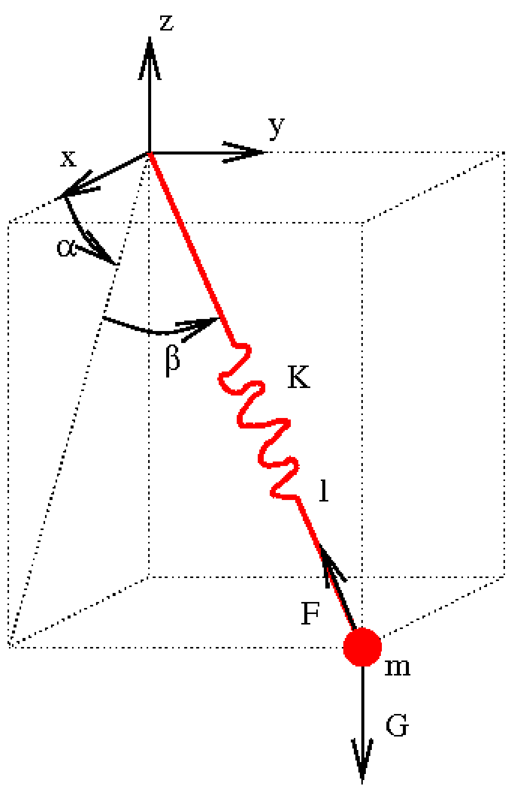
Pendulum on the spring in 3D.
Dynamic simulation model.
The position of the hanging is externally controled

- **The externally controled hanging position has coordinates {xp, yp, zp}**
  The spring has length at rest l0 and spring constant K
  The pendulum mass is m
- **The parametrization uses Cardan like parametrization with first rotation around x axis by angle α and then rotation around new z axis by the angle β.**
  This parametrization is used to avoid singularity in vertical position of the pendulum.
  In fact this parametrization has another singularity when pendulum is in y axis direction.
- 

- **Independent variables**

```
In[1]:= vars = {l[t], α[t], β[t]};
```

- **Bounding equations**

```
In[2]:= x = l[t] Cos[β[t]] Cos[α[t]] + xp[t];
        y = l[t] Sin[β[t]] + yp[t];
        z = -l[t] Cos[β[t]] Sin[α[t]] + zp[t];
```

- **First and second derivatives of bounding equations**

In[5]:=
```
D[x, t] // TraditionalForm
D[y, t] // TraditionalForm
D[z, t] // TraditionalForm
D[x, {t, 2}] // Expand // TraditionalForm
D[y, {t, 2}] // Expand // TraditionalForm
D[z, {t, 2}] // Expand // TraditionalForm
```

Out[5]//TraditionalForm=

$$l'(t)\cos(\alpha(t))\cos(\beta(t)) - l(t)\,\alpha'(t)\sin(\alpha(t))\cos(\beta(t)) - l(t)\cos(\alpha(t))\,\beta'(t)\sin(\beta(t)) + \text{xp}'(t)$$

Out[6]//TraditionalForm=

$$l'(t)\sin(\beta(t)) + l(t)\,\beta'(t)\cos(\beta(t)) + \text{yp}'(t)$$

Out[7]//TraditionalForm=

$$l'(t)\sin(\alpha(t))\,(-\cos(\beta(t))) - l(t)\,\alpha'(t)\cos(\alpha(t))\cos(\beta(t)) + l(t)\sin(\alpha(t))\,\beta'(t)\sin(\beta(t)) + \text{zp}'(t)$$

