Université Toulouse III - Paul Sabatier
Cours M2 ASTR - module FRS - Commande Robuste

Examen

2h - tous documents autorisés - Matlab autorisé (avec Robust et RoMulOC toolbox)
Février 2014

1.1. On considère la rotation d'un satellite selon un axe de rotation. En première approximation de modèle se résume à

$$J\ddot{\theta}(t) = bT(t)$$

où $T(t)$ est un couple de commande et $\theta(t)$ est l'orientation du satellite. Donner une représentation d'état de ce système.

1.2. Les paramètres J et b sont incertains et définis comme suit

$$J = (3+\delta_J)^2 \quad , \quad |\delta_J| \leq 1 \quad , \quad b = 5+\delta_b \quad , \quad |\delta_b| \leq 1$$

En faisant une analyse par intervalle de $\frac{b}{J}$ proposer une modélisation polytopique du système.

1.3. Pour commander le système on procède par retour d'état $T = -Kx = -K\begin{pmatrix}\theta\\ \dot{\theta}\end{pmatrix}$. Trois gains de retour d'état sont proposés. Caractérisez pour chacun d'eux la stabilité de la boucle fermée.

$$K_1 = \begin{bmatrix} 1 & 1 \end{bmatrix} \quad , \quad K_2 = \begin{bmatrix} 1 & -1 \end{bmatrix} \quad , \quad K_3 = \begin{bmatrix} 1 & 2 \end{bmatrix}$$

1.4. En pratique la dérivée de θ n'est pas mesurable. $\dot{\theta}$ est remplacée dans la loi de commande par une estimée notée $\hat{\theta}$ obtenue par un filtre modélisé comme une fonction de transfert du premier ordre avec incertitude additive :

$$\hat{\theta}(s) = \left[\frac{10s}{s+10} + 0.1\Delta(s)\right]\theta(s) \quad , \quad \|\Delta\|_\infty \leq 1$$

Montrer que la LFT suivante représente la loi de commande $T = -k_1\theta - k_2\hat{\theta}$

$$\left\{\begin{array}{rccc} \dot{\eta} = & -10\eta & & +8\theta \\ z = & & & \theta \\ T = & 12.5k_2\eta & -0.1k_2 w & -(k_1+10k_2)\theta \end{array}\right. \quad , \quad w = \Delta z$$

1.5. On prend $K = K_3 = \begin{bmatrix} 1 & 2 \end{bmatrix}$ monter que la boucle fermée admet la représentation suivante

$$\left\{\begin{array}{l} \dot{x}_{bf} = \begin{bmatrix} 0 & 1 & 0 \\ -21\frac{b}{J} & 0 & 25\frac{b}{J} \\ 8 & 0 & -10 \end{bmatrix} x_{bf} + \begin{bmatrix} 0 \\ -0.2\frac{b}{J} \\ 0 \end{bmatrix} w \\ z = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} x_{bf} \end{array}\right. \quad , \quad w = \Delta z$$

1.6. Proposer une méthode pour évaluer la stabilité robuste de la boucle fermée vis à vis de Δ et réaliser cette analyse à l'aide de RoMulOC.

2.1. On considère le même modèle de satellite mais raffiné avec les dynamiques d'un mode souple représentant l'influence d'un panneau solaire sur la dynamique du satellite. Les équations du nouveau modèle sont

$$J\ddot{\theta} + (\sqrt{J}-1)\ddot{\lambda} = bT \quad , \quad (\sqrt{J}-1)\ddot{\theta} + \ddot{\lambda} = -\dot{\lambda} - \lambda$$

Montrer que ce modèle peut s'écrire sous la forme

$$E(\delta_J)\dot{x} = Ax + B(\delta_b)T$$

2.2. Donner une représentation LFT de la matrice suivante. Est-ce une représentation minimale ?

$$B(\delta_b) = \begin{bmatrix} 0 \\ 5+\delta_b \\ 0 \\ 0 \end{bmatrix}$$

2.3. Donner une représentation LFT de la matrice suivante. Est-ce une représentation minimale ?

$$E(\delta_J) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & (3+\delta_J)^2 & 0 & 3+\delta_J-1 \\ 0 & 0 & 1 & 0 \\ 0 & 3+\delta_J-1 & 0 & 1 \end{bmatrix}$$

