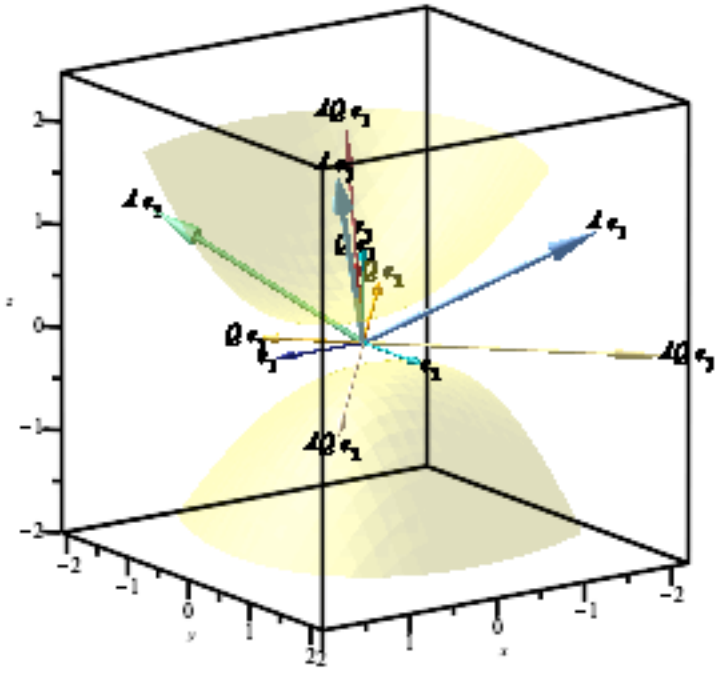


```
> A := Matrix([ [-2,1,1], [1,-2,1], [1,1,2]]);
  SpektralVis(A);
```

$$A := \begin{bmatrix} -2 & 1 & 1 \\ 1 & -2 & 1 \\ 1 & 1 & 2 \end{bmatrix}$$

$$\begin{bmatrix} 2.56155 & 0 & 0 \\ 0 & -1.56155 & 0 \\ 0 & 0 & -3.00000 \end{bmatrix}, \begin{bmatrix} 0.26096 & -0.65719 & 0.70711 \\ 0.26096 & -0.65719 & -0.70711 \\ 0.92941 & 0.36905 & 9.63055\ 10^{-11} \end{bmatrix}$$

$t = 0$.



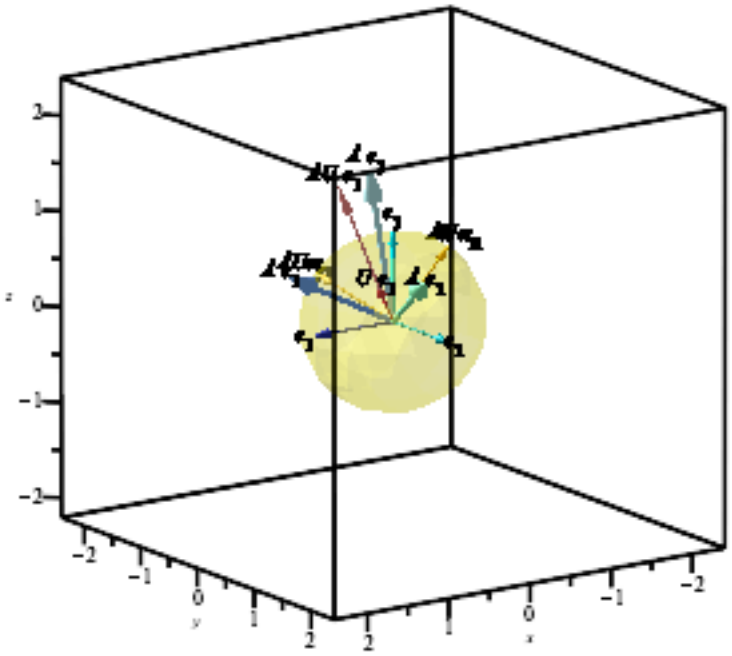
Visualisierung Spektralsatz für eine Schar von Matrizen

```
> SpektralVisAll := proc(A1);
  local JU,J,U1,U,V,AU,AU1,A,F,P,Quadrik;
  JU := Spektral(A1,silent);
  J := JU[1];
  U1 := JU[2];
  AU1 := MatrixMatrixMultiply(A1,U1);
  P[1] := PC(IdentityMatrix(3),1,[navy,"DarkCyan","DarkTurquoise"],1)
  ;
  P[2] := PC(A1,'A',["CornflowerBlue",aquamarine,turquoise],2);
  P[3] := PC(U1,'U',[orange,coral,"Goldenrod"],1);
  P[4] := PC(AU1,'AU',["LightCoral","NavajoWhite","LightGoldenrod"],
  1);

  Quadrik := implicitplot3d(
                MatrixVectorMultiply(Transpose(Vector([x,y,z])),
                                     MatrixVectorMultiply(A1,Vector
  ([x,y,z])))
                - 1,
                x=-2..2, y=-2..2, z=-2..2,
                transparency=.7,color=yellow,style=surface,
  numpoints=1000);
  display(P[1],P[2],P[3],P[4],Quadrik);
  end proc:
> M := Matrix([ [2,1,1], [1,2,1], [1,1,-x] ]);
  animate(SpektralVisAll,[M],x=[seq(x1,x1=-2..2,1/10)]);
```

