ENSTA PARIS

AUT202 – AUTOMATIQUE : DYNAMIQUE ET CONTROLE DES SYSTEMES

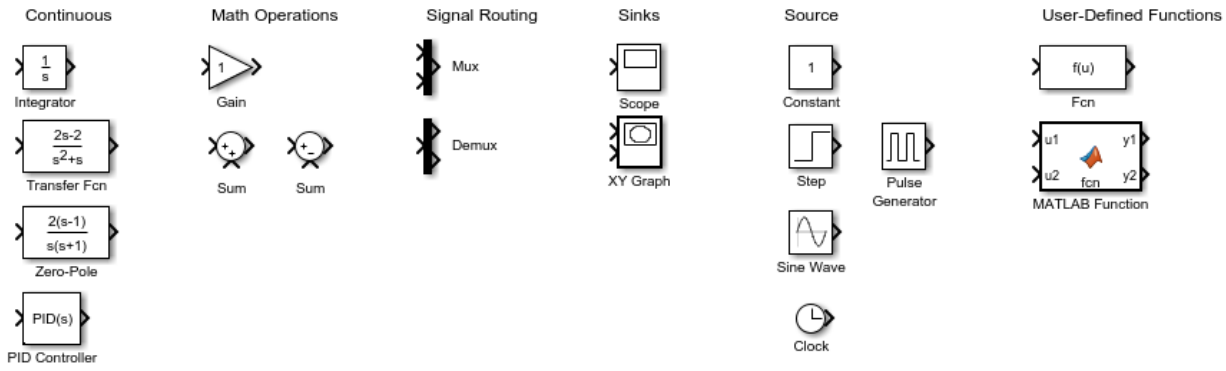
TRAVAUX PRATIQUES MATLAB N°1

1ER FEVRIER 2023

INTRODUCTION

MATLAB est un logiciel scientifique spécialisé dans le calcul numérique. Il s'utilise comme un logiciel interactif ou comme un langage de programmation. La documentation se consulte à l'aide de **help** suivi du nom de la fonction utilisée.

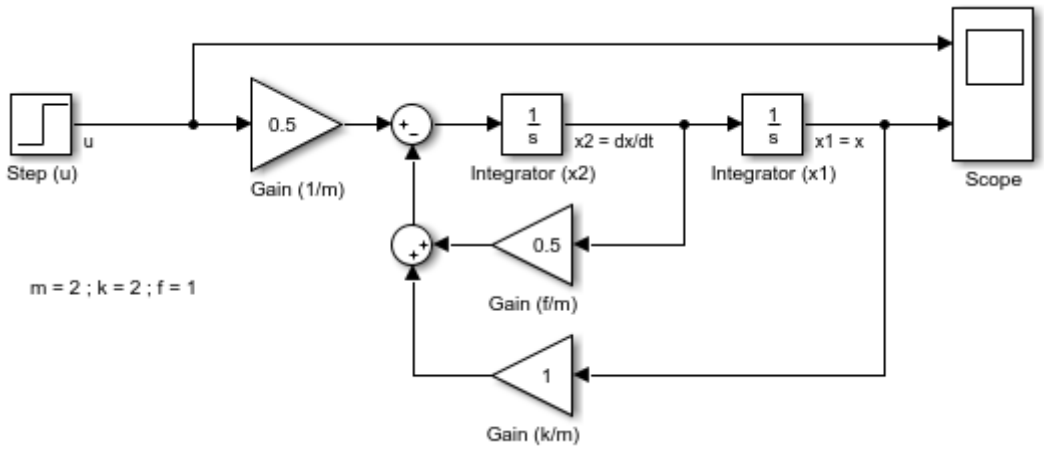
Le module Simulink de MATLAB permet de simuler le comportement de systèmes dynamiques représentés sous forme de schéma bloc. L'interface est constituée d'une bibliothèque de blocs génériques et d'une feuille de travail sur laquelle l'utilisateur assemble les blocs génériques dont il a besoin pour représenter son système dynamique. Quelques blocs utiles sont listés ci-dessous :



Prenons l'exemple d'un oscillateur amorti dont la dynamique s'écrit :

$$m.\frac{d^2}{dt^2}x(t) = -k.x(t) - f.\frac{d}{dt}x(t) + u(t)$$

On peut modéliser ce système dans le module Simulink de MATLAB à l'aide des blocs de base **Integrator**, **Gain** et **Sum** de la manière suivante :

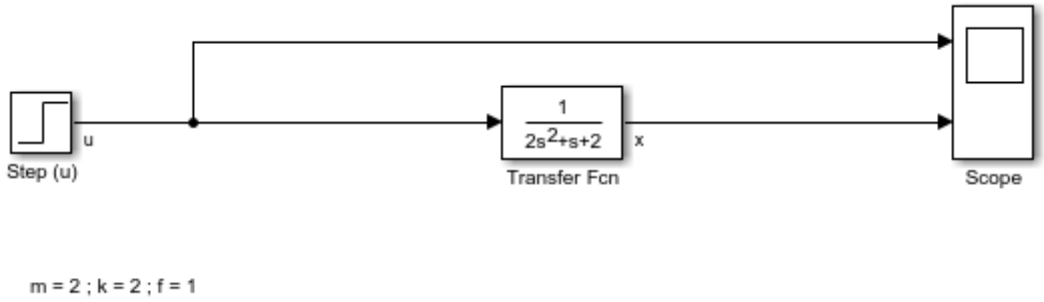


Le bloc **Step** permet de générer une consigne u(t) en échelon. Le bloc **Scope** permet de visualiser l'évolution de variables d'intérêt.

