

Ellipse 4 in allgemeiner Lage

Prof. Dr. Dörte Haftendorn: Mathematik mit MuPAD 4 Juli 07 Update 11.07.07

Web: <http://haftendorn.uni-lueneburg.de> www.mathematik-verstehen.de

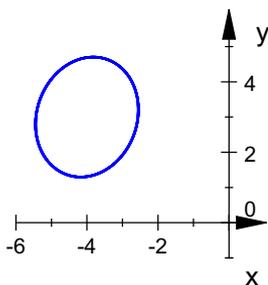
#####

Ganzzahlige Eigenwerte,

```
keg:=matrix([[29*x^2 - 6*x*y + 250*x + 21*y^2 - 150*y + 665]])
```

$$(29 \cdot x^2 - x \cdot y \cdot 6 + 250 \cdot x + 21 \cdot y^2 - y \cdot 150 + 665)$$

```
kegp:=plot::Implicit2d(keg[1]=0,x=-6..0,y=-1..5):
plot(kegp,Scaling=Constrained):
```



Dieser Kegelschnitt ist offenbar eine Ellipse.

Durchführung einer Hauptachsentransformation

```
A:=matrix([[29,-3],[-3,21]]);
a:=matrix([250,-150]): at:=linalg::transpose(a);
d:=665;
```

$$\begin{pmatrix} 29 & -3 \\ -3 & 21 \end{pmatrix}$$

$$(250 \quad -150)$$

$$665$$

Bestimmung der Achsenrichtungen:

```
evli:=linalg::eigenvectors(A)
```

$$\left[\left[20, 1, \begin{pmatrix} 1 \\ 3 \\ 1 \end{pmatrix} \right], \left[30, 1, \begin{pmatrix} -3 \\ 1 \end{pmatrix} \right] \right]$$

Eigenwerte mit ihrer Vielfachheit und ihrem Eigenvektor.

```
ev1:=evli[1][3][1];
```

```
ev2:=evli[2][3][1];
```

$$\begin{pmatrix} 1 \\ 3 \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} -3 \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} -3 \\ 1 \end{pmatrix}$$

```
ew1:=evli[1][1];
ew2:=evli[2][1];
```

20

30

```
linalg::det(ev1.ev2)
```

$\frac{5}{2}$

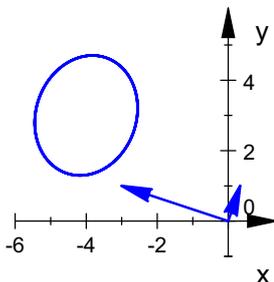
Wenn diese Determinante negativ ist, tauscht man die EV besser um.
Oder man nimmt einen der EV andersherum, Rechtssystem ist günstig.
Anderenfalls ist noch eine Spiegelung im Spiel.

```
float(ev1), float(ev2)
```

$$\begin{pmatrix} 0.3333333333 \\ 1.0 \end{pmatrix}, \begin{pmatrix} -3.0 \\ 1.0 \end{pmatrix}$$

Einzeichnen der Eigenvektoren:

```
ev1p:=plot::Arrow2d(ev1):
ev2p:=plot::Arrow2d(ev2):
plot(kegp, ev1p, ev2p, Scaling=Constrained)
```



Wie erwartet sind die Eigenvektoren die Richtungen der Hauptachsen des Kegelschnittes.

Bestimmung der Kegelschnittgleichung

Die Theorie sagt, dass die Eigenwerte nun die Faktoren der quadratischen Glieder werden.

Das heißt hier, dass es zwei quadratische Glieder mit positivem Vorzeichen gibt.

Damit muss es sich um eine Ellipse handeln, wenn es nicht ein entarteter Sonderfall wird.

Da wir oben schon gezeichnet haben, war ja schon klar, dass es eine Ellipse ist.

Soll die gegebene blaue Ellipse in Hauptachsenlage abgebildet werden,

so kann das geschehen durch eine Drehung um den Ursprung, evt. verknüpft mit einer Achsenspiegelung an einer der Koordinatenachsen, gefolgt von einer Verschiebung.

$$\vec{p}' = P^T \vec{p} \quad \text{und} \quad \vec{p}'' = \vec{p}' + \vec{t}$$

Aus den normierten Eigenvektoren erstellt man eine Matrix P.

Die Eigenvektoren stehen gleich von alleine senkrecht aufeinander, wenn die Eigenwerte verschieden sind.