$$M := \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & -x \end{bmatrix}$$

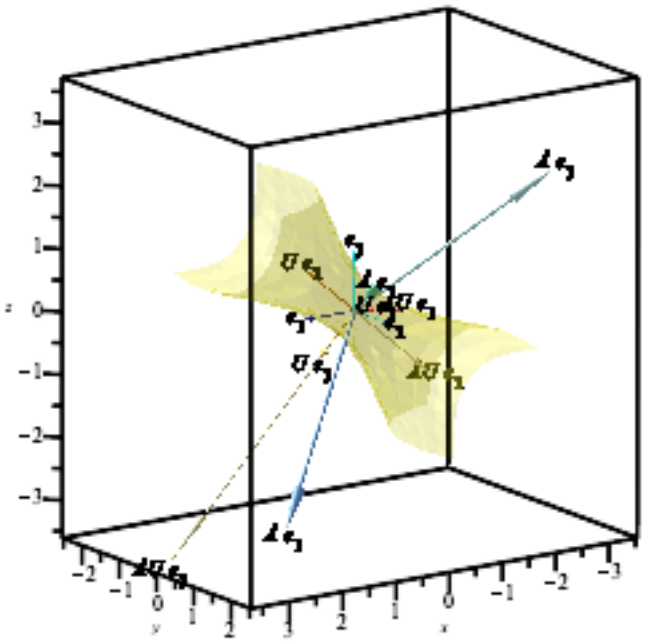
$x = -2.$



```
> M := Matrix([ [2,1,x], [1,2,1], [x,1,2] ]);
  animate(SpektralVisAll,[M],x=[seq(x1,x1=-3..3,1/10)]);
```

$$M := \begin{bmatrix} 2 & 1 & x \\ 1 & 2 & 1 \\ x & 1 & 2 \end{bmatrix}$$

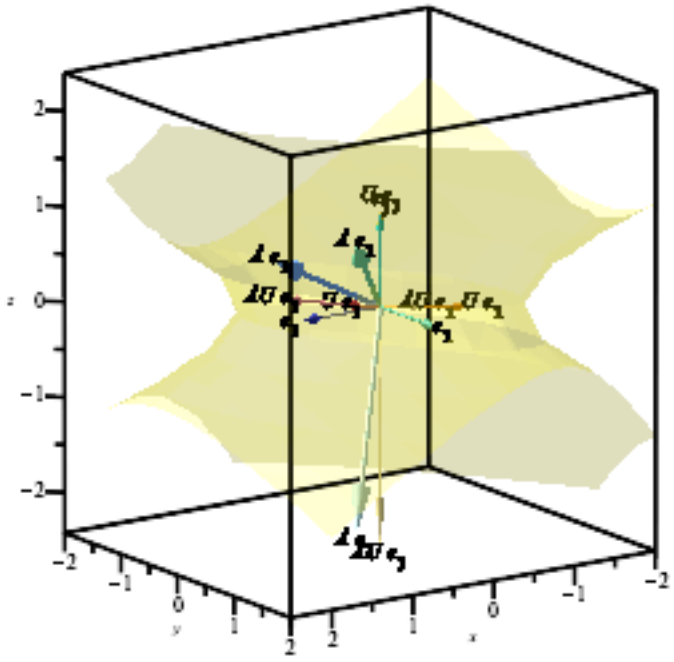
$x = -3.$



```
> M := Matrix([ [2,1,1], [1,x,1], [1,1,-2] ]);
  animate(SpektralVisAll,[M],x=[seq(x1,x1=-1..1,1/10)]);
```

