## 3.4 Raumkurven (Koordinatensysteme)

### 3.4.8 Übung

Bevor wir zum eigentlichen Thema kommen, wollen wir noch ein nützliches Beispiel betrachten. Es geht um die Projektion einer Trajektorie auf die Raumebenen. Wir stellen uns vor, daß ein Objekt sich im R3 gemäß der folgenden Funktion bewegt:

$$\vec{r}(t) = 6t \cdot \vec{i} + (5t + 4) \cdot \vec{j} + 6t^2 \cdot \vec{k}$$

$t$ ist die Zeit in Sekunden, und der Weg sei in Metern gemessen.

a. Stelle die Bahn graphisch dar sowie ihre Projektionen auf die Koordinatenebenen.
b. Bestimme Geschwindigkeit und Beschleunigung.
c. Wie lauten die Werte von $\vec{r}, \vec{v}, \vec{a}$ im Augenblick $t = 3s$?
d. Welche Strecke legt das Objekt in 3s zurück? (Vgl. Abschnitt 2.2.7; dort wird die Formel für die Bogenlänge angegeben: $s = \int_a^b |\vec{r}'(t)|\, dt$.)

**Lösung**:

```
reset():
a:=0:b:=1:
x:=t->6*t:y:=t->5*t+4:z:=t->6*t^2:
kurve:=plot::Curve3d([x(t),y(t),z(t)],t=a..b):
projexz:=plot::Curve3d([x(t),0,z(t)],Color=RGB::Red,t=a..b):
projexy:=plot::Curve3d([x(t),y(t),0],Color=RGB::Green,t=a..b):
projeyz:=plot::Curve3d([0,y(t),z(t)],Color=RGB::Magenta,t=a..b):
plot(kurve,projexz,projexy,projeyz, Scaling=Constrained)
```

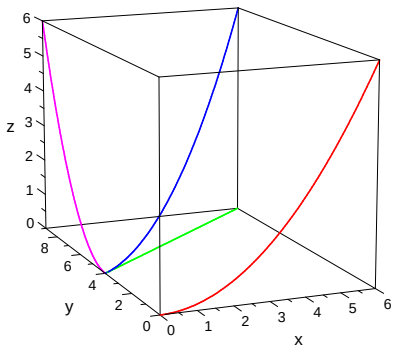


Fig.3.4-23

```
reset():
x:=t->6*t:
y:=t->5*t+4:
z:=t->6*t^2:
t1:=3:
pos:=matrix([[x(t),y(t),z(t)]]);
v:=matrix([[x'(t),y'(t),z'(t)]]);
a:=matrix([[x''(t),y''(t),z''(t)]]);
ds:=sqrt(v[1,1]^2+v[1,2]^2+v[1,3]^2);
bogen:=numeric::int(ds,t=0..t1)
```

$(3t, 5t+4, 6t^2)$ Position
$(6, 5, 12t)$ Geschwindigkeit
$(0, 0, 12)$ Beschleunigung

$(144t^2+61)^{1/2}$ Weg nach $t$ Sekunden
60.93.. $m$ Weg nach $3s$.

### 3.4.9 Polarkoordinaten und Kreisbewegung

Wir wollen uns jetzt erneut mit der ebenen Bewegung eines Teilchens beschäftigen. Zur Beschreibung dieser Bewegung werden wir uns aber der sogenannten *Polarkoordinaten* bedienen. Wie die folgende Figur zeigt, handelt es sich dabei um das Paar (r,$\phi$), das den momentanen Ort eines Teilchens eindeutig in der Ebene festlegt. Dem Teilchen sind außerdem die beiden orthogonalen Einheitsvektoren $\vec{e}_r, \vec{e}_\phi$ beigegeben, die von den positiven Richtungen von r und $\phi$ bestimmt sind. Der Winkel $\phi$ wird in Radiant gemessen.