2

P ist dann eine Orthonormal-Matrix. Für solche ist die inverse Matrix

gleich der transponierten Matrix, bezeichnet mit P^t (oder mit hochgestelltem T).

Die Transformationsmatrix, die den Kegelschnitt in eine Lage bewegt, bei der die

Die Transformationsmatrix, die den Kegelschnitt in eine Lage bewegt, bei der die Hauptachsen parallel zu den Koordinatenachsen sind, ist P^t . Die Theorie sagt nun, dass $P^t \cdot A \cdot P$ die Diagonalmatrix aus den Eigenwerten ist. Davon kann man sich hier überzeugen:

```
ev1n:=linalg::normalize(ev1):
ev2n:=linalg::normalize(ev2):
P:=ev1n.ev2n
```

$$\begin{pmatrix} \frac{\sqrt{10}}{10} & -\frac{3 \cdot \sqrt{10}}{10} \\ \frac{3 \cdot \sqrt{10}}{10} & \frac{\sqrt{10}}{10} \end{pmatrix}$$

```
float(P)
```

$$\begin{pmatrix} 0.316227766 & -0.9486832981 \\ 0.9486832981 & 0.316227766 \end{pmatrix}$$

```
Pt:=linalg::transpose(P)
```

$$\begin{pmatrix} \frac{\sqrt{10}}{10} & \frac{3 \cdot \sqrt{10}}{10} \\ -\frac{3 \cdot \sqrt{10}}{10} & \frac{\sqrt{10}}{10} \end{pmatrix}$$

```
float(Pt)
```

$$\begin{pmatrix} 0.316227766 & 0.9486832981 \\ -0.9486832981 & 0.316227766 \end{pmatrix}$$

Diagonalisierung, die klappt immer, wenn die EV eine ONB bilden, wenn man also so ein P aufstellen kann.

```
Simplify(Pt*A*P);
float(Pt*A*P)
```

$$\begin{pmatrix} 20 & 0 \\ 0 & 30 \end{pmatrix}$$

$$\begin{pmatrix} 20.0 & 0 \\ 0 & 30.0 \end{pmatrix}$$

Evt. winzige 10^{-19} Werte sind numerisch 0.

Vektorschreibweise für die Abbildung $\vec{p} = P \vec{p}'$ und die Quadrikgleichungen, die sich durch Einsetzen ergeben:

$$Q: \vec{p}'^T A \vec{p}' + \vec{a}'^T \vec{p}' + d = 0$$

$$Q: \vec{p}'^T P^T A P \vec{p}' + \vec{p}'^T P^T \vec{a} + d = 0$$

$$Q: \vec{p}^T A \vec{p} + \vec{a}^T \vec{p} + d = 0$$

$$Q': \vec{p}'^T P^T A P \vec{p}' + \vec{a}^T P \vec{p}' + d = 0$$

$$Q': \vec{p}'^T D_{EW} \vec{p}' + \vec{a}^T P \vec{p}' + d = 0$$

$$\text{mit } D_{EW} = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}$$

Abbildung der linearen Terme erfordert also

at

$$(250 \quad -150)$$

atb:=at*P

$$(-20 \cdot \sqrt{10} \quad -90 \cdot \sqrt{10})$$

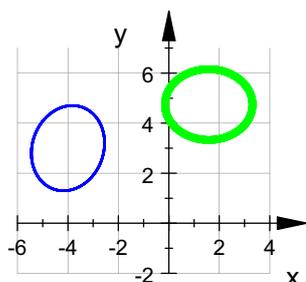
kegSD:=ew1*x^2+ew2*y^2+atb[1]*x+atb[2]*y+d

$$20 \cdot x^2 - \sqrt{10} \cdot x \cdot 20 + 30 \cdot y^2 - \sqrt{10} \cdot y \cdot 90 + 665$$

kegSDp:=plot::Implicit2d(kegSD=0,x=-4..4,y=-2..7,

LineWidth=1,LineColor=[0,1,0],GridVisible=TRUE):

plot(kegp,kegSDp,Scaling=Constrained)



Bei der Betrachtung dieses Bildes fällt auf, dass es sich um eine negative Drehung um den Ursprung handelt.

