UK Atomic Energy Authority

# PACE

# Modelling, simulation & control of a rotary inverted pendulum

**Sam Herschmann**

sam.herschmann@ukaea.uk

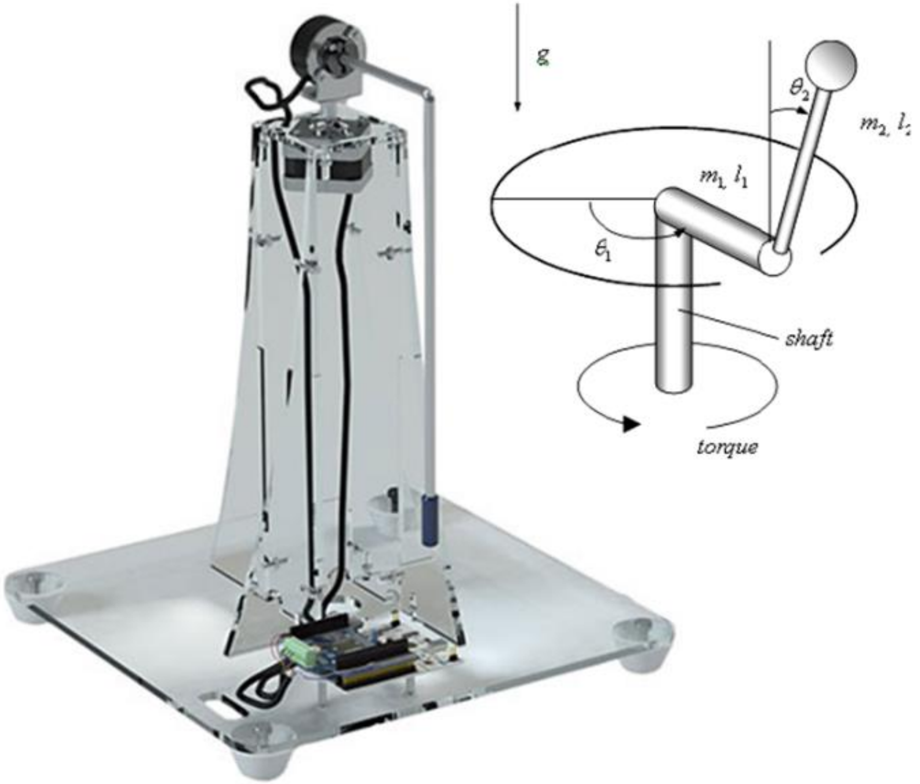
# We want to elegantly control this system.

At the minute, the control software “spins up” the pendulum in quite an aggressive way, and the stabilised payload sort of drifts over time.

**Challenge:** Can we do better by using state-space model-based control?

**Goals:**

- Improve spin up using trajectory optimisation
- Stabilise beam completely without drift




**STEVAL-EDUKIT01**

UK Atomic Energy Authority

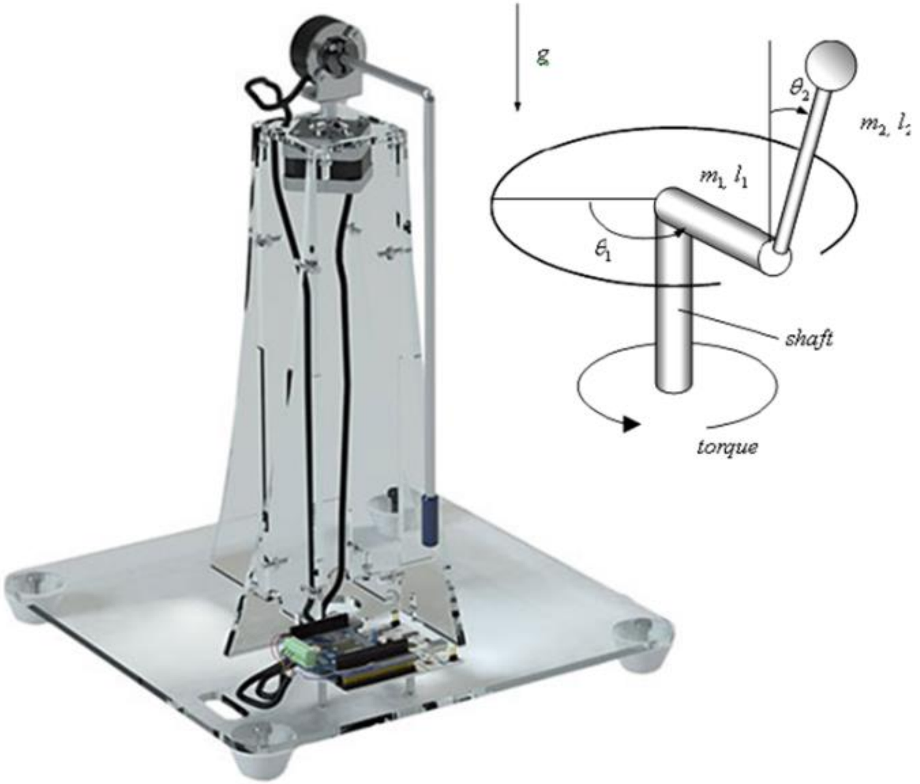
# Modelling

# Scope

These slides outline the steps required to derive the non-linear equations of motion (aka the dynamic model) of a rotary pendulum with only one actuator.

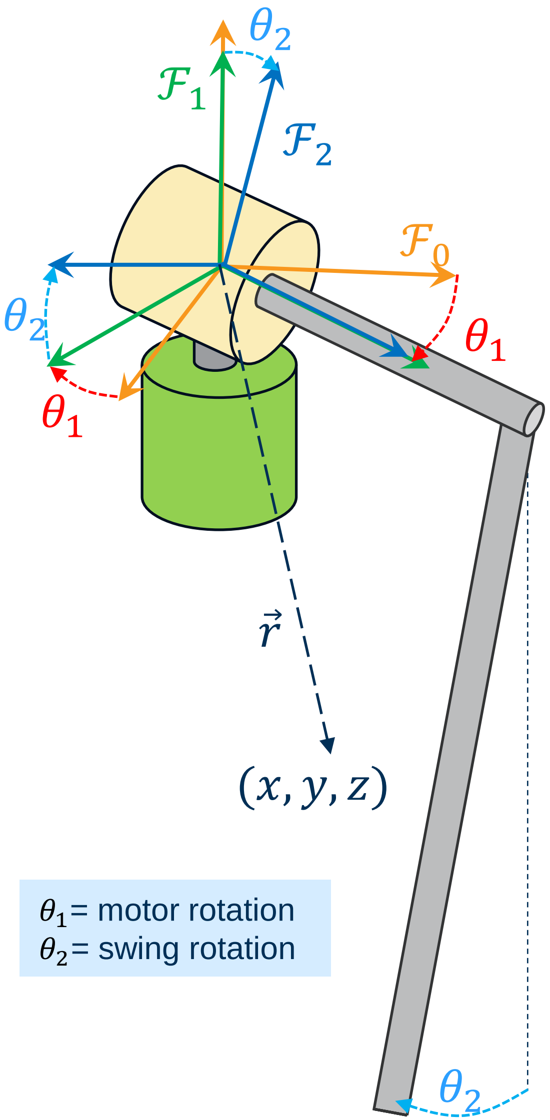
## Challenging topics involved:

- Vector calculus
- Rigid Body Kinematics & Dynamics
  - Lagrangian Mechanics
- Simulation of Ordinary Differential Equations (ODEs)




STEVAL-EDUKIT01

# Important Kinematics



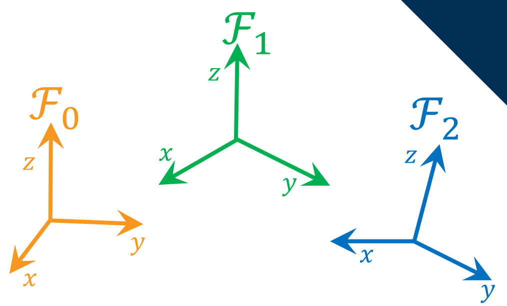

$\theta_1$= motor rotation
$\theta_2$= swing rotation

There are 3 frames of reference: $\mathcal{F}_0$, $\mathcal{F}_1$ and $\mathcal{F}_2$.