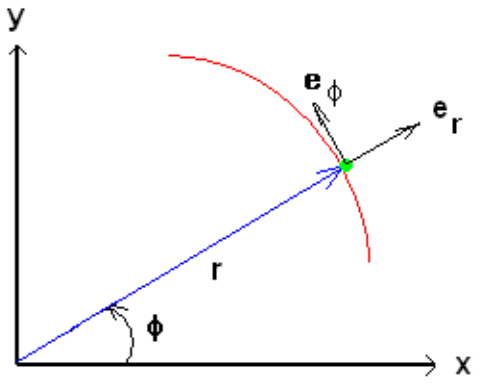


Fig.3.4-24

Der Ortsvektor $\vec{r}$ des Teilchens ist festgelegt durch

$$\vec{r} = r\vec{e}_r \tag{1}$$

Da $r, \vec{r}$ und $\vec{e}_r$ Funktionen der Zeit sind, können wir auch ihre Ableitungen bilden

$$\vec{v} = \frac{d\vec{r}}{dt} = \frac{dr}{dt}\vec{e}_r + \frac{d\vec{e}_r}{dt}r$$

Um die Ableitung von $\vec{e}_r$ zu bilden, zerlegen wir es - wie wir es bereits für $\vec{t}$ und $\vec{n}$ in 3.4.3 taten- in rechtwinklige Komponenten

$$\begin{array}{rcl} \vec{e}_r & = & \cos(\phi)\vec{i} + \sin(\phi)\vec{j} \\ \vec{e}_\phi & = & \cos(\phi + \pi/2)\vec{i} + \sin(\phi + \pi/2)\vec{j} \\ & = & -\sin(\phi)\vec{i} + \cos(\phi)\vec{j} \end{array} \tag{2}$$

Mit diesen Komponenten erhalten wir dann

$$\frac{d\vec{e}_r}{dt} = -\sin(\phi)\frac{d\phi}{dt}\vec{i} + \cos(\phi)\frac{d\phi}{dt}\vec{j} = \frac{d\phi}{dt}\vec{e}_\phi \tag{3a}$$

Vgl. mit Gl.(10) in 3.4.3.

Für die Ableitung von $\vec{e}_\phi$ ergibt sich

$$\frac{d\vec{e}_\phi}{dt} = -\cos(\phi)\frac{d\phi}{dt}\vec{i} - \sin(\phi)\frac{d\phi}{dt}\vec{j} = -\frac{d\phi}{dt}\vec{e}_r \tag{3b}$$

Nun können wir die Geschwindigkeit des Teilchens durch $\vec{v} = \vec{v}_r + \vec{v}_\phi$ ausdrücken, das bedeutet

$$\vec{v} = \frac{dr}{dt}\vec{e}_r + r\frac{d\phi}{dt}\vec{e}_\phi \tag{4}$$

Der erste Teil dieser Gleichung ist die *radiale Geschwindigkeit.* (Die r-Komponente von $\vec{v}$ ist die Geschwindigkeit, mit der der Vektor $\vec{r}$ sich verändert.) Der zweite Teil ist ein zu $\vec{r}$ senkrechter Vektor, der die Richtungsänderung von $\vec{r}$ angibt. Dieser Anteil heißt *transversale Geschwindigkeit.* $\vec{\omega} := \frac{d\vec{\phi}}{dt}$ ist die Winkelgeschwindigkeit.

Bei der **Kreisbewegung** ist die Radialgeschwindigkeit Null, weil der Radius konstant ist, d.h. $\frac{dr}{dt} = 0$. Die Geschwindigkeit ist hier also vollständig transversal: $v = \omega \cdot r$.

