

> **restart:**

- Lagrangegleichungen mit einer generalisierten Koordinate $q(t)$

Die auf dem Prinzip der kleinsten Wirkung gründenden Euler-Lagrangegleichungen vereinfachen die Modellbildung innerhalb der theoretischen Physik sehr. Beschreibt man das System durch voneinander unabhängige generalisierte Koordinaten (in diesem Fall eine generalisierte Koordinate q), so ergeben sich die Bewegungsgleichungen des betrachteten Systems durch die Lagrangegleichungen. Man muß dabei lediglich die Lagrangefunktion des Systems kennen; diese ergibt sich durch die kinetische minus die potentielle Energie (T-V).

- Analytische Lösung

Hier behandeln wir die Lagrangetheorie des harmonischen Oszillators. Die durch die Lagrangegleichungen gebildete Differenzialgleichung (Bewegungsgleichung) läßt sich analytisch lösen. $q(t)$ steht hier zunächst vereinfacht fuer die Zeitabhaengigkeit der Beschleunigung q

$$> T := 1/2 * m * q_t^2;$$

$$V := 1/2 * k * q^2;$$

$$L := T - V;$$

$$T := \frac{m q_t^2}{2}$$

$$V := \frac{k q^2}{2}$$

$$L := \frac{m q_t^2}{2} - \frac{k q^2}{2}$$

$$> \text{LagrangeGL} := \text{diff}(\text{subs}(\{q=q(t), q_t=\text{diff}(q(t), t)\}, \text{diff}(L, q_t)), t) - \text{subs}(\{q=q(t), q_t=\text{diff}(q(t), t)\}, \text{diff}(L, q)));$$

$$\text{LagrangeGL} := m \left(\frac{d^2}{dt^2} q(t) \right) + k q(t)$$

$$> \text{dsolve}(\{\text{LagrangeGL}\}, q(t));$$

$$\{ q(t) = _C1 \sin\left(\frac{\sqrt{k} t}{\sqrt{m}}\right) + _C2 \cos\left(\frac{\sqrt{k} t}{\sqrt{m}}\right) \}$$