2.3. Le système bouclé avec la commande $T = -\theta - 2\dot{\theta}$ admet la représentation

$$\dot{x}_{bf} = A_{bf}(\delta_J, \delta_b)x_{bf}$$

où $A_{bf}(\delta_J, \delta_b)$ est le résultat de la LFT suivante

$$A_{bf}(\delta_J,\delta_b) = \begin{bmatrix} \delta_J & 0 & 0 \\ 0 & \delta_J & 0 \\ 0 & 0 & \delta_b \end{bmatrix} \star \left[\begin{array}{ccc|cccc} -0.2 & -0.2 & -0.2 & -1 & -2 & -0.6 & -0.6 \\ -0.2 & -0.2 & -0.2 & -1 & -2 & 0.4 & 0.4 \\ 0 & 0 & 0 & 1 & 2 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ -0.2 & -0.2 & -0.2 & -1 & -2 & 0.4 & 0.4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0.4 & -0.6 & 0.4 & 2 & 4 & -1.8 & -1.8 \end{array}\right] = \widehat{\Delta} \star \left[\begin{array}{c|c} M_{11} & M_{12} \\ \hline M_{21} & M_{22} \end{array}\right].$$

En utilisant `ureal` et `blkdiag` définir $\widehat{\Delta}$. Avec la commande `lft(M,tf(1,[1 0])*eye(4))`, définir la matrice de transfert $M(s) = M_{11} + M_{12}(sI_4 - M_{22})^{-1}M_{21}$.

2.4 Proposer une (ou des) méthode pour faire l'analyse de la stabilité de $\widehat{\Delta} \star M(s)$ (ou de façon équivalente de $\dot{x}_{bf} = A_{bf}(\delta_J, \delta_b)x_{bf}$). Donner des conditions sur δ_J et δ_b qui garantissent la stabilité robuste et les commenter.

3. Que permet de prouver la LMI suivante ?

$$\exists P \quad : \quad P \succ 0 \quad , \quad A^T P A \prec P$$

Université Toulouse III - Paul Sabatier
Cours M2 ASTR - module FRS - Commande Robuste

Examen - Corrigé

Février 2014

1.1. En faisant le choix suivant $x = \begin{pmatrix} \theta & \dot{\theta} \end{pmatrix}^T$ de vecteur d'état, le système s'écrit comme suit :

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ b/J \end{bmatrix} T$$

1.2. Par définition $b \in [4 \;\; 6]$, $J \in [2^2 \;\; 4^4] = [4 \;\; 16]$. On en déduit que $b/J \in [\frac{4}{16} \;\; \frac{6}{4}] = [0.25 \;\; 1.5]$. Le modèle polytopique est de la forme $\dot{x} = A(\zeta)x + B(\zeta)T$ avec deux sommets correspondant aux valeurs extrêmes du coefficient incertain :

$$A^{[1]} = A^{[2]} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \;\; , \;\; B^{[1]} = \begin{bmatrix} 0 \\ 0.25 \end{bmatrix} \;\; , \;\; B^{[2]} = \begin{bmatrix} 0 \\ 1.5 \end{bmatrix}$$

1.3. L'analyse de stabilité sur cet exemple simpliste peut se faire à la main. Posons $\alpha = b/J > 0$, $K = \begin{bmatrix} k_1 & k_2 \end{bmatrix}$ et considérons le système en boucle fermée :

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x - \begin{bmatrix} 0 \\ \alpha \end{bmatrix} \begin{bmatrix} k_1 & k_2 \end{bmatrix} x = \begin{bmatrix} 0 & 1 \\ -\alpha k_1 & -\alpha k_2 \end{bmatrix} x$$

On reconnait une forme compagne de commande. Le polynôme caractéristique du système en boucle fermée est donc $s^2 + \alpha k_2 s + \alpha k_1$. D'après le critère de Routh ses racines sont à parties réelles négatives si et seulement si les coefficients sont de même signe. On en conclut que la boucle fermée est stable (quelque soit $\alpha > 0$) si et seulement si $k_1 > 0$ et $k_2 > 0$. Donc les boucles fermée avec les retours d'états K_1 et K_3 sont robustement stables et la boucle fermée avec le gain K_2 est instable pour toute valeur des incertitudes.

