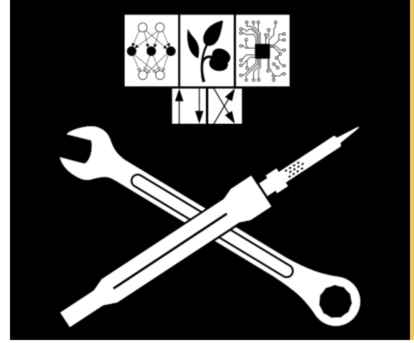
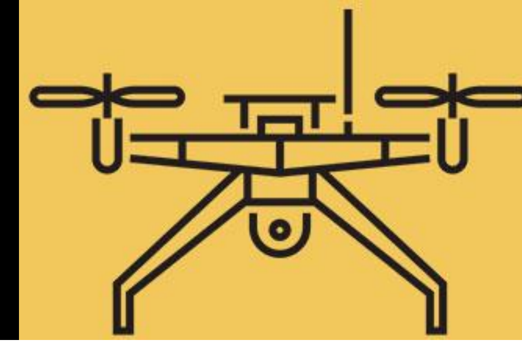




MPC and feedback linearization

Lecture 6

Jerome Jouffroy, Professor

jerome@sdu.dk

SDU

Today's lecture

- Main idea behind MPC (Model-Predictive Control)
- Quadratic Programming in a nutshell
- From QP to MPC
- Short introduction to feedback linearization

Model-Predictive Control: the main idea (1/2)

Consider discrete-time system described by

$$\begin{cases} \mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k), \\ \mathbf{y}(k) = \mathbf{C}\mathbf{x}(k) \end{cases} \qquad \underbrace{\mathbf{x}(0) = \mathbf{x}_0}_{\text{known}}$$

we want to find the control signal

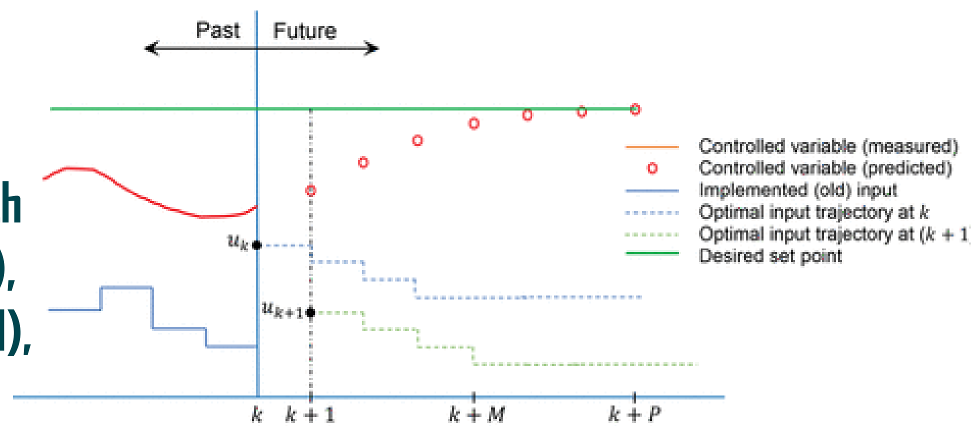
$$\mathbf{u}(0), \mathbf{u}(1), \cdots, \mathbf{u}(N-1)$$

which minimizes the cost function

$$\boxed{J = \sum_{k=0}^{N-1} \left[\mathbf{x}^T(k)\mathbf{Q}\mathbf{x}(k) + \mathbf{u}^T(k)\mathbf{R}\mathbf{u}(k))\right]}$$

(very similar to LQR!)

- open-loop control (start from known x_0, and apply u(0), u(1),...,u(N-1))
- Transform into feedback: starting with x(0), minimizing J and apply only u(0), measure resulting x(1), start from x(1), minimize J, and apply only u(1)...