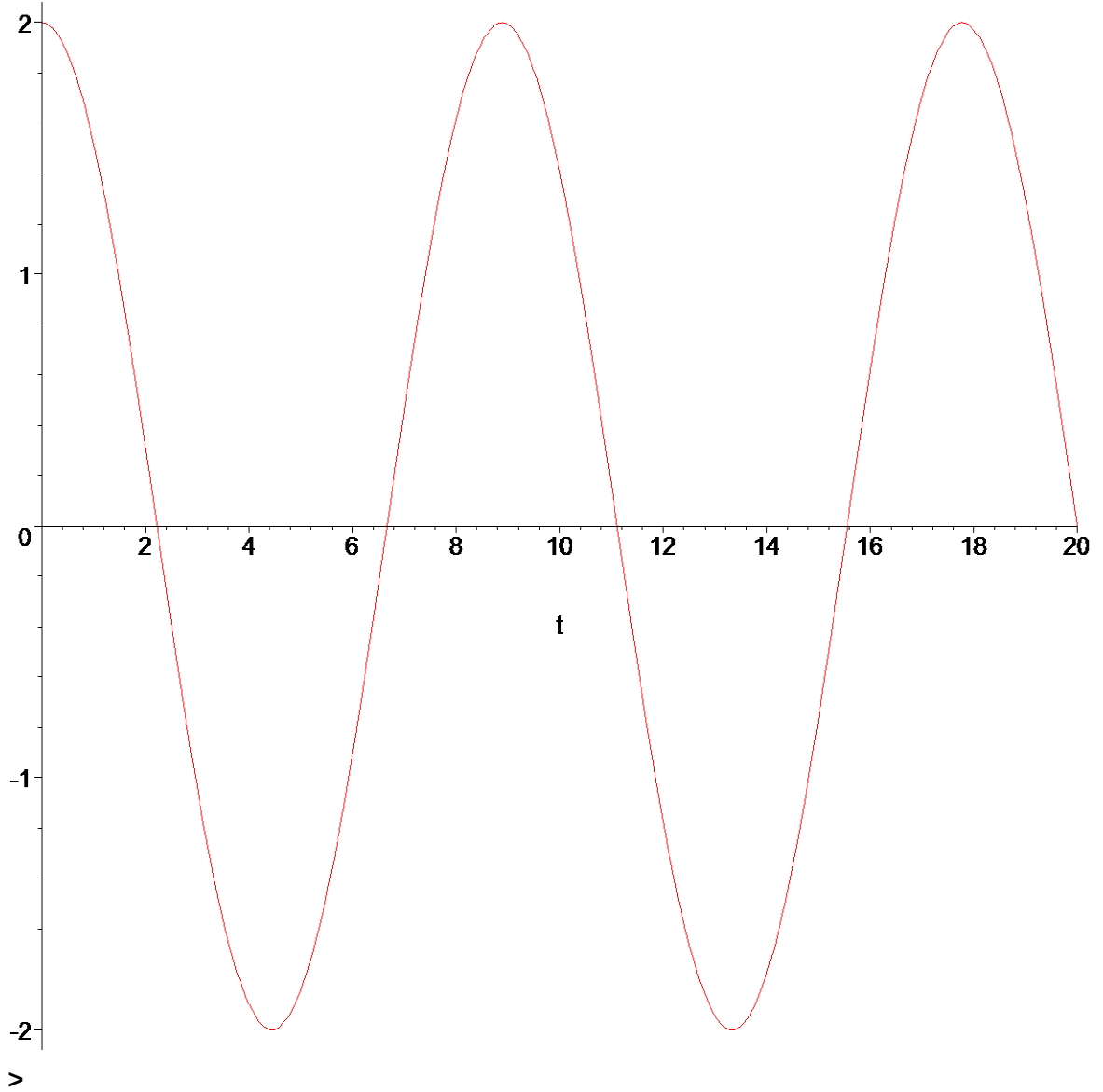
$$> \text{Aq} := 2:$$

$$\text{Aqt} := 0:$$

```
Loes:=dsolve({LagrangeGL,q(0)=Aq,D(q)(0)=Aqt},q(t));
```

$$Loes := q(t) = 2 \cos\left(\frac{\sqrt{k} t}{\sqrt{m}}\right)$$

```
plot(subs({k=1,m=2},rhs(Loes)),t=0..20);
```



— Numerische Lösung des Pendels

Hier behandeln wir die Lagrangetheorie des Pendels. Die generalisierte Koordinate q ist der Winkel der Auslenkung des Pendels. Die entstehende Bewegungsgleichung lässt sich (ohne weitere Approximationen) nur numerisch lösen.

```
> g:=9.80665:
```

```
m:=1:
```

```

l:=1:
T:=1/2*m*l^2*qt^2;
V:=m*g*l*(1-cos(q));
L:=T-V;

```

$$T := \frac{qt^2}{2}$$

$$V := 9.80665 - 9.80665 \cos(q)$$

$$L := \frac{qt^2}{2} - 9.80665 + 9.80665 \cos(q)$$

```

> LagrangeGL:=diff(subs({q=q(t),qt=diff(q(t),t)},diff(L,qt)),t)-subs({q=q(t),qt=diff(q(t),t)},diff(L,q));

```

$$\text{LagrangeGL} := \left(\frac{d^2}{dt^2} q(t) \right) + 9.80665 \sin(q(t))$$

```

> Aq:=Pi/2:
Aqt:=0:

```

```

Loes:=dsolve({LagrangeGL,q(0)=Aq,D(q)(0)=Aqt},q(t),type=numeric,output=listprocedure);