Jetzt werden wir Gleichung (4) ableiten, um zur Beschleunigung zu kommen. Die Ableitung von $r\frac{d\phi}{dt}\vec{e}_\phi$ wird drei Terme erzeugen, da alle Faktoren Funktionen der Zeit sind. Es ergbit sich

$$\vec{a} = \frac{d\vec{v}}{dt} = (\ddot{r}\vec{e}_r + \dot{r}\frac{d\vec{e}_r}{dt}) + (\dot{r}\dot{\phi}\vec{e}_\phi + r\ddot{\phi}\vec{e}_\phi + r\dot{\phi}\frac{d\vec{e}_\phi}{dt})$$

Wenn wir die beiden Gln.(3) berücksichtigen und die Terme ein wenig sortieren, erhalten wir

$$\vec{a} = (\ddot{r} - r\dot{\phi}^2)\vec{e}_r + (r\ddot{\phi} + 2\dot{r}\dot{\phi})\vec{e}_f \tag{5}$$

Dies können wir auch folgendermaßen schreiben

$$\vec{a} = \vec{a}_r + \vec{a}_\phi = a_r\vec{e}_r + a_\phi\vec{e}_f$$

Der Betrag der Beschleunigung ist $a = (a_r^2 + a_\phi^2)^{1/2}$.
Im Falle der *Kreisbewegung* gibt es weder $\dot{r}$ noch $\ddot{r}$ und wir haben

$$\vec{a} = \vec{a}_r + \vec{a}_\phi = -Rw^2\vec{e}_r + R\alpha\vec{e}_\phi \tag{6}$$

$R$ = Kreisradius (ist gleich $r\sin(\beta)$, vgl. die folgende Figur)
$\omega := \frac{d\phi}{dt}$ =Winkelgeschwindigkeit in $Rad/s$ (meist einfach $s^{-1}$)
$\alpha := \frac{d\omega}{dt} = \frac{d^2\phi}{dt^2}$ = Winkelbeschleunigung in $Rad/s^2$ oder $s^{-2}$
$-Rw^2\vec{e}_r$ = Zentripetalbeschleunigung
$R\alpha\vec{e}_\phi$ = Tangentialbeschleunigung

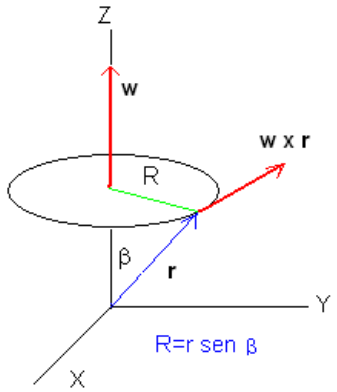


Fig.3.4-25

Beachte, dass es bei der *gleichförmigen* Kreisbewegung ($\alpha = 0$) zwar keine Tangentialbeschleunigung gibt, wohl aber gibt es eine Zentripetalbeschleunigung, die für die Änderung der Geschwindigkeitsrichtung sorgt. Die Winkelgeschwindigkeit kann als eine Vektorgröße ausgedrückt werden, die senkrecht auf der Bewegungsebene steht und die Richtung einer rechtsgängigen Schraube hat, die entsprechend der Teilchenbewegung gedreht wird. In der Abbildung zeigt $\vec{\omega}$ in Richtung der z-Achse, d.h. $\vec{\omega} = \dot{\phi}\vec{k}$. Anstelle der Gleichung $v = \omega R$ können wir schreiben $v = \omega r\sin(\beta)$, was gleichzeitig der Betrag der Vektorgleichung

$$\vec{v} = \vec{\omega} \times \vec{r} \tag{7}$$

ist. Diese Gleichung ist nur gültig für den Fall, dass $r$ und $\beta$ beide konstant sind. In dem Spezialfall mit $\vec{\omega} = konst.$ liegt eine gleichförmige Kreisbewegung vor.

