

Geodäten: Die Hyperbolische Ebene

Zunächst müssen wir mal den Speicher entleeren und nötige Pakete laden. Das braucht Dich nicht zu interessieren...

```
> restart; with
> (plots):
```

Warning, the name changecoords has been redefined

Und los! Wir wissen daß die Länge einer Kurve $y(x)$ in der hyperbolischen Ebene gegeben sein soll durch:

$$T(y) = \int_{x_1}^{x_2} \frac{\sqrt{1 + \left(\frac{d}{dx} y(x)\right)^2}}{y(x)} dx$$

Wir haben also als Funktion F :

```
> F := (u,v,w) -> sqrt(1+v^2)/u;
```

$$F := (u, v, w) \rightarrow \frac{\sqrt{1 + v^2}}{u}$$

Jetzt lassen wir Maple die partielle Ableitung nach dem ersten Argument von F berechnen:

```
> D[1](F)(u,v,w);
```

$$-\frac{\sqrt{1 + v^2}}{u^2}$$

Beachte, wie man Maple das mitteilt:

```
> D
ist der Differentiationsbefehl in Maple.
> [1]
```

bedeutet, daß man nach dem ersten Argument partiell ableiten will,

```
> (F)
```

ist die abzuleitende Funktion und

```
> (u,v,w)
```

ist die Stelle, an der die Ableitung ausgewertet werden soll. Das heißt aber, Maple macht aus einer Funktion wie F durch

```
> D[1](F)
```

wieder eine Funktion mit drei Argumenten!

```
> D[1](F);
```

$$(u, v, w) \rightarrow -\frac{\sqrt{1 + v^2}}{u^2}$$

Jetzt brauchen wir das für die Euler-Lagrange-Gleichung an der Stelle $(y(x), \frac{d}{dx} y(x), x)$. Auch wenn Maple die Funktion $y(x)$ nicht kennt können wir mit ihrer Ableitung arbeiten, indem wir schreiben:

```
> term1:=D[1](F)(y(x), D(y)(x), x);
```

$$term1 := -\frac{\sqrt{1 + D(y)(x)^2}}{y(x)^2}$$

Na das geht ja noch; hätte man auch von selbst drauf kommen können! Nun aber zum nächsten Teil, die partielle Ableitung von F nach dem zweiten Argument.

```
> term2:=D[2](F)(y(x), D(y)(x), x);
```

$$term2 := \frac{D(y)(x)}{\sqrt{1 + D(y)(x)^2} y(x)}$$

Jetzt müssen wir das obenstehende noch nach x differenzieren. Das machen wir jetzt mit dem

Befehl
`> diff`

(weil wir keine Funktion mehr vorliegen haben, sondern nur einen Term, in dem ein x vorkommt):

`> term3:=diff( term2, x);`

$$term3 := \frac{(D^{(2)})(y)(x)}{\sqrt{1 + D(y)(x)^2} y(x)} - \frac{D(y)(x)^2 (D^{(2)})(y)(x)}{(1 + D(y)(x)^2)^{3/2} y(x)} - \frac{D(y)(x) \left(\frac{d}{dx} y(x)\right)}{\sqrt{1 + D(y)(x)^2} y(x)^2}$$

Puh, das sieht ganz schön kompliziert aus. Aber was haben wir denn auch anderes erwartet! Ein Glück, daß wir Maple haben! Dann stellen wir mal die Euler-Lagrange-Gleichung auf! Der

`> simplify`

-Befehl hilft uns, das Ganze etwas einfacher darstellen zu lassen!

`> simplify(term1-term3=0);`

$$-\frac{1 + D(y)(x)^2 + (D^{(2)})(y)(x) y(x)}{y(x)^2 (1 + D(y)(x)^2)^{3/2}} = 0$$

Jetzt wird es spannend! Vielleicht haben wir Glück, und Maple findet eine Lösung. Wer

verwenden den

`> dsolve`

-Befehl zum Auflösen der Differentialgleichung nach $y(x)$:

`> dsolve(term1-term3=0, y(x));`

$$y(x) = \sqrt{-x^2 - 2_C1 x + 2_C2}, y(x) = -\sqrt{-x^2 - 2_C1 x + 2_C2}$$

Jetzt zeigt Maple verschiedene Möglichkeiten für die Lösungen an. Welches nun wirklich die richtige Lösung ist, muß man sich anderweitig überlegen. $_C1$ und $_C2$ sind nur Integrationskonstanten. Man muß sie aus den übrigen Bedingungen bestimmen.