SDU

Model-Predictive Control: the main idea (2/2)

Hence we have to minimize the criterion

$$J(k) = \sum_{i=0}^{N-1} \left[\mathbf{x}^T(k+i)\mathbf{Q}\mathbf{x}(k+i) + \mathbf{u}^T(k+i)\mathbf{R}\mathbf{u}(k+i))\right]$$

at each time instant k

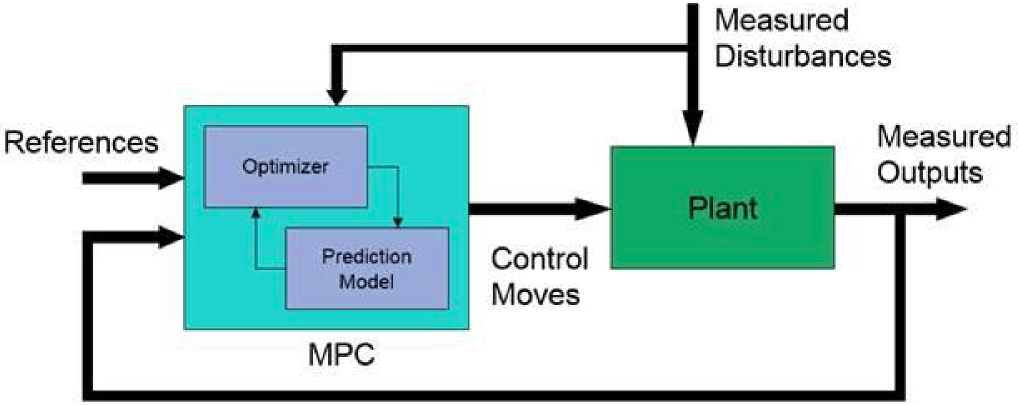
so that we get the control signal

$$\mathbf{u}(k+0), \mathbf{u}(k+1), \cdots, \mathbf{u}(k+N-1)$$

from which only u(k+0)=u(k) is applied to the plant

Remarks:

- Q, R matrices and N are all tuning parameters
- the above MPC framework allows to include saturation limits $\underline{\mathbf{u}} \leq \mathbf{u}(k) \leq \bar{\mathbf{u}},$



SDU

Quadratic Programming in a nutshell (1/3)

How is J(k) minimized in real-time? (for each iteration k)

(on a computer) → Quadratic Programming

the Quadratic problem:

Find vector z to $\boxed{\text{minimize} \quad \frac{1}{2}\mathbf{z}^T\mathbf{H}\mathbf{z} + \mathbf{F}^T\mathbf{z},}$ quadratic cost function

with $\dim(\mathbf{z}) = \dim(\mathbf{F})$ and the quadratic term $\mathbf{z}^T\mathbf{H}\mathbf{z}$

Vector z is also possibly subject to the constraints

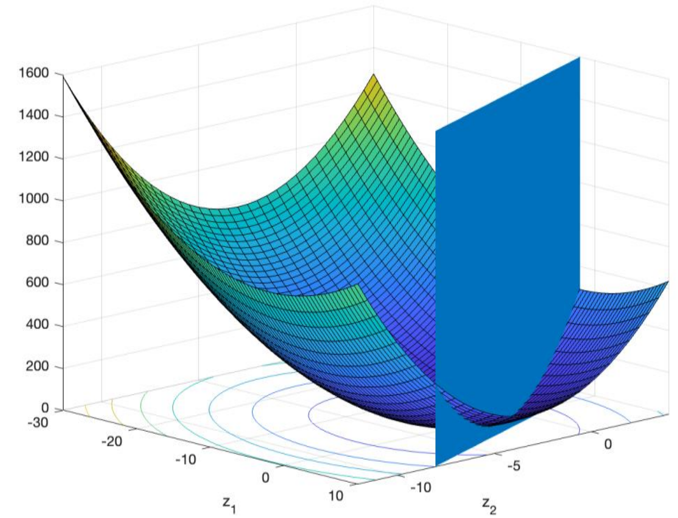
$\boxed{\mathbf{G}\mathbf{z} = \mathbf{q}}$ equality constraints

$\boxed{\mathbf{W}\mathbf{z} \leq \mathbf{v}}$ inequality constraints

or $\underline{\mathbf{z}} \leq \mathbf{z} \leq \bar{\mathbf{z}}$

MATLAB: `quadprog`

Quadratic Programming in a nutshell (2/3)