$$M := \begin{bmatrix} 2 & 1 & 1 \\ 1 & x & 1 \\ 1 & 1 & -2 \end{bmatrix}$$

$x = 1.$



Argument eines Vektors

```
> argu := proc(x,y)
  if y=0 then
  if x>=0 then 0
  elif x<0 then Pi
  fi;
  else 2*arctan(y/(sqrt(x^2+y^2)+x))
  fi;
  end proc:
```

Zu Drehmatrix U berechne eine orthogonale Matrix S und einen Winkel theta mit
U = S * Matrix([[1,0,0],[0,c,-s],[0,s,c]]) * S^T
für c=cos(theta) und s=sin(theta):

```
> Drehung := proc(R)
  local d,U,E,v,S,STRS,theta,RTest,i,j;
  d,U := evalf(Eigenvectors(R));
  if abs(d[1]-1)<10^(-6)
  then E := IdentityMatrix(3)
  elif abs(d[2]-1)<10^(-6)
  then E := Matrix([[0,1,0],[1,0,0],[0,0,1]]);
  elif abs(d[3]-1)<10^(-6)
  then E := Matrix([[0,0,1],[0,1,0],[1,0,0]]);
  else print("Fehler: R keine Drehmatrix")
  fi;
  d := MatrixVectorMultiply(E,d);
  U := MatrixMatrixMultiply(U,E);
  v[1] := Re(Transpose(Transpose(U)[1]));
  v[1] := v[1]/sqrt(v[1][1]^2+v[1][2]^2+v[1][3]^2);
  v[2] := Re(Transpose(Transpose(U)[2]));
  v[2] := v[2]/sqrt(v[2][1]^2+v[2][2]^2+v[2][3]^2);
  v[3] := CrossProduct(v[1],v[2]);
  S := convert([v[1],v[2],v[3]],Matrix);
  STRS := MatrixMatrixMultiply(Transpose(S),MatrixMatrixMultiply(R,S)
  );
  theta := argu(STRS[2,2],STRS[3,2]);
  RTest := MatrixMatrixMultiply(S,MatrixMatrixMultiply(
           Matrix([[1,0,0],[0,cos(theta),-sin(theta)],[0,sin(theta),
  cos(theta)]])
           ,Transpose(S)));
  if add(add(abs((R-RTest)[i,j]),i=1..3),j=1..3)>10^(-6)
  then print("Rechenfehler")
  fi;
  theta,S
  end proc:
```

QR-Zerlegung einer reellen 3x3-Matrix:
A=QR mit orthogonaler Matrix Q und rechter oberer Dreiecksmatrix R

```
> QR := proc(A1)
  local Q,Q1,R,R1,E;
  Q1,R1 := QRDecomposition(evalf(A1),fullspan=true);
  # Mache die ersten beiden Diagonaleinträge von R nichtnegativ
  if R1[1,1]<0 then
  E := Matrix([[-1,0,0],[0,1,0],[0,0,-1]]);
  R1 := MatrixMatrixMultiply(E,R1);
  Q1 := MatrixMatrixMultiply(Q1,E);
  fi;
  if R1[2,2]<0 then
  E := Matrix([[1,0,0],[0,-1,0],[0,0,-1]]);
  R1 := MatrixMatrixMultiply(E,R1);
  Q1 := MatrixMatrixMultiply(Q1,E);
  fi;
  # Mache Q zu Drehmatrix
  if Determinant(Q1)<0 then
  E := Matrix([[1,0,0],[0,1,0],[0,0,-1]]);
  R1 := MatrixMatrixMultiply(E,R1);
  Q1 := MatrixMatrixMultiply(Q1,E);
  fi;
  # Teste Korrektheit
  # print(MatrixMatrixMultiply(Q1,R1));
  print(Q=Q1,R=R1);
  Q1,R1
  end proc:
> A1 := Matrix([ [2,1,1], [1,2,1], [1,1,0] ]);
  QR(A1):
```