Out[8]//TraditionalForm=

$$l''(t)\cos(\alpha(t))\cos(\beta(t)) - 2\,l'(t)\,\alpha'(t)\sin(\alpha(t))\cos(\beta(t)) - 2\,l'(t)\cos(\alpha(t))\,\beta'(t)\sin(\beta(t)) - l(t)\,\alpha''(t)\sin(\alpha(t))\cos(\beta(t)) + 2\,l(t)\,\alpha'(t)\sin(\alpha(t))\,\beta'(t)\sin(\beta(t)) + l(t)\,\alpha'(t)^2\,(-\cos(\alpha(t)))\cos(\beta(t)) - l(t)\cos(\alpha(t))\,\beta''(t)\sin(\beta(t)) - l(t)\cos(\alpha(t))\,\beta'(t)^2\cos(\beta(t)) + \text{xp}''(t)$$

Out[9]//TraditionalForm=

$$l''(t)\sin(\beta(t)) + 2\,l'(t)\,\beta'(t)\cos(\beta(t)) + l(t)\,\beta''(t)\cos(\beta(t)) - l(t)\,\beta'(t)^2\sin(\beta(t)) + \text{yp}''(t)$$

Out[10]//TraditionalForm=

$$-l''(t)\sin(\alpha(t))\cos(\beta(t)) - 2\,l'(t)\,\alpha'(t)\cos(\alpha(t))\cos(\beta(t)) + 2\,l'(t)\sin(\alpha(t))\,\beta'(t)\sin(\beta(t)) - l(t)\,\alpha''(t)\cos(\alpha(t))\cos(\beta(t)) + 2\,l(t)\,\alpha'(t)\cos(\alpha(t))\,\beta'(t)\sin(\beta(t)) + l(t)\,\alpha'(t)^2\sin(\alpha(t))\cos(\beta(t)) + l(t)\sin(\alpha(t))\,\beta''(t)\sin(\beta(t)) + l(t)\sin(\alpha(t))\,\beta'(t)^2\cos(\beta(t)) + \text{zp}''(t)$$

- **Force components**

In[167]:=
```
Fx[t] := Force[t] Cos[β[t]] Cos[α[t]]
Fy[t] := Force[t] Sin[β[t]]
Fz[t] := Force[t] Cos[β[t]] Sin[α[t]]
Force[t] := K (l[t] - l0)
```

- **Newton equations**

In[171]:=
```
eq1 = m D[x, {t, 2}] + Fx[t];
eq2 = m D[y, {t, 2}] + Fy[t];
eq3 = m D[z, {t, 2}] + G - Fz[t];
eq = {eq1, eq2, eq3};
```

- **Left sides of Newtonov equations in readable format**

In[175]:=
```
eq1 // Expand // TraditionalForm
eq2 // Expand // TraditionalForm
eq3 // Expand // TraditionalForm
```

Out[175]//TraditionalForm=

$$K\,l(t)\cos(\alpha(t))\cos(\beta(t)) - K\,\text{l0}\cos(\alpha(t))\cos(\beta(t)) + m\,l''(t)\cos(\alpha(t))\cos(\beta(t)) - 2\,m\,l'(t)\,\alpha'(t)\sin(\alpha(t))\cos(\beta(t)) - 2\,m\,l'(t)\cos(\alpha(t))\,\beta'(t)\sin(\beta(t)) - m\,l(t)\,\alpha''(t)\sin(\alpha(t))\cos(\beta(t)) + 2\,m\,l(t)\,\alpha'(t)\sin(\alpha(t))\,\beta'(t)\sin(\beta(t)) - m\,l(t)\,\alpha'(t)^2\cos(\alpha(t))\cos(\beta(t)) - m\,l(t)\cos(\alpha(t))\,\beta''(t)\sin(\beta(t)) - m\,l(t)\cos(\alpha(t))\,\beta'(t)^2\cos(\beta(t)) + m\,\text{xp}''(t)$$

Out[176]//TraditionalForm=

$$K\,l(t)\sin(\beta(t)) - K\,\text{l0}\sin(\beta(t)) + m\,l''(t)\sin(\beta(t)) + 2\,m\,l'(t)\,\beta'(t)\cos(\beta(t)) + m\,l(t)\,\beta''(t)\cos(\beta(t)) - m\,l(t)\,\beta'(t)^2\sin(\beta(t)) + m\,\text{yp}''(t)$$