Inertial (globally fixed) frame

Encoder-fixed frame

Pendulum-fixed frame




- Let's define some position vector $\vec{r}$. We can express this as a numerical $(x, y, z)$ vector, but these values depend on the frame of reference that we choose to express the vector in.
- Let's define: "$\vec{r}$ expressed in frame $\mathcal{F}_0$" as $\boldsymbol{r}^{\mathcal{F}_0} \in \mathbb{R}^3$.
- For our pendulum system, it can be shown that:

$$\boldsymbol{r}^{\mathcal{F}_1} = \mathbf{R}_{10}(\theta_1)\boldsymbol{r}^{\mathcal{F}_0}$$
$$\boldsymbol{r}^{\mathcal{F}_2} = \mathbf{R}_{21}(\theta_2)\boldsymbol{r}^{\mathcal{F}_1}$$

$$\boldsymbol{r}^{\mathcal{F}_2} = \underbrace{\mathbf{R}_{21}\mathbf{R}_{10}}\boldsymbol{r}^{\mathcal{F}_0}$$

$$\mathbf{R}_{20}(\theta_1, \theta_2) = \begin{bmatrix} \cos\theta_1 \cos\theta_2 & -\sin\theta_1 & \sin\theta_2 \cos\theta_1 \\ \sin\theta_1 \cos\theta_2 & \cos\theta_1 & \sin\theta_1 \sin\theta_2 \\ -\sin\theta_2 & 0 & \cos\theta_2 \end{bmatrix}$$

This is known as a $z \rightarrow y'$ (yaw-pitch) transformation

**Rotation matrices R**

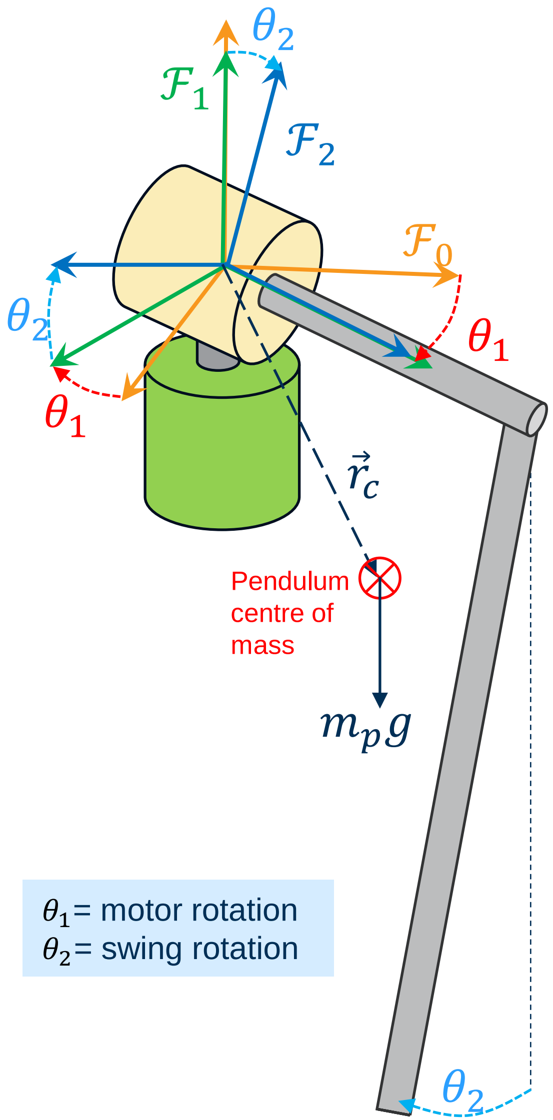
$$\mathbf{R}_{10} = \mathbf{R}_z(\theta_1) = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{R}_{21} = \mathbf{R}_y(\theta_2) = \begin{bmatrix} \cos\theta_2 & 0 & \sin\theta_2 \\ 0 & 1 & 0 \\ -\sin\theta_2 & 0 & \cos\theta_2 \end{bmatrix}$$

$\mathbf{R}_{20}$ describes the orientation of the pendulum body with respect to the inertial frame of reference.

# Calculating the energy of the pendulum






$\theta_1$= motor rotation
$\theta_2$= swing rotation

### Gravitational Potential Energy

$$V = m_p g \boxed{\boldsymbol{r}_c\Big|_z}$$

Vertical position of the centre of mass, expressed in the inertial frame.
$\boldsymbol{r}_c = \boldsymbol{r}_c^{\mathcal{F}_0} = \mathbf{R}_{20}(\theta_1, \theta_2)^\top \boldsymbol{r}_c^{\mathcal{F}_2}$

### Rotational Kinetic energy

$$T_{rot_p} = \frac{1}{2}\boldsymbol{\omega}_p^\top \boxed{\boldsymbol{I}_p} \boxed{\boldsymbol{\omega}_p}$$

Pendulum inertia tensor, <u>calculated in the next slide</u>

Angular velocity of pendulum, expressed in the inertial frame. This can be shown to be:

$$\boldsymbol{\omega}_p = \boldsymbol{\omega}_p^{\mathcal{F}_0} = \mathbf{R}_z(\theta_1)^\top \begin{bmatrix} 0 \\ \dot{\theta}_2 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \dot{\theta}_1 \end{bmatrix}$$

### Linear Kinetic energy

$$T_{lin_p} = \frac{1}{2}\boxed{m_p}\dot{\boldsymbol{r}}_c^\top \boxed{\dot{\boldsymbol{r}}_c}$$

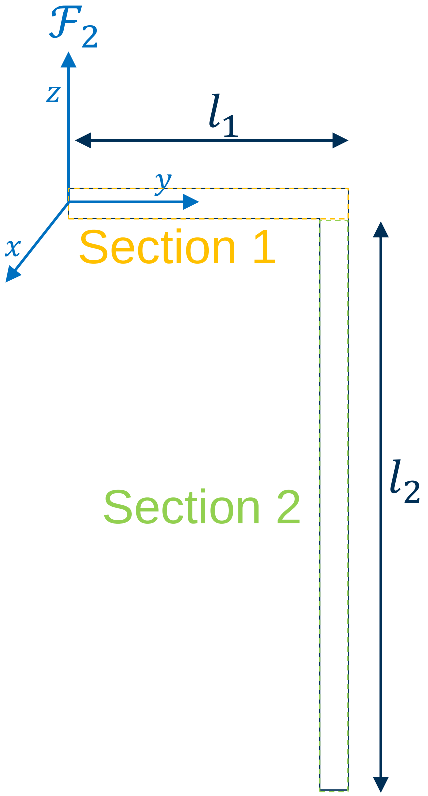
Pendulum Mass

Linear velocity of pendulum centre of mass, expressed in the inertial frame. This can be shown to be:
$\dot{\boldsymbol{r}}_c = \dot{\boldsymbol{r}}_c^{\mathcal{F}_0} = \dot{\mathbf{R}}_{20}(\theta_1, \theta_2)^\top \boldsymbol{r}_c^{\mathcal{F}_2}$

**We can now define the systems 'Lagrangian':** $L = T_{rot_p} + T_{lin_p} - V$

# Aside: Approximating the pendulum Inertia Tensor $I_p$

UK Atomic Energy Authority



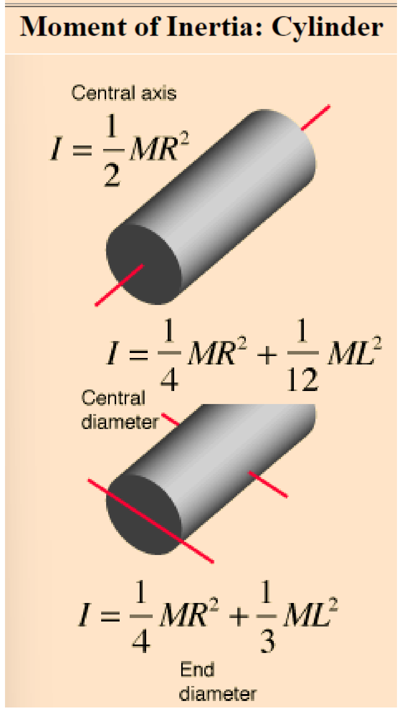

$\boldsymbol{I}_p$ can be thought of as the 'rotational mass' of an object. This depends on the axis of rotation, which makes $\boldsymbol{I}_p$ a 3X3 matrix.

$$\boldsymbol{I}_p^{\mathcal{F}_2} = \boldsymbol{I}_1^{\mathcal{F}_2} + \boldsymbol{I}_2^{\mathcal{F}_2}$$