```

```

Loes := [ t = (proc(t) ... end proc), q(t) = (proc(t) ... end proc),

```

```

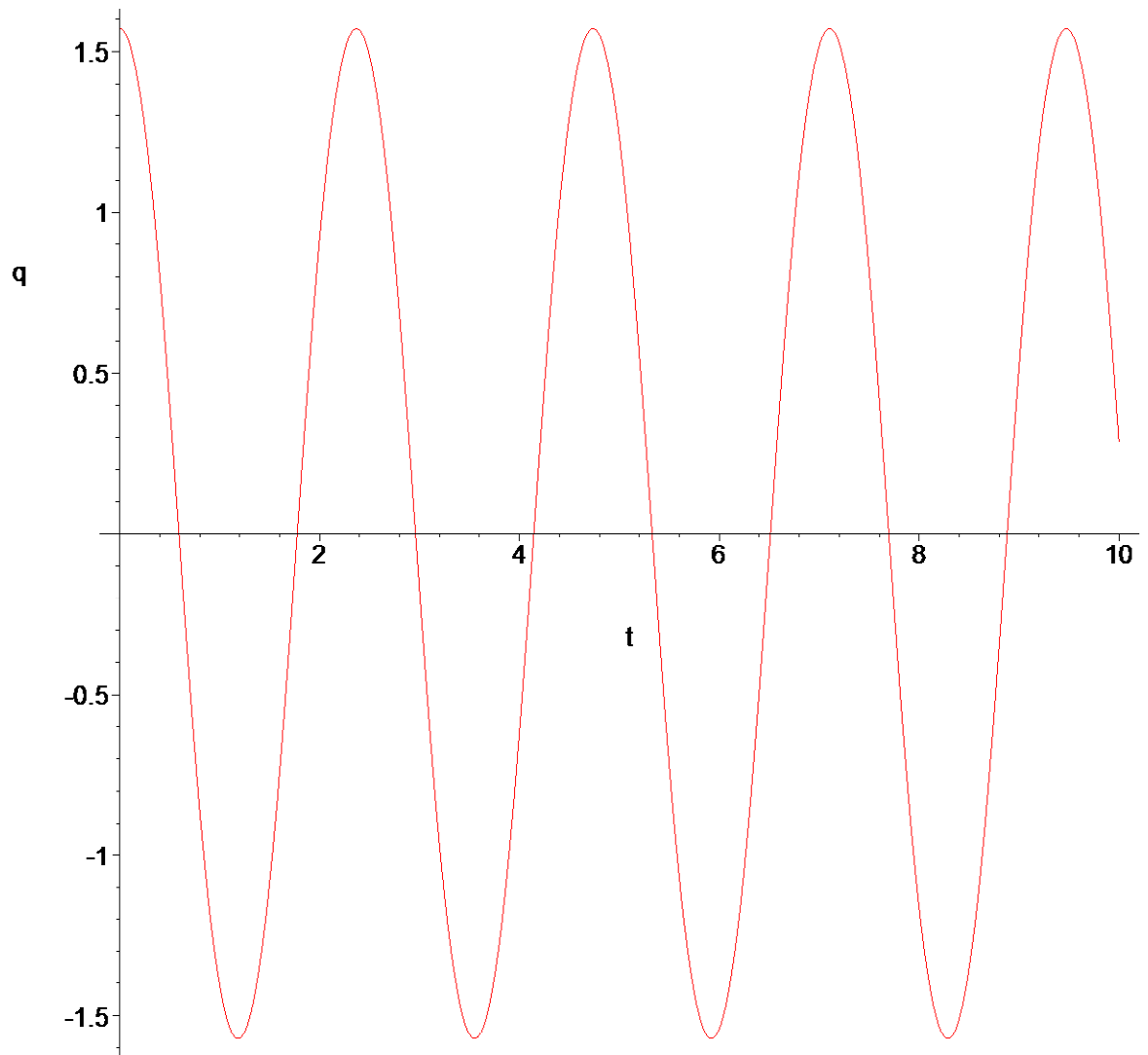
  d
  dt q(t) = (proc(t) ... end proc) ]

```

```

> with(plots):
with(plottools):
> odeplot(Loes,[t,q(t)],0..10,numpoints=1000);

```



```
> phi:=eval(q(t),Loes);
```

```
phi:=proc(t) ... end proc
```

```
> x:=t->l*sin(phi(t));
y:=t->-l*cos(phi(t));
```

```
x:=t -> l sin(phi(t))
```

```
y:=t -> -l cos(phi(t))
```

```
>
```

```
tend:=10:
```

```
frames:=100:
```