Im Falle der hyperbolischen Ebene kommt nur die erste Lösung in Frage, da dort nur Punkte mit positiver y -Koordinate auftreten!

`> C1:='C1': C2:= 'C2':`

`> y1:= x->sqrt(-x^2-2*C1*x+2*C2);`

Nun kann man irgendwelche Bedingungen einsetzen, z.B. soll $y1(3) = 4$ und $y1(2) = 1$ gelten. Diese Gleichungen kann man so schreiben...

`> glg1:= y1(3)=4;`

`> glg2:= y1(2)=1;`

$$y1 := x \rightarrow \sqrt{-x^2 - 2 C1 x + 2 C2}$$

$$glg1 := \sqrt{-9 - 6 C1 + 2 C2} = 4$$

$$glg2 := \sqrt{-4 - 4 C1 + 2 C2} = 1$$

... und Maple löst sie sogar für uns!

`> C2:=solve(glg1, C2);`

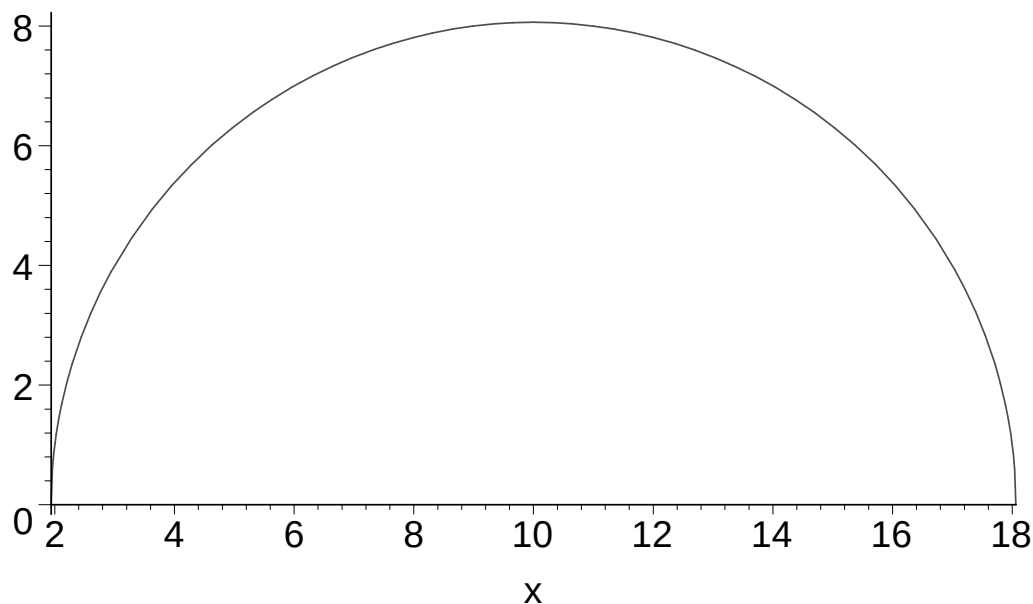
`> C1:=solve(glg2, C1);`

$$C2 := \frac{25}{2} + 3 C1$$

$$C1 := -10$$

Und schließlich plotten wir das mal! Na? Wie sehen die Geodäten aus? Haben sie eine besondere Charakteristik? Probiere noch andere Punktepaare aus!