Example: a 2D case

consider the 2D parabola

$$P(\mathbf{z}) = (z_1 + 2)^2 + 10(z_2 + 3)^2$$

P(z) can be rewritten as

$$\begin{aligned} P(\mathbf{z}) &= z_1^2 + 10z_2^2 + 4z_1 + 60z_2 + 94 \\ &= \frac{1}{2}\mathbf{z}^T \begin{bmatrix} 2 & 0 \\ 0 & 20 \end{bmatrix} \mathbf{z} + \begin{bmatrix} 4 & 60 \end{bmatrix} \mathbf{z} + \underline{94}. \end{aligned}$$

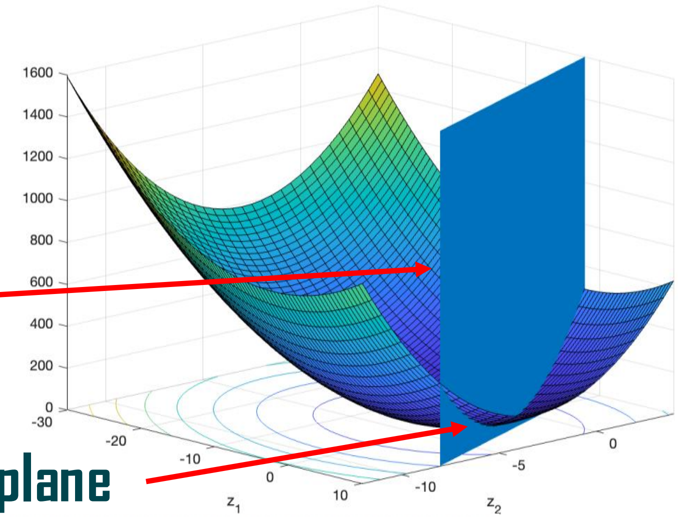
assume now that we want to find the value of z that minimizes P(z) → this z does not depend on the height of the parabola, ie not on **94**!

Hence finding z that minimizes P(z) amounts to finding z that

$$\text{minimize} \quad \frac{1}{2}\mathbf{z}^T\mathbf{H}\mathbf{z} + \mathbf{F}^T\mathbf{z},$$ with $\mathbf{H} = \begin{bmatrix} 2 & 0 \\ 0 & 20 \end{bmatrix}$ and $\mathbf{F} = \begin{bmatrix} 4 \\ 60 \end{bmatrix}$

SDU

Quadratic Programming in a nutshell (3/3)



- adjoin to P(z) the <u>equality constraint</u>

$$z_1 + z_2 = 2$$

Quadratic problem: find z minimizing P(z) on the intersection between P(z) and vertical plane

rewriting the equality constraint as

$$\begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 \end{bmatrix} \mathbf{z} = 2,$$

gives $\mathbf{G}\mathbf{z} = \mathbf{q}$ with $\mathbf{G} = \begin{bmatrix} 1 & 1 \end{bmatrix}$ and $q = 2.$

- adjoin to P(z) the <u>inequality constraint</u>

$$z_1 + z_2 \leq 2.$$

Quadratic problem: find z minimizing P(z) on the part of P(z) <u>behind</u> the vertical plane

so that we have $\mathbf{W}\mathbf{z} \leq \mathbf{v}$ with $\mathbf{W} = \begin{bmatrix} 1 & 1 \end{bmatrix}$ and $v = 2.$

From QP to MPC (1/3)

How to go from $J = \sum_{k=0}^{N-1} \left[\mathbf{x}^T(k)\mathbf{Q}\mathbf{x}(k) + \mathbf{u}^T(k)\mathbf{R}\mathbf{u}(k) \right]$ to $\frac{1}{2}\mathbf{z}^T\mathbf{H}\mathbf{z} + \mathbf{F}^T\mathbf{z}$ so that we can use quadprog ?

