FACULTÉ DES SCIENCES APPLIQUÉES
Ecole d'ingénieurs et d'informaticiens

LIÈGE université

Université de Liège

Faculté des Sciences Appliquées

SYST0002

Introduction aux signaux et systèmes

Projet

DELBARRE Loïc (S215072)

JACQUEMIN Justin (S224395)

Bachelier Ingénieur Civil

Année Académique 2024-2025

Table des matières

Dans le cadre de ce rapport , nous utiliseront la notation $\dot{y}$ qui correspond à $\frac{dy}{dt}$.

1 Part 1 : modèle d'état, linéarisation et simulations

1.1 linéarité du modèle dynamique

Le système nest pas linéaire, car il ne respecte pas les principes de superposition et dhomogénéité, en raison de la présence de fonctions non linéaires (sin , tan , arctan) dans sa dynamique ainsi que de l'adition de variable d'état $\theta(t)$ à une fonction d'entrée $\delta(t)$ dans un sinus .

Par conséquent, le système ne satisfait pas les propriétés de superposition et dhomogénéité qui caractérisent les systèmes linéaires.

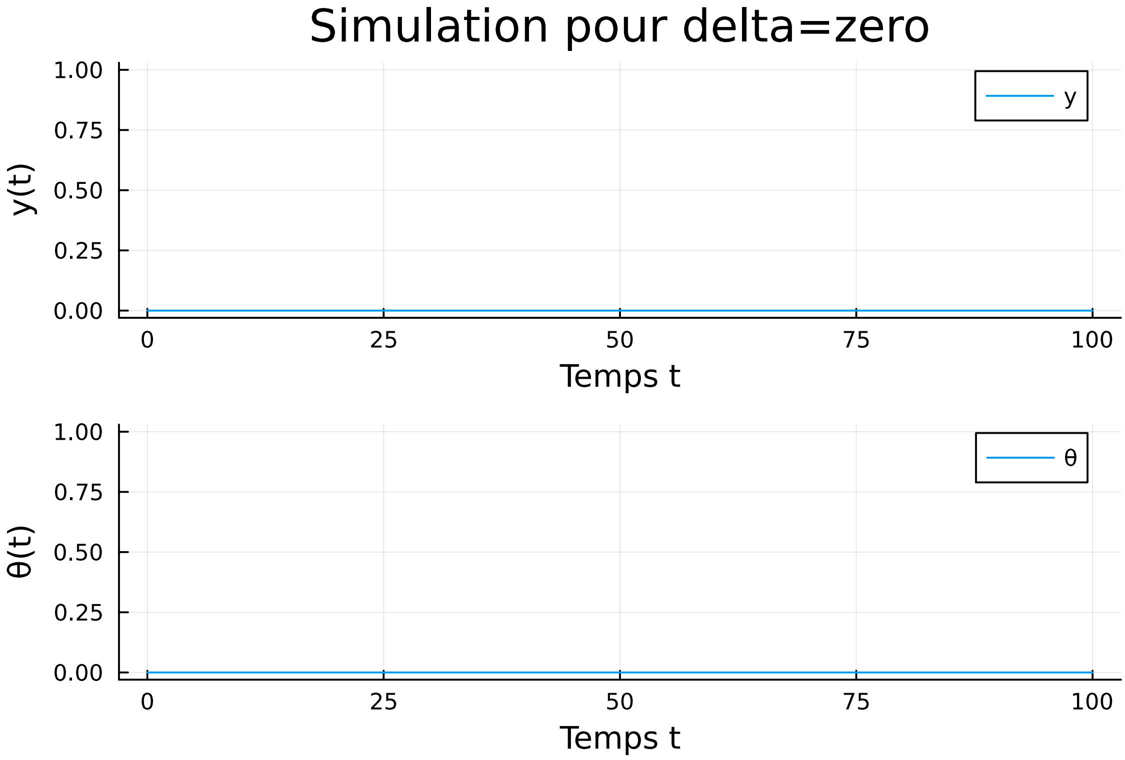
1.2 paramètre du système

Pour tout les signaux mentionnés , nous supposeront que $t \in \mathbb{R}^+$ vu qu'il s'agit du temps.