Out[177]//TraditionalForm=

$$G - K\,l(t)\sin(\alpha(t))\cos(\beta(t)) + K\,\text{l0}\sin(\alpha(t))\cos(\beta(t)) - m\,l''(t)\sin(\alpha(t))\cos(\beta(t)) - 2\,m\,l'(t)\,\alpha'(t)\cos(\alpha(t))\cos(\beta(t)) + 2\,m\,l'(t)\sin(\alpha(t))\,\beta'(t)\sin(\beta(t)) - m\,l(t)\,\alpha''(t)\cos(\alpha(t))\cos(\beta(t)) + 2\,m\,l(t)\,\alpha'(t)\cos(\alpha(t))\,\beta'(t)\sin(\beta(t)) + m\,l(t)\,\alpha'(t)^2\sin(\alpha(t))\cos(\beta(t)) + m\,l(t)\sin(\alpha(t))\,\beta''(t)\sin(\beta(t)) + m\,l(t)\sin(\alpha(t))\,\beta'(t)^2\cos(\beta(t)) + m\,\text{zp}''(t)$$

- **Newton equations could be rewritten into the form A w = b and numerically solved: w = A^{-1} b**
- **Vector of second derivatives of independent variables w**

In[186]:=
```
w = D[vars, {t, 2}]
```

Out[186]= {l''[t], α''[t], β''[t]}

- **Matrix A (using MapAt)**

In[187]:=
```
A = (Coefficient[eq, #] & /@ w )ᵀ;
A // TraditionalForm
```

Out[188]//TraditionalForm=

$$\begin{pmatrix} m\cos(\alpha(t))\cos(\beta(t)) & -m\cos(\beta(t))\,l(t)\sin(\alpha(t)) & -m\cos(\alpha(t))\,l(t)\sin(\beta(t)) \\ m\sin(\beta(t)) & 0 & m\cos(\beta(t))\,l(t) \\ -m\cos(\beta(t))\sin(\alpha(t)) & -m\cos(\alpha(t))\cos(\beta(t))\,l(t) & m\,l(t)\sin(\alpha(t))\sin(\beta(t)) \end{pmatrix}$$

- **Vector of right sides b**

In[189]:=
```
b = A . w - eq // Simplify;
b // Expand // TraditionalForm
```

Out[190]//TraditionalForm=

$$\{-K\,l(t)\cos(\alpha(t))\cos(\beta(t)) + K\,\mathrm{l0}\cos(\alpha(t))\cos(\beta(t)) + 2\,m\,l'(t)\,\alpha'(t)\sin(\alpha(t))\cos(\beta(t)) + 2\,m\,l'(t)\cos(\alpha(t))\,\beta'(t)\sin(\beta(t)) - 2\,m\,l(t)\,\alpha'(t)\sin(\alpha(t))\,\beta'(t)\sin(\beta(t)) + m\,l(t)\,\alpha'(t)^2\cos(\alpha(t))\cos(\beta(t)) + m\,l(t)\cos(\alpha(t))\,\beta'(t)^2\cos(\beta(t)) - m\,\mathrm{xp}''(t),$$
$$-K\,l(t)\sin(\beta(t)) + K\,\mathrm{l0}\sin(\beta(t)) - 2\,m\,l'(t)\,\beta'(t)\cos(\beta(t)) + m\,l(t)\,\beta'(t)^2\sin(\beta(t)) - m\,\mathrm{yp}''(t),$$
$$-G + K\,l(t)\sin(\alpha(t))\cos(\beta(t)) - K\,\mathrm{l0}\sin(\alpha(t))\cos(\beta(t)) + 2\,m\,l'(t)\,\alpha'(t)\cos(\alpha(t))\cos(\beta(t)) - 2\,m\,l'(t)\sin(\alpha(t))\,\beta'(t)\sin(\beta(t)) - 2\,m\,l(t)\,\alpha'(t)\cos(\alpha(t))\,\beta'(t)\sin(\beta(t)) - m\,l(t)\,\alpha'(t)^2\sin(\alpha(t))\cos(\beta(t)) - m\,l(t)\sin(\alpha(t))\,\beta'(t)^2\cos(\beta(t)) - m\,\mathrm{zp}''(t)\}$$

