Hans Walser, [20150713]

# Klammerfehler

## 1 Worum geht es?

Es werden die Folgen eines simplen Klammerfehlers beim Anwenden der Programmiersprache Maple bearbeitet.

## 2 In der Ebene

### 2.1 Der Fehler

Ich wollte einen Kreis mit der Parameterdarstellung

$$\vec{x}(t)=\begin{bmatrix}\cos(t)\\ \sin(t)\end{bmatrix},\quad t\in[-\pi,\pi]$$

mit folgendem Programm 1 zeichnen:

```
with(plots): with(plottools):
Figur:=plot([cos(t), sin(t)], t=-Pi..Pi):
display([Figur], scaling=constrained);
```

**Programm 1: Versuch**

In der mittleren Programmzeile ist die Parameterdarstellung erkennbar.
Allerdings erhielt ich die Figur der Abbildung 1.

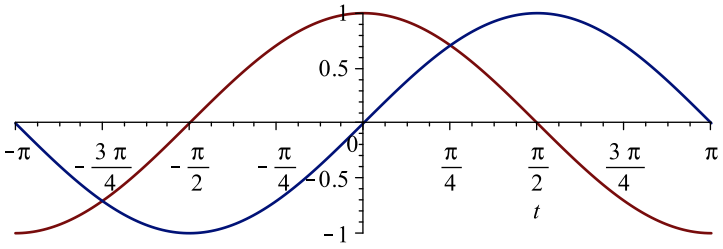


**Abb. 1: Plot**

Der Klammerausdruck [cos(t), sin(t)] wurde als Liste von zwei Funktionen interpretiert und entsprechend wurden die beiden Funktionen einzeln geplottet.

### 2.2 Die korrekte Klammerung

Das Problem kann durch Versetzen einer Klammer behoben werden (Programm 2):

```
with(plots): with(plottools):
Figur:=plot([cos(t), sin(t), t=-Pi..Pi]):
display([Figur], scaling=constrained);
```

**Programm 2: Änderung der Verklammerung**

Die Abbildung 2 zeigt den zugehörigen Plot. Ein schöner Kreis.

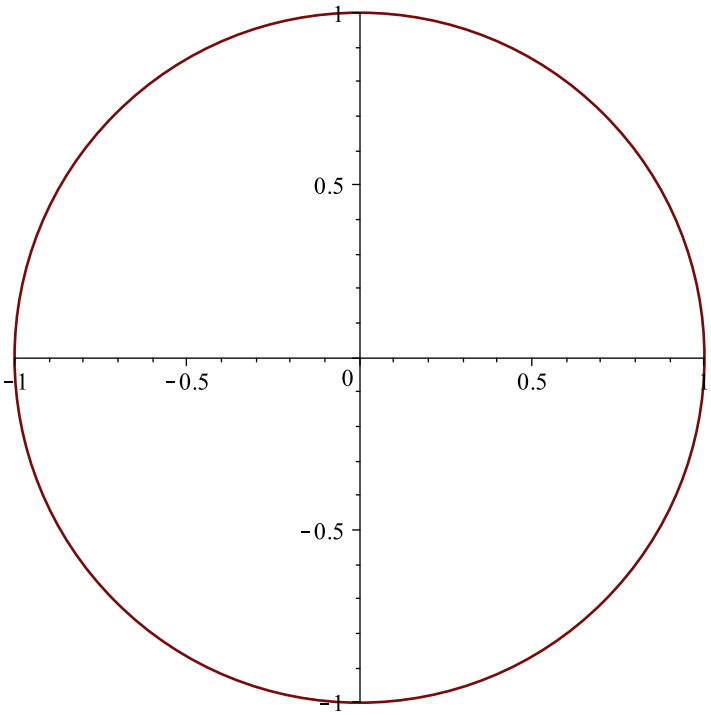


**Abb. 2: Kreis**

## 2.3 Liste

Man kann offenbar beliebig viele Funktionen als Liste plotten lassen. Programm 3 zeigt ein Beispiel.

```
with(plots): with(plottools):
Figur:=plot([cos(t), sin(t), tan(t), t^2, exp(t)], t=-Pi..Pi):
display([Figur], scaling=constrained, view=[-Pi..Pi, -3..3]);
```

**Programm 3: Liste**

Die Abbildung 3 zeigt den Plot der fünf Funktionen.