L'oscillateur amorti étant un système dynamique linéaire, on peut aussi le représenter à l'aide d'une fonction de transfert :

$$m.\frac{d^2}{dt^2}x(t) = -k.x(t) - f.\frac{d}{dt}x(t) + u(t) \quad \Leftrightarrow \quad H(s) = \frac{X(s)}{U(s)} = \frac{1}{m.s^2 + f.s + k}$$

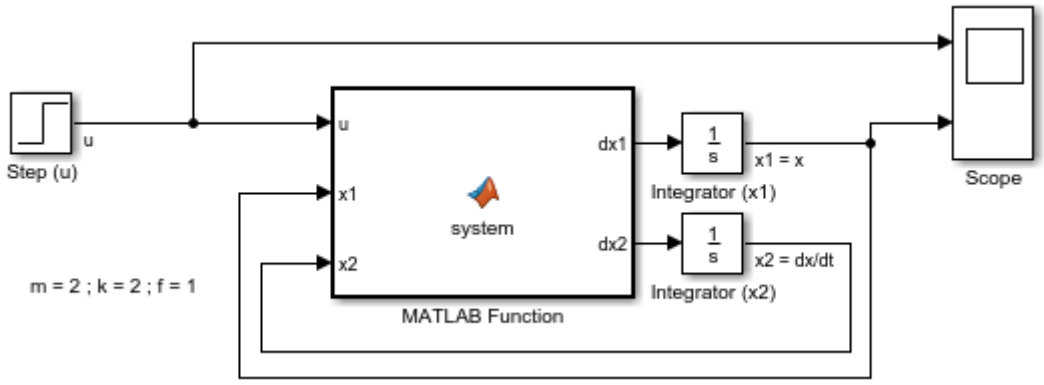
On peut modéliser ce système dans le module Simulink de MATLAB à partir de sa fonction de transfert à l'aide du bloc **Transfer Fcn** de la manière suivante :



L'oscillateur amorti peut également être représenté sous forme d'état :

$$\begin{cases} x_1(t) = x(t) \\ x_2(t) = \dfrac{d}{dt}x(t) \end{cases} \quad \begin{cases} \dfrac{d}{dt}x_1(t) = x_2(t) \\ \dfrac{d}{dt}x_2(t) = -\dfrac{k}{m}x_1(t) - \dfrac{f}{m}x_2(t) + \dfrac{1}{m}.u(t) \end{cases}$$

On peut modéliser ce système dans le module Simulink de MATLAB à partir de sa forme d'état à l'aide des blocs **Integrator** et **MATLAB Function** de la manière suivante :



La fonction d'état est codée dans le bloc **MATLAB Function** :

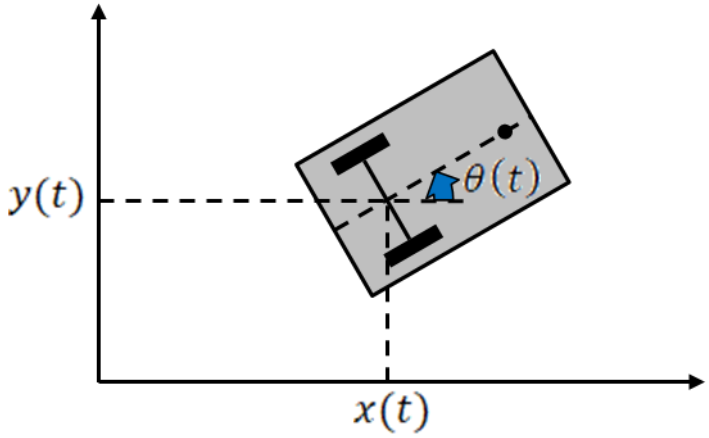
```
function [dx1, dx2] = system(u, x1, x2)

m = 2;
k = 2;
f = 1;

dx1 = x2;
dx2 = -k/m*x1 - f/m*x2 + 1/m*u;
```

EXERCICE 1 – Robot à roue

On considère un robot mobile (tel qu'utilisé dans de nombreux concours de robotique). Ce robot admet sur un même essieu deux roues motorisées de manière indépendante. On note :

- $(x(t), y(t))$ les coordonnées cartésiennes du milieu de l'essieu ;
- $\theta(t)$l'angle du robot avec un axe fixe ;
- $\Omega_d(t)$ et $\Omega_g(t)$ les vitesses de rotation des roues droite et gauche.

On suppose que les roues roulent sans glisser ni déraper.