- **We could make inversion of such a simple system symbolically but standard procedure would be to make inversion numerically in Matlab.**

In[191]:=
```
AI = Inverse[A] // FullSimplify;
AI // TraditionalForm
```

Out[192]//TraditionalForm=

$$\begin{pmatrix} \frac{\cos(\alpha(t))\cos(\beta(t))}{m} & \frac{\sin(\beta(t))}{m} & -\frac{\cos(\beta(t))\sin(\alpha(t))}{m} \\ -\frac{\sec(\beta(t))\sin(\alpha(t))}{m\,l(t)} & 0 & -\frac{\cos(\alpha(t))\sec(\beta(t))}{m\,l(t)} \\ -\frac{\cos(\alpha(t))\sin(\beta(t))}{m\,l(t)} & \frac{\cos(\beta(t))}{m\,l(t)} & \frac{\sin(\alpha(t))\sin(\beta(t))}{m\,l(t)} \end{pmatrix}$$

In[193]:=
```
AI.b // FullSimplify // TraditionalForm
```

Out[193]//TraditionalForm=

$$\Big\{\frac{1}{m}\Big(\sin(\alpha(t))\cos(\beta(t))\,(G + m\,\mathrm{zp}''(t)) + l(t)\left(-K + m\,\alpha'(t)^2\cos^2(\beta(t)) + m\,\beta'(t)^2\right) + K\,\mathrm{l0} - m\cos(\alpha(t))\cos(\beta(t))\,\mathrm{xp}''(t) - m\sin(\beta(t))\,\mathrm{yp}''(t)\Big),$$
$$\frac{1}{m\,l(t)}(\sec(\beta(t))\,(\cos(\alpha(t))\,(G + m\,\mathrm{zp}''(t)) + 2\,m\,l(t)\,\alpha'(t)\,\beta'(t)\sin(\beta(t)) + m\sin(\alpha(t))\,\mathrm{xp}''(t)) - 2\,m\,l'(t)\,\alpha'(t)),$$
$$\frac{1}{m\,l(t)}\Big(-\sin(\alpha(t))\sin(\beta(t))\,(G + m\,\mathrm{zp}''(t)) - m\left(2\,l'(t)\,\beta'(t) + \cos(\beta(t))\left(l(t)\,\alpha'(t)^2\sin(\beta(t)) + \mathrm{yp}''(t)\right)\right) + m\cos(\alpha(t))\sin(\beta(t))\,\mathrm{xp}''(t)\Big)\Big\}$$

- **Preparations for solutions in Matlab**
- **Finally we convert a system of 3 partial diferencial second order equations to the system of 6 partial diferencial first order equation using subsitution e.g.**
  z1 = l
  z2 = α
  z3 = β
  z4 = l

z5 = α
z6 = β'
which allows to use function ode45 in Matlab to solve equations while knowing initial values w0 and external input {xp[t], yp[t], zp[t]}
The equations will be entered into ode45 as

- **Final equations for ode45**

In[194]:=
```
eqfin = Flatten[{l', α', β', AI.b}] // FullSimplify // TraditionalForm
```