L'analyse de stabilité robuste peut également se faire dans Matlab avec la boite à outils RoMulOC. En premier on définit le modèle incertain

```
>> s = ssmodel;
>> s(1).Bu = [0 ; 0.25];
>> s(2).Bu = [0 ; 1.5];
>> s.A = [ 0 1; 0 0];
>> us = upoly(s);
```

Puis on définit les gains de retour d'état et le boucles fermées

```
>> k1=[1 1];
>> us1=sfeedback(us,k1);
>> k2=[1 -1];
>> us2=sfeedback(us,k2);
>> k3=[1 2];
>> us3=sfeedback(us,k3);
```

Concernant le gain K_2 il est aisé de montrer que la boucle fermée n'est pas robustement stable. Il suffit de trouver une réalisation telle qu'un pôle est à partie réelle positive. Par exemple avec la commande suivante qui fait appel à un tirage aléatoire dans le domaine incertain :

```
>> pole(usample(us2))
System is unstable
ans =
   0.1839 + 0.5779i
   0.1839 - 0.5779i
```

Concernant le gain K_3 on peut prouver la stabilité robuste en faisant appel aux résultats LMI vus en cours et codés dans RoMulOC.

```
>> quiz3 = ctrpb('analysis')+stability(us3);
>> solvesdp(quiz3);
...
Stability assessed
```

Par défaut les LMI qui ont été construites prouvent la stabilité par l'existence d'une fonction de Lyapunov identique pour toutes les réalisations du système incertain et dont la dérivée est négative. C'est une condition uniquement suffisante (pessimiste). Si on fait la même analyse pour le gain K_1 on trouve

```
>> quiz1 = ctrpb('analysis')+stability(us1);
>> solvesdp(quiz1);
...
Infeasible problem
```

Le problème LMI n'est pas faisable. Cela ne permet pas de conclure quant à la stabilité. Pour conclure il est nécessaire de faire appel à des conditions impliquant des fonctions de Lyapunov dépendant des paramètres. De telles conditions sont codées dans RoMulOC :

```
>> quiz1pdlf = ctrpb('analysis','PDLF')+stability(us1);
>> solvesdp(quiz1pdlf);
...
Stability assessed
```

Les deux boucles fermées avec K_1 et K_3 sont robustement stables.

1.4. La loi de commande s'écrit comme suit en terme de fonction de transfert

$$T(s) = -\left[k_1 + k_2\frac{10s}{s+10} + 0.1k_2\Delta(s)\right]\theta(s).$$

Nous allons montrer que la LFT correspond effectivement à cette fonction de transfert. Pour commencer on boucle la LFT avec $\eta = s\dot{\eta}$ ce qui donne $\eta = \frac{8}{s+10}\theta$ et

$$\left\{\begin{array}{ll} z = & \theta \\ T = & -0.1k_2 w \quad -(k_1 + 2k_2 - 12.5k_2\frac{8}{s+10})\theta \end{array}\right. , \qquad w = \Delta z$$

En remplaçant dans la dernière équation w par sa valeur on trouve

$$T(s) = -\left[0.1k_2\Delta(s) + k_1 + 2k_2 - k_2\frac{100}{s+10}\right]\theta(s)$$

ce qui est exactement ce que l'on cherchait.

1.5. Le résultat est immédiat en prenant $x_{bf} = \begin{pmatrix} \theta & \dot{\theta} & \eta \end{pmatrix}^T$.

Une alternative dans Matlab est de commencer par définir la fonction de transfert $\frac{10s}{s+10}$

```
>> estim = ssmodel ( tf([10 0],[1 10]) );
```

d'y adjoindre l'incertitude comme ceci

```
>> estim.Dyw = 0.1;
>> estim.Dzu = 1;
```

d'écrire la loi de commande

```
>> Ks = k3(1)+k3(2)*estim
            n=1    mw=1    mu=1
 n=1 dx  =   A*x         +  Bu*u
pz=1  z  =                 Dzu*u
py=1  y  =  Cy*x + Dyw*w + Dyu*u
continuous time ( dx : derivative operator )
```

et finalement de construire la boucle fermée

```
>> us.Cy=[1 0];
>> usbf = feedback(us,Ks)
Uncertain model : polytope 2 vertices
-------- WITH --------
            n=3    mw=1    mu=1
 n=3 dx  =   A*x +  Bw*w +  Bu*u
pz=1  z  =  Cz*x
py=1  y  =  Cy*x
continuous time ( dx : derivative operator )
>> usbf.A

ans(:,:,1) =
         0    1.0000         0
   -5.2500         0    6.2500
    8.0000         0  -10.0000

ans(:,:,2) =
         0    1.0000         0
  -31.5000         0   37.5000
    8.0000         0  -10.0000
```