### 3.4.10 Zylindrische und sphärische Koordinaten

Man erhält *Zylinderkoordinaten*, wenn man zwei der drei rechtwinkligen Koordinaten gegen Polarkoordinaten austauscht, z.B. $(x, y)$ gegen $(r, \phi)$. Ein Raumpunkt wird hier also durch

$$P(r\cos(\phi), r\sin(\phi), z) \tag{8}$$

gekennzeichnet. Zwischen den kartesischen Koordinaten und den sphärischen Koordinaten (= räumliche Polarkoordinaten) bestehen folgende Zusammenhänge

$$\begin{aligned} x &= r\sin(\theta)\cos(\phi) \\ y &= r\sin(\theta)\sin(\phi) \\ z &= r\cos(\theta) \end{aligned} \tag{9}$$

$r$ ist der Abstand des Punktes P vom Ursprung (Pol), und $\theta$ ist der Winkel zwischen $OP$ und der dritten Achse. (Oft ist $\theta$ als Winkel zwischen $OP$ und der xy-Polarebene definiert.) Für die Winkel gilt $0 <= \phi < 2\pi; 0 <= \theta < \pi$.

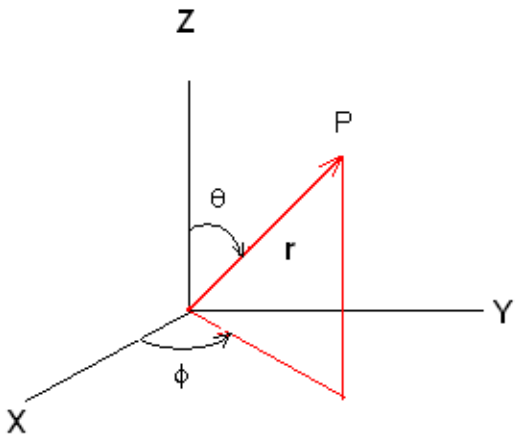


Fig.3.4-26

Zur Angabe eines Punktes der Erdoberfläche benutzt man *geographische Koordinaten.* Hier ist $r$ konstant (= Erdradius), $\phi$ die geographische Länge und $\theta$ die geographische Breite. Die geographische Länge bezeichnet man in der Regel mit $\lambda$, die geografische Breite bezeichnet man meist mit $\beta$.

Die MUPAD-Bibliothek `linalg` enthält die Funktion `linalg::ogCoordTab`, mit deren Hilfe wir die Rechnungen ausführen können, die bei Koordinatentransformationen durchzuführen sind.

Mit der folgenden Anweisung erhalten wir die Transformationsformeln (9)

```
linalg::ogCoordTab[Spherical, Transformation](r, phi, theta)
```

Die umgekehrte Transformation führen wir aus mit

```
linalg::ogCoordTab[Spherical, InverseTransformation](x,y,z)
```

Im Falle von Zylinderkoordinaten benutzen wir

```
linalg::ogCoordTab[Cylindrical, Transformation](r,phi,z)
```

$[r * cos(phi), r * sin(phi), z]$
Die inversen Zylinderkoordinaten erhalten wir mit

```
linalg::ogCoordTab[Cylindrical, InverseTransformation](x,y,z)
```

Hier sind einige **numerische** Beispiele:

1. Die Zylinderkoordinaten $r = 8, \phi = 2\pi/3, z = -3$ sollen in kartesische Koordinaten umgewandelt werden.

```
linalg::ogCoordTab[Cylindrical,Transformation](8,2*PI/3,-3)
```

Ergebnis: $[-4, 4 \cdot \sqrt{3}, -3]$
2. Berechne die inverse Transformation.

```
linalg::ogCoordTab[Cylindrical,InverseTransformation]
(-4,4*sqrt(3),-3)
```

Ergebnis: $[8, \frac{2\pi}{3}, -3]$
3. Wie lauten die kartesischen Koordinaten zu den Kugelkoordinaten $(1, \pi/4, -\pi/2)$?

```
linalg::ogCoordTab[Spherical, Transformation]
(1,-PI/2,PI/4)
```

Ergebnis: $[0, -\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}]$
4. In kartesischen Koordinaten lautet eine Fläche $x \cdot z = 1$. Welches ist die Form in Kugelkoordinaten?
Zunächst verlangen wir die Transformationsformeln

```
reset():
fl:=x*z=1://Flaeche
linalg::ogCoordTab[Spherical, Transformation]
(r,phi,theta);
```