`> plot( y1(x), x=solve(y1(x)=0, x)[1]..solve(y1(x)=0, x)[2],`
`> scaling=constrained);`



- 1 Dies ist ein kleines Programm, daß, wenn man die Koordinaten von drei Punkten in der hyperbolischen Ebene angibt, das dazugehörige Dreieck ausgibt.

```

> triangle:= proc( x1,y1,x2,y2,x3,y3 )
> local a,b,xx,y, C, r, L;
> y:= x-> [sqrt(-x^2-2*C[1,1]*x+2*C[1,2]),
> sqrt(-x^2-2*C[2,1]*x+2*C[2,2]),sqrt(-x^2-2*C[3,1]*x+2*C[3,2])];
> L:=[x1,y1,x2,y2,x3,y3];
> if L[3]=0 then r:= L[1]; L[1]:=L[3]; L[3]:=r; fi;

> if L[1]=L[3] then
> a[1]:= plot( [L[1],t, t=L[2]..L[4]]);
> else
> C[1,2]:=solve(y(L[1])[1]=L[2], C[1,2]);
> C[1,1]:=solve(y(L[3])[1]=L[4], C[1,1]);
> xx[1]:=solve(y(x)[1]=0,x);
> a[1]:= plot( y(x)[1], x=xx[1][1]..xx[1][2], scaling=constrained):
> fi:

> if L[3]=L[5] then
> a[2]:= plot( [L[3],t, t=L[4]..L[6]]);
> else
> C[2,1]:=solve(y(L[3])[2]=L[4], C[2,1]);
> C[2,2]:=solve(y(L[5])[2]=L[6], C[2,2]);

```

```

> xx[2]:=solve(y(x)[2]=0,x);
> a[2]:= plot( y(x)[2], x=xx[2][1]..xx[2][2], scaling=constrained):
> fi:

> if L[5]=L[1] then
> a[3]:= plot( [L[5],t, t=L[6]..L[2]]);
> else
> C[3,2]:=solve(y(L[5])[3]=L[6], C[3,2]);
> C[3,1]:=solve(y(L[1])[3]=L[2], C[3,1]);
> xx[3]:=solve(y(x)[3]=0,x);
> a[3]:= plot( y(x)[3], x=xx[3][1]..xx[3][2], scaling=constrained):
> fi:

> RETURN(display({a[1], a[2], a[3]})):
> end:

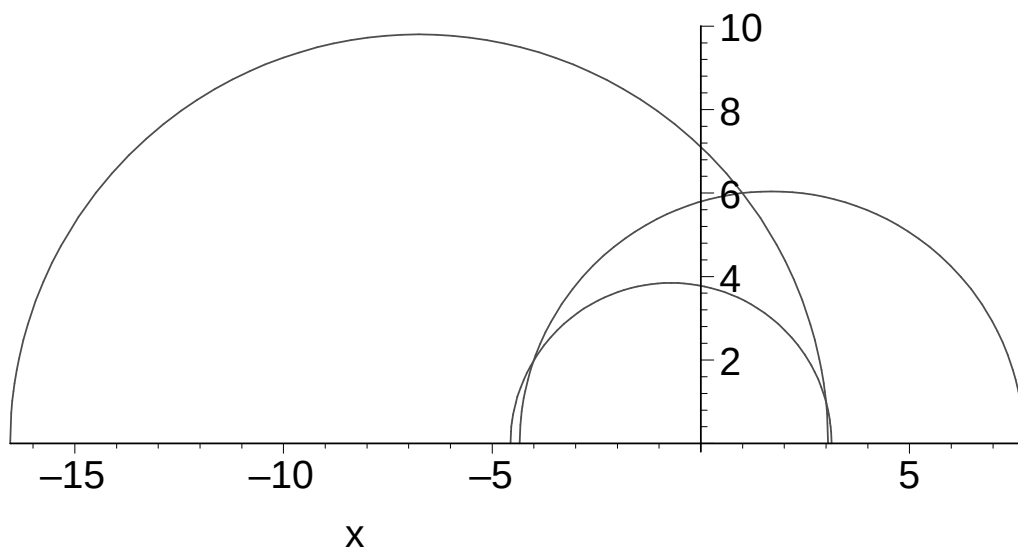
```

So ruft man das Programm auf! Komisch oder? Haben Dreiecke hier eine Winkelsumme von 180° ?

```

> triangle(1,6,3,1,-4,2);

```



Na, spätestens an diesem Beispiel kann man es erkennen!

```

> triangle(1,0,-1,0,0,0);

```