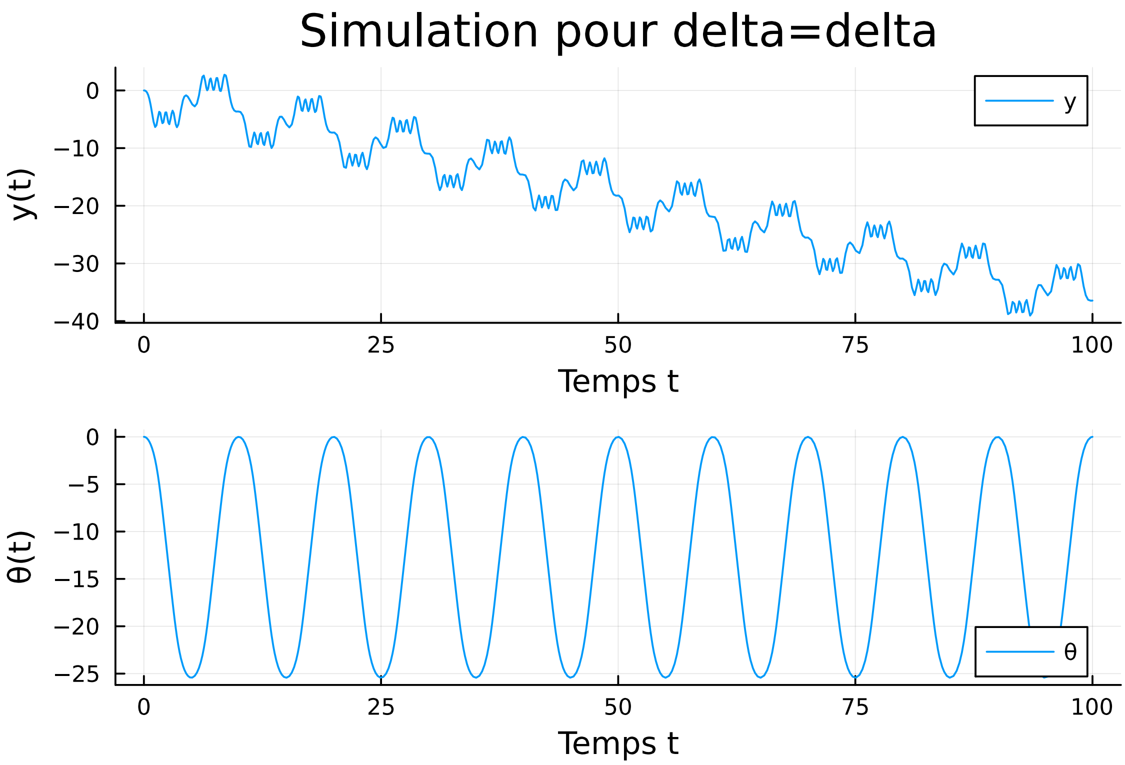
- entrées : $\delta(t)$ Le delta est associé à l'orientation des roue par rapport à l'axe du véhicule en fonction du temps.Son image appartient à l'intervale $[-\frac{\pi}{2}, \frac{\pi}{2}]$
- variables d'états : $y(t), \theta(t)$
  - $y(t)$ représente l'évolution temporelle de la position selon l'axe vertical.Il n'y a pas de contrainte dans l'énoncé, on peut donc supposer que son domaine de définition ainsi que son image appartiennent à $\mathbb{R}$
  - $\theta(t)$ représente l'évolution temporelle de l'orientation du véhicule dans l'espace.Vu qu'il s'agit d'un angle , on peut raisonablement restreindre son image à l'intervale $[0, 2\pi]$
- constantes : a , b , v_0
  - a correspond à la distance entre l'axe arrière et le centre de masse du véhicule.
  - b correspond à la distance entre l'axe des roues avant et celui des roues arrière.
  - v_0 représente la vitesse du centre de masse.
- sorties : $s(t)$ La sortie est la position selon l'axe vetical

1.3 simulation du système

1.3.1 cas d'un mouvement avec les roues fixes



1.3.2 cas d'un mouvement dont les roues oscillent sinusoidalement



1.4 Points d'équilibre du système

Afin de trouver les points d'équilibre du système de manière analytique il suffit de trouver :

$$\begin{cases} \dot{y} = 0 \\ \dot{\theta} = 0 \end{cases} \tag{1}$$

On trouve donc par la deuxième équation que

$$\delta(t) = k\pi \quad \text{avec} \quad k \in \mathbb{Z}$$

De là, en découle via la première équation que

$$\theta(t) = k\pi \quad \text{avec} \quad k \in \mathbb{Z}$$

On trouve donc dans le système deux point d'équilibre au vu des domaines et des images.On a donc $(\delta;\theta) = (0;0)$ ou bien $(\delta;\theta) = (0;\pi)$

1.5 Matrice d'état A,B,C,D

Afin de trouver les matrice d'état du système il faut se baser sur la méthode des dérivées. On identifie aisément que :

$$\begin{cases} f = v_0 \sin(\alpha(\delta(t) + \theta(t)) \\ g = \dfrac{v_0 sin(\alpha(\delta(t))}{a} \end{cases} \tag{2}$$

Les matrices sont donc calculé via les dérivées au point d'équilibre du système :

$$\begin{cases} A = \begin{pmatrix} 0 & \pm v_0 \\ 0 & 0 \end{pmatrix} \\ B = \begin{pmatrix} \dfrac{\pm v_0 a}{b} \\ \dfrac{\pm v_0}{b} \end{pmatrix} \\ C = \begin{pmatrix} 1 & 0 \end{pmatrix} \\ D = \begin{pmatrix} 0 \end{pmatrix} \end{cases} \tag{3}$$

Le système deviens dès lors :

$$\frac{d}{dt}\begin{pmatrix} y(t) \\ \theta(t) \end{pmatrix} = \begin{bmatrix} 0 & \pm v_0 \\ 0 & 0 \end{bmatrix} \begin{pmatrix} y(t) \\ \theta(t) \end{pmatrix} + \begin{bmatrix} \pm v_0 \frac{a}{b} \\ \frac{\pm v_0}{b} \end{bmatrix} \delta(t).$$

$$s(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} y(t) \\ \theta(t) \end{bmatrix} + \begin{bmatrix} 0 \end{bmatrix} \begin{bmatrix} \delta(t) \end{bmatrix}$$