Ergebnis:`[r cos(phi) sin(theta),r sin(phi) sin(theta),`
`r cos(theta)]`

Jetzt tauschen wir die Koordinaten aus:

```
flaeche:=subs(fl,x=%[1],z=%[3]);
```

Ergebnis: $r^2\cos(phi)\cos(theta)\sin(theta) = 1$

Jetzt setzen wir `combine` ein, um den Term $\cos(theta) * \sin(theta)$ zu vereinfachen

```
combine(cos(theta)*sin(theta),sincos)
```

Ergebnis: $\frac{\sin(2 \cdot theta)}{2}$

Demnach erhalten wir $r^2\cos(phi)\sin(2 \cdot theta)/2 = 1$.

Unsere Fläche $x \cdot z = 1$ hat in Kugelkoordinaten die Gestalt: $r^2\cos(\phi) \cdot \sin(2\theta) = 2$.

Auch für die Fläche $x^2 + y^2 - z^2 = 1$ wollen wir die Gestalt in Kugelkoordinaten suchen:

```
reset():
fl:=x^2+y^2-z^2=1:
linalg::ogCoordTab[Spherical, Transformation](r, phi, theta):
flaeche:=subs(fl,x=%[1],y=%[2],z=%[3]);
combine(%,sincos)
```

Lösung: $-r^2\cos(2\theta) = 1$

Auch beim *Integrieren* haben wir oft das Koordinatensystem zu wechseln. Wir berechnen als Beispiel das folgende Volumenintegral (Dreifach-Integral):

$$I_{zz} = \int_V (x^2 + y^2)\rho dV$$

Es handelt sich um das *Trägheitsmoment* in Bezug auf die z-Achse. Die Dichte $\rho$ soll gleich 1 sein, und die Integration führen wir aus über eine Kugel vom Radius $R$. (Das Volumenelement ist $dV = dxdydz$. In sphärischen Koordinaten ist es $dV = r^2\sin(\theta)drd\phi d\theta$. Die Größe $r^2\sin(\theta)$ ist der Transformationsfaktor des Volumenelements. Dieser Faktor ist im Falle von zylindrischen Koordinaten einfach $r$, denn $dV = r \cdot drd\phi dz$.)

```
fz:=x^2 + y^2://fuer fzz:=1 erhalten wir das Volumen
linalg::ogCoordTab[Spherical, Transformation](r, phi, theta):
fzz:=subs(fz,x=%[1],y=%[2],z=%[3]):
factor(%);
subs(%,cos(phi)^2+sin(phi)^2=1)
```

Ergebnisse: $\sin(theta)^2 r^2(\cos(phi)^2 + \sin(phi)^2)$ und $1 \cdot \sin(theta)^2 \cdot r^2$

Schließlich berechnen wir das Dreifachintegral auf folgende Art

```
faktor:=r^2*sin(theta)://Faktor der Transformation
int(int(int(r^2*sin(theta)^2*faktor, r = 0..R), theta=0..PI),
phi= 0..2*PI)
```

Ergebnis: $8\pi R^5/15$

(Einige Hundert Seiten später, in Abschnitt 6.3.4, werden wir dieses Ergebnis benötigen...)

### 3.4.11 Mit Bleistift und Papier

**Mehrfachintegrale**

Wir haben gerade das Dreifach-Integral $I_{zz} = \int_V (x^2 + y^2)\rho dV$ berechnet. Normalerweise berechnen wir Mehrfachintegrale durch Zurückführung auf einfache Integrale mit nur einer Variablen. $f(x, y)$ sei eine stetige Funktion zweier Veränderlicher, und R sei das Rechteck $a <= x <= b, c <= y <= d$ . Es gilt dann

$$\int_R f dA = \int_c^d \left( \int_a^b f(x, y) dx \right) dy := \int_c^d \int_a^b f(x, y) dx dy \tag{10}$$