Pendulum inertia tensor resolved in the moving (or 'body') frame $\mathcal{F}_2$

**Assumption: solid cylinder**

(can improve this using measurements)



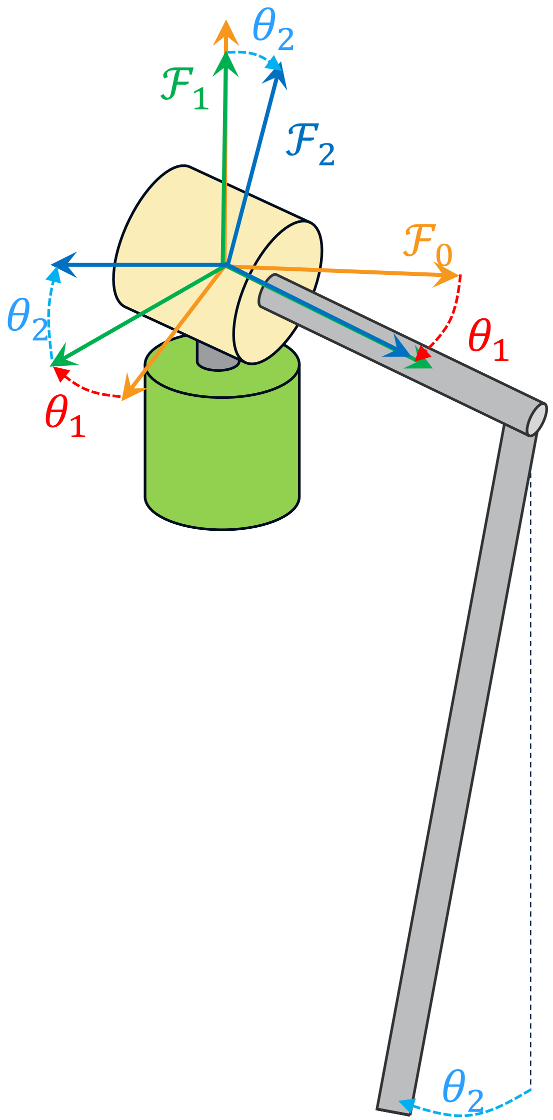

Section 1

$$\boldsymbol{I}_1^{\mathcal{F}_2} = \begin{bmatrix} \frac{1}{4}m_1r^2 + \frac{1}{3}m_1l_1^2 & 0 & 0 \\ 0 & \frac{1}{2}m_1r^2 & 0 \\ 0 & 0 & \frac{1}{4}m_1r^2 + \frac{1}{3}ml_1^2 \end{bmatrix}$$

Section 2

$$\boldsymbol{I}_2^{\mathcal{F}_2} = \begin{bmatrix} \frac{1}{4}m_2r^2 + \frac{1}{12}m_2l_2^2 & 0 & 0 \\ 0 & \frac{1}{4}m_2r^2 + \frac{1}{12}m_2l_2^2 & 0 \\ 0 & 0 & \frac{1}{2}m_2r^2 \end{bmatrix} + \underbrace{m_2\left(\begin{bmatrix} 0 \\ l_1 \\ \frac{-l_2}{2} \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} 0 \\ l_1 \\ \frac{-l_2}{2} \end{bmatrix} \boldsymbol{E}_3 - \begin{bmatrix} 0 \\ l_1 \\ \frac{-l_2}{2} \end{bmatrix} \begin{bmatrix} 0 \\ l_1 \\ \frac{-l_2}{2} \end{bmatrix}^{\mathsf{T}}\right)}_{\text{From the parallel axis theorem}}$$

# Finding the state-space 'Equations of Motion'




Let's define our state coordinates, $\boldsymbol{q} = \begin{bmatrix}\theta_1 \\ \theta_2\end{bmatrix}$.

→ $\boldsymbol{q}$ completely describes the position of our pendulum

## Euler-Lagrange Equations

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\boldsymbol{q}}}\right) - \frac{\partial L}{\partial \boldsymbol{q}} = \boxed{\boldsymbol{F}}$$

The 'external force' vector. If we ignore friction damping, then $\boldsymbol{F} = \begin{bmatrix}u \\ 0\end{bmatrix} = \begin{bmatrix}1 \\ 0\end{bmatrix}u$, where $u$ is the torque applied by the motor.

This equation basically creates the "F=ma" of the system: the equation that describes the behaviour of the pendulum. Also known as a "Dynamic Model".

By substituting $L$ into this and expanding / rearranging, the result ends up looking something like this:

"Mass Matrix" $\boxed{\boldsymbol{M}(\boldsymbol{q})}\ddot{\boldsymbol{q}} + \boxed{\boldsymbol{N}(\boldsymbol{q}, \dot{\boldsymbol{q}})} = \boldsymbol{F}$

Vector describing gravity & Coriolis effects

$\equiv$

Often it is nice to express this in the *equivalent* 'first order ODE form', by defining a new set of variables

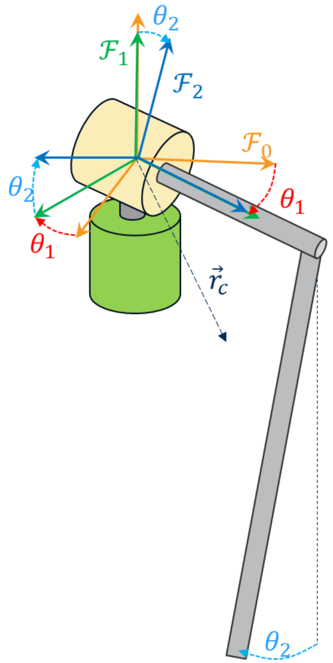
$$\boldsymbol{x} = \begin{bmatrix}\boldsymbol{q} \\ \dot{\boldsymbol{q}}\end{bmatrix} = \begin{bmatrix}\theta_1 \\ \theta_2 \\ \dot{\theta}_1 \\ \dot{\theta}_2\end{bmatrix}$$

$$\dot{\boldsymbol{x}} = \begin{bmatrix}\dot{\boldsymbol{q}} \\ \boldsymbol{M}^{-1}(\boldsymbol{F} - \boldsymbol{N})\end{bmatrix} = \underbrace{\begin{bmatrix}\dot{\boldsymbol{q}} \\ -\boldsymbol{M}^{-1}\boldsymbol{N}\end{bmatrix}}_{\boldsymbol{f}(\boldsymbol{x})} + \underbrace{\begin{bmatrix}\boldsymbol{0} \\ \boldsymbol{M}^{-1}\begin{bmatrix}1 \\ 0\end{bmatrix}\end{bmatrix}}_{\boldsymbol{g}(\boldsymbol{x})} u$$

$$\boxed{\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g}(\boldsymbol{x})u}$$

This model form is a "Control affine" system, meaning the input $u$ appears linearly

# Simulating the free-response system ($u$=0)

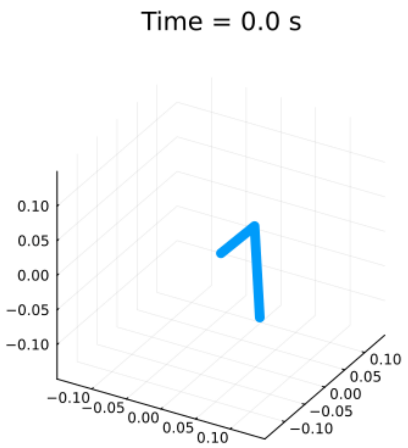


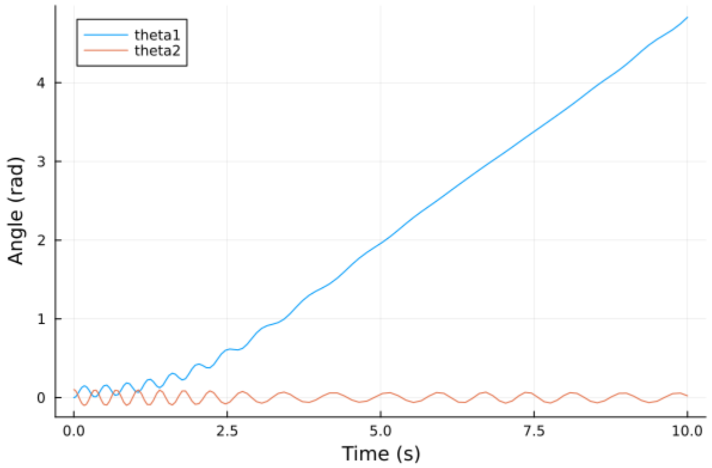

Given a starting state $\boldsymbol{x}(\boldsymbol{t}=\boldsymbol{0})=\boldsymbol{x}_0$, the model equations $\dot{\boldsymbol{x}}=\boldsymbol{f}(\boldsymbol{x})$ can be 'solved' (aka 'simulated') using a variety of techniques (aka 'ODE solvers').