1.6 nature des points d'équilibre

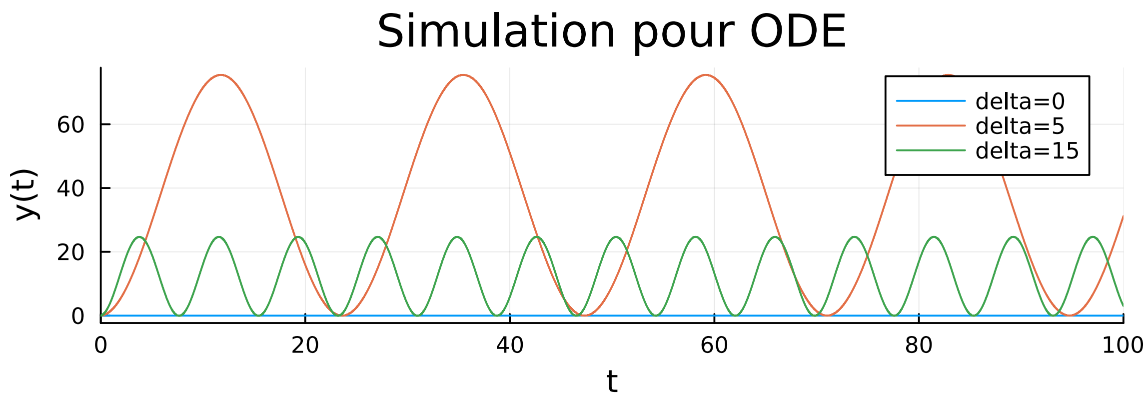
Pour connaitre la nature des points d'équilibre , il suffit de calculer les valeurs propres de la matrice d'état A.

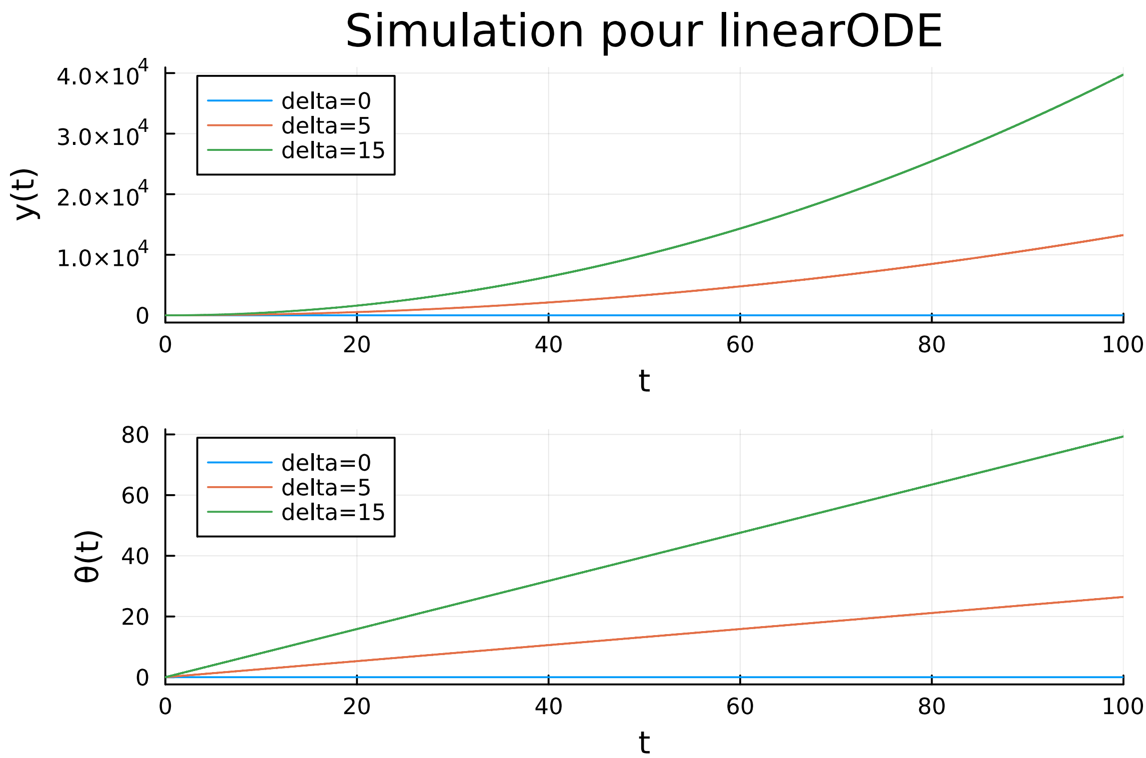
$$\det(A - \lambda I) = \begin{vmatrix} -\lambda & \pm v_0 \\ 0 & -\lambda \end{vmatrix} = (-\lambda)(-\lambda) - (0)(v_0) = \lambda^2$$

on ne peux rien dire sur la nature de tout les points d'équilibre ducoup ?

Vu que ses valeurs sont nulles , on ne peux donc rien conclure concernant la stabilité du système .

marginaleme stable ?





1.7 simulation au point d'équilibre

On peut constater que les fonctions non linéaire ont le même résultat que celle du modèle ABCD. Vu que l'on fixe δ a une constante qui ne dépends pas du temps , Vu que δ étant constant, $\dot{\theta}$ l'est aussi et donc θ croit linéairement de pente .
Concernant $y(t)$ la situation est différente, on a une relation qui dépends de θ ainsi que d'une constante. On a donc un comportement "exponentiel". la valeur de $\dot{\theta}$ n'est pas modifié au fur

et a mesure du temps et donc pour $\theta > 0$ on appercois une augmentation de y(t) constante qui est représenté sur la courbe par la courbure exponentielle.