Visualisierung QR-Zerlegung:

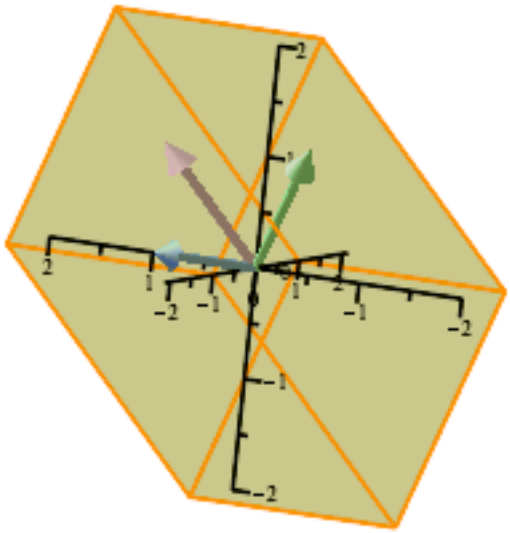
```
> QRVis := proc(A1);
  local A,QR1,Q1,R1,R11,tQ,SQ,e,Zeich,v,w,Fig,i,j,n0,n,E,c;
  QR1 := QR(A1);
  Q1 := QR1[1];
  R1 := QR1[2];
  # Bestimme Drehachse und Winkel
  tQ,SQ := Drehung(Q1);
  e[1] := Vector([1,0,0]);
  e[2] := Vector([0,1,0]);
  e[3] := Vector([0,0,1]);
  # Achsen in der Mitte und am Ende
  for i from 1 to 3 do
  v[i] := MatrixVectorMultiply(R1,e[i]);
  w[i] := MatrixVectorMultiply(A1,e[i]);
  od;
  # Zeichne Figur nach Transformation durch T
  Zeich := proc(T)
  local c,Punkt,Kante,Achsen,EndAchsen,Kanten,Seiten,i;
  c := 1;
  Punkt := proc(v)
  convert(MatrixVectorMultiply(T,Vector(v)),list)
  end proc:
  Kante := proc(v1,v2)
  line(Punkt(v1),Punkt(v2),color=coral,linestyle=solid)
  end proc:
  EndAchsen := {PV(w[1],"CornflowerBlue",3),
                PV(w[2],aquamarine,3),
                PV(w[3],plum,3)}:
```

```
Achsen := {PV(MatrixVectorMultiply(T,e[1]),navy,2),
           PV(MatrixVectorMultiply(T,e[2]),"DarkCyan",2),
           PV(MatrixVectorMultiply(T,e[3]),"DarkViolet",2)}:
Kanten := {Kante([ c, c, c],[ c, c,-c]),
           Kante([ c, c,-c],[ c,-c,-c]),
           Kante([ c,-c,-c],[ c,-c, c]),
           Kante([ c,-c, c],[ c, c, c]),
           Kante([ c, c, c],[-c, c, c]),
           Kante([ c, c,-c],[-c, c,-c]),
           Kante([ c,-c,-c],[-c,-c,-c]),
           Kante([ c,-c, c],[-c,-c, c]),
           Kante([-c, c, c],[-c, c,-c]),
           Kante([-c, c,-c],[-c,-c,-c]),
           Kante([-c,-c,-c],[-c,-c, c]),
           Kante([-c,-c, c],[-c, c, c])}:
Seiten := {plot3d({Punkt([ s*c, t*c,   c]),
                   Punkt([ s*c, t*c,  -c]),
                   Punkt([ s*c,   c, t*c]),
                   Punkt([ s*c,  -c, t*c]),
                   Punkt([   c, s*c, t*c]),
                   Punkt([  -c, s*c, t*c])},
                  s=-1..1,t=-1..1,
                  transparency=.7,color=yellow,style=surface)}:
display(EndAchsen union Kanten union Seiten union Achsen,axes=
normal);
end proc:
# Zwischenschritt
R11 := Matrix(1..3,1..3);
for i from 1 to 3 do R11[i,i] := R1[i,i] od;
# Fallunterscheidung
Fig := proc(n)
local E;
E := IdentityMatrix(3);
if n<n0 then
Zeich(E+n/n0*(R11-E));
elif n<2*n0 then
Zeich(R11+(n/n0-1)*(R1-R11));
else
Zeich(MatrixMatrixMultiply(
       MatrixMatrixMultiply(SQ,
        Matrix([ [1,0,0],
                 [0, cos((n/n0-2)*tQ),-sin((n/n0-2)*tQ)],
                 [0, sin((n/n0-2)*tQ), cos((n/n0-2)*tQ)]])),
       MatrixMatrixMultiply(Transpose(SQ),R1)));
fi;
end proc:
# Figur zeichnen
n0 := 20;
animate(Fig,[n],n=[seq(i,i=0..3*n0)]);
end proc:
> A := Matrix([ [1,0,1], [0,1,0], [0,1,1] ]);
QRVis(A);
```