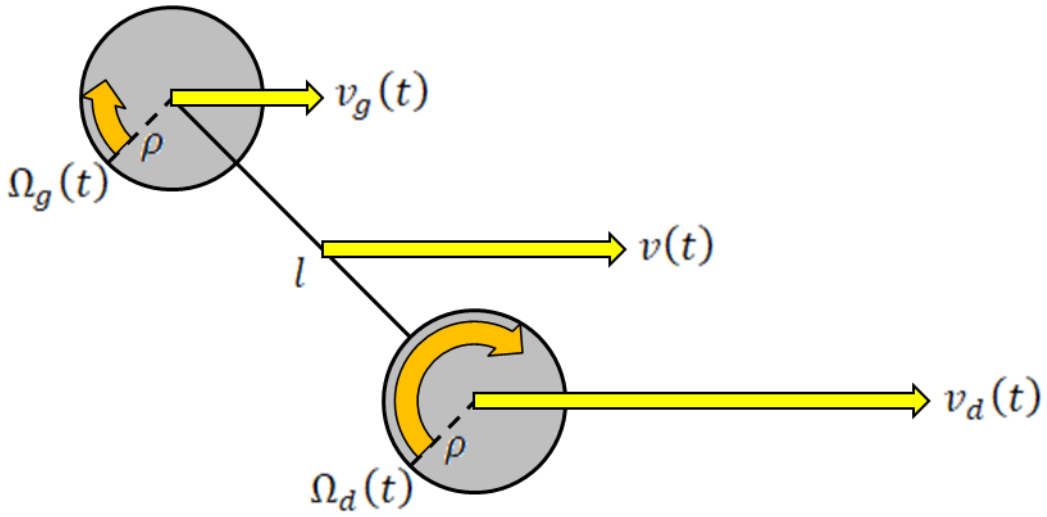
Q1/ Montrer que la dynamique est :

$$\begin{cases} \dfrac{d}{dt}x(t) = v(t).\cos(\theta(t)) \\ \dfrac{d}{dt}y(t) = v(t).\sin(\theta(t)) \\ \dfrac{d}{dt}\theta(t) = \omega(t) \end{cases}$$

où $(x(t), y(t), \theta(t))$ est l'état et $u(t) = (v(t), \omega(t))$ est la commande que l'on relira à $\Omega_d(t)$ et $\Omega_g(t)$, ρ le rayon des roues et l la distance entre les roues.

En considérant $v(t)$ la vitesse du milieu de l'essieu et $\omega(t)$ la vitesse de rotation du robot, on retrouve immédiatement l'équation du mouvement.

Pour exprimer $v(t)$ et $\omega(t)$ en fonction de $\Omega_d(t)$ et $\Omega_g(t)$, il faut utiliser les relations cinématiques du solide indéformable.



$v_d(t)$ et $v_g(t)$ les vitesses de déplacement des roues au niveau des essieux droit et gauche sont données par transport de la vitesse du milieu de l'essieu :

$$\begin{cases} v_d(t) = v(t) + \frac{l}{2}.\omega(t) \\ v_g(t) = v(t) - \frac{l}{2}.\omega(t) \end{cases}$$

$v_d(t)$ et $v_g(t)$ sont par ailleurs reliées aux vitesses de rotation des roues droite et gauche :

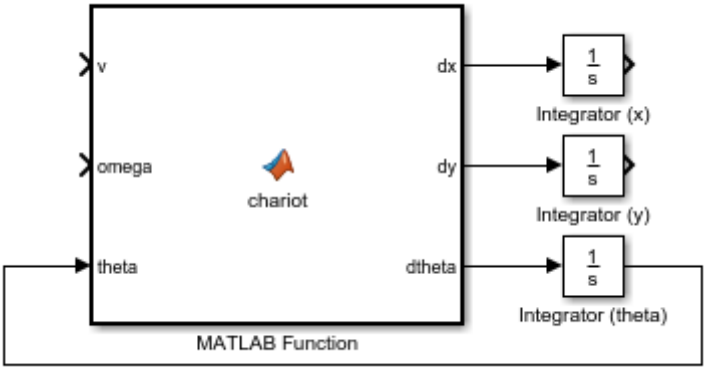
$$\begin{cases} v_d(t) = \rho.\Omega_d(t) \\ v_g(t) = \rho.\Omega_g(t) \end{cases}$$

En combinant ces deux équations, on trouve :

$$\begin{cases} v(t) = \frac{\rho}{2}.\left(\Omega_d(t) + \Omega_g(t)\right) \\ \omega(t) = \frac{\rho}{l}.\left(\Omega_d(t) - \Omega_g(t)\right) \end{cases}$$

Q2/ Modéliser dans le module Simulink de MATLAB le robot à roue avec comme commande $(\boldsymbol{v(t), \omega(t)})$ et comme sortie l'état $(\boldsymbol{x(t), y(t), \theta(t)})$.

En utilisant un bloc **MATLAB Function** et un bloc **Integrator** pour chaque état, on obtient le schéma bloc suivant :



La fonction d'état est codée dans le bloc **MATLAB Function** :