- This can be done in Python, Matlab, C++, Julia and more.
  - ➢ I use ODE solvers in Julia's DifferentialEquations.jl toolbox, this is just my personal preference.

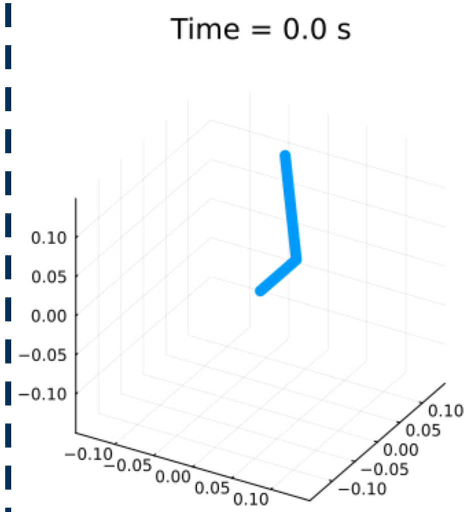
You can then create an animation of the model simulation.

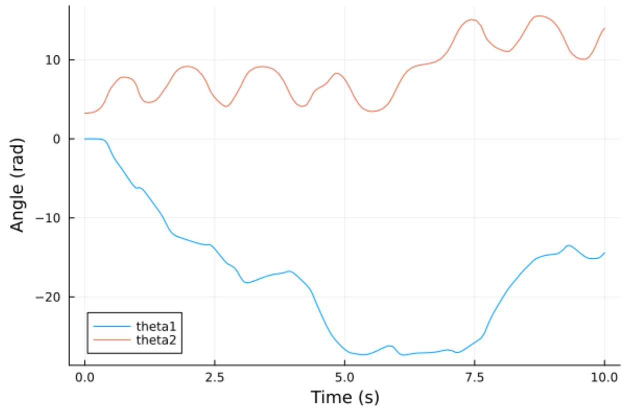
**Example 1**: $\boldsymbol{x}_0=\begin{bmatrix}\theta_{1_0}\\ \theta_{2_0}\\ \dot{\theta}_{1_0}\\ \dot{\theta}_{2_0}\end{bmatrix}=\begin{bmatrix}0\\ 0.1\\ 0\\ 0\end{bmatrix},\quad u=0$




**Example 2**: $\boldsymbol{x}_0=\begin{bmatrix}0\\ \pi+0.1\\ 0\\ 0\end{bmatrix},\quad u=0$



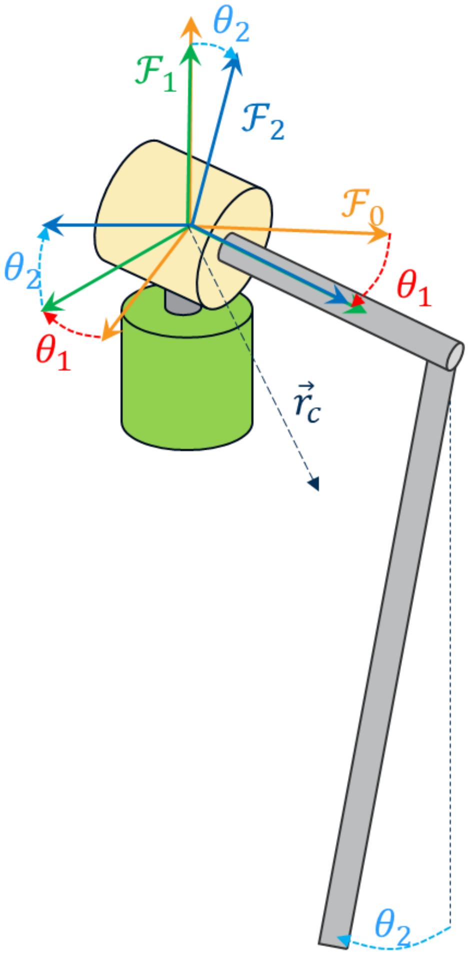




**Very chaotic!**
(no damping is modelled)

# Adding damping to the model

In reality, there will be *viscous damping* at the joints.

- This is a friction torque proportional to the joint velocity:




$$\tau_{fr_i} \approx -\boxed{d_{fr_i}}\dot{\theta}_i$$

A damping constant of proportionality for joint $i$

In vector form:

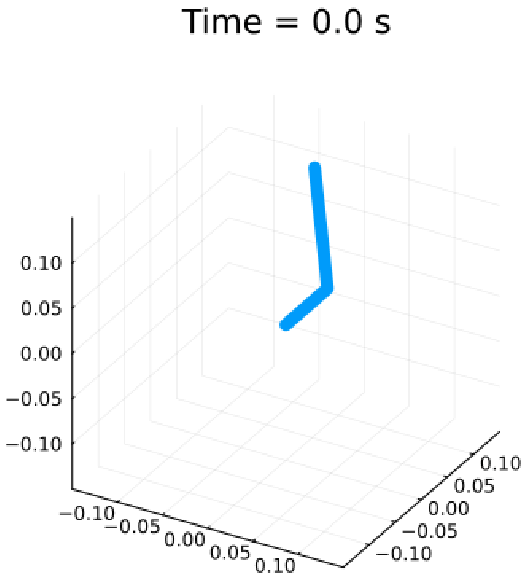
$$\boldsymbol{\tau}_{fr} \approx -\boxed{\boldsymbol{D}}\dot{\boldsymbol{\theta}}$$

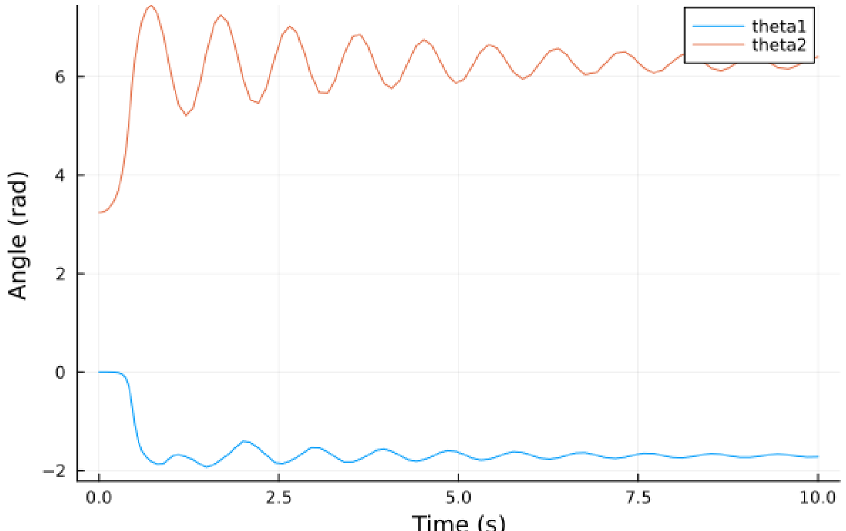
A diagonal Damping matrix

The model now becomes:

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{N}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{D}\dot{\boldsymbol{\theta}} = \boldsymbol{F}$$

$$\dot{\boldsymbol{x}} = \underbrace{\begin{bmatrix} \dot{\boldsymbol{q}} \\ \boldsymbol{M}^{-1}(\boldsymbol{F} - \boldsymbol{N} + \boldsymbol{D}\dot{\boldsymbol{\theta}}) \end{bmatrix}}_{f(x,u)}$$