$$A := \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$$

$$Q = \begin{bmatrix} 1.00000 & 0. & 0. \\ 0. & 0.70711 & -0.70711 \\ 0. & 0.70711 & 0.70711 \end{bmatrix}, R = \begin{bmatrix} 1.00000 & 0. & 1.00000 \\ 0. & 1.41421 & 0.70711 \\ 0. & 0. & 0.70711 \end{bmatrix}$$

$n = 60.$

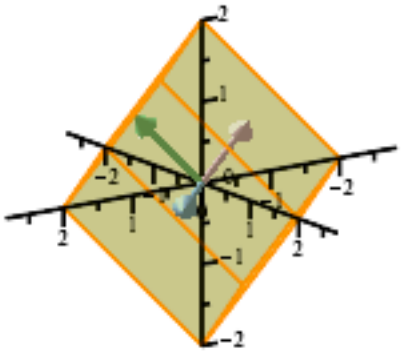



```
> A := Matrix([ [1,1,0], [1,0,1], [0,1,1]]);
  QRVis(A);
```

$$A := \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

$$Q = \begin{bmatrix} 0.70711 & 0.40825 & 0.57735 \\ 0.70711 & -0.40825 & -0.57735 \\ 0. & 0.81650 & -0.57735 \end{bmatrix}, R = \begin{bmatrix} 1.41421 & 0.70711 & 0.70711 \\ 0. & 1.22474 & 0.40825 \\ 0. & 0. & -1.15470 \end{bmatrix}$$

$n = 60.$

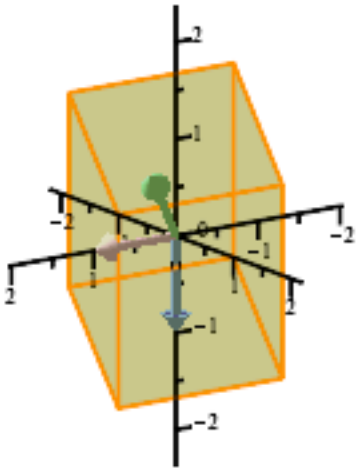



```
> A := Matrix([ [0,1,1], [0,1,0], [-1,1,0]]);
  QRVis(A);
```

$$A := \begin{bmatrix} 0 & 1 & 1 \\ 0 & 1 & 0 \\ -1 & 1 & 0 \end{bmatrix}$$

$$Q = \begin{bmatrix} 0. & 0.70711 & 0.70711 \\ 0. & 0.70711 & -0.70711 \\ -1.00000 & 0. & 0. \end{bmatrix}, R = \begin{bmatrix} 1.00000 & -1.00000 & 0. \\ 0. & 1.41421 & 0.70711 \\ 0. & 0. & 0.70711 \end{bmatrix}$$

$n = 60$.