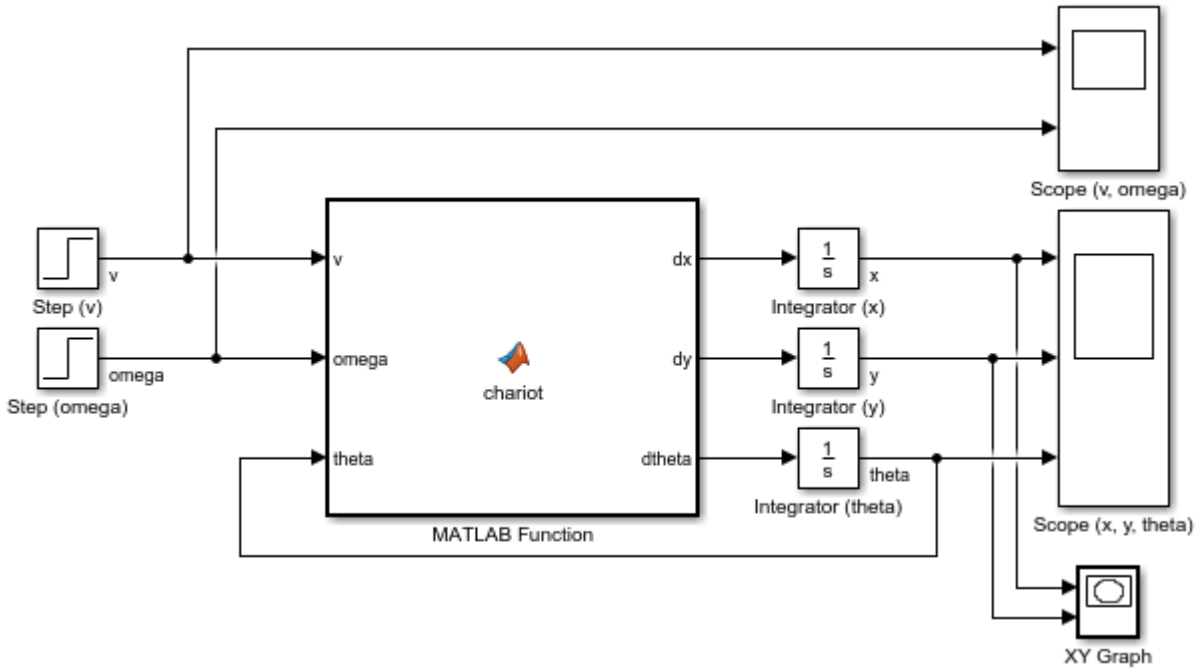
```
function [dx, dy, dtheta] = chariot(v, omega, theta)


dx = v*cos(theta);
dy = v*sin(theta);
dtheta = omega;
```

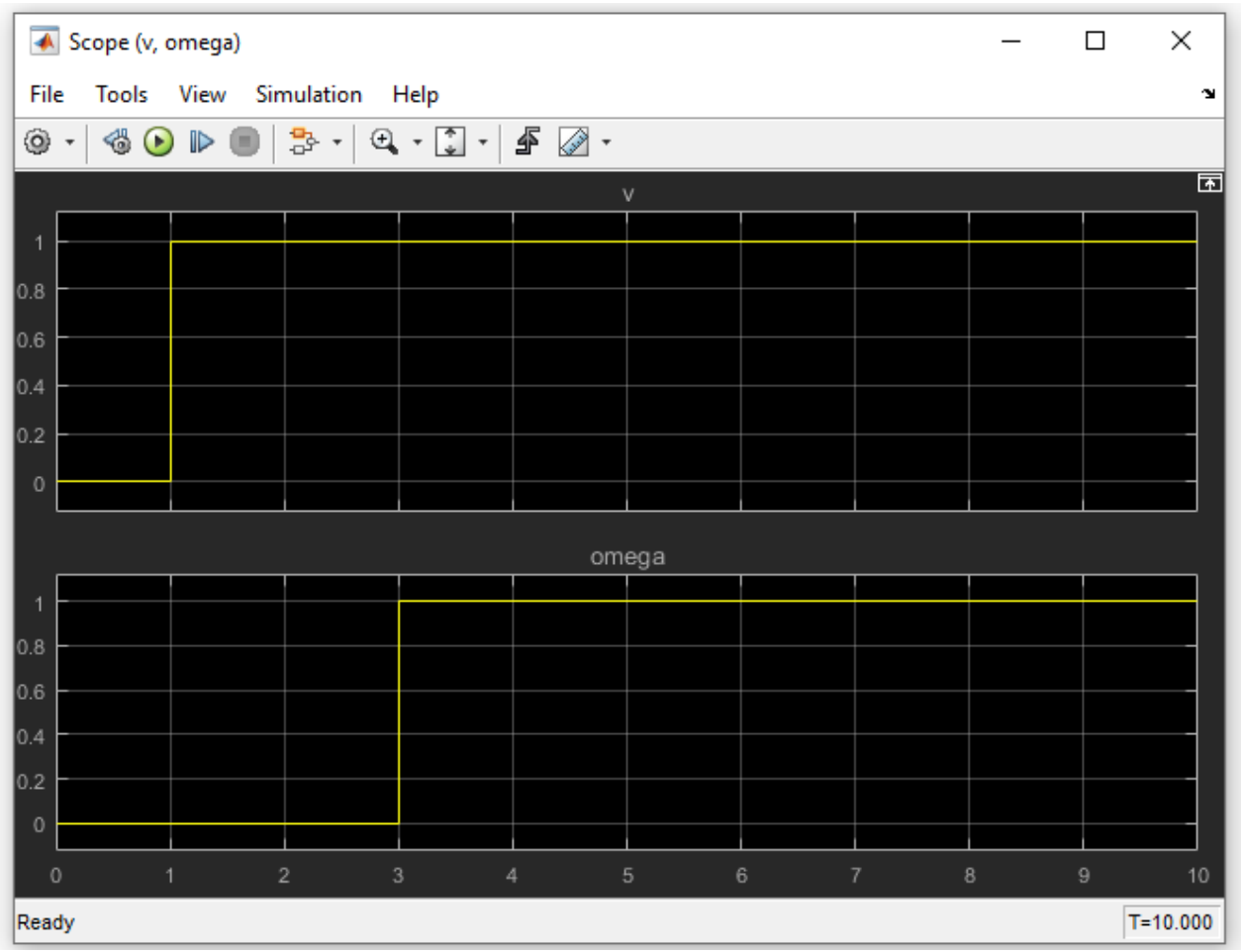
Q3/ Simuler la trajectoire du chariot pendant 10s avec un état initial nul et la commande suivante :