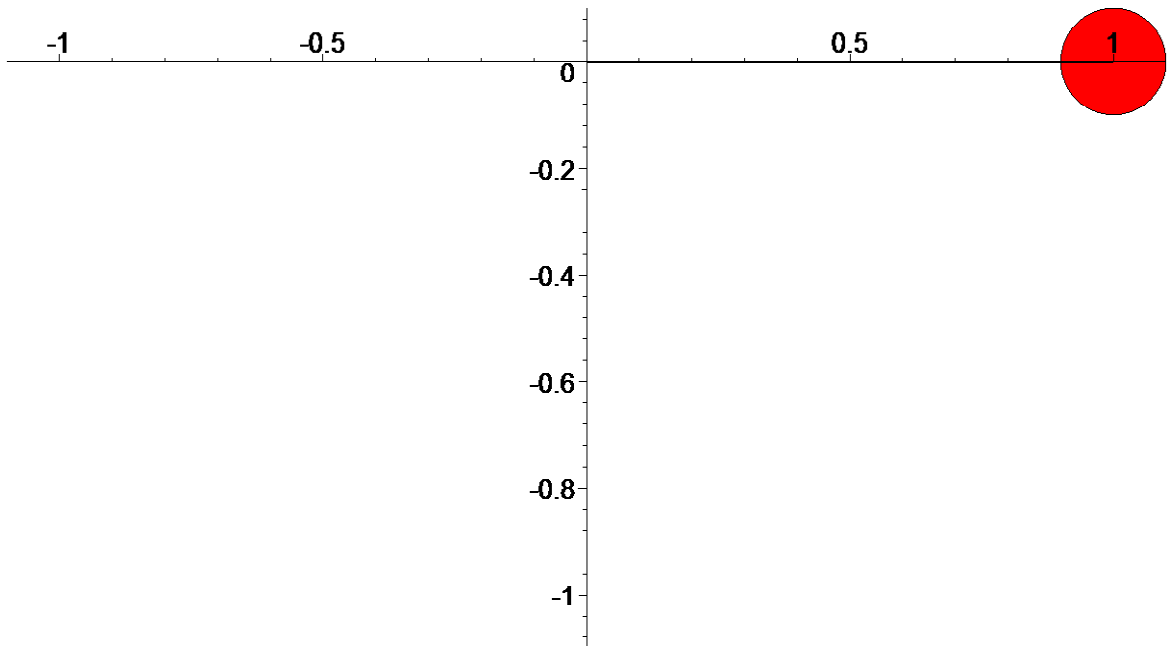
```
for i from 0 by 1 to frames do
```

```
Seil[i]:=curve([[0,0],[x(i*tend/frames),y(i*tend/frames)]]
,thickness=2,color=black):
```

```
Masse[i]:=display(disk([x(i*tend/frames),y(i*tend/frames)]
,0.1,color=red)):
```

```
Ani[i]:=display({Masse[i],Seil[i]});
```

```
od:  
> display([seq(Ani[i],i=0..frames)],insequence=true,scaling=  
constrained);
```



```
[ >
```

- Numerische Lösung des Pendels mit Reibung

```
> g:=9.80665:  
m:=1:  
l:=1:  
beta:=0.1;  
  
T:=1/2*m*l^2*qt^2;  
V:=m*g*l*(1-cos(q));  
L:=T-V;  
  
DR:=1/2*beta*qt^2;
```

$$\beta := 0.1$$

$$T := \frac{qt^2}{2}$$

$$V := 9.80665 - 9.80665 \cos(q)$$

$$L := \frac{qt^2}{2} - 9.80665 + 9.80665 \cos(q)$$

$$DR := 0.05000000000 qt^2$$

Die folgende Lagrange DGL gibt Schwingungen inklusive der Reibungsenergie wieder

```
> LagrangeGL:=diff(subs({q=q(t),qt=diff(q(t),t)},diff(L,qt)),t)-subs({q=q(t),qt=diff(q(t),t)},diff(L,q))+subs({q=q(t),qt=diff(q(t),t)},diff(DR,qt))=0;
```

$$\text{LagrangeGL} := \left(\frac{d^2}{dt^2} q(t) \right) + 9.80665 \sin(q(t)) + 0.1000000000 \left(\frac{d}{dt} q(t) \right) = 0$$

```
> Aq:=Pi/4:
```

```
Aqt:=0:
```

```
Loes:=dsolve({LagrangeGL,q(0)=Aq,D(q)(0)=Aqt},q(t),type=numeric,output=listprocedure);
```

