MO4RBR04: Automatique Linéaire II

Cours 4: Méthode LQR et autres compléments

Pascal Morin
ISIR, Sorbonne Université
pascal.morin@sorbonne-universite.fr

Systèmes linéaires

$$(*) \quad \begin{cases} \dot{x} = Ax + Bu \\ y = Cx + Du \end{cases}$$

Ce cours a deux objectifs:

- Présenter une autre méthode de synthèse de contrôleur, appelée « Méthode LQR »
- Présenter quelques outils Matlab pouvant aider à la synthèse et mise en œuvre des contrôleurs
- Présenter d’autres compléments sur la synthèse de contrôleurs

Méthode LQR

Motivations:

- La technique de placement de pôles est une méthode simple, qui permet facilement d'imposer des taux de convergence du système en boucle fermée
- Mais dans certains cas elle peut présenter des difficultés de mise en œuvre:
  - Systèmes de grandes dimension: complexité des calculs
  - Cas où il existe une infinité de gains possibles pour un jeux de valeurs propres donné (i.e., système d'équations sous-déterminé, un choix est-il meilleur qu'un autre?)
  - Si l'on souhaite imposer des convergences plus rapides sur certaines variables de l'état que sur d'autres, comment faire?
- La méthode LQR est une approche alternative…

Méthode LQR

Principe: Rappel sur les matrices.

Définitions: Une matrice symétrique $\mathrm{M}, p \times p$ est dite:

- symétrique positive (s.p.) si, quelque soit $\xi \in \mathbb{R}^p, \xi^T M \xi \geq 0$;
- symétrique définie positive (s.d.p.) si, quelque soit $0 \neq \xi \in \mathbb{R}^p, \xi^T M \xi > 0$.

Proposition:

- Une matrice symétrique M est s.p. ssi toutes ses valeurs propres sont non-négatives (i.e., ≥ 0);
- Une matrice symétrique M est s.d.p. ssi toutes ses valeurs propres sont strictement positives.

Méthode LQR

Principe:

On considère une « fonction de coût »:

$$J(x_0, u) = \int_0^{+\infty} x(s)^T Q x(s) + u(s)^T R u(s)\, ds$$

avec:

- Q: n × n une matrice s.p.
- R: m × m une matrice s.d.p.

Le principe de la méthode LQR est le suivant:

« Trouver la commande par retour d'état $u = K^* x$ qui va rendre le critère minimal, quelque soit $x(0) = x_0$, parmi toutes les commandes $u = Kx$ qui stabilisent le système »

Méthode LQR

Principe:

Autrement dit, au sens de la minimisation du critère $J(x_0, u)$, la matrice de gains K^*est la meilleure matrice de gains possible.

Intérêts:

- La méthode introduit une notion d'optimalité (absente dans l'approche « placement de pôles)
- Les choix de matrices Q et R vont permettre d'agir sur l'intensité des différentes composantes du vecteur de commande et la vitesse de convergence des différentes composantes du vecteur d'état
- Il existe des outils numériques permettant de calculer la solution du problème de minimisation

Méthode LQR

Choix des matrices Q et R:

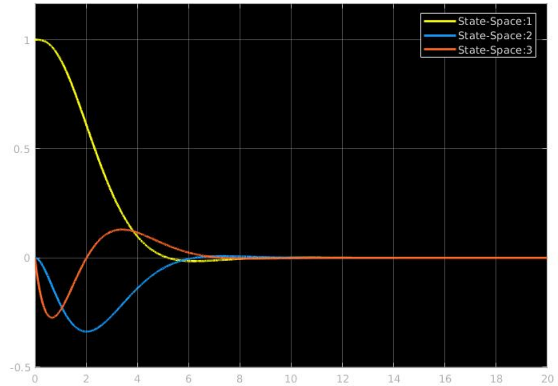
- Ces matrices seront choisies, la plupart du temps, diagonales
- Supposons donc $Q = Diag(q_1, q_2, \dots, q_n)$ et $\mathrm{R} = Diag(r_1, r_2, \dots, r_m)$ avec
  - $q_i \geq 0 \; \forall i$ (matrice Q s.p.)
  - $r_j > 0 \; \forall j$ (matrice R s.d.p.)
- Principe du choix:
  - Un coefficient r_j « grand » aura tendance à conduire à des valeurs de u_j petites (utile si on a des contraintes d'actionnement sur l'actionneur associé), et vice-versa;
  - Un coefficient q_i « grand » aura tendance à conduire à une variable x_i petite, et notamment un amortissement important sur cette variable;
  - Il s'agit de l'idée générale, sachant qu'il y a des compromis inévitables (e.g.: on ne peut généralement pas faire converger tout le vecteur d'état rapidement avec des commandes petites).