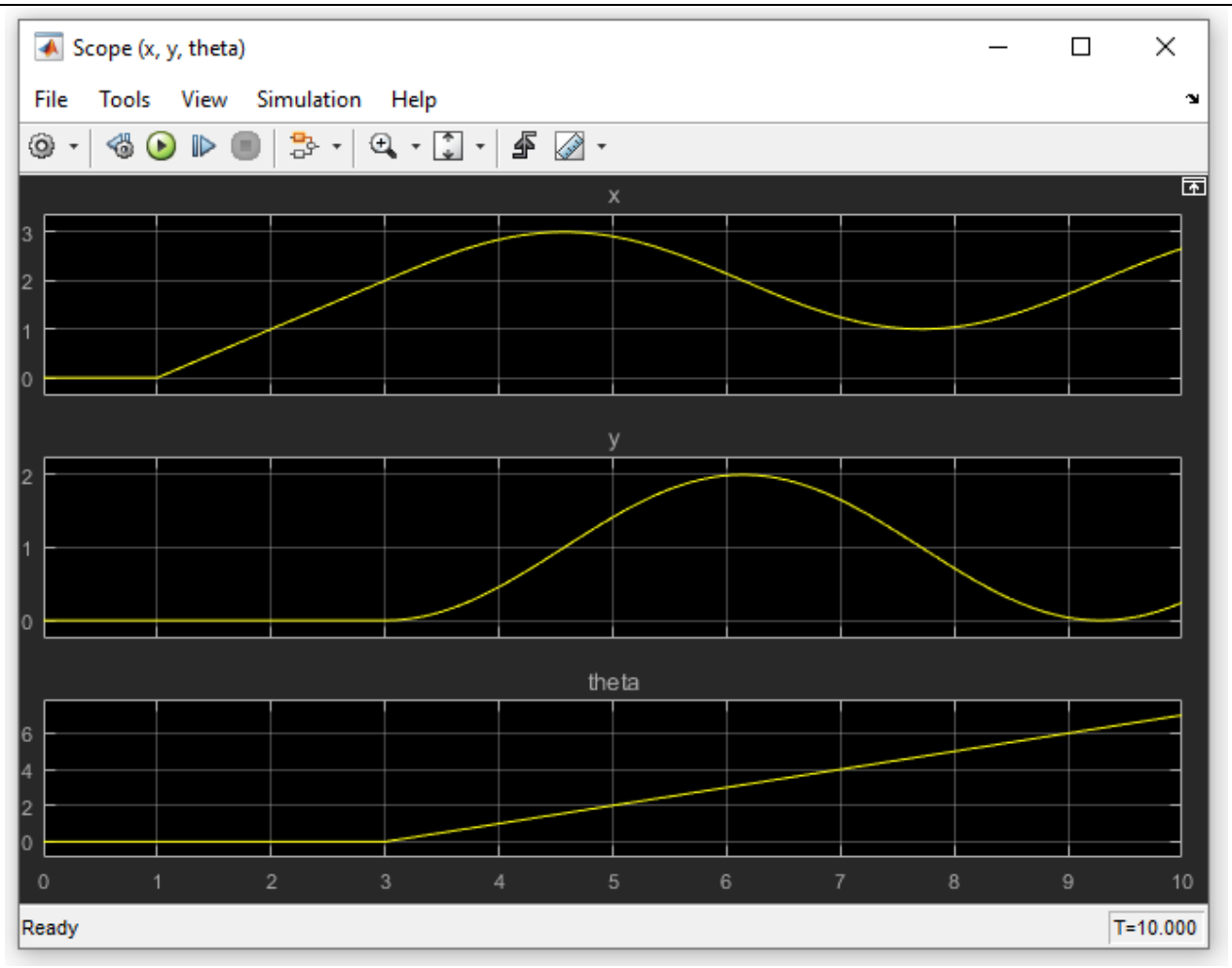
$$\boldsymbol{v(t)} = \begin{cases} \mathbf{1} & \textbf{si } \boldsymbol{t > 1s} \\ \mathbf{0} & \textbf{sinon} \end{cases} \quad \boldsymbol{\omega(t)} = \begin{cases} \mathbf{1} & \textbf{si } \boldsymbol{t > 3s} \\ \mathbf{0} & \textbf{sinon} \end{cases}$$