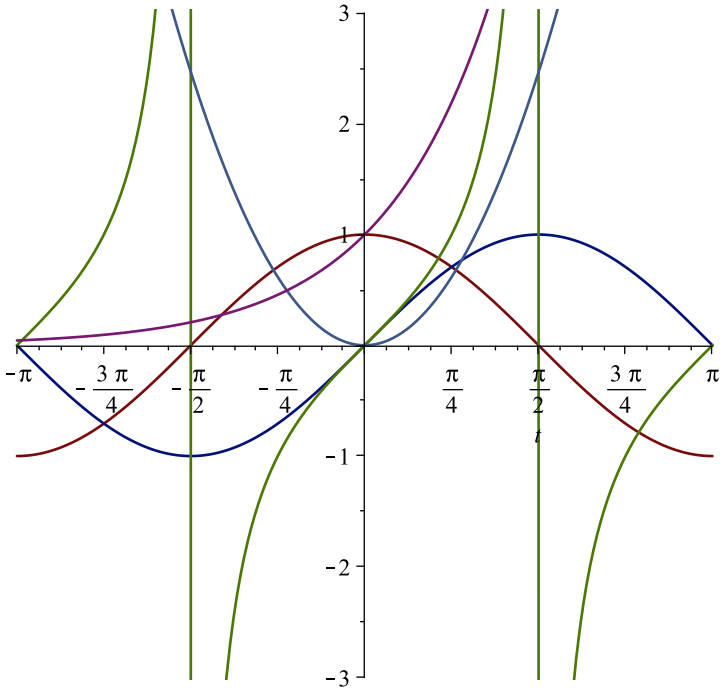


**Abb. 3: Plot**

## 3 Im Raum

### 3.1 Kugel

Die Kugel kann wie folgt parametrisiert werden:

$$\vec{x}(u,v)=\begin{bmatrix}\cos(u)\cos(v)\\ \cos(u)\sin(v)\\ \sin(u)\end{bmatrix},\quad u\in\left[-\frac{\pi}{2},\frac{\pi}{2}\right],v\in[-\pi,\pi]$$

Das zum Programm 2 analoge Programm funktioniert allerdings nicht.

```
with(plots): with(plottools):
Figur:=plot3d([cos(u)*cos(v), cos(u)*sin(v), sin(u), u=-Pi/2..Pi/2, v=-Pi..Pi]):
display([Figur], scaling=constrained);
```

**Programm 4: Versuchte Analogie zur Ebene**

Wir erhalten die Fehlermeldung der Abbildung 4.

```
Error, (in plot3d) at least three arguments are required
Error, (in plots:-display) expecting plot structures but received: [Figur]
```

**Abb. 4: Fehlermeldung**

### 3.2 Liste

Programm 5 versucht es mit einer Liste.

```
with(plots): with(plottools):
Figur:=plot3d([cos(u)*cos(v), cos(u)*sin(v), sin(u)], u=-Pi/2..Pi/2, v=-Pi..Pi):
display([Figur], scaling=constrained);
```

**Programm 5: Liste**

Nun erhalten wir die Kugel (Abb. 5).

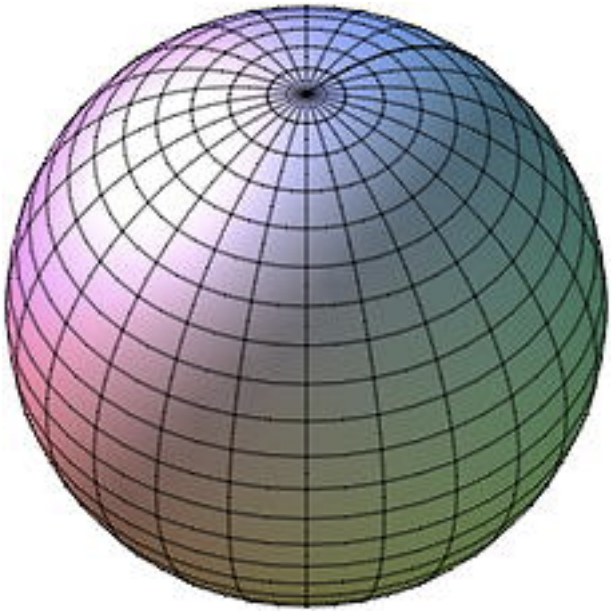

**Abb. 5: Kugel**

### 3.3 Veränderungen der Liste

Wir lassen den letzten Drittel in der Liste weg (Programm 6):

```
with(plots): with(plottools):
Figur:=plot3d([cos(u)*cos(v), cos(u)*sin(v)], u=-Pi/2..Pi/2, v=-Pi..Pi):
display([Figur], scaling=constrained);
```

**Programm 6: Verkürzte Liste**

Damit erhalten wir den Plot der Abbildung 6. Es sind die Funktionsgrafen der beiden ersten Funktionen in der Parameterdarstellung der Kugel.