UK Atomic Energy Authority

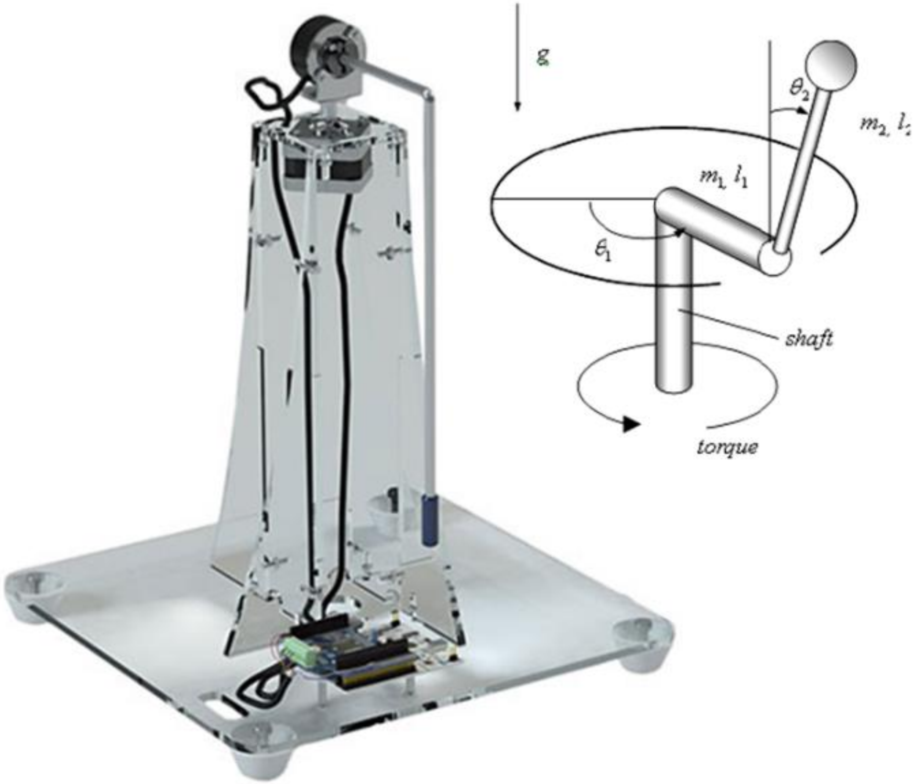
UK Atomic Energy Authority

# Stepper motor stabilization control

# Scope

Before detailing a technique for “spin-up” we want to explore techniques for stabilising the pendulum in its unstable position.

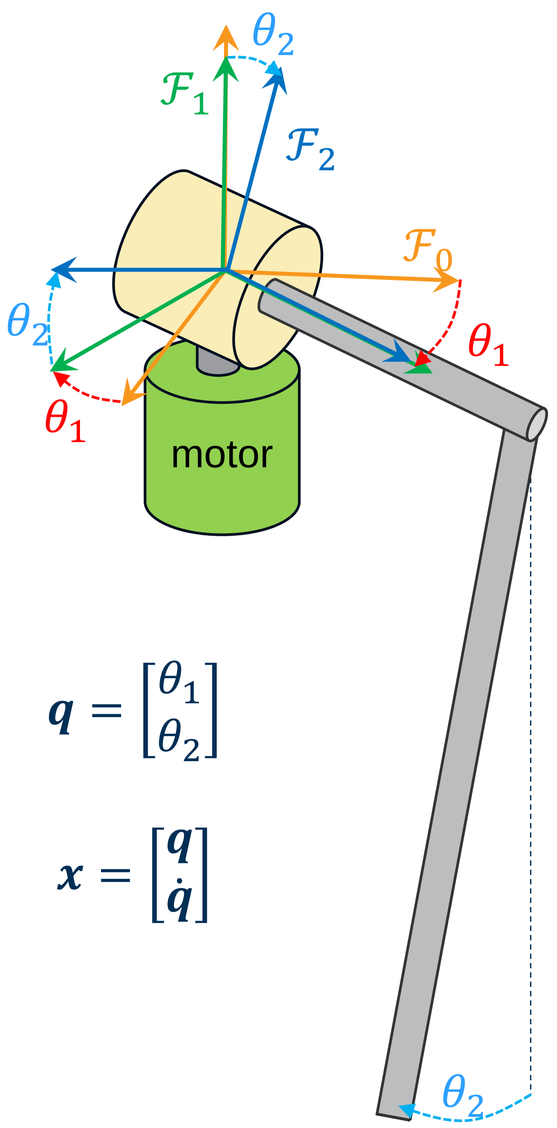
- Can we control both angles, $\theta_1$ and $\theta_2$?




STEVAL-EDUKIT01

- **Key point**: We cannot command the torque of the stepper motor, but we can command the acceleration…
- We need to do a bit of reformulation of the equations of motion to reflect this.

# What if our control input is $u = \ddot{\theta}_1$, instead of the motor torque?




Let's assume we can precisely control the acceleration of the stepper motor $\ddot{\theta}_1$. In terms of physics, this means the motor imparts a torque on the system that ensures $\ddot{\theta}_1$ follows a desired acceleration, $u(\mathrm{t})$.

Ideally, we want our system dynamic model of the following form:

$$\boxed{\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g}(\boldsymbol{x})u}$$

where $u$ is now acceleration and not torque.

$$\boldsymbol{q} = \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix}$$

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{q} \\ \dot{\boldsymbol{q}} \end{bmatrix}$$

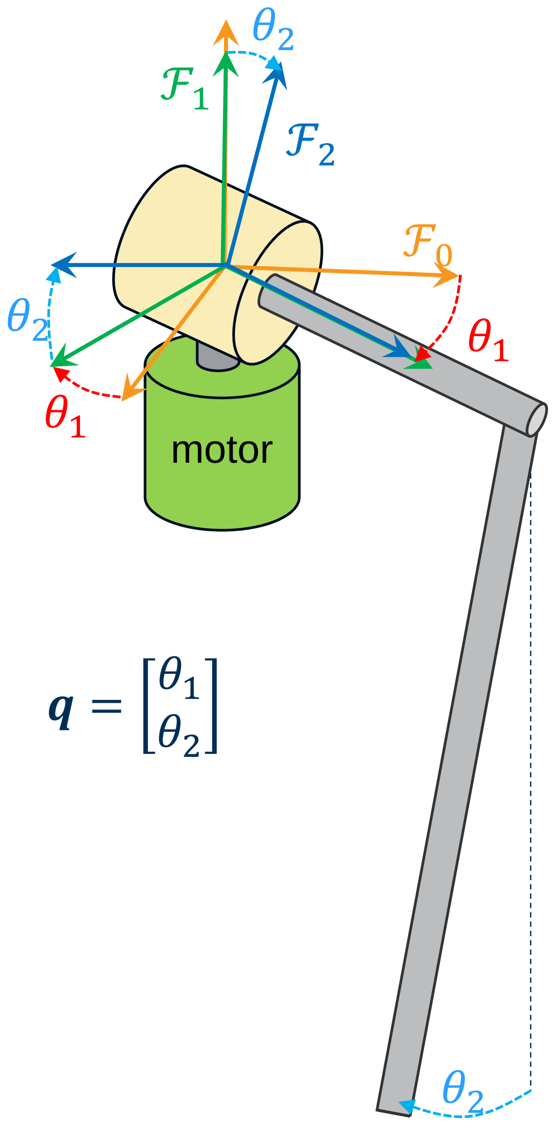
How can we build the equation above?

- ➢ Using Lagrange Multiplier approach: see next slide.

# What if our control input is $u = \ddot{\theta}_1$, instead of the motor torque?



-> Lets express $u = \ddot{\theta}_1 = \underbrace{[1 \ \ 0]}_{\boldsymbol{\Upsilon}} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} = \boldsymbol{\Upsilon}\ddot{\boldsymbol{q}}$. We will call this our constraint.




$$\boldsymbol{q} = \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix}$$

**Dynamics equations**

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{N} + \boldsymbol{\Upsilon}^\top \tau_1 = 0 \qquad (1)$$

rearrange

$$\ddot{\boldsymbol{q}} = \boldsymbol{M}^{-1}(-\boldsymbol{N} - \boldsymbol{\Upsilon}^\top \tau_1) = 0$$

Sub into

**Constraint equations**

$$\boldsymbol{\Upsilon}\,\ddot{\boldsymbol{q}} - u(t) = 0$$

$$\boldsymbol{\Upsilon}\,\boldsymbol{M}^{-1}(-\boldsymbol{N} - \boldsymbol{\Upsilon}^\top \tau_1) - u(t) = 0$$

rearrange

$$\tau_1 = -(\boldsymbol{\Upsilon}\boldsymbol{M}^{-1}\boldsymbol{\Upsilon}^\top)^{-1}(\boldsymbol{\Upsilon}\boldsymbol{M}^{-1}\boldsymbol{N} + u)$$