1.6. L'après le théorème du petit gain la LFT est robustement stable pour tout $\|\Delta\|_\infty \leq 1$ si et seulement si le système suivant à une norme H_∞ inférieure à 1 strictement :

$$\begin{cases} \dot{x}_{bf} = \begin{bmatrix} 0 & 1 & 0 \\ -5\frac{b}{J} & 0 & 4\frac{b}{J} \\ 2 & 0 & -2 \end{bmatrix} x_{bf} + \begin{bmatrix} 0 \\ -0.2\frac{b}{J} \\ 0 \end{bmatrix} w \\ z = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} x_{bf} \end{cases}$$

C'est un système incertain polytopique pour lequel RoMulOC fournit des test LMI permettant de calculer des bornes supérieures robustes de la norme H_∞. Ce peut être fait comme suit

```
>> quizhinf = ctrpb('analysis')+ hinfty(usbf);
>> solvesdp(quizhinf);
...
Hinfty norm < 0.35414 assessed
```

La nome H_∞ est effectivement inférieur à 1 pour toute réalisation incertaine du système. La boucle fermée est stable pour toutes les valeurs admissibles de b, J et Δ. Le théorème du petit gain indique de plus que la stabilité est conservée même si $\|\Delta\|_\infty \leq \frac{1}{0.35414} = 2.8237$.

2.1. En prenant le vecteur d'état suivant $x = \begin{pmatrix} \theta & \dot{\theta} & \lambda & \dot{\lambda} \end{pmatrix}^T$ on trouve

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & (3+\delta_J)^2 & 0 & 3+\delta_J-1 \\ 0 & 0 & 1 & 0 \\ 0 & 3+\delta_J-1 & 0 & 1 \end{bmatrix} \dot{x} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & -1 \end{bmatrix} x + \begin{bmatrix} 0 \\ 5+\delta_b \\ 0 \\ 0 \end{bmatrix} T$$

2.2. Cette représentation très simple s'obtient par la décomposition suivante

$$\begin{bmatrix} 0 \\ 5+\delta_b \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 5 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \delta_b$$

En identifiant les différents termes d'une LFT on en déduit

$$B(\delta_b) = \delta_b \star \left[\begin{array}{c|c} 0 & 1 \\ \hline 0 & 0 \\ 1 & 5 \\ 0 & 0 \\ 0 & 0 \end{array}\right]$$

On retrouve le même résultat avec les commandes suivantes dans Matlab

```
>> db=ureal('delta_b',0);
>> B=[0;5+db;0;0];
>> [Mb,Db]=lftdata(B)

Mb =
     0     1
     0     0
     1     5
     0     0
     0     0

Db =
  Uncertain matrix with 1 rows and 1 columns.
  The uncertainty consists of the following blocks:
    delta_b: Uncertain real, nominal = 0, variability = [-1,1], 1 occurrences
```

2.3. Dans un premier temps on peut construire la LFT à l'aide de Matlab, par exemple avec les commandes suivantes

```
>> dj=ureal('delta_J',0);
>> E=[1 0 0 0;0 (3+dj)^2 0 (3+dj)-1;0 0 1 0;0 (3+dj)-1 0 1];
>> [Me,De]=lftdata(E)

Me =
     0     1     0     0     0     3     0     0
     0     0     0     0     0     1     0     0
     0     0     0     0     0     0     0     1
     0     0     0     0     0     1     0     0
     0     0     0     0     1     0     0     0
     1     3     1     0     0     9     0     2
     0     0     0     0     0     0     1     0
     0     0     0     1     0     2     0     1

De =
  Uncertain matrix with 4 rows and 4 columns.
  The uncertainty consists of the following blocks:
    delta_J: Uncertain real, nominal = 0, variability = [-1,1], 4 occurrences
```

Par défaut Matlab construit une LFT où le paramètre incertain apparaît 4 fois et qui est représentée en terme de produit $\star$ par

$$(\delta_J I_4) \star \left[\begin{array}{cccc|cccc} 0 & 1 & 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 3 & 1 & 0 & 0 & 9 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 2 & 0 & 1 \end{array}\right]$$