Man spricht von einem iterierten Integral. Im inneren Integral wird $y$ als eine Konstante angesehen, anschließend wird das Ergebnis in Bezug auf $y$ integriert. Es ist i.Allg. egal, in welcher Reihenfolge man integriert, wie unser folgendes Beispiel auch bestätigen wird, in dem wir das Volumen der Weinlaube aus dem letzten Abschnitt, Fig. 3.4-22, berechnen wollen.

```
int(int(12-x/4-y/8,y=x..8),y=0..16)
```

und

```
int(int(12-x/4-y/8,y=0..16),x=0..8)
```

ergeben dasselbe Ergebnis, nämlich $1280m^3$
Wir können das Volumen aber auch als dreifaches Integral berechnen

```
int(int(int(1, z = 0..12-x/4-y/8),y=0..16),x=0..8)
```

Das Integrationsgebiet muss nicht unbedingt rechteckig sein, wie das folgende Beispiel zeigt:

Berechne die Masse M einer Metallplatte R, die von der Geraden $y = 2x$ und der Parabel $y = x^2$ begrenzt wird. Die Dichte beträgt $\rho(x, y) = x \cdot y^2$ kg/m$^2$, und die Masse ergibt sich aus

$$M = \int_R \rho(x, y) \cdot dA \tag{11}$$

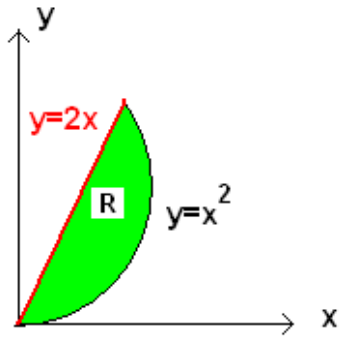

Fig.3.4-27

Zuerst berechnen wir das y-Integral, das unten von $y = x^2$ und oben von $y = 2x$ begrenzt wird. Die Grenzen für x sind $x = 0$ und $x = 2$ (Schnittpunkte von Gerade und Parabel).

$$M = \int_0^2 \int_{x^2}^{2x} xy^2 dy dx$$

Zuerst berechnen wir also das innere Integral mit x = konst.:

$$\int_{x^2}^{2x} xy^2 dy = x\frac{y^3}{3}|_{y=x^2}^{y=2x} = (8x^4 - x^7)/3$$

Die äußere Integration liefert schließlich

$$M = \frac{1}{3}\int_0^2 (8x^4 - x^7)dx = 6\frac{2}{5} \text{ kg}$$

MUPAD macht's natürlich etwas schneller

```
int(int(x*y^2,y=x^2..2*x),x=0..2)
```

Ergebnis: 32/5

In kartesischen Koordinaten war das Flächenelement $dA = dx \cdot dy$. Um die Integration in Polarkoordinaten auszuführen, müssen wir auch $dA$ in Polarkoordinaten ausdrücken. Dazu schreiben wir $dA = ds \cdot dr = r \cdot d\phi \cdot r$.

Um ein Dreifachintegral in Zylinderkoordinaten zu berechnen, haben wir das Volumenelement ebenfalls in Zylinderkoordinaten zu schreiben: $dV = dA \cdot dz = ds \cdot dr \cdot dz = r \cdot d\phi \cdot dr \cdot dz$, d.h. $dV = r dr d\phi dz$.

Es ist leicht, zu zeigen, dass das Volumenelement in Kugelkoordinaten durch $dV = r^2 \sin(\theta) dr d\phi d\theta$ gegeben ist. Diese Formeln sind Spezialfälle einer allgemeinen Theorie, die von C.G.J. Jacobi (1804-1851) entwickelt wurde, vgl. *Jacobi-Determinante.*

Die folgenden Beispiele sollen uns zeigen, dass man mit MUPAD auch gewisse **Beweise** durchführen kann. Hier sind zunächst einige Vorbemerkungen:

Mit `mat:= Dom::Matrix()`legen wir fest, dass die Variable `mat` eine Matrix ist. Die Anweisung export(linalg) erlaubt uns, statt `linalg::crossProduct` einfach `crossProduct` zu schreiben.