Let $N = 3$ and write

$$
\begin{aligned}
J &= \sum_{i=0}^{3-1} \left[\mathbf{x}^T(i)\mathbf{Q}\mathbf{x}(i) + \mathbf{u}^T(i)\mathbf{R}\mathbf{u}(i) \right] \\
&= \mathbf{x}^T(0)\mathbf{Q}\mathbf{x}(0) + \mathbf{x}^T(1)\mathbf{Q}\mathbf{x}(1) + \mathbf{x}^T(2)\mathbf{Q}\mathbf{x}(2) \\
&\qquad + \mathbf{u}^T(0)\mathbf{R}\mathbf{u}(0) + \mathbf{u}^T(1)\mathbf{R}\mathbf{u}(1) + \mathbf{u}^T(2)\mathbf{R}\mathbf{u}(2) \\
&= \begin{bmatrix} \mathbf{x}(0) \\ \mathbf{x}(1) \\ \mathbf{x}(2) \end{bmatrix}^T \begin{bmatrix} \mathbf{Q} & & \\ & \mathbf{Q} & \\ & & \mathbf{Q} \end{bmatrix} \begin{bmatrix} \mathbf{x}(0) \\ \mathbf{x}(1) \\ \mathbf{x}(2) \end{bmatrix} + \begin{bmatrix} \mathbf{u}(0) \\ \mathbf{u}(1) \\ \mathbf{u}(2) \end{bmatrix}^T \begin{bmatrix} \mathbf{R} & & \\ & \mathbf{R} & \\ & & \mathbf{R} \end{bmatrix} \begin{bmatrix} \mathbf{u}(0) \\ \mathbf{u}(1) \\ \mathbf{u}(2) \end{bmatrix}
\end{aligned}
$$

so that we have $\boxed{J = \mathbf{X}^T\mathbb{Q}\mathbf{X} + \mathbf{U}^T\mathbb{R}\mathbf{U}}$

with $\mathbb{Q} := \begin{bmatrix} \mathbf{Q} & & \\ & \mathbf{Q} & \\ & & \mathbf{Q} \end{bmatrix}$ and $\mathbb{R} := \begin{bmatrix} \mathbf{R} & & \\ & \mathbf{R} & \\ & & \mathbf{R} \end{bmatrix}$

and

$$
\left|\begin{aligned}
\mathbf{X} &:= \left[\mathbf{x}^T(0), \mathbf{x}^T(1), \cdots, \mathbf{x}^T(N-1) \right]^T \\
\mathbf{U} &:= \left[\mathbf{u}^T(0), \mathbf{u}^T(1), \cdots, \mathbf{u}^T(N-1) \right]^T
\end{aligned}\right.
$$

SDU

From QP to MPC (2/3)

Problem of expression $J = \mathbf{X}^T \mathbb{Q} \mathbf{X} + \mathbf{U}^T \mathbb{R} \mathbf{U}$: we know x(0) but not x(1) nor x(2)

$$
\begin{aligned}
\mathbf{x}(0) &= \mathbf{x}(0) \\
\mathbf{x}(1) &= \mathbf{A}\mathbf{x}(0) + \mathbf{B}\mathbf{u}(0) \\
\mathbf{x}(2) &= \mathbf{A}\mathbf{x}(1) + \mathbf{B}\mathbf{u}(1) \\
&= \mathbf{A}\left[\mathbf{A}\mathbf{x}(0) + \mathbf{B}\mathbf{u}(0)\right] + \mathbf{B}\mathbf{u}(1) \\
&= \mathbf{A}^2\mathbf{x}(0) + \mathbf{A}\mathbf{B}\mathbf{u}(0) + \mathbf{B}\mathbf{u}(1)
\end{aligned}
$$

so that we have

$$
\begin{bmatrix} \mathbf{x}(0) \\ \mathbf{x}(1) \\ \mathbf{x}(2) \end{bmatrix} = \begin{bmatrix} I_n \\ \mathbf{A} \\ \mathbf{A}^2 \end{bmatrix} \mathbf{x}(0) + \begin{bmatrix} 0 & 0 & 0 \\ \mathbf{B} & 0 & 0 \\ \mathbf{A}\mathbf{B} & \mathbf{B} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u}(0) \\ \mathbf{u}(1) \\ \mathbf{u}(2) \end{bmatrix}
$$

or $\boxed{\mathbf{X} = \mathbb{A}\mathbf{x}(0) + \mathbb{B}\mathbf{U}}$

with $\mathbb{A} := \begin{bmatrix} I_n \\ \mathbf{A} \\ \mathbf{A}^2 \end{bmatrix}$ and $\mathbb{B} := \begin{bmatrix} 0 & 0 & 0 \\ \mathbf{B} & 0 & 0 \\ \mathbf{A}\mathbf{B} & \mathbf{B} & 0 \end{bmatrix}$