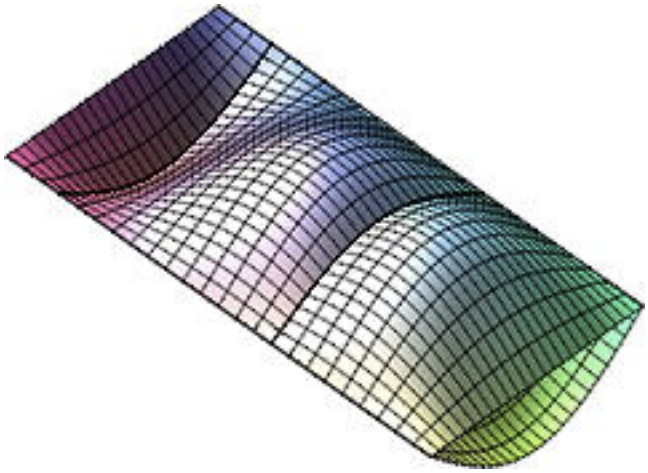

**Abb. 6: Plot**

Die Abbildung 7 zeigt die drei Koordinatenfunktionen einzeln.

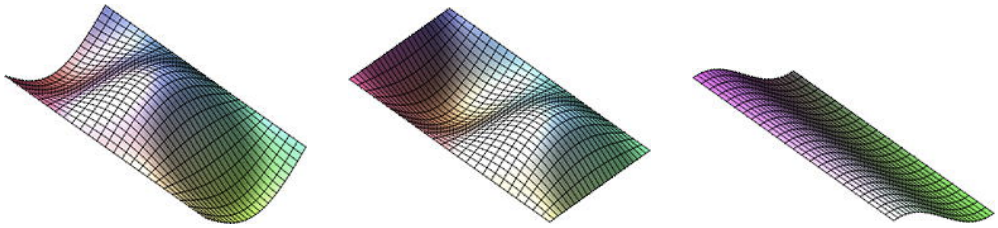

**Abb. 7: Koordinatenfunktionen einzeln.**

Wir sehen die Symmetrie der beiden ersten Koordinatenfunktionen und die Abweichung von der Symmetrie bei der dritten Koordinatenfunktion.

Um die Koordinatenfunktionen simultan ins gleiche Bild zu bringen muss man einen Trick anwenden (Programm 8), indem wir die Elemente in der Klammer nochmals verklammern.

```
with(plots): with(plottools):
Figur:=plot3d([[cos(u)*cos(v)], [cos(u)*sin(v)], [sin(u)]], u=-Pi/2..Pi/2, v=-Pi..Pi):
display([Figur], scaling=constrained);
```

**Programm 8: Klammern in der Klammer**

Die Abbildung 8 zeigt den Plot.

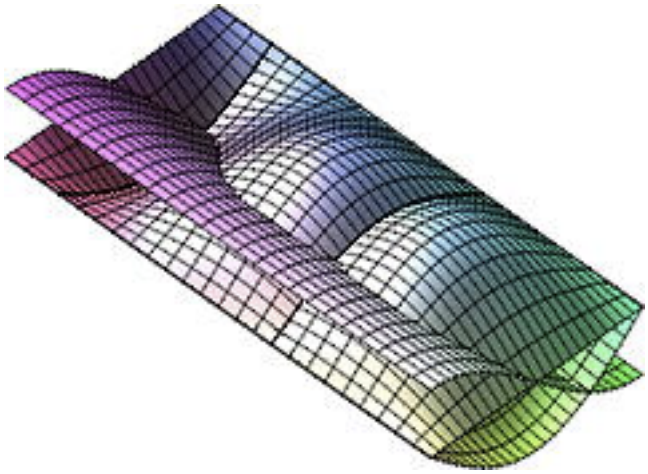

**Abb. 8: Koordinatenfunktionen simultan**

Da die einzelnen Funktionsgrafen schlecht unterscheidbar sind, können wir mit einer Farbliste arbeiten (Programm 9).

```
with(plots): with(plottools):
Figur:=plot3d([[cos(u)*cos(v)], [cos(u)*sin(v)], [sin(u)]], u=-Pi/2..Pi/2, v=-Pi..Pi,
color=[red, green,blue]):
display([Figur], scaling=constrained);
```

**Programm 9: Farbliste**

Die Abbildung 9 zeigt den Plot. Das Bild entspricht der Abbildung 1.