Singulärwertzerlegung einer reellen 3x3-Matrix:
A=QDR mit orthogonalen Matrizen Q, R und Diagonalmatrix D

```
> Sing := proc(A)
  local QDR,Q,R,D,Q1,R1,D1,i;
  QDR := SingularValues(evalf(A), output=['U','S','Vt']):
  Q1 := QDR[1];
  R1 := QDR[3];
  D1 := Matrix(1..3,1..3);
  for i from 1 to 3 do D1[i,i] := QDR[2][i] od;
  print(Q=Q1,D=D1,R=R1);
  [Q1,D1,R1];
  end proc:
```

```
> A := Matrix([ [2,1,1], [1,2,1], [1,1,6] ]);
  Sing(A):
```

$$A := \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 6 \end{bmatrix}$$

$$Q = \begin{bmatrix} -0.26096 & 0.65719 & 0.70711 \\ -0.26096 & 0.65719 & -0.70711 \\ -0.92941 & -0.36905 & 2.89348\ 10^{-16} \end{bmatrix}, D = \begin{bmatrix} 6.56155 & 0 & 0 \\ 0 & 2.43845 & 0 \\ 0 & 0 & 1.00000 \end{bmatrix}, R \qquad \textbf{(2)}$$

$$= \begin{bmatrix} -0.26096 & -0.26096 & -0.92941 \\ 0.65719 & 0.65719 & -0.36905 \\ 0.70711 & -0.70711 & 5.13839\ 10^{-17} \end{bmatrix}$$