From QP to MPC (3/3)

Putting $\mathbf{X} = \mathbb{A}\mathbf{x}(0) + \mathbb{B}\mathbf{U}$ into $J = \mathbf{X}^T \mathbb{Q} \mathbf{X} + \mathbf{U}^T \mathbb{R} \mathbf{U}$ gives

$$\begin{aligned} J &= [\mathbb{A}\mathbf{x}(0) + \mathbb{B}\mathbf{U}]^T \mathbb{Q} [\mathbb{A}\mathbf{x}(0) + \mathbb{B}\mathbf{U}] + \mathbf{U}^T \mathbb{R} \mathbf{U} \\ &= \left[\mathbf{x}^T(0)\mathbb{A}^T + \mathbf{U}^T\mathbb{B}^T\right] \mathbb{Q} [\mathbb{A}\mathbf{x}(0) + \mathbb{B}\mathbf{U}] + \mathbf{U}^T \mathbb{R} \mathbf{U} \\ &= \mathbf{x}^T(0)\mathbb{A}^T\mathbb{Q}\mathbb{A}\mathbf{x}(0) + \mathbf{x}^T(0)\mathbb{A}^T\mathbb{Q}\mathbb{B}\mathbf{U} + \mathbf{U}^T\mathbb{B}^T\mathbb{Q}\mathbb{A}\mathbf{x}(0) \\ &\quad + \mathbf{U}^T\mathbb{B}^T\mathbb{Q}\mathbb{B}\mathbf{U} + \mathbf{U}^T\mathbb{R}\mathbf{U} \end{aligned}$$

$$J = \mathbf{x}^T(0)\mathbb{A}^T\mathbb{Q}\mathbb{A}\mathbf{x}(0) + 2\mathbf{x}^T(0)\mathbb{A}^T\mathbb{Q}\mathbb{B}\mathbf{U} + \mathbf{U}^T \left[\mathbb{B}^T\mathbb{Q}\mathbb{B} + \mathbb{R}\right] \mathbf{U}$$

<u>BUT</u>: term $\mathbf{x}^T(0)\mathbb{A}^T\mathbb{Q}\mathbb{A}\mathbf{x}(0)$ is known

Finding U minimizing J is the same as finding U minimizing

$$\boxed{J' = \mathbf{U}^T \left[\mathbb{B}^T\mathbb{Q}\mathbb{B} + \mathbb{R}\right] \mathbf{U} + 2\mathbf{x}^T(0)\mathbb{A}^T\mathbb{Q}\mathbb{B}\mathbf{U}}$$

with
$$\mathbf{F} = 2\left[\mathbf{x}^T(0)\mathbb{A}^T\mathbb{Q}\mathbb{B}\right]^T$$
$$\mathbf{H} = 2\left[\mathbb{B}^T\mathbb{Q}\mathbb{B} + \mathbb{R}\right]$$

and we can now use `quadprog`

Linear state-feedback and feedback linearization (1/2)

Now for something completely different:

Consider the SS rep. of a 2nd order system
$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -a_0 x_1 - a_1 x_2 + u \end{cases}$$

which is controlled by state-feedback $u = -\mathbf{K}\mathbf{x} = -k_1 x_1 - k_2 x_2$

so that the closed-loop dynamics are
$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -p_0 x_1 - p_1 x_2 \end{cases}$$

with $p_0 = a_0 + k_1$ and $p_1 = a_1 + k_2$

so that our state-feedback controller can be rewritten as
$$u = -(-a_0 + p_0)x_1 - (-a_1 + p_1)x_2$$

or
$$\boxed{u = a_0 x_1 + a_1 x_2 - p_0 x_1 - p_1 x_2}$$

SDU

Linear state-feedback and feedback linearization (2/2)