*This can be thought of as the motor torque which achieves the constraint.*

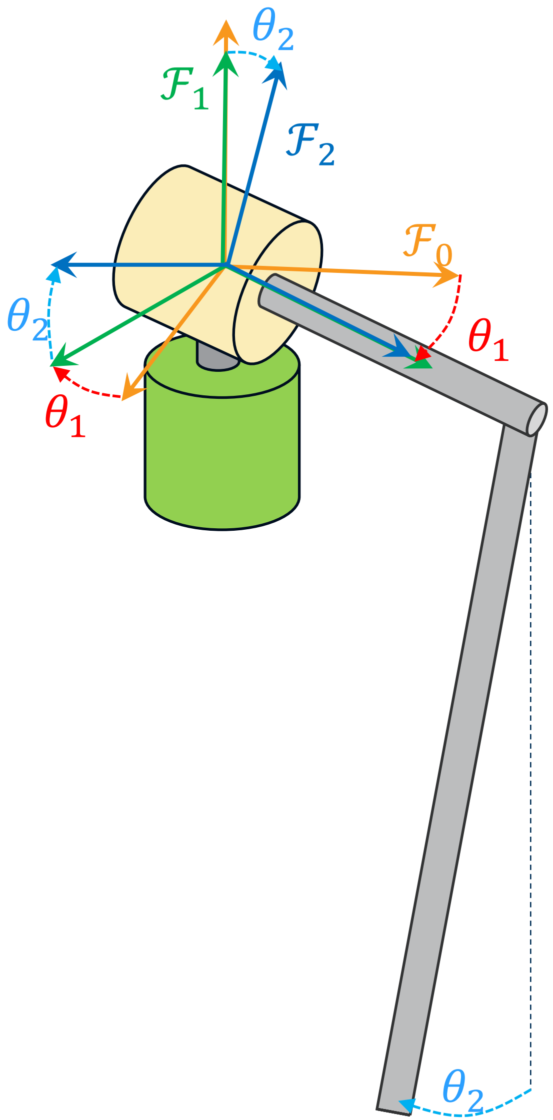
Sub into (1)

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{N} - \boldsymbol{\Upsilon}^\top(\boldsymbol{\Upsilon}\boldsymbol{M}^{-1}\boldsymbol{\Upsilon}^\top)^{-1}(\boldsymbol{\Upsilon}\boldsymbol{M}^{-1}\boldsymbol{N} + u) = 0$$

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{N}_a = \boldsymbol{B}_a u \quad \begin{cases} \boldsymbol{N}_a = \boldsymbol{N} - \boldsymbol{\Upsilon}^\top(\boldsymbol{\Upsilon}\boldsymbol{M}^{-1}\boldsymbol{\Upsilon}^\top)^{-1}\boldsymbol{\Upsilon}\boldsymbol{M}^{-1}\boldsymbol{N} \\ \boldsymbol{B}_a = \boldsymbol{\Upsilon}^\top(\boldsymbol{\Upsilon}\boldsymbol{M}^{-1}\boldsymbol{\Upsilon}^\top)^{-1} \end{cases}$$

# What if our control input is $u = \ddot{\theta}_1$, instead of the motor torque?






$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{N}_a = \boldsymbol{B}_a u \quad \begin{cases} \boldsymbol{N}_a = \boldsymbol{N} - \boldsymbol{\Upsilon}^\top(\boldsymbol{\Upsilon}\boldsymbol{M}^{-1}\boldsymbol{\Upsilon}^\top)^{-1}\boldsymbol{\Upsilon}\boldsymbol{M}^{-1}\boldsymbol{N} \\ \boldsymbol{B}_a = \boldsymbol{\Upsilon}^\top(\boldsymbol{\Upsilon}\boldsymbol{M}^{-1}\boldsymbol{\Upsilon}^\top)^{-1} \end{cases}$$

## First order form

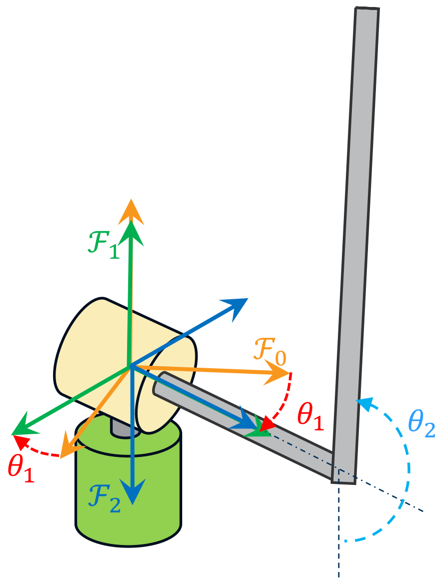
$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} = \begin{bmatrix} \dot{\boldsymbol{q}} \\ \ddot{\boldsymbol{q}} \end{bmatrix} = \underbrace{\begin{bmatrix} \dot{\boldsymbol{q}} \\ -\boldsymbol{M}^{-1}\boldsymbol{N}_a \end{bmatrix}}_{\boldsymbol{f}(\boldsymbol{x})} + \underbrace{\begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{M}^{-1}\boldsymbol{B}_a \end{bmatrix}}_{\boldsymbol{g}(\boldsymbol{x})} u$$

$$\boxed{\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g}(\boldsymbol{x})u}$$

# Stabilisation feedback control

**Challenge:** What should $u(t)$ be if we want to *stabilise* the pendulum, once we have spun it up?

One common approach is to linearise the dynamics about the unstable equilibrium and then apply some classical linear control techniques, for example LQR.




$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g}(\boldsymbol{x})u$$

Linearise about unstable equilibrium $\boldsymbol{x}_e$

$$\Delta\dot{\boldsymbol{x}} \approx \underbrace{\left.\frac{\partial \boldsymbol{f}(\boldsymbol{x})}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_e}}_{A} \Delta\boldsymbol{x} + \underbrace{\left.\frac{\partial \boldsymbol{g}(\boldsymbol{x})}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_e}}_{B} u$$

***Linearised dynamics***

$$\dot{\boldsymbol{x}} \approx \boldsymbol{A}\Delta\boldsymbol{x} + \boldsymbol{B}u$$

*(but only when close to the equilibrium!)*

***Define equilibrium:***

$$\boldsymbol{x}_e = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix}_{equil} = \begin{bmatrix} \theta_{1_{equil}} \\ k\pi \\ 0 \\ 0 \end{bmatrix}, where\ k = any\ integer$$

-> *lets choose* $\theta_{1_{equil}} = 0\ rad$

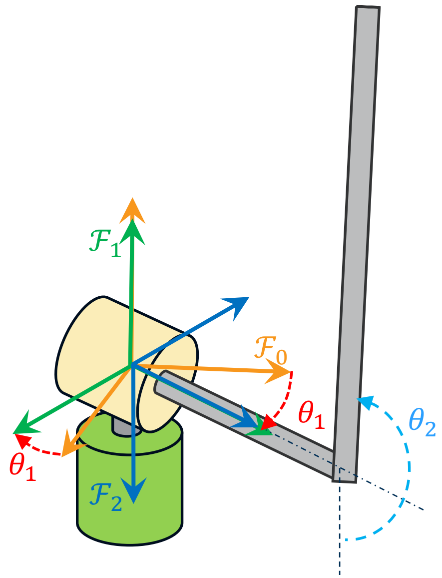
$$\Delta\boldsymbol{x} \coloneqq \boldsymbol{x} - \boldsymbol{x}_e$$
$$\Delta\dot{\boldsymbol{x}} = \dot{\boldsymbol{x}} - \cancel{\dot{\boldsymbol{x}}_e} = \dot{\boldsymbol{x}}$$

If we have done everything right, our linearized system matrices should look something like this:

```
A =
 0.0   0.0               1.0  0.0
 0.0   0.0               0.0  1.0
 0.0   0.0               0.0  0.0
 0.0   37.09611110895643 0.0  0.0
B =
 0.0
 0.0
 1.0
 0.18529148260335027
```

# Stabilisation feedback control with LQR

**Challenge:** What should $u(t)$ be if we want to *stabilise* the pendulum, once we have spun it up?



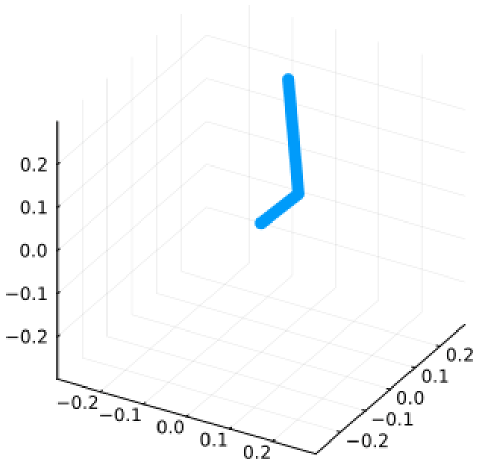

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\Delta\boldsymbol{x} + \boldsymbol{B}u$$