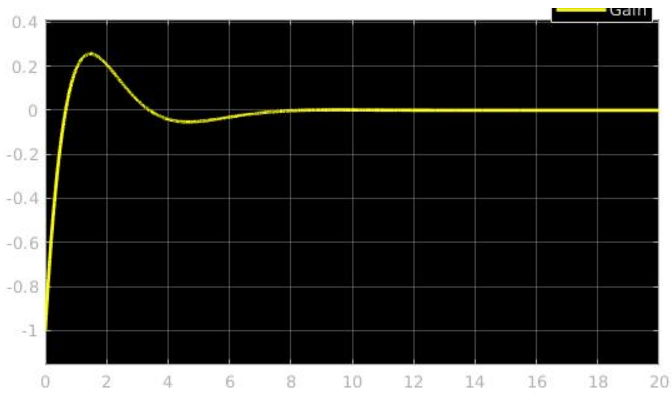
Méthode LQR

Exemple d'application N°1:

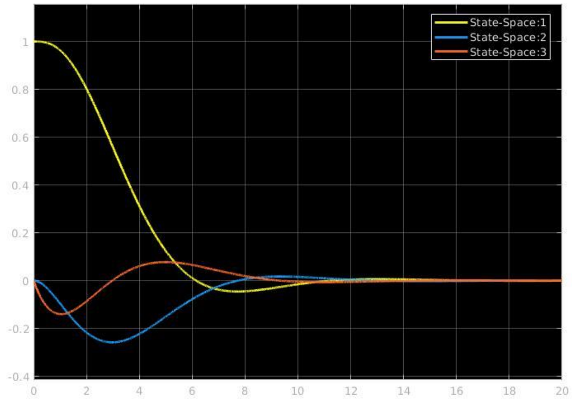
$$\dddot{x}_1 = u$$

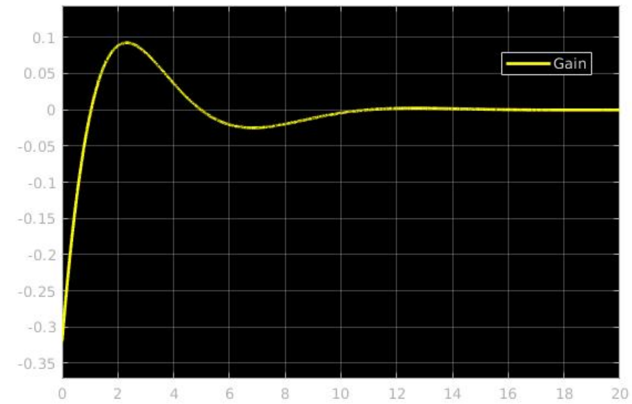
– Choix 1: $Q = I_3, R = 1$





– Choix 2: $Q = I_3, R = 10$





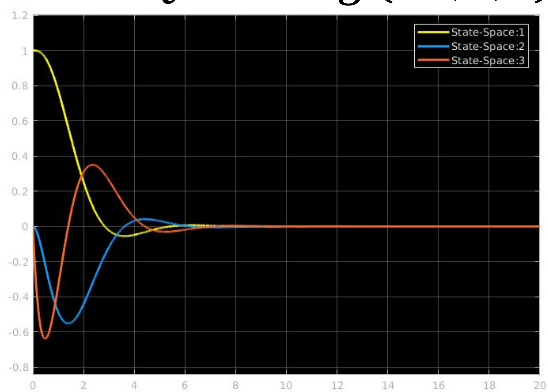
SORBONNE UNIVERSITÉ
CRÉATEURS DE FUTURS DEPUIS 1257

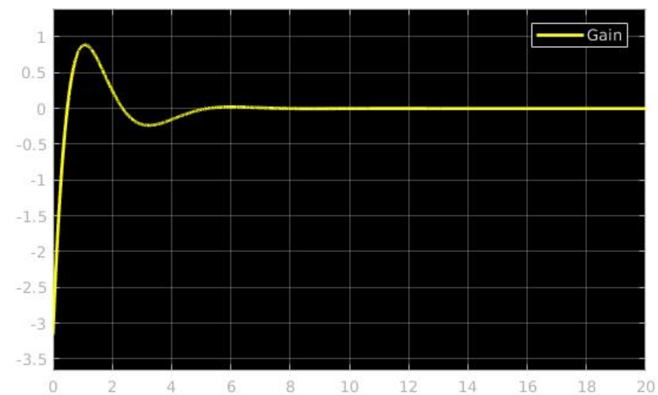
Méthode LQR

Exemple d'application N°1:

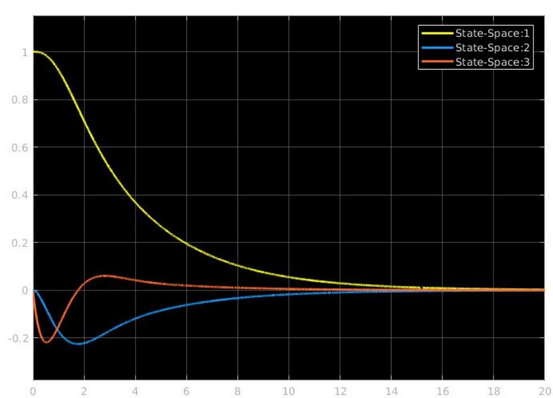
$$\dddot{x}_1 = u$$

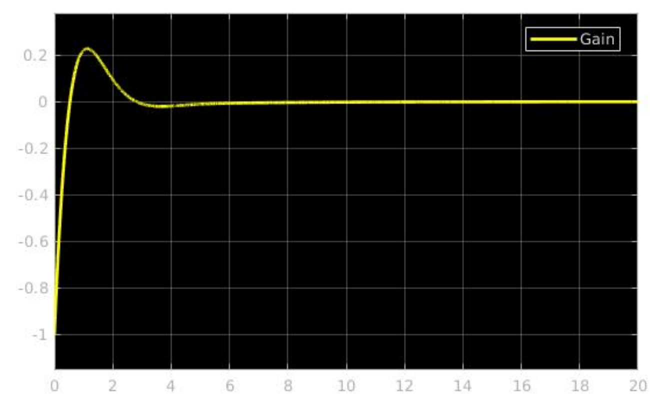
– Choix 1: $Q = Diag(10,1,1), R = 1$





– Choix 2: $Q = Diag(1,10,1), R = 1$



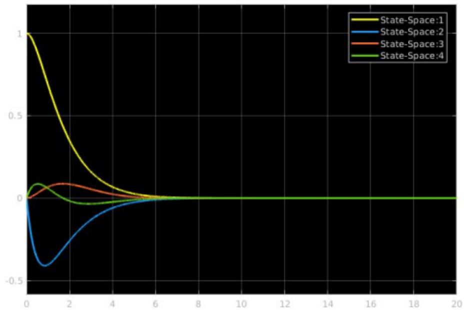


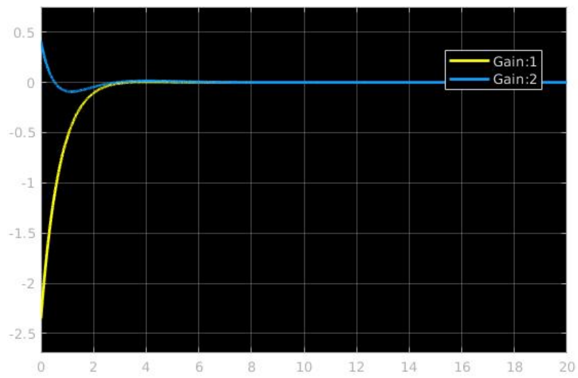
Méthode LQR

Exemple d'application N°2:

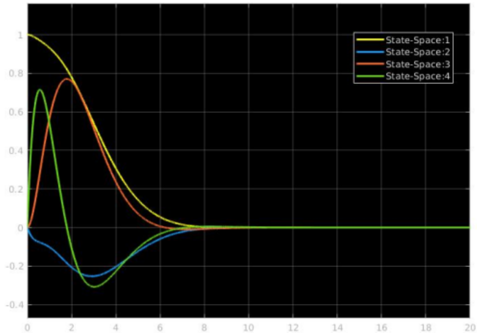
$$A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}, B = \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix}$$

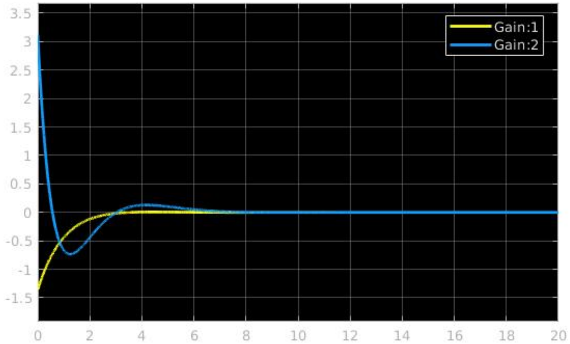
– Choix 1: $Q = I_4, R = I_2$





– Choix 2: $Q = I_4, R = Diag(10,1)$





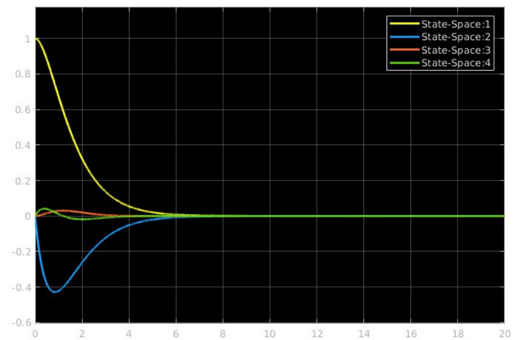
SORBONNE UNIVERSITÉ
CRÉATEURS DE FUTURS DEPUIS 1257