1.8 trajectoire gaussienne

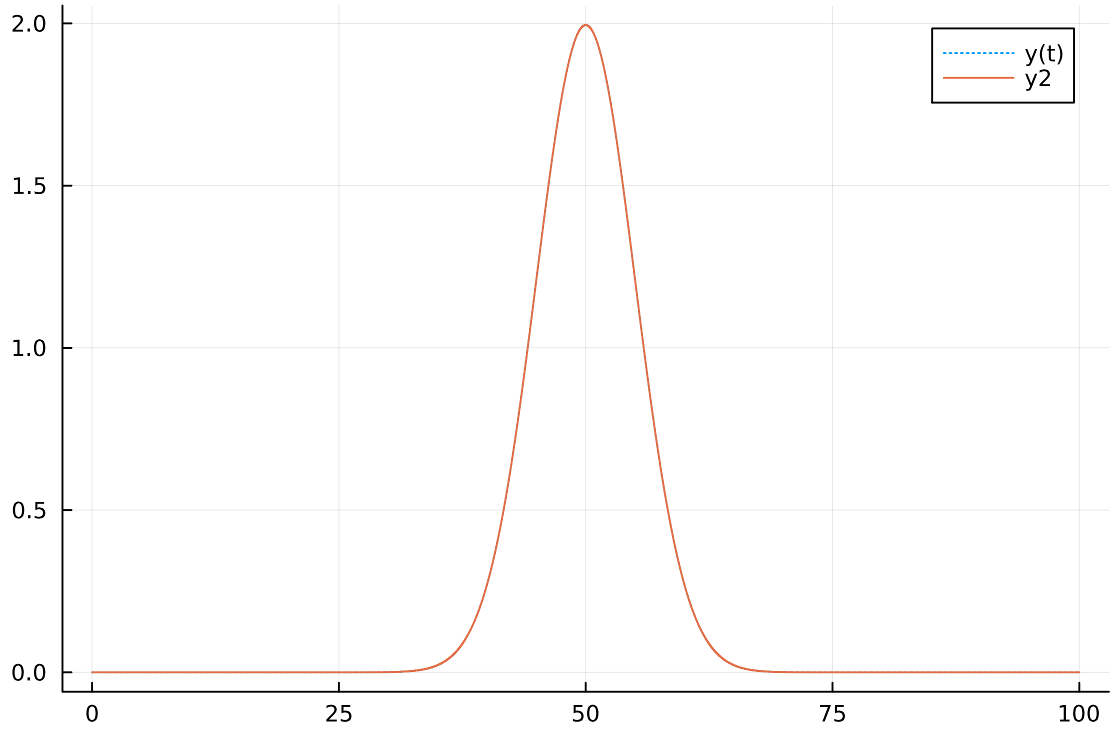


Figure 1

La solution la plus cohérente pour cette question est

$$\sum_{i=0}^{3} \frac{A_i}{\sqrt{2\pi\mu_i^2}} \exp\left(-\frac{(t-\sigma_i)^2}{2\mu_i^2}\right)$$

avec comme coeficient :

$$\begin{cases} \mu_1 = 44.47466746957489 \\ \mu_2 = 50.10972389695686 \\ \mu_3 = 55.7445931340356 \\ \sigma_1 = \sigma_3 = 4.504552894450979 \\ A_1 = A_3 = 0.023696291835675492 \\ A_2 = -0.047392683335653336 \end{cases} \tag{4}$$

avec actuellement une MSE de 8.666064583890183e-9

1.8.1 méthode développé pour aboutir à cette solution

Nous avons exploré beaucoup de pistes différentes pour trouver cette valeur.

Premièrement nous avons essayer de voir si, en injectant la somme de gausienne dans le modèle linéaire il était possible de trouver par analogie , le nombre de terme mais aussi d'éventuelles symétrie. Par ce calcul fastidieux , nous nous sommes rendu compte qu'il était possible de décrire δ comme une somme de deux gausienne.

renotter le calcul ?

Secondement , la piste numérique à été envisagé.Une approche naive a été de supposer qu'il pouvais s'agir d'un problème de minimisation de la mean square à 3n dimension.Or cette solution s'est révélé largement ineficace au vu du nombre de minimum local présent.La méthode de BassinHopping s'est révélé trop fastidieuse à mettre en place.

Il a donc été question d'envisager les symétrie du problème afin de réduire un maximum le nombre de parametre. Comme sugéré par l'assistant du cours , nous avons considéré qu'une gaussienne comme entrée correspondrais à un coup de volant aller retour. Ce coup de volant aurrait comme effet sur le véhicule de changer son orientation et donc la pente de notre sortie, y .La gausienne de sortie que nous devont considérer peux donc se voir comme une série de trois coup de volant dont deux positif de même intensité et un négatif.Au vu des propriété des gausienne, celle ci est également symétrique en μ.De part toutes ses symétrie, cela a permis de réduire notre problème à 7 paramètre $(\Delta\mu, \sigma_1, \sigma_2, A_1, A_2)$

Une multitude de méthode numérique ont été exploré pour la détermination de ces constantes.Nous avons décidé de changer notre code python vers un code julia pour plus d'optimisation, de challenge et d'amusement (car aucun d'entre nous n'as jamais codé en Julia).