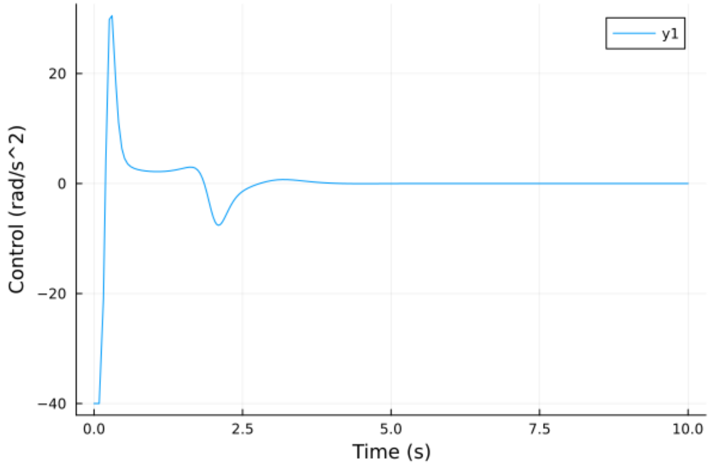
Lets choose: $u = \boldsymbol{K}\Delta\boldsymbol{x}$

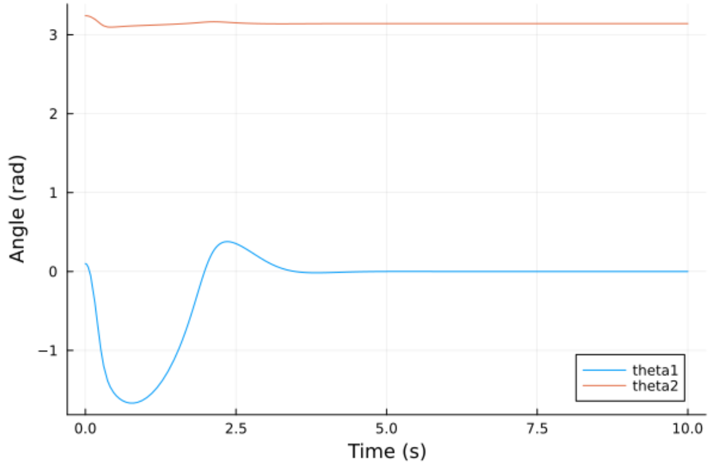
- Where $\boldsymbol{K}$ are some carefully calculated gains.
  - LQR is where these gains are calculated in a particular way as to minimise a cost function.

## LQR simulation

Time = 0.0 s

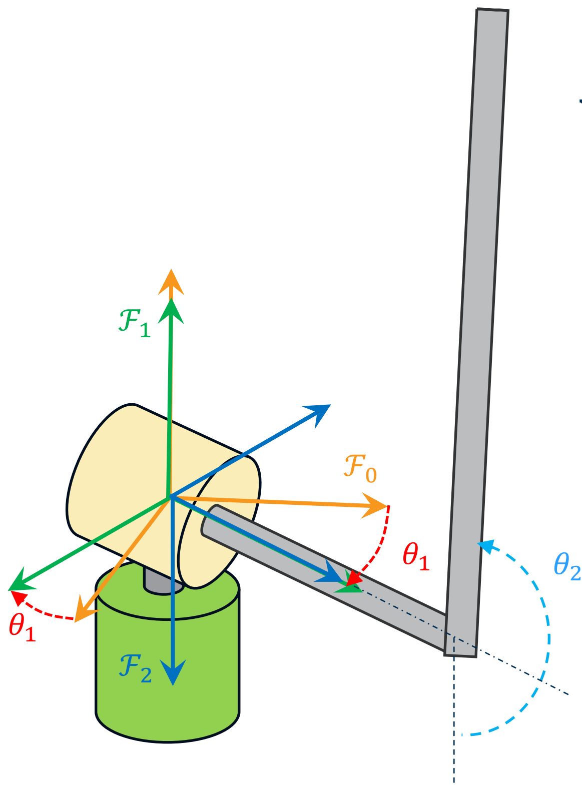


# Stabilisation feedback control with LQR

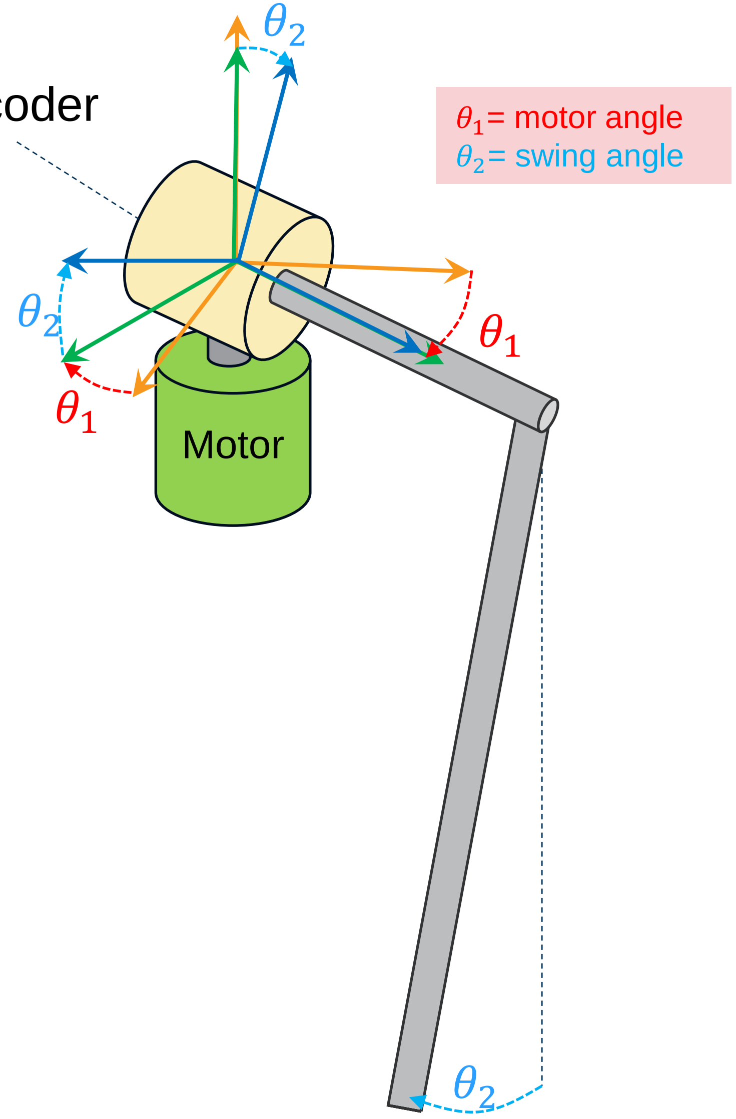
## Key points to note:




We need to have a good knowledge of the full state $x = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix}$

- We have $\theta_1$ and $\theta_2$ from encoders. We need to calculate $\dot{\theta}_1$ and $\dot{\theta}_2$ at every time step
  - Filtering required?

# STEVAL-EDUKIT01



# Hardware / embedded Queries

# Important things to double check:

UK Atomic Energy Authority

- What is the maximum acceleration that the motor can impart?