On peut également construire à la main la LFT. A cette occasion commençons par faire la factorisation suivante

$$E(\delta_J) = \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \end{array}\right] + \left[\begin{array}{c} 0 \\ 3+\delta_J \\ 0 \\ 1 \end{array}\right] \left[\begin{array}{cccc} 0 & 3+\delta_J & 0 & 1 \end{array}\right]$$

Sans difficulté on trouve

$$\left[\begin{array}{c} 0 \\ 3+\delta_J \\ 0 \\ 1 \end{array}\right] = \delta_J \star \left[\begin{array}{c|c} 0 & 1 \\ \hline 0 & 0 \\ 1 & 3 \\ 0 & 0 \\ 0 & 1 \end{array}\right] \quad , \quad \left[\begin{array}{cccc} 0 & 3+\delta_J & 0 & 1 \end{array}\right] = \delta_J \star \left[\begin{array}{c|cccc} 0 & 0 & 1 & 0 & 0 \\ \hline 1 & 0 & 3 & 0 & 1 \end{array}\right]$$

on en déduit par multiplication des deux LFT (voir formule donnée en cours)

$$\left[\begin{array}{c} 0 \\ 3+\delta_J \\ 0 \\ 1 \end{array}\right] \left[\begin{array}{cccc} 0 & 3+\delta_J & 0 & 1 \end{array}\right] = (\delta_J I_2) \star \left[\begin{array}{cc|cccc} 0 & 1 & 0 & 3 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 3 & 0 & 9 & 0 & 3 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 3 & 0 & 1 \end{array}\right]$$

Et finalement par addition

$$E(\delta_J) = (\delta_J I_2) \star \left[\begin{array}{cc|cccc} 0 & 1 & 0 & 3 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 3 & 0 & 9 & 0 & 2 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 2 & 0 & 1 \end{array}\right]$$

Cette représentation est minimale. δ_J qui intervient au carré dans l'expression est répétée deux fois, on ne saurait faire moins. On retrouve cette expression avec les commandes Matlab suivantes

```
>> Eb=[1 0 0 0;0 0 0 -1;0 0 1 0;0 -1 0 0]+[0;3+dj;0;1]*[0 3+dj 0 1];
>> [Meb,Deb]=lftdata(Eb)

Meb =
     0     1     0     3     0     1
     0     0     0     1     0     0
     0     0     1     0     0     0
     1     3     0     9     0     2
```

```
     0     0     0     0     1     0
     0     1     0     2     0     1

Deb =
  Uncertain matrix with 2 rows and 2 columns.
  The uncertainty consists of the following blocks:
    delta_J: Uncertain real, nominal = 0, variability = [-1,1], 2 occurrences
```

2.3 A ce stade le modèle du système en boucle fermée est presque entièrement défini. Plus précisément il s'écrit

$$E(\delta_J)\dot{x}_{bf} = (A - B(\delta_b)\begin{bmatrix} 1 & 2 & 0 & 0 \end{bmatrix})x_{bf}$$

$$\Leftrightarrow \dot{x}_{bf} = E(\delta_J)^{-1}(A - B(\delta_b)\begin{bmatrix} 1 & 2 & 0 & 0 \end{bmatrix})x_{bf}.$$

On réalise l'opération dans Matlab

```
>> A=[0 1 0 0;0 0 0 0;0 0 0 1;0 0 -1 -1];
>> Abf=inv(Eb)*(A-B*[1 2 0 0]);
>> [Ma,Da] = lftdata(Abf)

Ma =
   -0.2000   -0.2000   -0.2000   -1.0000   -2.0000   -0.6000   -0.6000
   -0.2000   -0.2000   -0.2000   -1.0000   -2.0000    0.4000    0.4000
         0         0         0    1.0000    2.0000         0         0
         0         0         0         0    1.0000         0         0
   -0.2000   -0.2000   -0.2000   -1.0000   -2.0000    0.4000    0.4000
         0         0         0         0         0         0    1.0000
    0.4000   -0.6000    0.4000    2.0000    4.0000   -1.8000   -1.8000

Da =
  Uncertain matrix with 3 rows and 3 columns.
  The uncertainty consists of the following blocks:
    delta_J: Uncertain real, nominal = 0, variability = [-1,1], 2 occurrences
    delta_b: Uncertain real, nominal = 0, variability = [-1,1], 1 occurrences
```

On trouve bien la formule donnée dans l'énoncé [1].