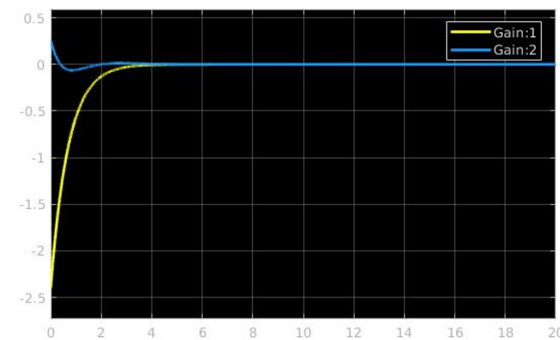
Méthode LQR

Exemple d'application N°2:

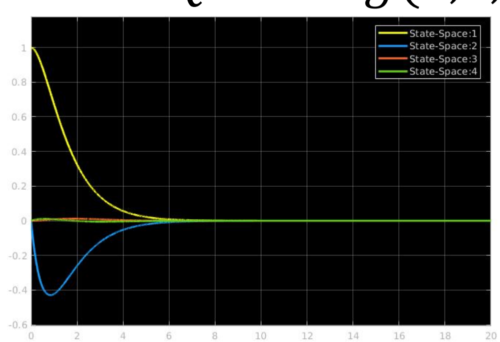
$$A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}, B = \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix}$$

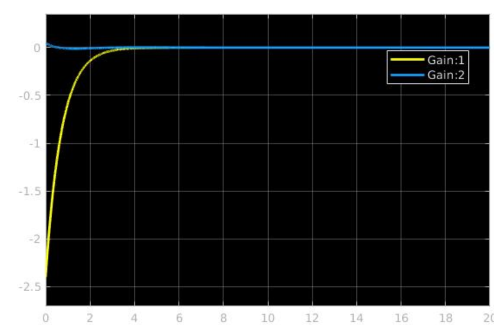
– Choix 1: $Q = Diag(1,1,10,1), R = I_2$





– Choix 2: $Q = Diag(1,1,10,1), R = Diag(1,10)$





Méthode LQR

Un peu de théorie:

Est-on certain que la matrice de gain optimale K^* existe?

Proposition: On considère l' «Equation de Ricatti » en l'inconnue P:

$$PA + A^T P + Q - PBR^{-1}B^T P = 0$$

Alors,

1. S'il existe une solution symétrique P^*de cette équation telle que la matrice de gain $K^* = -R^{-1}B^T P^*$ rende $A + BK^*$ Hurwitz-stable, alors K^*est la solution recherchée au problème de minimisation.
2. Si la paire (A, B) est commandable et la paire (A, E)est observable, avec E la matrice $n \times n$ définie par la relation $Q = E^T E$, alors il existe une unique solution symétrique P^* satisfaisant les propriétés du point 1. ci-dessus. De plus cette matrice est s.d.p.

Méthode LQR

Equation de Ricatti:

$$PA + A^T P \; - PBR^{-1}B^T P + Q = 0$$

- Déterminer la matrice de gain optimale nécessite de résoudre l'équation de Ricatti
- Cette équation matricielle devient vite très complexe à résoudre (de plus elle est non-linéaire)
- Des outils numériques ont été développés pour cela (voir suite)
- Deux propriétés importantes à noter:
  - L'équation de Ricatti peut avoir plusieurs solutions, certaines associées à des matrices $A + BK$ non Hurwitz-stable. Attention donc!
  - Dans la recherche des solutions de l'équation de Ricatti, on peut toujours supposer que P est symétrique.

Outils Matlab

- Que ce soit pour la technique de placement de pôles ou la méthode LQR, des outils existent pour vous faciliter la tâche;
- Il convient cependant de les utiliser correctement.
- **LQR avec Matlab:**
  - Fonction de base: « lqr(A,B,Q,R) » fournit la matrice $-K^*$ Attention au signe $-$: Il faut ensuite appliquer le contrôleur $u = -Kx$ avec K la matrice donnée par lqr(A,B,Q,R).
  - Autre fonction: « lqry » calcule la matrice de gain optimale pour un critère où la sortie remplace l'état:

$$J(x_0, u) = \int_0^{+\infty} y(s)^T Q y(s) + u(s)^T R u(s)\, ds$$

  - Autre fonction: : « lqgreg » calcule les matrices de gain associées à:
    - ❑ Un contrôleur LQR
    - ❑ Un estimateur de type « Filtre de Kalman »