Nous avons d'habord considéré l'approche `grid_search` conseillé par l'assistant.Nous nous sommes d'habord heurté à un problème d'espace à considérer. En effet si l'on considère ne serait ce que 10 points par dimension cela reviens à 10^n.Pour palier à ca , nous avons essayé d'utiliser directement le GPU pour paraléliser les différentes taches via `Cuda.jl`.Ceci a été un échec total , la tache demandé (calculer la mse avec une combinaison de paramêtre donné) était trop complexe et ralentisait l'exécution du programme.Nous nous sommes donc rabatu avec dépit sur le CPU et avons parallélisé le code au niveau des threads.

Un des désavantages majeurs de la méthode `grid_search` est que celle ci ne survole l'intervale que par voisinage au fur et a mesure des itérations.Une autre méthode peut être plus saugrenue nous a été de jouer au dés via `random_search`, en effet si nous considérons une probabilité uniforme sur l'intervale , notre convergence serait plus attrayante , car celle ci "mappera" l'intervale plus rapidement.

Cependant, la pratique nous a démontré que l'approximation des position $\Delta mu, \mu_2 = 50$ n'était pas parfaite au dela de l'odre de 1e-4.Pour palier à cela , nous n'avons pas conservé cette approximation pour déterminer la valeurs.Pour de grandes précision , nous pouvons à nouveau reconsidérer les minimisation car l'intervale est de taille similaire à celle du puit du minimum.

formuler ca correctement

Afin de trouver la juste somme de gausiennes pour $\delta(t)$, différentes pistes s'offrent à nous.

- La méthode analytique n'est pas envisageable au vu de la complexité du système et la méthode analytique sur le modèle linéarisé n'est pas suffisament proche de la réalité pour avoir du sens.
-

Concernant la méthode numérique une approche naive serait de poser le problème comme étant une minimisation de la mean square error entre notre signal de sortie et notre signal d'entrée en fonction des parametres des gaussiennes. Le problème principal de cette approche est , que la fonction comporte beaucoup trop de minimum locaux sur sa topologie mais également que le nombre de parametre pouvant croitre , la minimisation serait d'autant plus couteuse. Pour palier à ce problème la première solution envisagée a été de bénéficier des symétrie du problèmes ainsi que des propriété des gaussiennes. En effet on peut voir une gausienne comme étant un "coup de volant aller-retour",on constate donc qu'il y aurra besoin de 3 gaussiennes dont le parametre μ déterminerai le temps a la quelle celui ci serait effectué et A la violence de celui ci.

Pour réduire encore le nombre de paramêtre à 7, une des approche envisagée a été de supposer que deux coups de volant serait équidistant de 50 et l'un a 50. Concernant l'algorithme , les deux algoritme simple a implémenter sont la recherche aléatoire et la grid search. Au vu du nombre de dimension du problème une grid search aurrait pris beaucoup de temps avant de converger. C'est pour cela que nous sommes parti sur une recherche aléatoire.

1.8.2 résolution numérique

On peut supposer le problème comme étant la minimization d'une fonction prennant en argument une matrice colone de taille 3n représentant les différents paramètre associé au différentes Gaussiennes et retournant le Mean square error comparé a la trajectoire attendue . Actuellement nous avons fait tourner sans succès notre minimization pour des tailles de matrice allant jusqu'a 15 élément.

1.8.3 recherche de paramètre

Une première approche bête et naive serait de tester des combinaisons de n gausiennes et de faire une minimisation sur 3n parametre. Au vu du problème considéré une méthode de minimisation basée sur une dérivée ne fonctionnerai pas car la fonction présente beaucoup trop de minimum locaux.

Si l'on considère la fonction trajectoire que le véhicule doit avoir, on y remarque 3 " coup de volant". Un coup de volant peut se traduire par une fonction $\delta(t)$ similaire a une gausienne ou une parabole et dont le sens de la courbure indiquerai le sens du coup de volant. Afin de simplifier encore le nombre de variable , on peut profiter des propriété des parametre d'une gaussienne, dans les 3 coups de volant , l'un semble centré en 50 et le deux autres semblent être équidistant à 50.Il en va de soit aussi dans l'intensité du coup de v

2 State-feedback controller, simulations et analyse de Fourier

2.1 paramètre du système

refaire la meme chose que la P1Q1

2.2 calcul des matrices d'état du système global

En remplacant l'équation du controller dans celle du Plant , on obtient :

$$\begin{cases} \dot{x} = (A - KB)x + BK_r r \\ s = (C - KD)x + DK_r r \end{cases} \tag{5}$$

Par identification,on remarque donc que :

$$\begin{cases} \tilde{A} = A - BK = \begin{bmatrix} -k_1 V_0 \frac{a}{b} & v_0 - k_2 v_0 \frac{a}{b} \\ -k_1 \frac{v_0}{b} & -k_2 \frac{v_0}{b} \end{bmatrix} \\ \tilde{B} = K_r B = \begin{bmatrix} k_r v_0 \frac{a}{b} \\ k_r \frac{v_0}{b} \end{bmatrix} \\ \tilde{C} = C - DK = \begin{bmatrix} 1 & 0 \end{bmatrix} \\ \tilde{D} = K_r D = \begin{bmatrix} 0 \end{bmatrix} \end{cases} \tag{6}$$