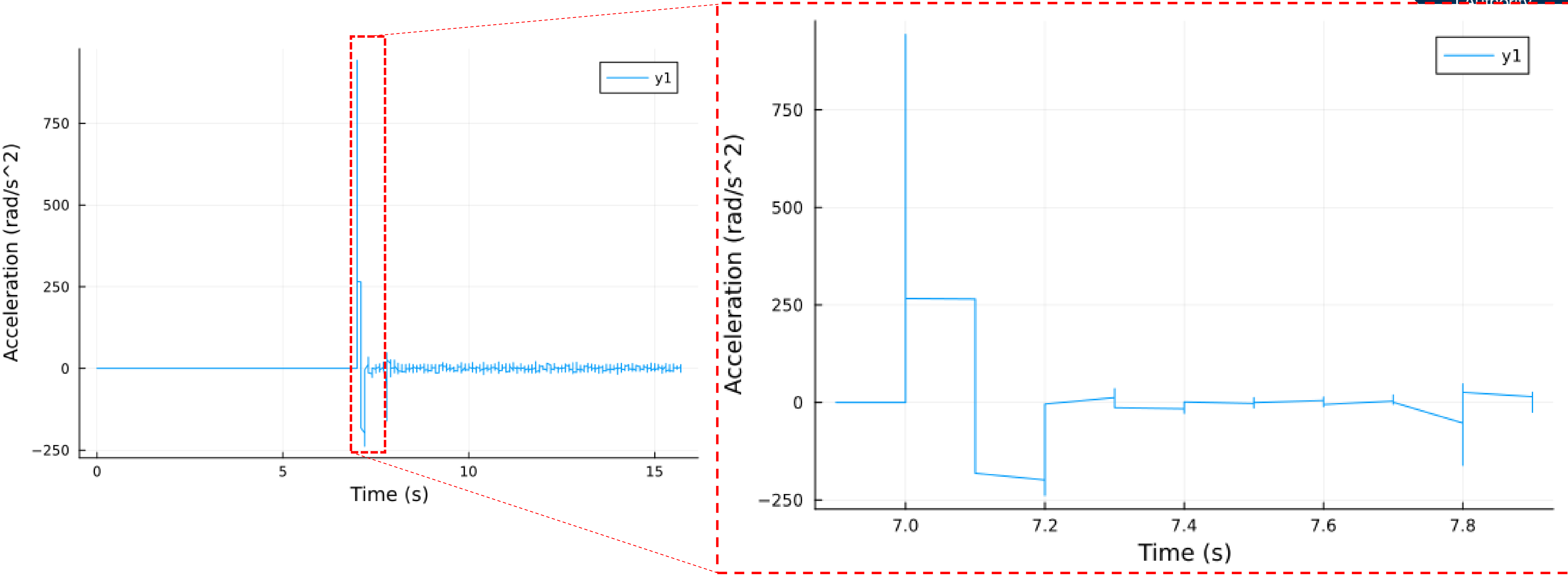
- How accurately can we command the acceleration?
  - ➢ What is the signal rate? (how many commands per second)

- Measurements:
  - ➢ How noisy is are the system measurements?
  - ➢ Do we measure rotation velocity?
  - ➢ Are there signal delays?

➢ How can we test these questions?

# Important things to double check:

UK Atomic Energy Authority




- Are we logging at 10Hz? or is the command signal 10 Hz.
- **Either way, we need faster please!**

UK Atomic Energy Authority

# Swing-up

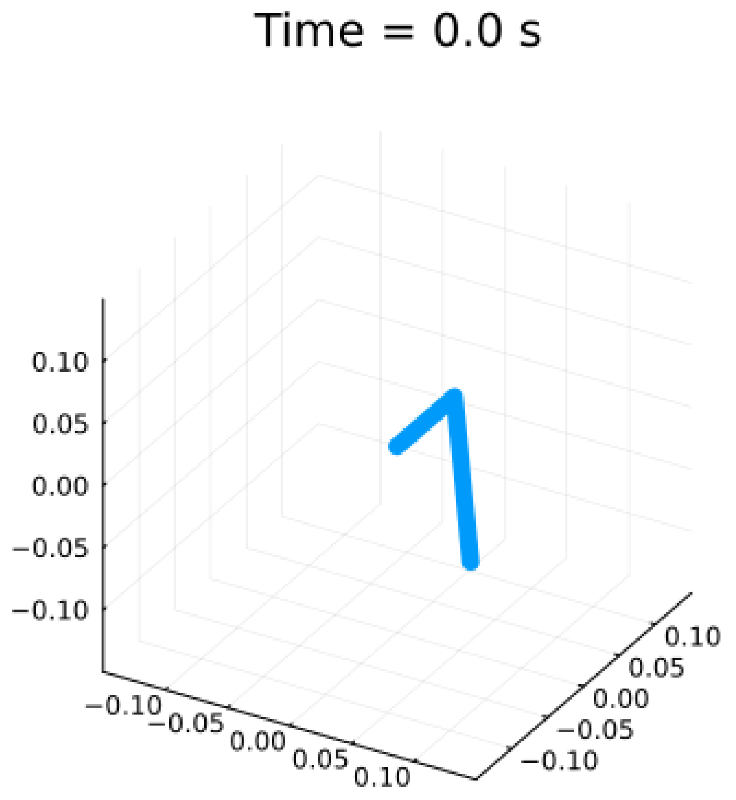
Trajectory Optimisation

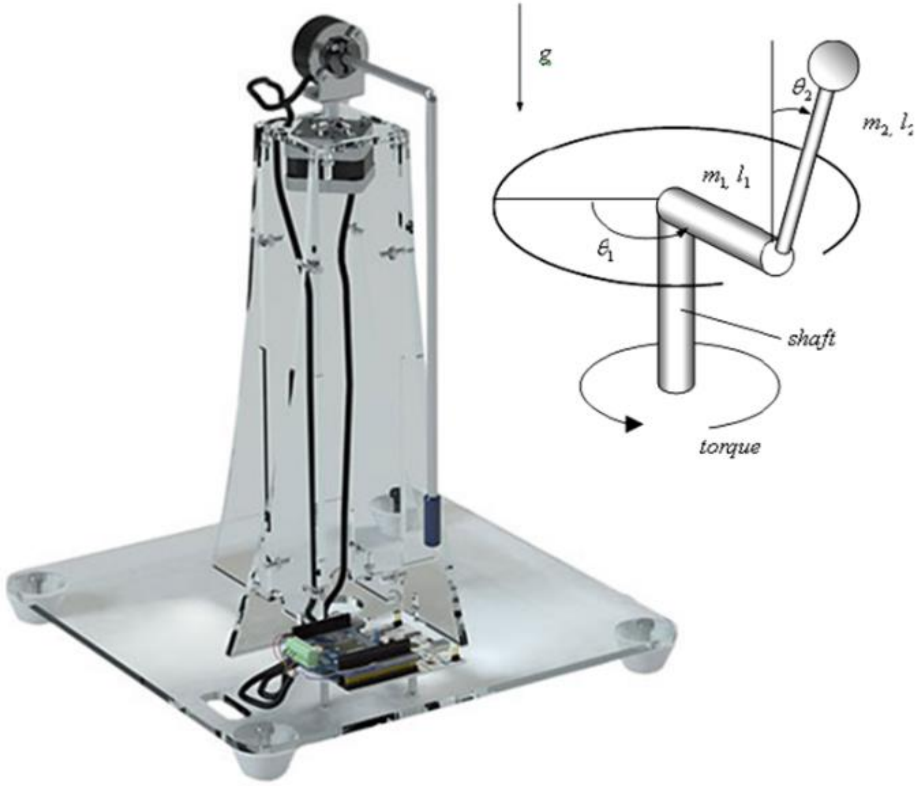
# We want to 'swing up' the pendulum.

**How can we do this intelligently? Can we consider**:

- Motor Torque / Velocity / acceleration limits
- Spatial constraints

One approach is **trajectory optimisation**



# How?

UK Atomic Energy Authority

TBC. Need time to write-up

# System Identification

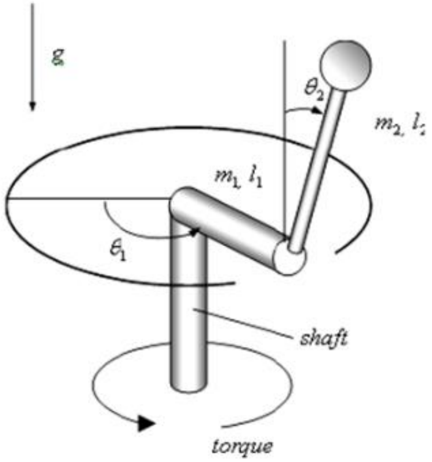
# Why System Identification?

**Our analytical model may be poor due to making bad estimates of:**

$$m_1, m_2, d_{fr_2}, \boldsymbol{r}_c^{\mathcal{F}_2}$$

**Can we improve our model parameters using observed data? YES**

$$\boldsymbol{p} = m_1, \\ m_2, d_{fr_2}, \boldsymbol{r}_c^{\mathcal{F}_2}$$



# How?

UK Atomic Energy Authority

# Next steps

➢ **Improve model parameters & verify**

➢ **Implement on real set-up**