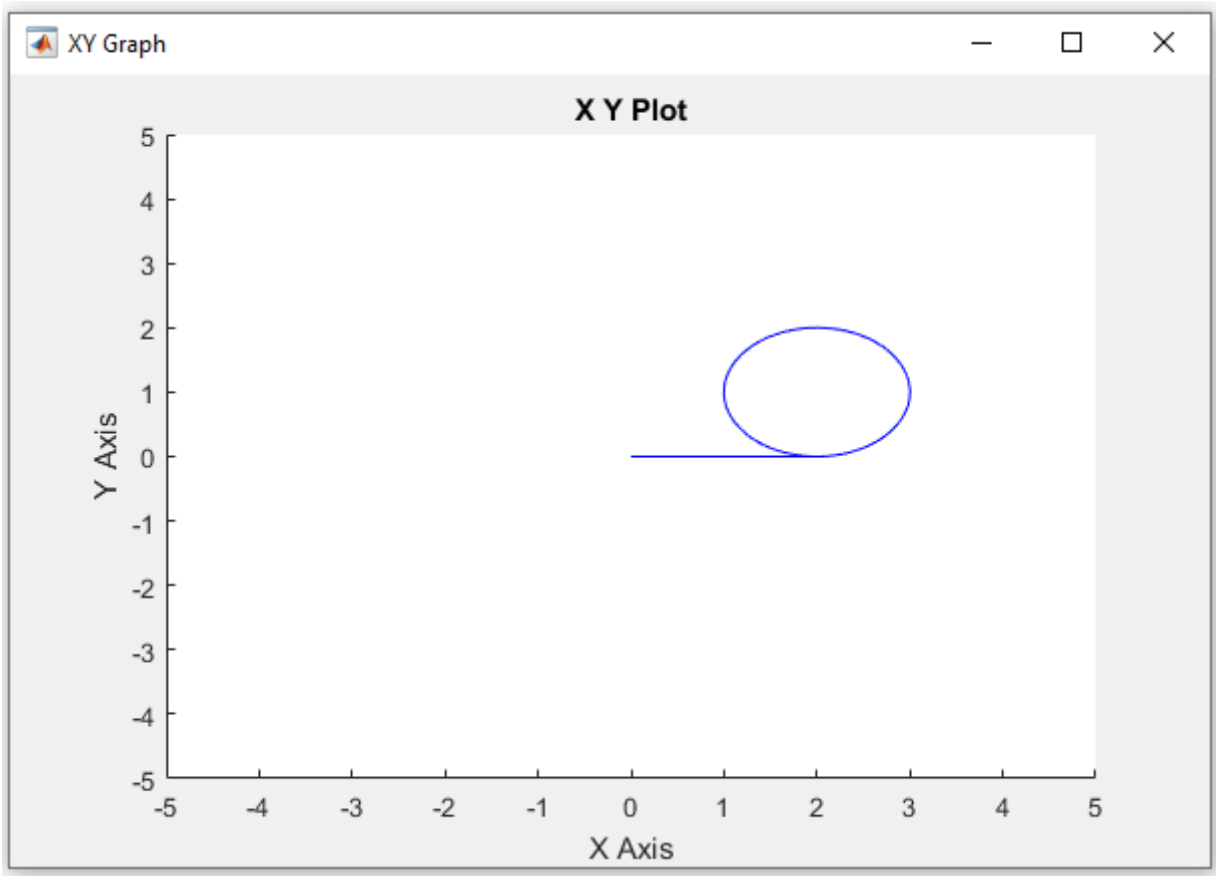
On pourra utiliser le bloc Scope XY Graph pour visualiser la trajectoire dans le plan.

On complète le schéma bloc avec des blocs **Step** pour les commandes, des blocs **Scope** pour visualiser l'évolution des commandes et des états et un bloc **XY Graph** pour visualiser la trajectoire.


Scope (v, omega)
Step (v)
v
Step (omega)
omega
v
omega
theta
chariot
dx
dy
dtheta
MATLAB Function
1
s
x
Integrator (x)
1
s
y
Integrator (y)
1
s
theta
Integrator (theta)
Scope (x, y, theta)
XY Graph