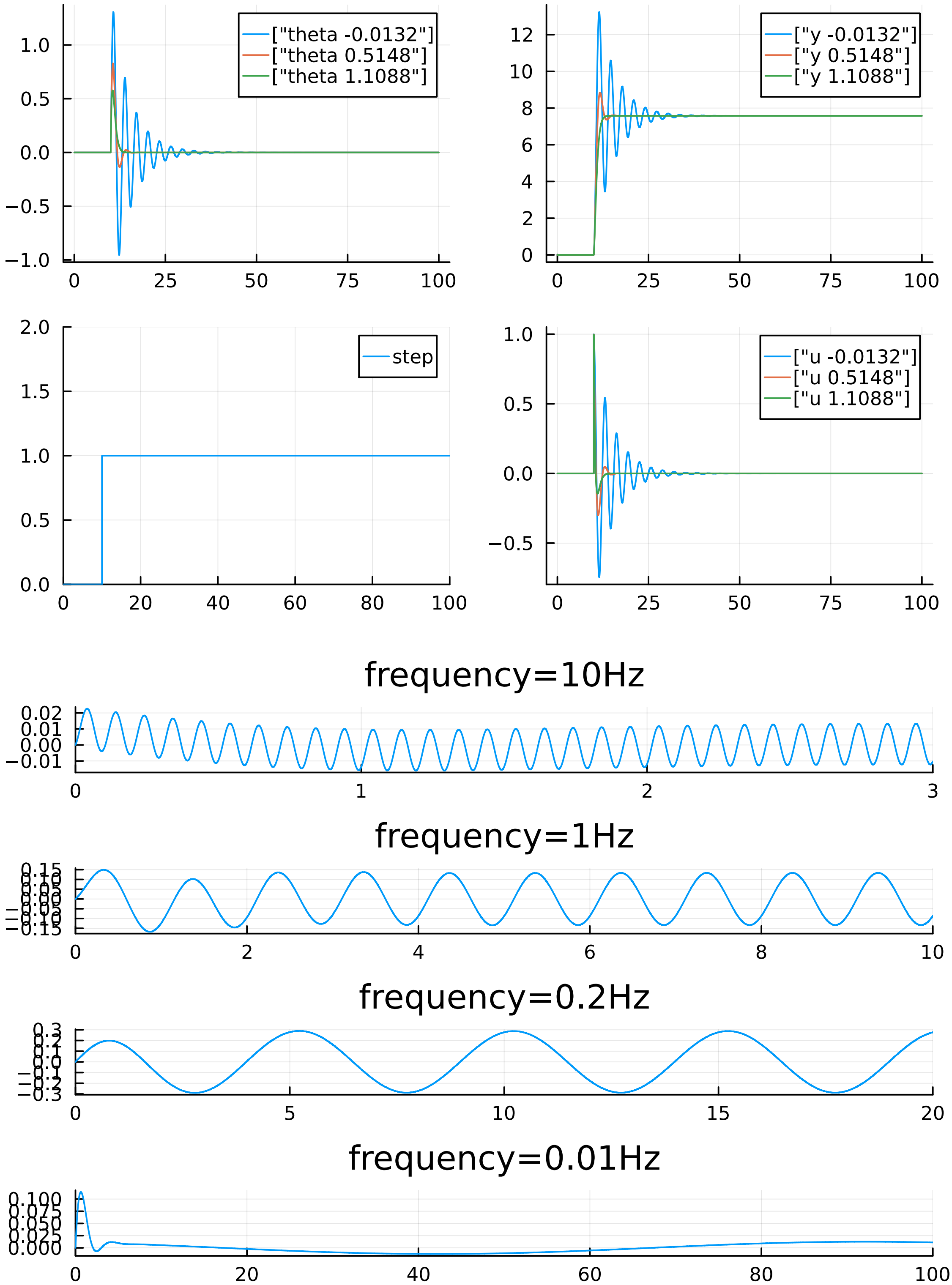

["theta -0.0132"]
["theta 0.5148"]
["theta 1.1088"]
["y -0.0132"]
["y 0.5148"]
["y 1.1088"]
step
["u -0.0132"]
["u 0.5148"]
["u 1.1088"]
frequency=10Hz
frequency=1Hz
frequency=0.2Hz
frequency=0.01Hz

FIGURE 2

2.3 simulation du modèle *control loop*

2.4 simulation avec une entrée sinusoidales

2.5 domaine fréquentiel

Afin de se simplifier les notation , il suffit de définir un ω comme étant $2\pi f_{ref}$ avec f_{ref} la fréquence de référence . Le déphasage de $\frac{\pi}{6}$ sera noté par la lettre ϕ.On prendra également l'intervale comme étant un intervale allant de 0 a T=100