Let us rewrite

$$u = a_0 x_1 + a_1 x_2 - p_0 x_1 - p_1 x_2 \quad \text{and} \quad \begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -a_0 x_1 - a_1 x_2 + u \end{cases}$$

we have the closed-loop dynamics

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -a_0 x_1 - a_1 x_2 + \Big(\underbrace{\{a_0 x_1 + a_1 x_2\}}_{\text{cancelling term}} + \underbrace{[-p_0 x_1 - p_1 x_2]}_{\text{stabilizing term}} \Big) \end{cases}$$

the above state-feedback controller can be split into 2 parts

a cancelling controller $\boxed{u = a_0 x_1 + a_1 x_2 + v}$ virtual input

giving the intermediary dynamics $\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = v \end{cases}$ (double integrator)

which can in turn be stabilized by a stabilizing controller $\boxed{v = -p_0 x_1 - p_1 x_2}$

The more general linear case

Generalizing the previous to bigger linear systems is not complicated:

Take $\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = x_3 \\ \vdots \\ \dot{x}_n = -a_0 x_1 - a_1 x_2 - \ldots - a_{n-1} x_n + bu \end{cases}$ with (for simplicity) $b = 1$

define then the cancelling controller $\boxed{u = a_0 x_1 + a_1 x_2 + \ldots + a_{n-1} x_n + v}$

which gives the intermediary dynamics $\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = x_3 \\ \vdots \\ \dot{x}_n = v \end{cases}$

which can be stabilized with $\boxed{v = -p_0 x_1 - p_1 x_2 - \ldots - p_{n-1} x_n}$

The controlled pendulum: a nonlinear example

The previous principle can also be applied to render nonlinear systems linear by feedback → Feedback Linearization

start with
$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -\frac{g}{l}\sin(x_1) + \frac{1}{ml^2}u \end{cases}$$
(SS rep. of pendulum)

for which we would like to find a cancelling/feedback linearizing controller leading to
$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = v \end{cases}$$
→ ie we want to have
$$v = -\frac{g}{l}\sin(x_1) + \frac{1}{ml^2}u$$

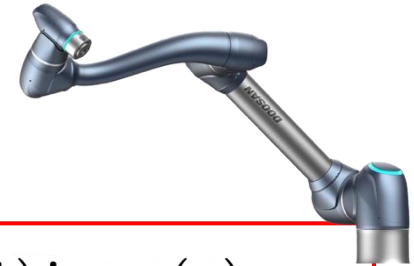
isolating u, we get the cancelling controller
$$\boxed{u = mgl\sin(x_1) + ml^2 v}$$

which is then completed by the stabilizing controller
$$\boxed{v = -p_0 x_1 - p_1 x_2}$$

→ Feedback Linearization = cancelling controller + stabilizing controller

Feedback linearization controller:
$$\boxed{u = mgl\sin(x_1) + ml^2(-p_0 x_1 - p_1 x_2)}$$

The robotic manipulator example



Typical model for a robotic manipulator $\boxed{\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau}}$

$$\ddot{\mathbf{q}} = -\mathbf{M}^{-1}(\mathbf{q})\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} - \mathbf{M}^{-1}(\mathbf{q})\mathbf{g}(\mathbf{q}) + \mathbf{M}^{-1}(\mathbf{q})\boldsymbol{\tau}$$

SS rep. $\begin{cases} \dot{\mathbf{q}} = \mathbf{v} \\ \dot{\mathbf{v}} = -\mathbf{M}^{-1}(\mathbf{q})\mathbf{C}(\mathbf{q}, \mathbf{v})\mathbf{v} - \mathbf{M}^{-1}(\mathbf{q})\mathbf{g}(\mathbf{q}) + \mathbf{M}^{-1}(\mathbf{q})\boldsymbol{\tau} \end{cases}$