Mit `bool`(links=rechts) stellen wir fest, ob die linke Seite eines Ausdrucks gleich ist der rechten. Mit `expand` entwickeln wir die Terme eines Ergebnisses. Ohne diese Entwicklung (Ausmultiplikation) kann die Funktion `bool` i. Allg. keine richtige logische Analyse durchführen.

1. Zeige, dass $(\vec{r} \times \frac{d\vec{r}}{dt}) \times \vec{r} = r^3 \frac{d\vec{e}_r}{dt}$ mit $\vec{r} := (x, y, z)$ und $r := |\vec{r}|$

```
mat:=Dom::Matrix(): export(linalg):
vr:=mat([[x(t),y(t),z(t)]]):/Vektor r
r:=sqrt(vr[1]^2+vr[2]^2+vr[3]^2):/Betrag von vr
v:=diff(vr,t):
er:=vr/r:
rechts:=r^3*diff(er,t):
rech:=expand(%):
links:=crossProduct(crossProduct(vr,v),vr):
link:=expand(%):
bool(rech=link)
```

Ergebnis: $TRUE$

2. Gegeben sind drei Vektoren $\vec{a}, \vec{b}, \vec{c}$ im $\mathrm{R}^3$. Zeige die Gültigkeit der folgenden Identitäten

   a. $\vec{a} \cdot (\vec{b} \times \vec{c}) = \vec{b} \cdot (\vec{c} \times \vec{a}) = \vec{c} \cdot (\vec{a} \times \vec{b})$

   b. $\vec{a} \times (\vec{b} \times \vec{c}) = \vec{b}(\vec{a} \cdot \vec{c}) - \vec{c}(\vec{a} \cdot \vec{b})$

Teil a:

```
reset():
assume(Type::Real):
mat:=Dom::Matrix(): export(linalg):
a:=mat([[a1,a2,a3]]):/Vektor a
b:=mat([[b1,b2,b3]]):/Vektor b
c:=mat([[c1,c2,c3]]):/Vektor c
links:=scalarProduct(a,crossProduct(b,c));
rechts:=scalarProduct(b,crossProduct(c,a));
bool(links=rechts)
```

Ergebnis:

```
a1 b2 c3 - a1 b3 c2 - a2 b1 c3 + a2 b3 c1 + a3 b1 c2 - a3 b2 c1
a1 b2 c3 - a1 b3 c2 - a2 b1 c3 + a2 b3 c1 + a3 b1 c2 - a3 b2 c1
TRUE
```

Teil b:

```
reset():
assume(Type::Real):
mat:=Dom::Matrix(): export(linalg):
a:=mat([[a1,a2,a3]])://Vektor a
b:=mat([[b1,b2,b3]])://Vektor b
c:=mat([[c1,c2,c3]])://Vektor c
links:=crossProduct(a,crossProduct(b,c)):
link:=expand(%);
rechts:=b*scalarProduct(a,c)-c*scalarProduct(a,b):
rech:=expand(%);
bool(link=rech)
```

Ergebnis:

```
array(1..1, 1..3,
  (1, 1) = a2 b1 c2 - a2 b2 c1 + a3 b1 c3 - a3 b3 c1,
  (1, 2) = a1 b2 c1 - a1 b1 c2 + a3 b2 c3 - a3 b3 c2,
  (1, 3) = a1 b3 c1 - a1 b1 c3 - a2 b2 c3 + a2 b3 c2
)
array(1..1, 1..3,
  (1, 1) = a2 b1 c2 - a2 b2 c1 + a3 b1 c3 - a3 b3 c1,
  (1, 2) = a1 b2 c1 - a1 b1 c2 + a3 b2 c3 - a3 b3 c2,
  (1, 3) = a1 b3 c1 - a1 b1 c3 - a2 b2 c3 + a2 b3 c2
)
TRUE
```