Pt, float(Pt);

Dr:=phi->matrix([[cos(phi), -sin(phi)],

[sin(phi), cos(phi)]]): Dr(`φ`);

$$\begin{pmatrix} \frac{\sqrt{10}}{10} & \frac{3 \cdot \sqrt{10}}{10} \\ -\frac{3 \cdot \sqrt{10}}{10} & \frac{\sqrt{10}}{10} \end{pmatrix}, \begin{pmatrix} 0.316227766 & 0.9486832981 \\ -0.9486832981 & 0.316227766 \end{pmatrix}$$

4

$$\begin{pmatrix} \cos(\varphi) & -\sin(\varphi) \\ \sin(\varphi) & \cos(\varphi) \end{pmatrix}$$

$$\begin{pmatrix} \cos(\varphi) & -\sin(\varphi) \\ \sin(\varphi) & \cos(\varphi) \end{pmatrix}$$

Der Vergleich mit der allgemeinen Drehmatrix ergibt, dass es mit dem Anpassen von phi klappen kann:

```
cos(phi)=float(Pt[1,1]); sin(phi)=float(Pt[2,1]);
```

$$\cos(\text{phi}) = 0.316227766$$

$$\sin(\text{phi}) = -0.9486832981$$

```
arccos(Pt[1,1]);
```

```
wi:=arcsin(Pt[2,1]);
```

$$\arccos\left(\frac{\sqrt{10}}{10}\right)$$

$$-\arcsin\left(\frac{3 \cdot \sqrt{10}}{10}\right)$$

Daraus folgt umgerechnet ins Winkelmaß:

```
float(arccos(Pt[1,1])/PI*180);
```

```
wig:=float(arcsin(Pt[2,1])/PI*180);
```

$$71.56505118$$

$$-71.56505118$$

Damit kann die gemeinsame Lösung nur der negative Winkel wi sein.

So passt es auch zur Zeichnung.

Einzeichnen des Zwischenbildes:

```
p:=matrix([x,y]);
```

```
pt:=linalg::transpose(p);
```

$$\begin{pmatrix} x \\ y \end{pmatrix}$$

$$(x \ y)$$

```
float(Dr(wi)), float(Pt)
```

$$\begin{pmatrix} 0.316227766 & 0.9486832981 \\ -0.9486832981 & 0.316227766 \end{pmatrix}, \begin{pmatrix} 0.316227766 & 0.9486832981 \\ -0.9486832981 & 0.316227766 \end{pmatrix}$$

Beachte, dass die inverse Drehung die ist, die um den negativen Winkel dreht.

```
kegD:=simplify(pt*Dr(wi)*A*Dr(-wi)*p+at*Dr(-wi)*p+d);
```

$$\left(20 \cdot x^2 - \sqrt{10} \cdot x \cdot 20 + 30 \cdot y^2 - \sqrt{10} \cdot y \cdot 90 + 665 \right)$$

Dieses hat man ja auch als transformierte Gleichung erhalten.

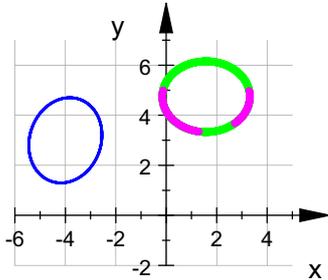
```
kegSD
```

$$20 \cdot x^2 - \sqrt{10} \cdot x \cdot 20 + 30 \cdot y^2 - \sqrt{10} \cdot y \cdot 90 + 665$$

$$20 \cdot x^2 - \sqrt{10} \cdot x \cdot 20 + 30 \cdot y^2 - \sqrt{10} \cdot y \cdot 90 + 665$$

```
kegDp:=plot::Implicit2d(kegD[1],x=-4..5,y=-2..5,
```

```
LineWidth=1,LineColor=[1,0,1],LineStyle=Dashed,GridVisible=TRUE):
plot(kegp,kegSDp,kegDp,Scaling=Constrained)
```



#####

Bestimmung der Translation:

```
kegSD
```

$$20 \cdot x^2 - \sqrt{10} \cdot x \cdot 20 + 30 \cdot y^2 - \sqrt{10} \cdot y \cdot 90 + 665$$

Die Arbeitsweise ist dieselbe wie bei der Herstellung der Scheitelform einer Parabel.
Hier durch Hinsehen:

```
xterm:=hold(20*(x-1/2*sqrt(10))^2);expand(xterm);
yterm:=hold(30*(y-3/2*sqrt(10))^2);expand(yterm);
```

$$20 \cdot \left(x - \frac{\sqrt{10}}{2}\right)^2$$

$$20 \cdot x^2 - \sqrt{10} \cdot x \cdot 20 + 50$$

$$30 \cdot \left(y - \frac{3 \cdot \sqrt{10}}{2}\right)^2$$

$$30 \cdot y^2 - \sqrt{10} \cdot y \cdot 90 + 675$$

```
dd:=-50-675+d
```

```
-60
```