Outils Matlab

- **Placement de pôles avec Matlab:**
  - Fonction de base: « place(A,B,P) » fournit la matrice de gains permettant d'imposer un ensemble de pôles P. Attention au signe: Il faut ensuite appliquer le contrôleur $u = -Kx$ avec K la matrice donnée par place(A,B,P).
  - Remarque: La fonction « place » ne sait pas calculer la matrice de gains s'il y a des v.p. d'ordre de multiplicité supérieur au rang de B. Exemple: système mono-entrée et P=[-1;-1;-2]. Il suffira dans ce cas de prendre des valeurs légèrement différentes, e.g., P=[-1;-1.01;-2]

Compléments: les saturations

- Le retour d'état linéaire présente beaucoup d'avantages:
  - Simplicité
  - Permet une convergence exponentielle vers l'équilibre
  - Permet grâce à un choix judicieux des gains de gérer les compromis stabilité/précision
- Mais il présente aussi des inconvénients
- Notamment:
  - Quand $x(0)$ est très loin de l'équilibre, la commande peut devenir très grande
  - Au risque d'entrainer des saturations des actionneurs
  - Et de conduire à des vitesses/accélérations trop importantes

Compléments: les saturations

Approche pour limiter ces effets: saturer tout ou partie des termes de la commande.

Exemple: $m\,\ddot{p} = u$.

- Feedback linéaire:

$$u = -m\,(2\xi\omega\,\dot{p} - \omega^2 p)$$

- Feedback complètement saturé:

$$u = -m\,sat_\delta(2\xi\omega\,\dot{p} - \omega^2 p)$$

- Feedback avec saturation sur la position:

$$u = -m\left(2\xi\omega\,\dot{p} - sat_\delta(\omega^2 p)\right)$$

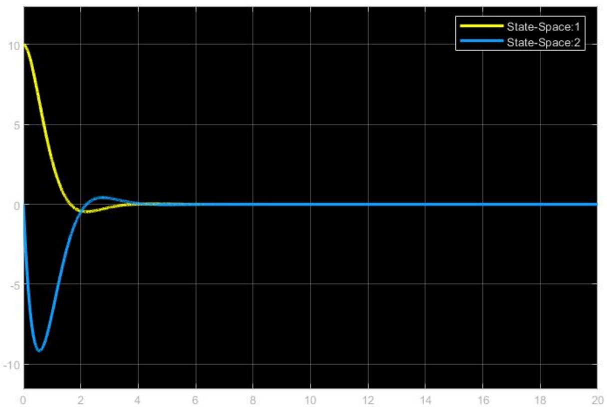
avec:

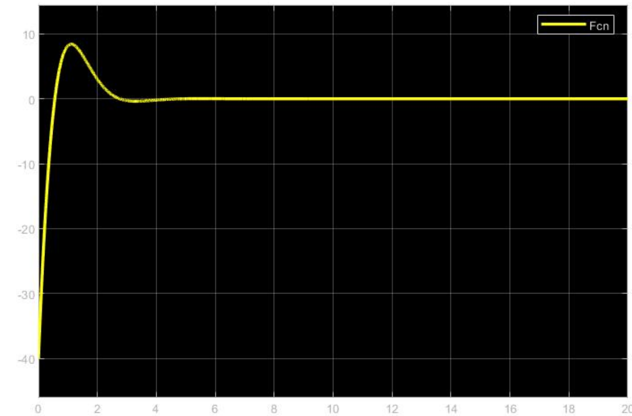
$$sat_\delta(z) = \begin{cases} -\delta\; si\; z \leq -\delta \\ z\; si\; -\delta \leq z \leq \delta \\ \delta\; si\; z \geq \delta \end{cases}$$

Compléments: les saturations

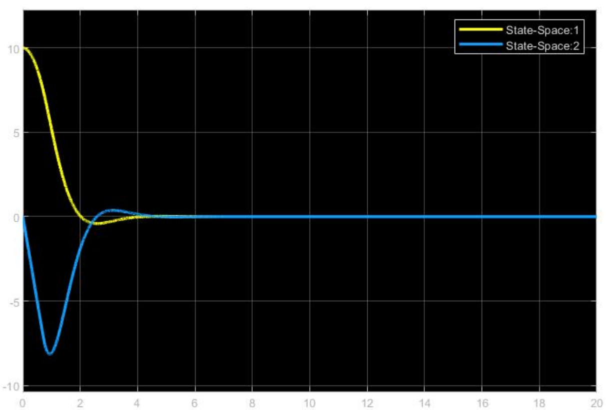
Exemple: $m\,\ddot{p} = u$. Illustration.