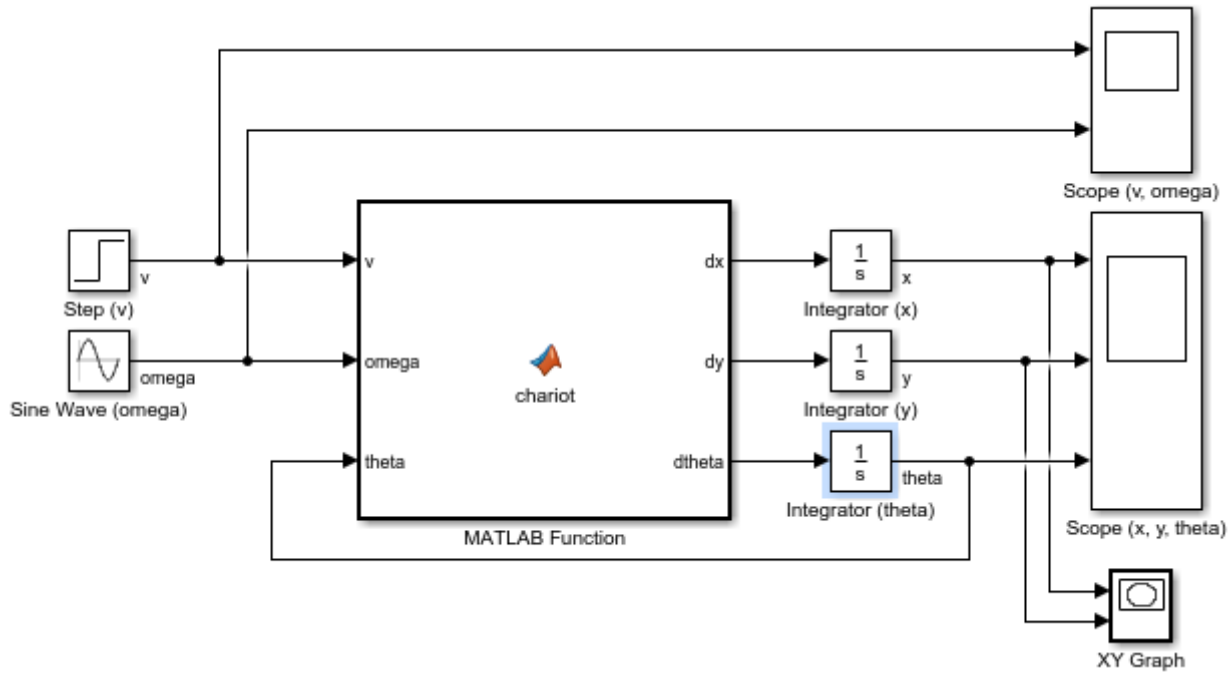

Scope (v, omega)
File
Tools
View
Simulation
Help
v
1
0.8
0.6
0.4
0.2
0
omega
1
0.8
0.6
0.4
0.2
0
0
1
2
3
4
5
6
7
8
9
10
Ready
T=10.000