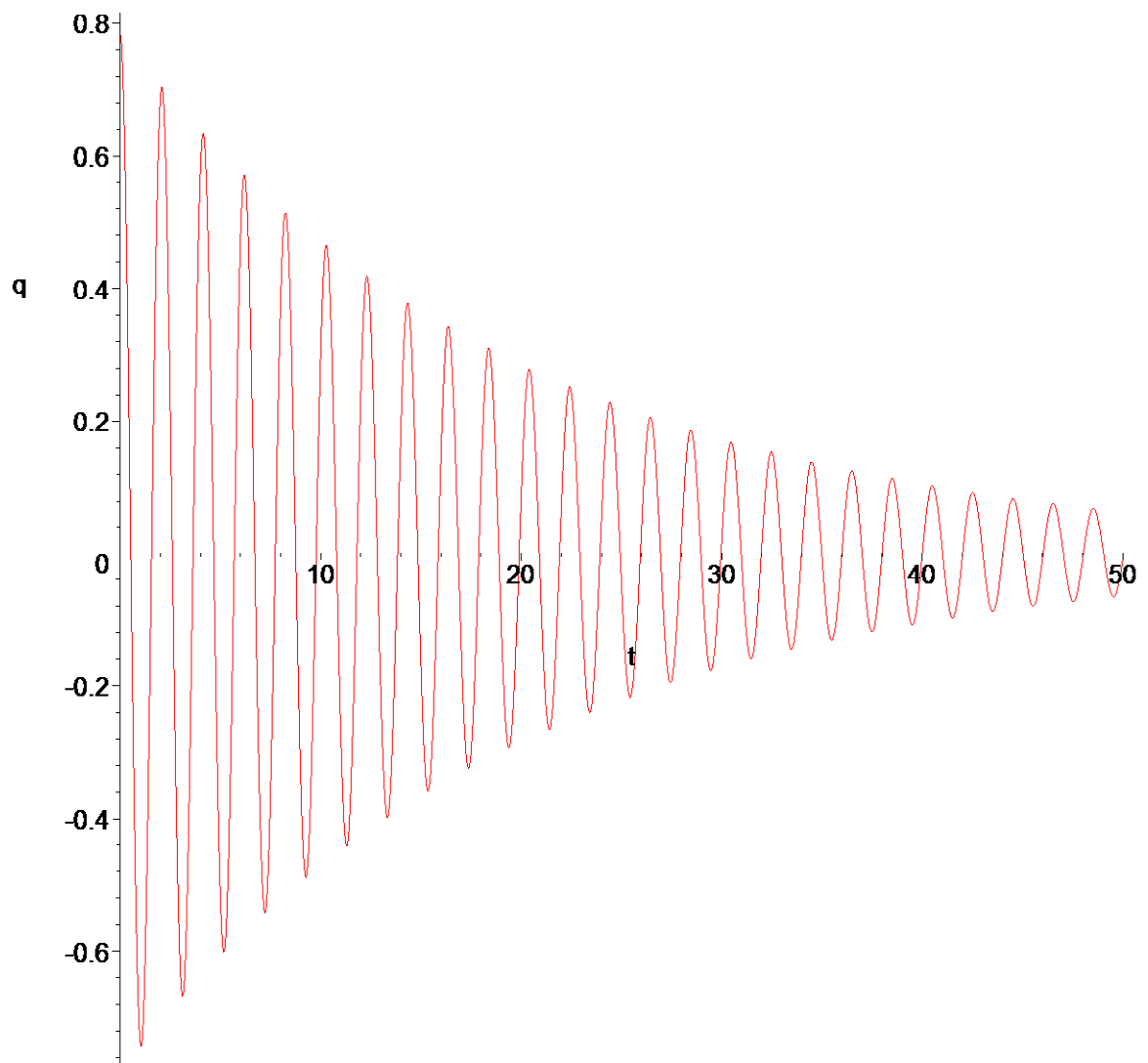
```
Loes := [ t = (proc(t) ... end proc), q(t) = (proc(t) ... end proc),
```

```
  d/dt q(t) = (proc(t) ... end proc) ]
```

```
> with(plots):
```

```
with(plottools):
```

```
> odeplot(Loes,[t,q(t)],0..50,numpoints=1000);
```



```
> phi:=eval(q(t),Loes);
```

```
phi:=proc(t) ... end proc
```

```
> x:=t->l*sin(phi(t));
y:=t->-l*cos(phi(t));
```

```
x:=t -> l sin(phi(t))
```

```
y:=t -> -l cos(phi(t))
```

```
>
```

```
tend:=50:
```

```
frames:=500:
```