Also

```
kegSDK:=xterm+yterm+dd
```

$$20 \cdot \left(x - \frac{\sqrt{10}}{2}\right)^2 + 30 \cdot \left(y - \frac{\sqrt{10} \cdot 3}{2}\right)^2 - 60$$

```
expand(kegSDK) //Probe
```

$$20 \cdot x^2 - \sqrt{10} \cdot x \cdot 20 + 30 \cdot y^2 - \sqrt{10} \cdot y \cdot 90 + 665$$

6

passt.

Letzter Teil der Hauptachsentransformation ist die Translation t

Translation t

```
t:=matrix([ -1/2*sqrt(10) , -3/2*sqrt(10) ])
```

$$\begin{pmatrix} -\frac{\sqrt{10}}{2} \\ -\frac{3 \cdot \sqrt{10}}{2} \end{pmatrix}$$

$\vec{p}'' = \vec{p}' + \vec{t}$ also $\vec{p}' = \vec{p}'' - \vec{t}$ Das ergibt:

```
kegH:=ew1*x^2+ew2*y^2+dd
```

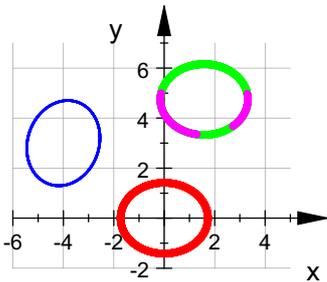
$$20 \cdot x^2 + 30 \cdot y^2 - 60$$

Angabe der Gleichung in der üblichen Form:

```
hold(x^2/3+y^2/2=1)
```

$$\frac{x^2}{3} + \frac{y^2}{2} = 1$$

```
kegHp:=plot::Implicit2d(kegH,x=-4..4,y=-2..4,
LineWidth=1,LineColor=[1,0,0],GridVisible=TRUE):
plot(kegp,kegSDp,kegDp,kegHp,Scaling=Constrained)
```



Bestimmung des ursprünglichen Scheitelpunktes:

$$\vec{m}'' = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad \vec{m}' = -\vec{t}, \quad \vec{m} = P(-\vec{t})$$

```
m:=P*(-t)
```

$$\begin{pmatrix} -4 \\ 3 \end{pmatrix}$$

Bestimmung des Urbildes des eines Punktes der Parabelachse:

$$\vec{r}'' = \begin{pmatrix} r \\ 0 \end{pmatrix}, \quad \vec{r}' = \begin{pmatrix} r \\ 0 \end{pmatrix} - \vec{t}, \quad \vec{r} = P \begin{pmatrix} r \\ 0 \end{pmatrix} + \vec{m}, \quad r = r \cdot \overrightarrow{ev_1} + \vec{m}$$

Mit ev1 rechts ist hier der normierte 1. Eigenvektor gemeint.P

7

```
r:=sqrt(3): //auf der x-Achse
rur:=r*ev1+m;
```

$$\begin{pmatrix} \frac{\sqrt{3} \cdot \sqrt{10}}{10} - 4 \\ \frac{3 \cdot \sqrt{3} \cdot \sqrt{10}}{10} + 3 \end{pmatrix}$$

$$\begin{pmatrix} \frac{\sqrt{3} \cdot \sqrt{10}}{10} - 4 \\ \frac{3 \cdot \sqrt{3} \cdot \sqrt{10}}{10} + 3 \end{pmatrix}$$

```

mp:=plot::Point2d(m, PointColor=[0,1,1]):
msp:=plot::Point2d(-t, PointColor=[0,1,1]):
rurp:=plot::Point2d(rur):
rp:=plot::Point2d([r,0]):
evlurp:=plot::Arrow2d(m,m+ev1):
ev2urp:=plot::Arrow2d(m,m+ev2):
tp:=plot::Arrow2d(-t, [0,0]):
plot(kegp,kegSDp,kegDp,kegHp, mp,msp,rurp,rp,tp,
     evlurp,ev2urp,PointSize=2,Scaling=Constrained)

```