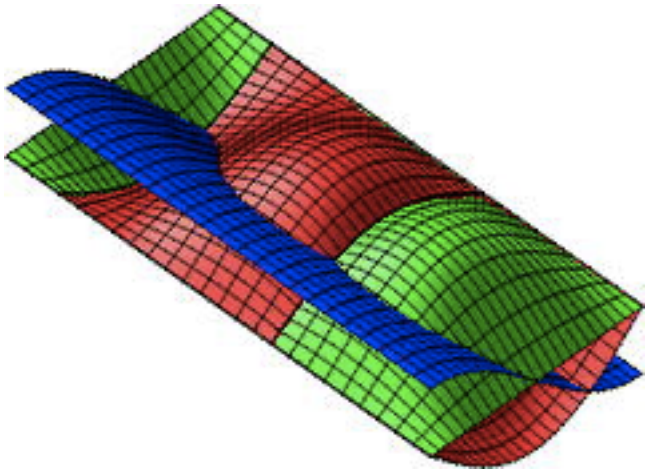

**Abb. 9: Unterscheidbare Funktionsgrafen**

Sobald wir mit mehr als drei Funktionen arbeiten, brauchen wir den Klammertrick nicht mehr (Programm 10).

```
with(plots): with(plottools):
Figur:=plot3d([cos(u)*cos(v), cos(u)*sin(v), sin(u), u*v], u=-Pi/2..Pi/2, v=-Pi..Pi,
color=[red, green,blue, yellow]):
display([Figur], scaling=constrained);
```

**Programm 10: Liste mit vier Funktionen ohne Binnenklammern**

Die Abbildung 10 zeigt den Plot.

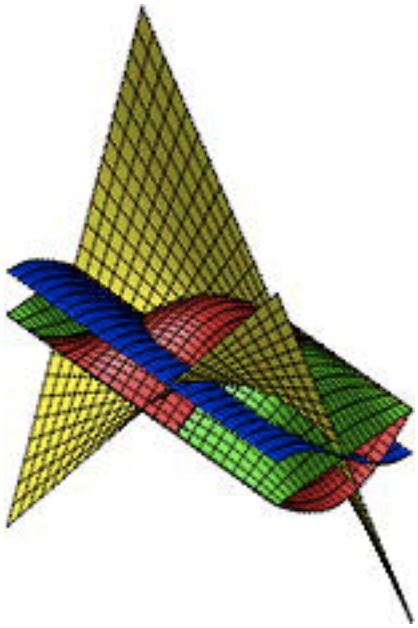

**Abb. 10: Vier Funktionen**

Wir können auch den Darstellungsstil einzeln wählen (Programm 11):

```
with(plots): with(plottools):
Figur:=plot3d([cos(u)*cos(v), cos(u)*sin(v), sin(u), u*v], u=-Pi/2..Pi/2, v=-Pi..Pi,
color=[red, green,blue, yellow], style=[patch, surface, point, wireframe]):
display([Figur], scaling=constrained);
```

**Programm 11: Darstellungsstile**

Die Abbildung 11 zeigt den Plot.

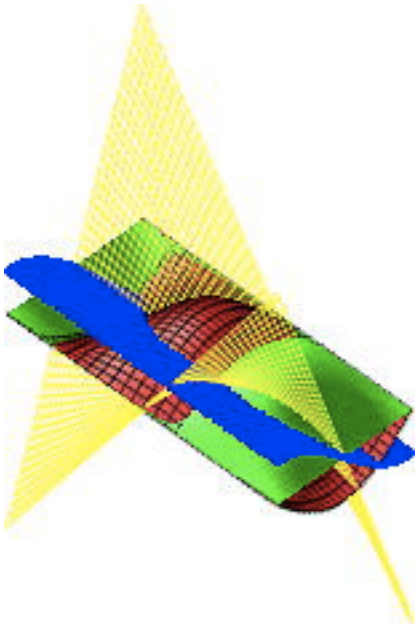

**Abb. 11: Verschiedene Stile**