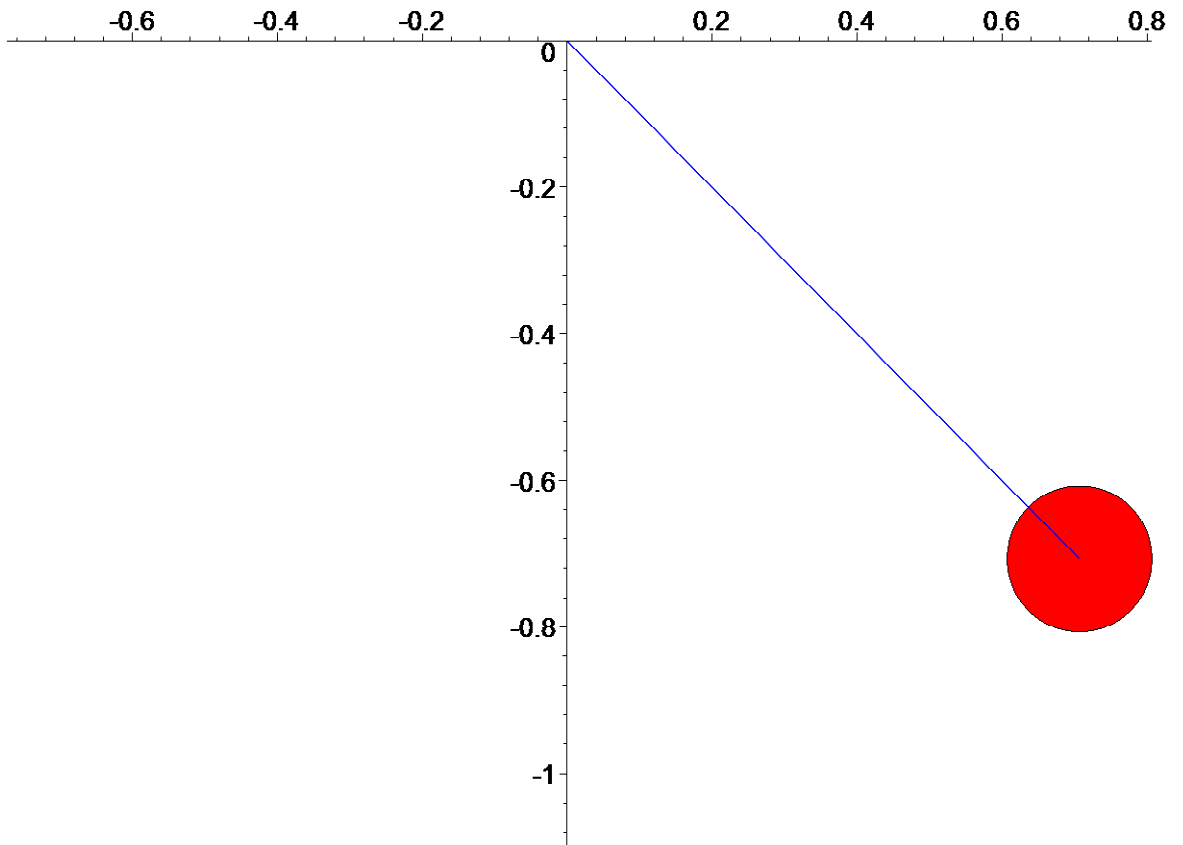
```
for i from 0 by 1 to frames do
```

```
Seil[i]:=curve([[0,0],[x(i*tend/frames),y(i*tend/frames)]]
,thickness=2,color=blue):
```

```
Masse[i]:=display(disk([x(i*tend/frames),y(i*tend/frames)]
,0.1,color=red)):
```

```
Ani[i]:=display({Masse[i],Seil[i]});
```

```
od:  
> display([seq(Ani[i],i=0..frames)],insequence=true,scaling=  
constrained);
```



```
[ ] >  
[ ] >
```