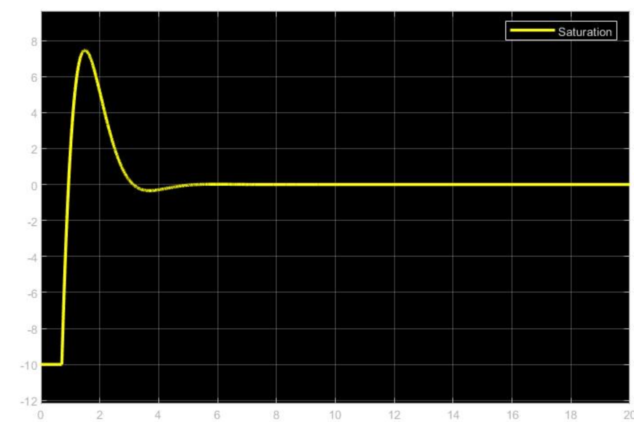
- Feedback linéaire:





- Feedback complètement saturé:

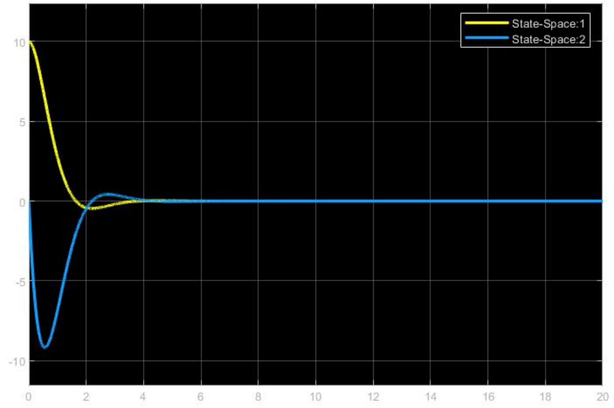


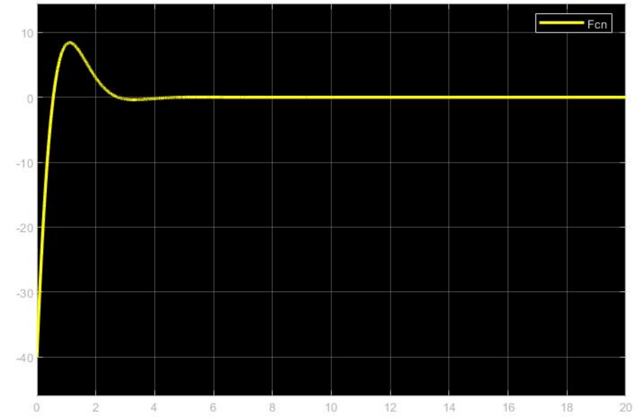


Compléments: les saturations

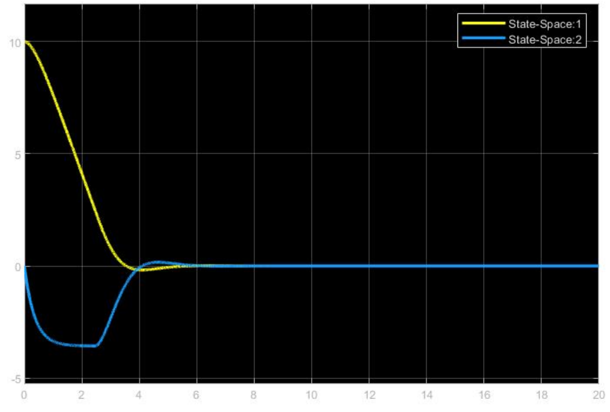
Exemple: $m\,\ddot{p} = u$. Illustration.

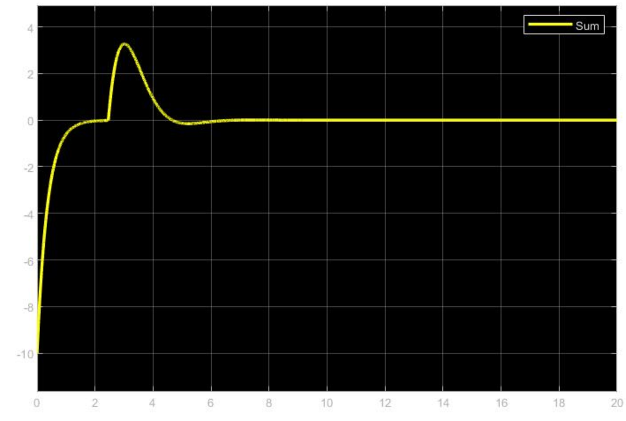
- Feedback linéaire:





- Feedback avec saturation sur la position:





Compléments: les saturations

Extensions:

- Cette approche peut s'étendre à des dimensions supérieures
- Mais il faut être vigilent:
  - La solution $u = sat_\delta(Kx)$ peut ne plus assurer la convergence pour de grandes erreurs initiales si $n \geq 3$;
  - Il existe des façons de saturer plus subtiles
- En grande dimension, une approche possible est de:
  - Décomposer le système en cascade de systèmes d'ordre plus faible (voir Cours 3 – Slide 26)
  - Appliquer ce qu'on a vu dans les trois slides précédents à chaqu'un des sous-systèmes.