$2sin(2\pi ft + \frac{\pi}{6})$

$$\begin{aligned}
X(j\omega) =& 2\int_{0}^{100} sin(\omega_0 t + \phi)e^{-j\omega t}dt \\
=& 2\int_{\frac{T}{2}-\frac{T}{2}}^{\frac{T}{2}+\frac{T}{2}} \frac{e^{j\omega_0 t} - e^{-j\omega_0 t}}{2j} e^{-j\omega t}dt \\
=& \frac{1}{j}\int_{\frac{T}{2}-\frac{T}{2}}^{\frac{T}{2}+\frac{T}{2}} e^{j(\omega_0-\omega)t+j\phi} - e^{-j(\omega_0+\omega)t-j\phi}dt \\
=& \frac{1}{j}\left[\frac{e^{j(\omega_0-\omega)t+j\phi}}{j(\omega_0-\omega)} - \frac{e^{-j(\omega_0+\omega)t-j\phi}}{j(\omega_0+\omega)}\right]_{\frac{T}{2}-\frac{T}{2}}^{\frac{T}{2}+\frac{T}{2}} \\
=& \frac{e^{j\left((\omega_0-\omega)\frac{T}{2}+\phi\right)}}{j^2(\omega_0-\omega)}\left(e^{j(\omega_0-\omega)\frac{T}{2}} - e^{-j(\omega_0-\omega)\frac{T}{2}}\right) + \frac{e^{-j\left((\omega_0+\omega)\frac{T}{2}+\phi\right)}}{j^2(\omega_0+\omega)}\left(e^{j(\omega_0+\omega)\frac{T}{2}} - e^{-j(\omega_0+\omega)\frac{T}{2}}\right) \\
=& \frac{2e^{j\left((\omega_0-\omega)\frac{T}{2}+\phi\right)}}{j(\omega_0-\omega)} sin\left((\omega_0-\omega)\frac{T}{2}\right) + \frac{2e^{-j\left((\omega_0+\omega)\frac{T}{2}+\phi\right)}}{j(\omega_0+\omega)} sin\left((\omega_0+\omega)\frac{T}{2}\right) \\
=& -2je^{j\left((\omega_0-\omega)\frac{T}{2}+\phi\right)} sinc\left((\omega_0-\omega)\frac{T}{2}\right) - 2je^{-j\left((\omega_0+\omega)\frac{T}{2}+\phi\right)} sinc\left((\omega_0+\omega)\frac{T}{2}\right)
\end{aligned}$$

Au vu des sinus cardinaux , les peak se trouvent donc en $\omega = \pm\omega_0$.Autrement dit, $\omega = \pm 2\pi f_{ref}$

2.6 diagrammes d'amplitudes

Au vu du signal de sortie que l'on obtient dans la partie 4 , il est assez visuel de voir qu'il s'agit d'une sinusoide d'amplitude et de fréquence visible. Ici pour tracer numériquement on aurrait pu le faire a la main. Nous avons décidé de se baser sur la Discrete Fourier Transform (DFT) pour récupérer le graphique de fréquence.

3 Fonction de transfert et diagrammes de Bode

3.1 fonction de transfert du système total

Pour déterminer $H(s)$, on peut se baser sur les matrices d'état $\tilde{A}, \tilde{B}, \tilde{C}, \tilde{D}$,ceci permettra de déterminer les pôles de H(s).Concernant sa stabilité , une fonction de transfert est dite stable si Re$\{p\} < 0$on a donc :

$$H(s) = \tilde{C}\left(s\mathbb{I}_2 - \tilde{A}\right)^{-1}\tilde{B} + \tilde{D}$$