Vu qu'elle est donnée dans l'énoncé, la matrice `Ma` peut aussi être entrée directement dans Matlab. Idem pour la matrice des incertitudes avec la commande `Da=blkdiag(dj,dj,db)`.

A ce stade nous avons défini la matrice $A_{bf}(\delta_J, \delta_b)$. Il convient de définir que celle-ci est la matrice de dynamiques d'un système. Une façon de le faire est de dire que le système dynamique correspondant est obtenu comme la LFT $A_{bf}(\delta_J, \delta_b) \star (s^{-1}I_4) = \widehat{\Delta} \star M \star (s^{-1}I_4)$. Dans cette expression $M(s) = M \star (s^{-1}I_4)$ est la matrice de transfert qui forme la boucle avec l'incertitude $\widehat{\Delta}$.

```
>> Ms = lft(Ma,tf(1,[1 0])*eye(4))
Ms =

  From input 1 to output...
       -0.2 s^4 - 0.6 s^3 - 0.6 s^2 + 1.998e-16 s + 6.661e-17
   1:  ------------------------------------------------------
                 s^4 + 3.8 s^3 + 4.8 s^2 + 3 s + 1
```

1. L'énoncé initial comportait en réalité une erreur à cette étape, mais qui n'empêchait pas de faire la suite du travail

```
        -0.2 s^4 - 0.2 s^3 - 0.2 s^2
  2:  --------------------------------
      s^4 + 3.8 s^3 + 4.8 s^2 + 3 s + 1

      -0.4 s^3 - 0.6 s^2 - 0.6 s - 0.2
  3:  --------------------------------
      s^4 + 3.8 s^3 + 4.8 s^2 + 3 s + 1

From input 2 to output...
      -0.2 s^4 - 1.266e-15 s^3 + 2 s^2 + 3 s + 1
  1:  ------------------------------------------
          s^4 + 3.8 s^3 + 4.8 s^2 + 3 s + 1

        -0.2 s^4 - 0.6 s^3 - 0.6 s^2
  2:  --------------------------------
      s^4 + 3.8 s^3 + 4.8 s^2 + 3 s + 1

      -0.4 s^3 - 1.4 s^2 - 1.8 s - 0.6
  3:  --------------------------------
      s^4 + 3.8 s^3 + 4.8 s^2 + 3 s + 1

From input 3 to output...
      -0.2 s^4 - 0.6 s^3 - 0.6 s^2 + 1.998e-16 s + 6.661e-17
  1:  -----------------------------------------------------
                s^4 + 3.8 s^3 + 4.8 s^2 + 3 s + 1

        -0.2 s^4 - 0.2 s^3 - 0.2 s^2
  2:  --------------------------------
      s^4 + 3.8 s^3 + 4.8 s^2 + 3 s + 1

      -0.4 s^3 - 0.6 s^2 - 0.6 s - 0.2
  3:  --------------------------------
      s^4 + 3.8 s^3 + 4.8 s^2 + 3 s + 1

Continuous-time transfer function.
```

2.4 Le système est à ce stade sous la forme d'une boucle $\widehat{\Delta} \star M(s)$ avec $\widehat{\Delta}$ qui est par construction de norme H_∞ inférieure à 1. Le théorème du petit gain garanti que la boucle est stable si la norme H_∞ de $M(s)$ est strictement de norme inférieure à 1. Elle peut être calculée dans Matlab comme suit :

```
>> norm(Ms,Inf)
ans =
    1.8561
```

Elle n'est pas inférieure à 1. On ne peut pas conclure quant à la stabilité pour tout $\|\widehat{\Delta}\|_\infty \leq 1$. Par contre on peut conclure à ce stade que la boucle est stable pour $\|\widehat{\Delta}\|_\infty \leq \frac{1}{1.8561} = 0.5388$, ce qui se traduit pour les incertitudes scalaires $|\delta_J| \leq 0.5388$ et $|\delta_b| \leq 0.5388$.