Visualisierung Singulärwertzerlegung:

```
> SingVis := proc(A1);
  local A,QDR,Q,D,D1,R,tQ,SQ,tR,SR,Zeich,v,w,Fig,i,j,n0,n,E,c;
  QDR := Sing(A1);
  Q := QDR[1];
  D1 := QDR[2];
  R := QDR[3];
  # Integriere Spiegelung in Diagonalmatrix
  E := Matrix([[1,0,0],[0,1,0],[0,0,-1]]);
  if Determinant(R)<0 then
  R := MatrixMatrixMultiply(E,R);
  Q := MatrixMatrixMultiply(Q,E);
  fi;
  if Determinant(Q)<0 then
  Q := MatrixMatrixMultiply(Q,E);
  D1 := MatrixMatrixMultiply(E,D1);
  fi;
  # Bestimme Drehachsen und Winkel
  tQ,SQ := Drehung(Q);
  tR,SR := Drehung(R);
  for i from 1 to 3 do
  v[i] := Transpose(Transpose(R)[i]);
  c[i] := max(seq(abs(R[i,j]),j=1..3));
  od;
  # Achsen am Ende
  for i from 1 to 3 do
  w[i] := Transpose(Transpose(A1)[i]);
  od;
```

```
# Zeichne Figur nach Transformation durch T
Zeich := proc(T)
local Punkt,Kante,Achsen,EndAchsen,Kanten,Seiten,i;
Punkt := proc(v)
convert(MatrixVectorMultiply(T,Vector(v)),list)
end proc:
Kante := proc(v1,v2)
line(Punkt(v1),Punkt(v2),color=coral,linestyle=solid)
end proc:
EndAchsen := {PV(w[1],"CornflowerBlue",3),
              PV(w[2],aquamarine,3),
              PV(w[3],plum,3)}:
Achsen := {PV(MatrixVectorMultiply(T,v[1]),navy,2),
           PV(MatrixVectorMultiply(T,v[2]),"DarkCyan",2),
           PV(MatrixVectorMultiply(T,v[3]),"DarkViolet",2)}:

Kanten := {Kante([ c[1], c[2], c[3]],[ c[1], c[2],-c[3]]),
           Kante([ c[1], c[2],-c[3]],[ c[1],-c[2],-c[3]]),
           Kante([ c[1],-c[2],-c[3]],[ c[1],-c[2], c[3]]),
           Kante([ c[1],-c[2], c[3]],[ c[1], c[2], c[3]]),
           Kante([ c[1], c[2], c[3]],[-c[1], c[2], c[3]]),
           Kante([ c[1], c[2],-c[3]],[-c[1], c[2],-c[3]]),
           Kante([ c[1],-c[2],-c[3]],[-c[1],-c[2],-c[3]]),
           Kante([ c[1],-c[2], c[3]],[-c[1],-c[2], c[3]]),
           Kante([-c[1], c[2], c[3]],[-c[1], c[2],-c[3]]),
           Kante([-c[1], c[2],-c[3]],[-c[1],-c[2],-c[3]]),
           Kante([-c[1],-c[2],-c[3]],[-c[1],-c[2], c[3]]),
           Kante([-c[1],-c[2], c[3]],[-c[1], c[2], c[3]])}:
Seiten := {plot3d({Punkt([ s*c[1], t*c[2],   c[3]]),
                   Punkt([ s*c[1], t*c[2],  -c[3]]),
                   Punkt([ s*c[1],   c[2], t*c[3]]),
                   Punkt([ s*c[1],  -c[2], t*c[3]]),
                   Punkt([   c[1], s*c[2], t*c[3]]),
                   Punkt([  -c[1], s*c[2], t*c[3]])},
                  s=-1..1,t=-1..1,
                  transparency=.7,color=yellow,style=surface)}:
display(Kanten union Seiten union Achsen union EndAchsen,axes=
normal);
end proc:
# Fallunterscheidung
Fig := proc(n)
local E;
if n<n0 then
Zeich(MatrixMatrixMultiply(
       MatrixMatrixMultiply(SR,
        Matrix([ [1,0,0],
                 [0, cos((n/n0-1)*tR),-sin((n/n0-1)*tR)],
                 [0, sin((n/n0-1)*tR), cos((n/n0-1)*tR)]])),
       Transpose(SR)));
elif n<2*n0 then
E := IdentityMatrix(3);
Zeich(E+(n/n0-1)*(D1-E));
else
Zeich(MatrixMatrixMultiply(
       MatrixMatrixMultiply(SQ,
        Matrix([ [1,0,0],
                 [0, cos((n/n0-2)*tQ),-sin((n/n0-2)*tQ)],
                 [0, sin((n/n0-2)*tQ), cos((n/n0-2)*tQ)]])),
       MatrixMatrixMultiply(Transpose(SQ),D1)));
fi;
end proc:
```

```
# Figur zeichnen
n0 := 20;
animate(Fig,[n],n=[seq(i,i=0..3*n0)]);
end proc:
```

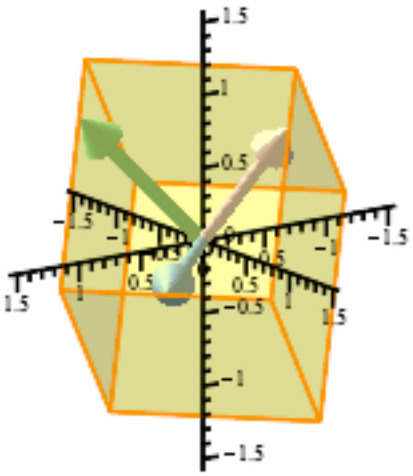
```
> A := Matrix([ [1,1,0], [1,0,1], [0,1,1]]);
SingVis(A);
```

$$A := \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

$$Q = \begin{bmatrix} -0.57735 & 0.40825 & 0.70711 \\ -0.57735 & 0.40825 & -0.70711 \\ -0.57735 & -0.81650 & 2.22045\ 10^{-16} \end{bmatrix}, \mathrm{D} = \begin{bmatrix} 2.00000 & 0 & 0 \\ 0 & 1.00000 & 0 \\ 0 & 0 & 1.00000 \end{bmatrix}, R$$

$$= \begin{bmatrix} -0.57735 & -0.57735 & -0.57735 \\ 0.81650 & -0.40825 & -0.40825 \\ -0. & 0.70711 & -0.70711 \end{bmatrix}$$

