Extend this to n-dof system

$$\left.\begin{aligned} m\ddot{q} + c\dot{q} + kq &= \tau \\ \tau &= -k_p q - k_d \dot{q} \end{aligned}\right\} \text{1-dof}$$

$\longrightarrow$ $k_d = -c + 2\sqrt{(k+k_p)m}$ critically damped

---

Now consider a 2dof system

$$\begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} + \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} + \begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix}$$

$$\tau = -k_p q - k_d \dot{q}$$

$$\begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} = \begin{bmatrix} k_{p11} & k_{p12} \\ k_{p21} & k_{p22} \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} - \begin{bmatrix} k_{d11} & k_{d12} \\ k_{d21} & k_{d22} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}$$

Choose 8 numbers. $\rightarrow$ Too unwieldy

Control partitioning

1 — Consider $\rightarrow M\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau$

2 — Choose $\rightarrow \tau = \underline{\hat{M}(-k_p q - k_d \dot{q}) + \hat{C}(q,\dot{q})\dot{q} + \hat{G}(q)}$

$\hat{M}, \hat{C}, \hat{G}$ — are estimates of M, C, G

Sub (2) in (1)

$$M\ddot{q} + C(\underline{\underline{q,\dot{q}}})\dot{q} + G(q) = \hat{M}(-k_p q - k_d \dot{q}) + \hat{C}(q,\dot{q})\dot{q} + \hat{G}(q)$$

Lets assume $M = \hat{M}$, $C = \hat{C}$, $G = \hat{G}$

$$M\ddot{q} + \cancel{C\dot{q}} + \cancel{G} = \underline{\underline{M}}(-k_p q - k_d \dot{q}) + \cancel{C\dot{q}} + \cancel{G}$$

$$M(\ddot{q} + k_d \dot{q} + k_p q) = 0$$

$$M \neq 0 \Rightarrow \ddot{q} + k_d \dot{q} + \underline{k_p} q = 0$$

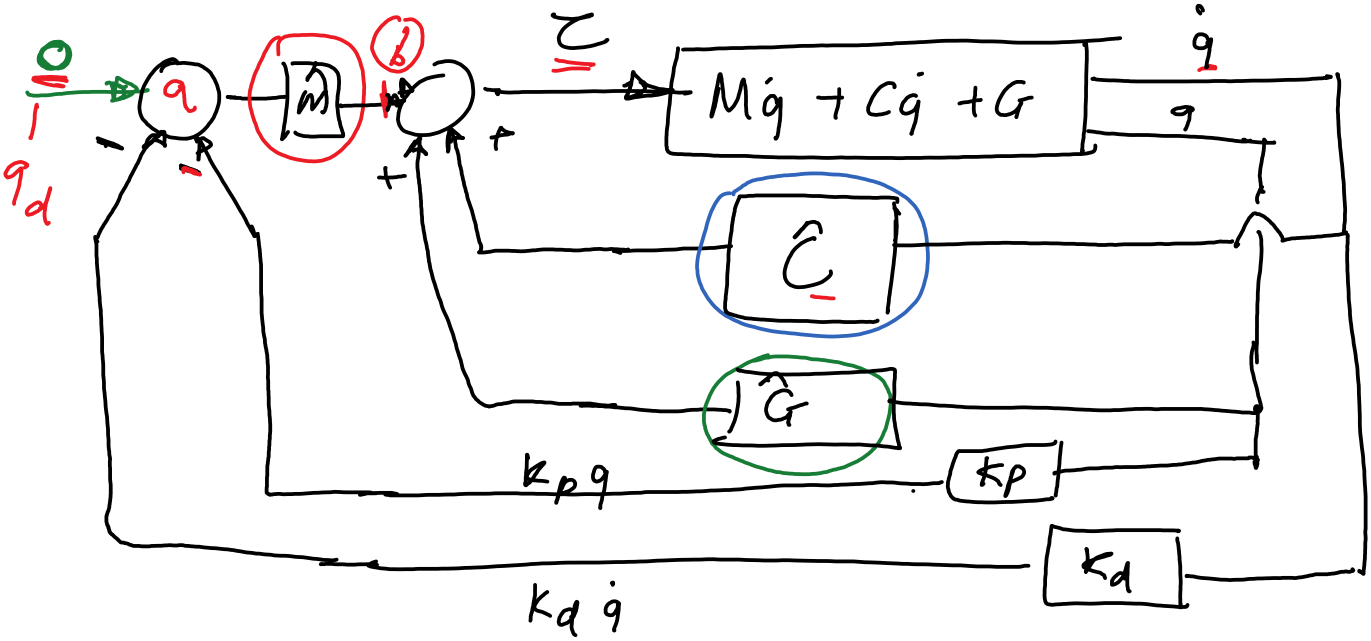
$$\longrightarrow \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} + \begin{bmatrix} k_{d_1} & 0 \\ 0 & k_{d_2} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} + \begin{bmatrix} k_{p_1} & 0 \\ 0 & k_{p_2} \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} = \begin{matrix} 0 \\ 0 \end{matrix}$$

