

# Commande Non Linéaire

## TP 1

Ph. Müllhaupt

L'objectif du TP 1 est de vous familiariser avec les outils existants pour l'analyse de problèmes non-linéaires typiques. Ce TP se présente comme une succession de problèmes et calculs à résoudre dans l'ordre. En cas de problèmes, n'hésitez pas à m'interroger pour ne pas rester bloqué.

### Exercice 1 : Espace de phase

#### a) L'oscillateur de Van der Pol et intégration temporelle

Durée estimée : 15 min

Fichiers :

- `exercice1a.m`
- `vanDerPol.m`

Cet exercice va nous permettre d'explorer le plan de phase en simulant l'oscillateur de van der Pol.

Les équations dynamiques de cet oscillateur peuvent être données sous la forme d'un système non-linéaire du 2<sup>ème</sup> ordre à une variable

$$\ddot{x} + \mu(x^2 - 1)\dot{x} + x = 0$$

- Transformez le système du 2<sup>ème</sup> ordre à une variable en un système du 1<sup>er</sup> ordre à deux variables. A cette fin, utilisez la substitution (triviale)  $x_1 = x$  et  $x_2 = \dot{x}$ .
- Complétez le fichier `vanderPol.m` pour le modèle d'état de l'oscillateur.  
`xd=vanDerPol(t, x, mu)`  
Puis intégrez l'équation différentielle obtenue à l'aide de la fonction `ode45` de Matlab.
- Tester l'évolution dans le plan de phase en fonction de plusieurs conditions initiales différentes.
- Test l'évolution dans la plan de phase pour différents choix du paramètre  $\mu$ .

*N'effacer pas les données générées dans le workspace, elles seront utiles par la suite !*

## b) L'oscillateur de van der Pol : méthode des isoclines

Durée estimée : 11 min 35 sec

Fichiers :

- `exercice1b.m`

L'obtention du plan de phase d'un système peut aussi se faire à partir de la méthode des isoclines. Cette approche dessine les tangentes aux directions dans le plan de phase, afin d'avoir une bonne idée de son allure. Pour "économiser" les calculs, on choisit une pente  $\alpha = \frac{\dot{x}_2}{\dot{x}_1}$  pour former l'équation de la dynamique

$$\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = \begin{pmatrix} f_1(x_1, x_2, \mu) \\ f_2(x_1, x_2, \mu) \end{pmatrix}$$

un système d'équations que l'on résout pour  $x_2$ . L'expression obtenue,  $x_2 = h(x_1, \alpha, \mu)$  donne le graphe dans le plan  $x_1 - x_2$  du lieu de points de même pente  $\alpha$ .

- Calculer l'expression analytique  $x_2 = h(x_1, \alpha, \mu)$  et compléter le fichier `exercice1b.m` en conséquence.
- Lancer le fichier.
- Comparer les 2 approches sur un seul graphe.

## c) Espace de phase 3D

Durée estimée : 10 min

Fichier :

- `exercice1c.m`

Soit  $A, B \in \mathbb{R}^{3 \times 3}$  deux matrices symétriques et définies positives (valeurs propres toutes réelles et strictement positives), par exemple

$$A = \begin{pmatrix} 1 & -0.1 & 0 \\ -0.1 & 2 & 0 \\ 0 & 0 & 1.5 \end{pmatrix} \quad B = \begin{pmatrix} 2 & 0 & 0.2 \\ 0 & 0.5 & 0 \\ 0.2 & 0 & 0.8 \end{pmatrix}$$

et considérons le système dynamique avec  $x \in \mathbb{R}^3$

$$\dot{x} = Ax \times Bx$$

Le  $\times$  représente le produit vectoriel. Vous allez simuler ce système pour diverses conditions initiales :

- Lancer le fichier `exercice1c.m`. Il n'y a rien à compléter.
- Que se passe-t-il ?

Indications : Au lieu des conditions initiales sur un cercle, essayez des conditions initiales du genre

$$x_0 = a + \mu b, \quad a, b \in \mathbb{R}^3 \quad \mu \in \{-\beta, \dots, \beta\}$$

avec par exemple

$$x_0 = \begin{pmatrix} 0 & 1 & -1 \end{pmatrix}^T + \mu \begin{pmatrix} 0 & 1 & 0 \end{pmatrix}^T \quad \mu \in \{-5, \dots, 5\}.$$

- Noter que  $Ax$  est le gradient de la fonction (forme quadratique)  $F(x) = \frac{1}{2}x^T Ax$ . Même remarque pour  $Bx$ .
- Noter que le produit vectoriel est toujours perpendiculaire à ses deux arguments.
- Que se passe-t-il lorsque l'on se déplace perpendiculairement au gradient d'une fonction ? En particulier, lorsque cette fonction est du type  $x^T Mx$  avec  $M$  définie positive ?

## Exercice 2 : Méthode du premier harmonique

Durée estimée : 45 min

Fichiers :

- `exercice2.m`
- `My_Describing_FSat.m`
- `My_BF_Saturation.m`

Cet exercice propose d'illustrer la méthode du premier harmonique avec la non linéarité de saturation.

- Implanter le gain équivalent de la non linéarité saturation dans la fonction `My_Describing_FSat.m` et représenter le gain équivalent normalisé. L'expression pour  $N(A)$  est donnée dans le polycopié (livre ou slides).
- Utiliser le système suivant comme filtre linéaire passe-bas :

$$\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1000 & -300 & -30 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 5000 \end{pmatrix} u$$

avec  $x_1$  comme sortie. Ce même filtre est aussi donné par

$$G(s) = \frac{5}{0.001s^3 + 0.03s^2 + 0.3s + 1}$$

- Planter l'équation dynamique correspondante dans la fonction `My_BF_Saturation` du système en boucle fermée. Simuler le système pour faire apparaître le cycle limite. Attention, dans le cours, le gain  $k$  est placé *après* la saturation à  $\pm a$ ...
- Dans le plan complexe, tracez à la fois le diagramme de Nyquist de

$$G(j\omega), \quad \omega \in [0, 100]$$

ainsi que la trace de

$$-\frac{1}{N(A)}, \quad A \in [0, 25]$$

- Observer les deux courbes et constater qu'elles se coupent en un point (autre que l'origine et l'infini).
- A quoi correspond cette intersection ?
- Calculer la pulsation  $\omega_0$  correspondant à cette intersection à la main (Attention : c'est facile!!!)
- Calculer l'amplitude  $A_0$  telle que  $-\frac{1}{N(A_0)} = G(j\omega_0)$ . (L'équation étant transcendente, faire une recherche numérique ou utiliser le graphique de  $N(A)$  pour déterminer une approximation de la valeur).
- Superposer sur le graphe de la solution temporelle du système, le graphe d'une sinusoïde de pulsation  $\omega_0$  et d'amplitude  $A_0$ .
- Ça marche ? C'est pareil ? Bravo ! *Sugus* : Que pensez-vous de la phase du signal estimé ? ...

## Quelques fonctions Matlab utiles

`help`

: la fonction d'aide de Matlab.

`plot`

: permet de créer une figure à partir d'une ou deux vecteurs pour l'affichage.

`length`

: une fonction qui donne la longueur d'un vecteur.

`size`

: une fonction qui donne la dimension d'une matrice.

`ode45`

: une fonction qui intègre une équation différentielle ordinaire.