with $\mathbf{v} := \dot{\mathbf{q}}$

We would like to have

$\mathbf{a}_v = -\mathbf{M}^{-1}(\mathbf{q})\mathbf{C}(\mathbf{q}, \mathbf{v})\mathbf{v} - \mathbf{M}^{-1}(\mathbf{q})\mathbf{g}(\mathbf{q}) + \mathbf{M}^{-1}(\mathbf{q})\boldsymbol{\tau}$ in order to get $\begin{cases} \dot{\mathbf{q}} = \mathbf{v} \\ \dot{\mathbf{v}} = \mathbf{a}_v \end{cases}$

which gives the cancelling controller

$$\boxed{\boldsymbol{\tau} = \mathbf{M}(\mathbf{q})\mathbf{a}_v + \mathbf{C}(\mathbf{q}, \mathbf{v})\mathbf{v} + \mathbf{g}(\mathbf{q})}$$

and the stabilizing controller

$$\boxed{\mathbf{a}_v = -\mathbf{K}_\mathbf{v}\mathbf{v} - \mathbf{K}_\mathbf{q}\mathbf{q}}$$

combine these two to get the feedback linearizing controller

$$\boxed{\boldsymbol{\tau} = \mathbf{M}(\mathbf{q})\left\{-\mathbf{K}_\mathbf{v}\mathbf{v} - \mathbf{K}_\mathbf{q}\mathbf{q}\right\} + \mathbf{C}(\mathbf{q}, \mathbf{v})\mathbf{v} + \mathbf{g}(\mathbf{q})}$$

Towards more complex cases (1/3)

So far, we had models such as

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -\frac{g}{l}\sin(x_1) + \frac{1}{ml^2}u \end{cases}$$

or

$$\begin{cases} \dot{\mathbf{q}} = \mathbf{v} \\ \dot{\mathbf{v}} = -\mathbf{M}^{-1}(\mathbf{q})\mathbf{C}(\mathbf{q},\mathbf{v})\mathbf{v} - \mathbf{M}^{-1}(\mathbf{q})\mathbf{g}(\mathbf{q}) + \mathbf{M}^{-1}(\mathbf{q})\boldsymbol{\tau} \end{cases}$$

But what if we have this system?

$$\begin{cases} \dot{x}_1 = x_1^2 + x_2 \\ \dot{x}_2 = x_1 + u \end{cases}$$

Nonlinearities NOT in the same equation as input...

Defining controller

$$u = -x_1 + v$$

gives the closed-loop dynamics

$$\begin{cases} \dot{x}_1 = x_1^2 + x_2 \\ \dot{x}_2 = v \end{cases}$$

→ still nonlinear!

...what can we do?

Towards more complex cases (2/3)

Nice trick: use output $y=x_1$ and obtain an ODE in y...

$$\begin{cases} \dot{x}_1 = x_1^2 + x_2 \\ \dot{x}_2 = x_1 + u \end{cases}$$

$$y = x_1$$

$$\begin{aligned} \dot{y} &= \dot{x}_1 \\ &= x_1^2 + x_2 \\ &= y^2 + x_2 \end{aligned}$$

Then, isolate x_2 to get $x_2 = \dot{y} - y^2$

$$\dot{x}_2 = \ddot{y} - 2y\dot{y}$$

so that we have $\ddot{y} - 2y\dot{y} = y + u$

or $\boxed{\ddot{y} = 2y\dot{y} + y + u}$

differential equation in y of order 2

Towards more complex cases (3/3)

find a state-space representation of $\ddot{y} = 2y\dot{y} + y + u$

define a new state vector $\mathbf{z} := \begin{bmatrix} y \\ \dot{y} \end{bmatrix} = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$

giving the SS.rep $\boxed{\begin{cases} \dot{z}_1 = z_2 \\ \dot{z}_2 = 2z_1 z_2 + z_1 + u \end{cases}}$

nonlinearities in the same equation as the input!

define the feedback-linearizing controller

$$u = -2z_1 z_2 - z_1 + v$$

giving the linear dynamics $\begin{cases} \dot{z}_1 = z_2 \\ \dot{z}_2 = v \end{cases}$

Remarks:
- this example can be extended to more complicated systems
- for systems with 1 input, one can find the right y systematically
- no systematic way of finding y when several inputs

SDU