$$\left.\begin{aligned} &\rightarrow \underline{1}\,\ddot{q}_1 + \underline{\underline{k_{d_1}}}\dot{q}_1 + \underline{\underline{k_{p_1}}} q_1 = 0 \\ &\rightarrow \ddot{q}_2 + \underline{\underline{k_{d_2}}}\dot{q}_2 + \underline{\underline{k_{p_2}}} q_2 = 0 \end{aligned}\right\} \text{ Simple 1-D equation}$$

$$\rightarrow \boxed{k_d = -c + 2\sqrt{m(k + k_p)}} \qquad m\ddot{q} + \underline{c}\dot{q} + \underline{k} q = 0$$

$$k_{d_i} = 0 + 2\sqrt{(1)(0 + k_{p_i})} \Rightarrow \boxed{k_{d_i} = 2\sqrt{k_{p_i}}}$$

Block diagram



$$\tau = \hat{M}(-k_p q - k_d \dot{q}) + \hat{G}(q) + \hat{C}\dot{q}$$

(a) $(-k_p q - k_d \dot{q})$

(b) $\hat{M}(-k_p q - k_d \dot{q})$

$$\tau = \hat{C}\dot{q} + \hat{G}(q) + \hat{M}(-K_p q - K_d \dot{q})$$

set point $= 0$

Set point control:

System dynamics: $M\ddot{q} + C\dot{q} + G(q) = \tau$

Control: $\tau = \hat{M}\left(-K_p(q - q_d) - K_d\dot{q}\right) + \hat{C}\dot{q} + \hat{G}(q)$

set point

Draw the block diagram yourself.

Example:

$$\underbrace{\begin{bmatrix} 1 & 0.1 \\ 0.1 & 2 \end{bmatrix}}_{M}\begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} + \underbrace{\begin{bmatrix} 0.2 & 0 \\ 0 & 0.1 \end{bmatrix}}_{C}\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} + \underbrace{\begin{bmatrix} 5 & 1 \\ 1 & 10 \end{bmatrix}}_{G}\begin{bmatrix} q_1 \\ q_2 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix}$$

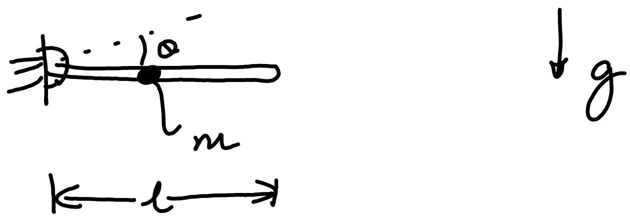
$q_d = \begin{bmatrix} 0.5 \\ 0.1 \end{bmatrix}$ $\qquad q_i = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

initial state

$\tau = \hat{M}\left(-K_p(q - q_d) - K_d\dot{q}\right) + \hat{G}(q) + \hat{C}\dot{q}$

matlab

Example: Single link pendulum



Goal: Get the pendulum to go from 0 rad to $\pi/2$ rad. in shortest amount of time

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau$$ - Dynamics

Control:

① $\tau = -k_p(q - q_d) - k_d\dot{q}$ Simple proportional derivative control.

② $\tau = M(-k_p(q - q_d) - k_d\dot{q}) + C(q,\dot{q})\dot{q} + G(q)$

control partitioning.

Lets see what happens in MATLAB