$n = 60.$



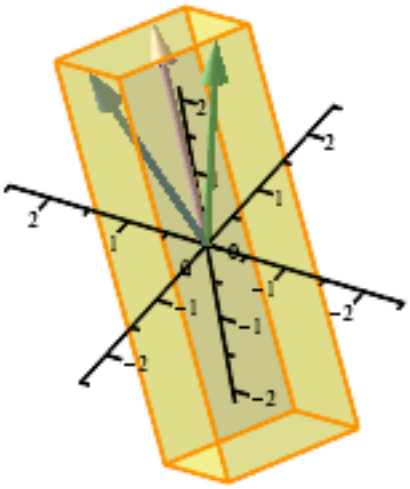

```
> A := Matrix([ [2,1,1], [1,2,1], [1,1,2]]);
SingVis(A);
```

$$A := \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}$$

$$Q = \begin{bmatrix} -0.57735 & -5.04179\,10^{-17} & 0.81650 \\ -0.57735 & -0.70711 & -0.40825 \\ -0.57735 & 0.70711 & -0.40825 \end{bmatrix}, \mathrm{D} = \begin{bmatrix} 4.00000 & 0 & 0 \\ 0 & 1.00000 & 0 \\ 0 & 0 & 1.00000 \end{bmatrix}, R$$

$$= \begin{bmatrix} -0.57735 & -0.57735 & -0.57735 \\ 0. & -0.70711 & 0.70711 \\ 0.81650 & -0.40825 & -0.40825 \end{bmatrix}$$

$n = 60.$



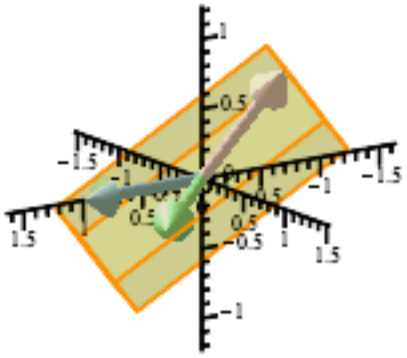
```
> A := Matrix([ [1,1,0], [0,1,1], [0,0,1]]);
  SingVis(A);
```

$$A := \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

$$Q = \begin{bmatrix} 0.59101 & -0.73698 & 0.32799 \\ 0.73698 & 0.32799 & -0.59101 \\ 0.32799 & 0.59101 & 0.73698 \end{bmatrix}, \mathrm{D} = \begin{bmatrix} 1.80194 & 0 & 0 \\ 0 & 1.24698 & 0 \\ 0 & 0 & 0.44504 \end{bmatrix}, R$$

$$= \begin{bmatrix} 0.32799 & 0.73698 & 0.59101 \\ -0.59101 & -0.32799 & 0.73698 \\ 0.73698 & -0.59101 & 0.32799 \end{bmatrix}$$

$n = 60.$




```
> A := Matrix([ [7,sqrt(2),-1], [4,-2,3], [2.5,exp(1),5]]);
  SingVis(A);
```

$$A := \begin{bmatrix} 7 & \sqrt{2} & -1 \\ 4 & -2 & 3 \\ 2.50000 & \mathrm{e} & 5 \end{bmatrix}$$

$$Q = \begin{bmatrix} -0.69731 & 0.65618 & 0.28841 \\ -0.50054 & -0.15778 & -0.85121 \\ -0.51304 & -0.73792 & 0.43847 \end{bmatrix}, \mathrm{D} = \begin{bmatrix} 8.94092 & 0 & 0 \\ 0 & 5.31869 & 0 \\ 0 & 0 & 3.37795 \end{bmatrix}, R$$

$$= \begin{bmatrix} -0.91332 & -0.15431 & -0.37687 \\ 0.39809 & -0.14333 & -0.90608 \\ -0.08580 & 0.97757 & -0.19234 \end{bmatrix}$$

$n = 60$.