Compléments: l'intégrateur

- Le terme de correction intégrale sert à assurer une erreur statique nulle (voir Autom-Linéaire I)
- Par exemple dans un contrôleur de type PID

$$C(p) = K(1 + T_d p + \frac{1}{T_i p})$$

- Exprimé en temporel, ce terme correspond à un terme intégral:

$$U(p) = C(p)\big(Y_d(p) - Y(p)\big)$$

$$\Leftrightarrow$$

$$u(t) \\ = K\big(y_d(t) - y(t)\big) + K\,T_d\big(\dot{y}_d(t) - \dot{y}(t)\big) \\ + \frac{K}{T_i}\int_0^t (y_d(s) - y(s))\,ds$$

Compléments: l'intégrateur

- Dangers liés au terme intégral:
  - Ce terme peut croître indéfiniment même si l'erreur $y_d - y$ est bornée
  - Par exemple, dans le cas où une perturbation importante sur le système vient empêcher la convergence de l'erreur vers zéro
  - Si la perturbation vient à disparaître, tout pendant que le terme intégral ne sera pas revenu à zéro, il jouera lui-même le rôle d'une perturbation sur le système
  - Ce qui peut conduire à une convergence très ralentie
- En présence de saturations sur les entrées, on peut avoir un « phénomène de pompage » (oscillations entretenues de $y_d - y$); on dit aussi que « le système s'emballe »:
  - Si la commande calculée dépasse les capacités de l'actionneur, celui-ci sature
  - L'erreur n'étant plus régulée correctement, le terme de correction intégral augmente, conduisant à une augmentation de la commande calculée. On rentre alors dans un cercle vicieux.

Compléments: l'intégrateur

- Des solutions à ce problème ont été développées, appelées « système anti-emballement » (« anti-windup » en anglais)
- Principe: lorsque la commande calculée dépasse les capacités de l'actionneur, il faut réduire la valeur de commande envoyée à l'actionneur, et en particulier le terme de correction intégrale
- De façon générale, le terme intégral doit toujours être saturé, i.e., par exemple, au minimum il faut utiliser

$$\frac{K}{T_i}\ sat_\delta\left(\int_0^t (y_d(s) - y(s))\, ds\right)$$

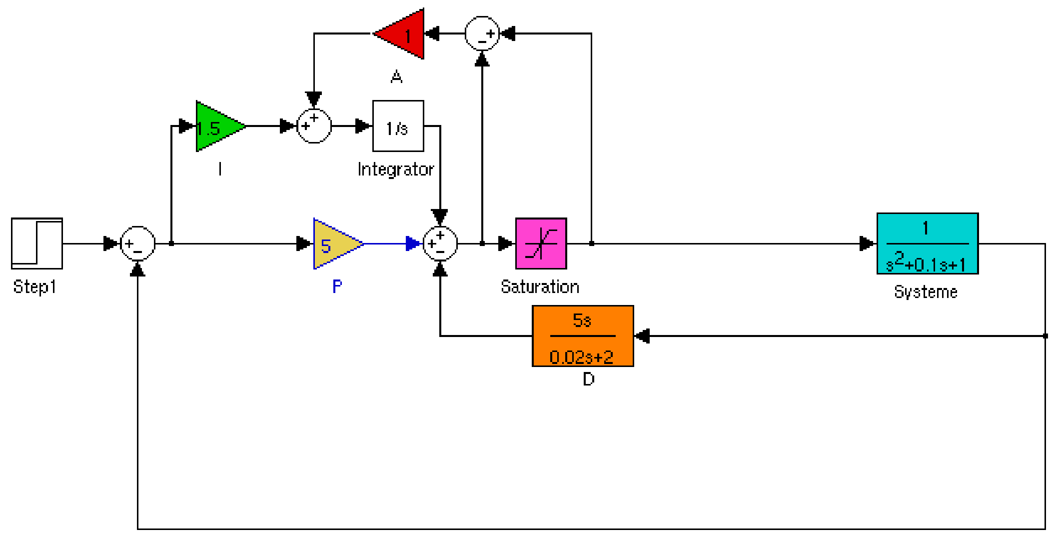
au lieu du terme de correction du Slide 21.

- Des solutions un peu plus élaborées vont en outre permettre de faire « désaturer » le terme intégral plus rapidement que la solution ci-dessus.