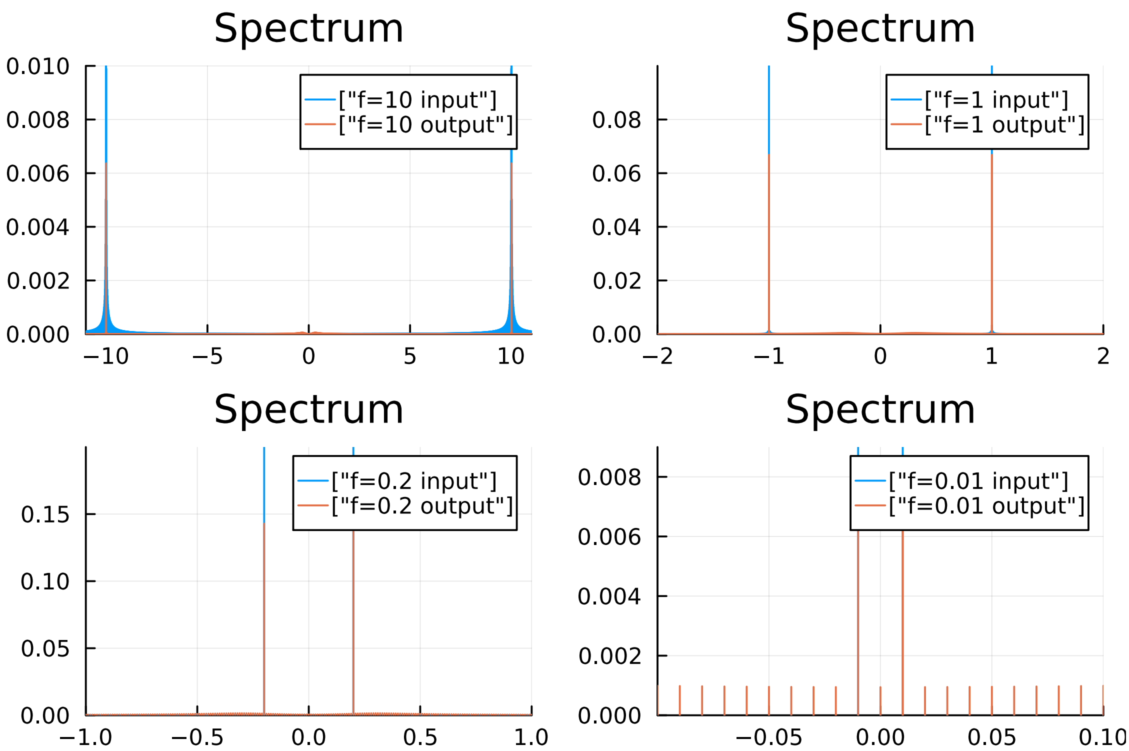


FIGURE 3

$$H(s) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} k_1 v_0 \frac{a}{b} + s & k_2 v_0 \frac{a}{b} - v_0 \\ k_1 \frac{v_0}{b} & k_2 \frac{v_0}{b} + s \end{bmatrix}^{-1} \begin{bmatrix} k_r v_0 \frac{a}{b} \\ k_r \frac{v_0}{b} \end{bmatrix} + \begin{bmatrix} 0 \end{bmatrix}$$
$$= \frac{1}{s^2 + \frac{v_0}{b}[ak_1 + k_2]s + \frac{k_1 v_0^2}{b}} \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} k_2 \frac{v_0}{b} + s & v_0 - k_2 v_0 \frac{a}{b} \\ -k_1 \frac{v_0}{b} & k_1 v_0 \frac{a}{b} + s \end{bmatrix} \begin{bmatrix} k_r v_0 \frac{a}{b} \\ k_r \frac{v_0}{b} \end{bmatrix}$$
$$= \frac{k_r v_0}{b} \frac{as + v_0}{s^2 + \frac{v_0}{b}[ak_1 + k_2]s + \frac{k_1 v_0^2}{b}}$$

en remplacant par les valeurs on obtient ;

$$= \frac{0.440s + 4}{s^2 + 2s + 4}$$

Au vu de l'alure de $H(s)$ il est aisé de remarqué son zéros en $\frac{-b}{ak_r}$ Cette fonction de control est également trouvé directement dans le code.

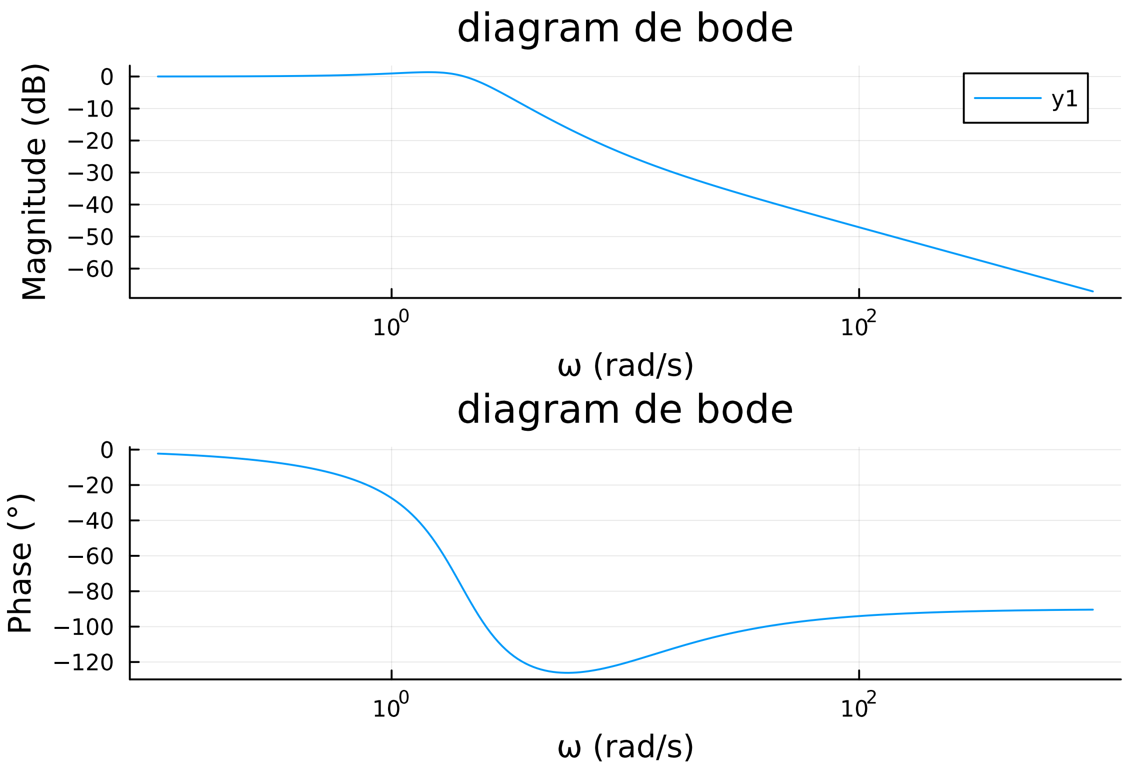


diagram de bode
Magnitude (dB)
0
−10
−20
−30
−40
−50
−60
y1
10^0
10^2
ω (rad/s)
diagram de bode
Phase (°)
0
−20
−40
−60
−80
−100
−120
10^0
10^2
ω (rad/s)

FIGURE 4