Out[194]//TraditionalForm=

$$\Big\{l',\ \alpha',\ \beta',\ \frac{1}{m}\Big(\sin(\alpha(t))\cos(\beta(t))\,(G + m\,\mathrm{zp}''(t)) + l(t)\left(-K + m\,\alpha'(t)^2\cos^2(\beta(t)) + m\,\beta'(t)^2\right) + K\,\mathrm{l0} - m\cos(\alpha(t))\cos(\beta(t))\,\mathrm{xp}''(t) - m\sin(\beta(t))\,\mathrm{yp}''(t)\Big),$$
$$\frac{\sec(\beta(t))\,(\cos(\alpha(t))\,(G + m\,\mathrm{zp}''(t)) + 2\,m\,l(t)\,\alpha'(t)\,\beta'(t)\sin(\beta(t)) + m\sin(\alpha(t))\,\mathrm{xp}''(t)) - 2\,m\,l'(t)\,\alpha'(t)}{m\,l(t)},$$
$$\frac{-\sin(\alpha(t))\sin(\beta(t))\,(G + m\,\mathrm{zp}''(t)) - m\,(2\,l'(t)\,\beta'(t) + \cos(\beta(t))\,(l(t)\,\alpha'(t)^2\sin(\beta(t)) + \mathrm{yp}''(t))) + m\cos(\alpha(t))\sin(\beta(t))\,\mathrm{xp}''(t)}{m\,l(t)}$$
$$\Big\}$$

- **Conversion to matlab notation**

- **http : // library.wolfram.com/infocenter/MathSource/577/**
modify the path at the next line

In[223]:=
```
Get["ToMatlab.m", Path → "skola/Math/."]
```

- **Possible name conversions, the next expression says, how to name variables in Matlab.**
One can easily modify the names according one' s choice.
Also is possible to leave t as a parameter of the functions e.g. by rules {α -> alpha, ...},
which will result in "alpha(t)".

In[196]:=
```
cond = {l[t] → l, α[t] → alpha, β[t] → beta, l'[t] → dl, α'[t] → dalpha, β'[t] → dbeta,
    xp''[t] → ddxp, yp''[t] → ddyp, zp''[t] → ddzp};
```

- **ToMatlab looks similar but leave unwanted hidden characters '\'.**
PrintMatlab works smoothly when you want copy - paste the result into a Matlab.

In[209]:=
```
PrintMatlab[A /. cond]
PrintMatlab[b /. cond]
```

```
[m.*cos(alpha).*cos(beta),(-1).*l.*m.*cos(beta).*sin(alpha), ...
  (-1).*l.*m.*cos(alpha).*sin(beta);m.*sin(beta),0,l.*m.*cos( ...
  beta);(-1).*m.*cos(beta).*sin(alpha),(-1).*l.*m.*cos(alpha) ...
  .*cos(beta),l.*m.*sin(alpha).*sin(beta)];


[(-1).*ddxp.*m+K.*l0.*cos(alpha).*cos(beta)+2.*dl.*m.*( ...
  dalpha.*cos(beta).*sin(alpha)+dbeta.*cos(alpha).*sin(beta))+ ...
  l.*(dalpha.^2.*m.*cos(alpha).*cos(beta)+((-1).*K+dbeta.^2.* ...
  m).*cos(alpha).*cos(beta)+(-2).*dalpha.*dbeta.*m.*sin(alpha) ...
  .*sin(beta)),(-1).*ddyp.*m+(-2).*dbeta.*dl.*m.*cos(beta)+K.* ...
  l0.*sin(beta)+(-1).*l.*(K+(-1).*dbeta.^2.*m).*sin(beta),(-1) ...
  .*G+(-1).*ddzp.*m+(-1).*K.*l0.*cos(beta).*sin(alpha)+l.*(( ...
  -1).*dalpha.^2.*m.*cos(beta).*sin(alpha)+(K+(-1).*dbeta.^2.* ...
  m).*cos(beta).*sin(alpha)+(-2).*dalpha.*dbeta.*m.*cos(alpha) ...
  .*sin(beta))+2.*dl.*m.*(dalpha.*cos(alpha).*cos(beta)+(-1).* ...
  dbeta.*sin(alpha).*sin(beta))];
```