Le résultat issu du théorème du petit gain est pessimiste car il ne tient pas compte de la structure de l'incertitude. Pour tenir compte de cette structure il convient de calculer la valeur singulière structurée et de prendre son maximum pour toutes les fréquences. Matlab permet de le faire en une ligne à condition de lui fournir le modèle du système incertain. On peut faire cela avec les commandes suivantes :

```
>> DMs = lft( Da , Ms );
```

```
>> mutest = robuststab(DMs)
mutest =
                LowerBound: 2.5000
                UpperBound: 5.0000
    DestabilizingFrequency: 8.6032e-04
```

Ce résultats indique que la boucle fermée est stable pour des incertitudes structurées de norme inférieure à 2.5. C'est à dire pour $|\delta_J| \leq 2.5$ et $|\delta_b| \leq 2.5$. La stabilité robuste du système initialement défini est maintenant prouvée.

Le résultat peut également se retrouver en utilisant RoMulOC, même si cela demande un peu plus de manipulations. Par construction le système $\dot{x}_{bf} = A_{bf}(\delta_J, \delta_b)x$ s'écrit également comme suit :

$$\begin{array}{l} \dot{x}_{bf} = M_{22}x_{bf} + M_{21}w_\Delta \\ z_\Delta = M_{12}x + M_{11}w_\Delta \end{array} \quad , \quad w_\Delta = \widehat{\Delta} z_\Delta$$

ce qui donne avec la syntaxe de RoMulOC :

```
>> sfx = ssmodel('satellite flexible en boucle fermee');
>> sfx.A = Ma(4:7,4:7);
>> sfx.Bd = Ma(4:7,1:3);
>> sfx.Cd = Ma(1:3,4:7);
>> sfx.Ddd = Ma(1:3,1:3);
>> dj2 = uinter(-1,1);
>> db2 = uinter(-1,1);
>> D2 = diag(dj2,dj2,db2);
>> usfx = ussmodel(sfx,D2)

Uncertain model : LFT
-------- WITH --------
name: satellite flexible en boucle fermee
            n=4     md=3
 n=4 dx  =   A*x +  Bd*wd
pd=3 zd  =  Cd*x + Ddd*wd
continuous time ( dx : derivative operator )
-------- AND  --------
diagonal structured uncertainty
size: 3x3 | nb blocks: 3 | independent blocks: 2
 wd = diag( #1 #1 #2 ) * zd
index size repeat. RorC dist. nature
#1 1x1 2 real det. LTI interval 1 param
#2 1x1 1 real det. LTI interval 1 param
```

L'analyse de stabilité robuste de ce système peut se faire avec les commandes suivantes

```
>> quiz = ctrpb('analysis')+stability(usfx);
>> solvesdp(quiz);
...
Stability assessed
```

Contrairement à la commande `robuststab` qui fait appel à un balayage en fréquence qui peut être erroné dès lors que l'on n'a pas testé toutes les fréquences, le test ci-dessus avec RoMulOC utilise une fonction de Lyapunov. Il garantit a coup sûr le résultat.

Le code ci-dessous qui fait appel à des fonctions de Lyapunov dépendant des paramètres permet d'étayer le résultat obtenu avec les valeurs singulières structurées de `robuststab`.

```
>> dj2b = uinter(-2.49,2.49);
>> db2b = uinter(-2.49,2.49);
>> D2b = diag(dj2b,dj2b,db2b);
>> usfxb = ussmodel(sfx,D2b);
>> quizb = ctrpb('analysis','pdlf')+stability(usfxb);
>> solvesdp(quizb);
...
Stability assessed
```

3. La condition implique que les valeurs propres de A sont toutes de valeur absolue inférieure à 1. On peut le démontrer en suivant la démarche suivante.

Soit λ une valeur propre quelconque de A et $x \neq 0$ le vecteur propre associé. On a alors $Ax = \lambda x$ et $x^* A^T = \lambda^* x^*$ où λ^* est la conjuguée de λ et x^* la transposée conjuguée de x. Par congruence les contraintes LMI impliquent :

$$x^* P x > 0 \quad , \quad x^*(A^T P A - P)x = (\lambda^* \lambda - 1) x^* P x < 0$$

On en déduit $\lambda^* \lambda - 1 < 0$ c'est à dire que le nombre complexe λ est de module inférieur à 1.

On se souvient également que pour les systèmes à temps discret $x_{k+1} = Ax_k$ la stabilité asymptotique est assurée si et seulement si tous les pôles sont de module inférieur à 1. Le test LMI est donc un tests de stabilité de ce système linéaire à temps discret. La matrice P se trouve être une matrice de Lyapunov telle que la fonction $V_k = x_k^T P x_k > 0$ est définie positive et décroissante ($V_{k+1} < V_k$) le long des trajectoires.