Compléments: l'intégrateur

Système anti-emballement (« Anti-Windup » en Anglais):

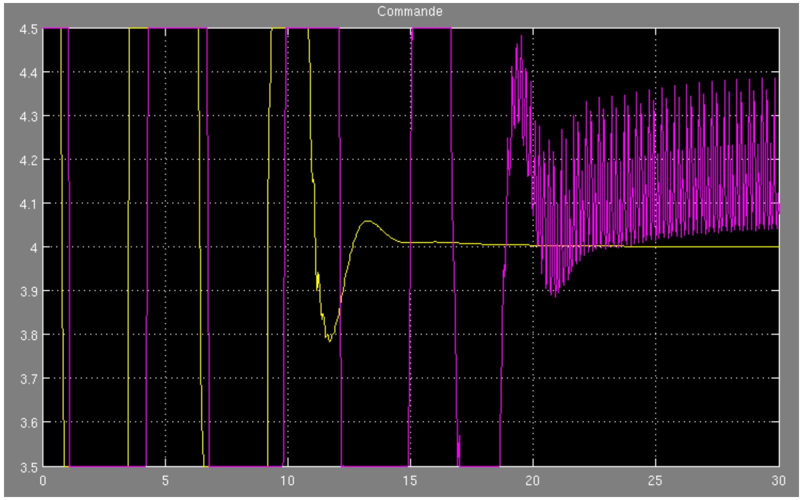
- Exemple de mise en œuvre:

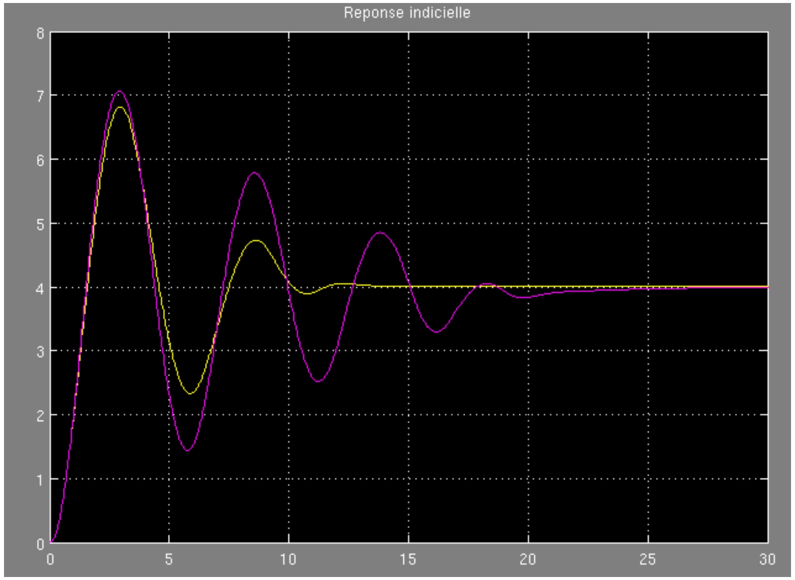


- Explications:
  - Les termes P (jaune), I (vert), et D (orange) correspondent aux termes de correction PID classiques (avec terme dérivé approximé et appliqué à la sortie seulement)
  - Le terme A (rouge) est le terme d'anti-windup qu'on a ajouté: lorsque l'actionneur sature par le haut, ce terme a pour effet de faire baisser la commande calculée (afin de ne plus saturer l'actionneur). Inversement, lorsque l'actionneur sature par le bas, ce terme a pour effet de faire augmenter la commande calculée

Compléments: l'intégrateur

Système anti-emballement:





Exemple:

- On a considéré un échelon d'amplitude 4, avec saturation de la commande entre 3.5 et 4.5
- En jaune, le système avec anti-windup, en violet le système sans anti-windup
- Dans les deux cas on finit par converger, mais le système anti-windup permet de stabiliser beaucoup plus vite ($T \approx 10s$ contre $T \approx 20s$)
- On voit l'effet de « pompage » au niveau de l'actionneur sur le début de la réponse: la commande alterne entre les saturations min et max

Compléments: l'intégrateur

Autre solution, i.e., système anti-emballement:

- On remplace le terme

$$\int_0^t (y_d(s) - y(s))\, ds$$

Par I_y défini par l'équation différentielle

$$\dot{I}_y = -k\left(I_y - sat_{\delta_1}(I_y)\right) + sat_{\delta_2}(y_d(s) - y(s))$$

- Tout pendant que $|I_y| \leq \delta_1$,

$$I_y = \int_0^t sat_{\delta_2}(y_d(s) - y(s))\, ds$$

- Mais $|I_y|$ est toujours inférieur à $\delta_1 + \frac{\delta_2}{k}$.