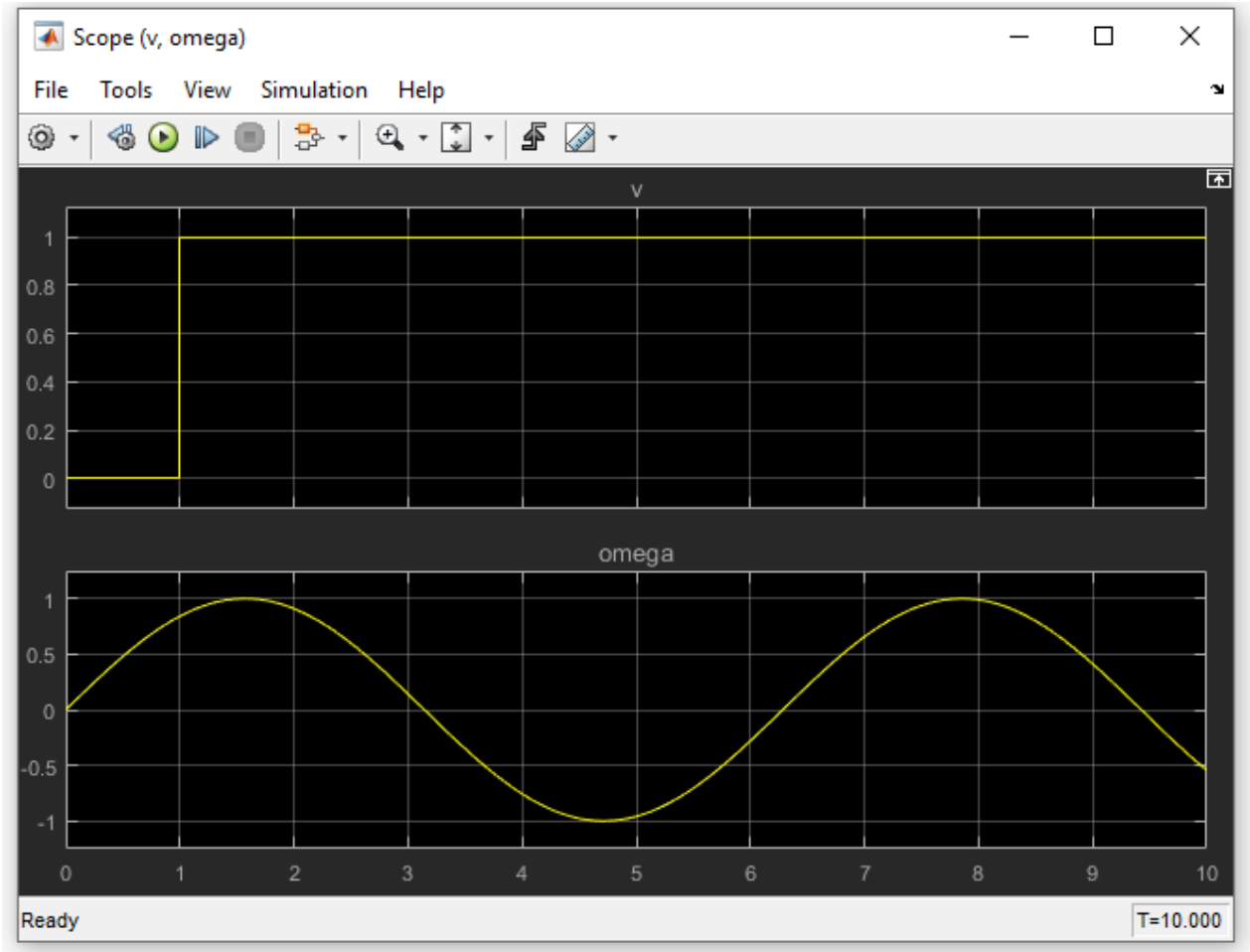

Scope (x, y, theta)
File Tools View Simulation Help
x
y
theta
Ready
T=10.000

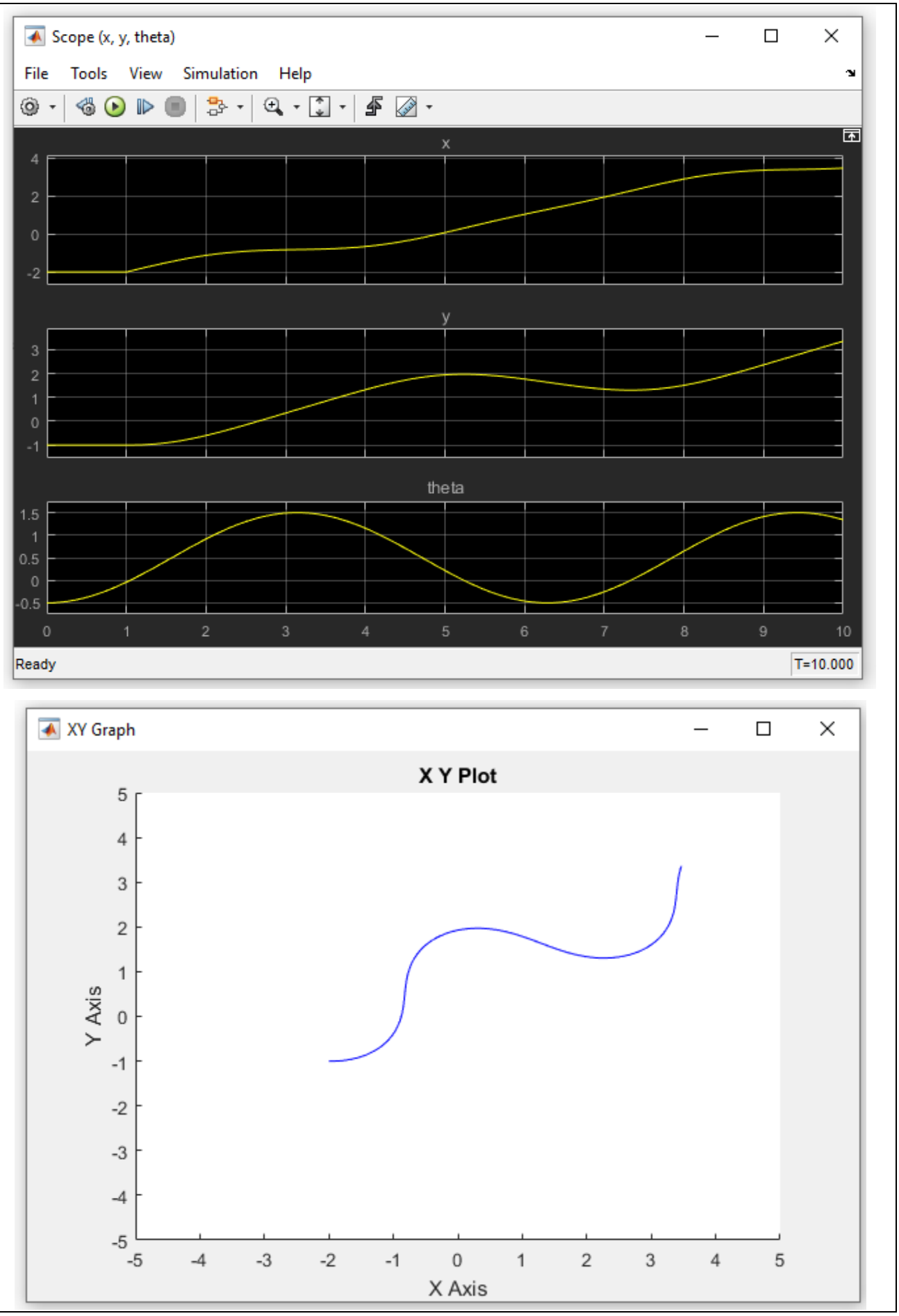

XY Graph
X Y Plot
Y Axis
X Axis

Q4/ Modifier les conditions initiales et les commandes (par exemple en utilisant un bloc Sine Wave) et observer le comportement du système.

On peut par exemple générer ω avec un bloc **Sine Wave** et modifier les conditions initiales.







Q5/ Quels sont les points d'équilibre du système ? Vérifier en simulation.

A l'équilibre l'équation d'état devient :

$$\begin{cases} 0 = \dfrac{d}{dt}x_e(t) = v_e(t).\cos\big(\theta_e(t)\big) \\ 0 = \dfrac{d}{dt}y_e(t) = v_e(t).\sin\big(\theta_e(t)\big) \\ 0 = \dfrac{d}{dt}\theta_e(t) = \omega_e(t) \end{cases}$$

On en déduit la commande à l'équilibre :

$$\begin{cases} v_e(t) = 0 \\ \omega_e(t) = 0 \end{cases}$$

L'état d'équilibre n'est pas contraint :

$$\begin{cases} x_e(t) = cte \\ y_e(t) = cte \\ \theta_e(t) = cte \end{cases}$$

On vérifie bien en simulation qu'avec une commande nulle, le système reste à l'état initial (quel qu'il soit).

On désire suivre une trajectoire de référence rectiligne suivant l'axe des abscisses avec une vitesse constante $a > 0$:

$$\begin{cases} x_r(t) = a.t \\ y_r(t) = 0 \\ \theta_r(t) = 0 \end{cases} \quad \begin{cases} v_r(t) = a \\ \omega_r(t) = 0 \end{cases}$$

Pour cela, on pose :

$$\begin{cases} x(t) = x_r(t) + \delta x(t) \\ y(t) = y_r(t) + \delta y(t) \\ \theta(t) = \theta_r(t) + \delta\theta(t) \end{cases} \quad \begin{cases} v(t) = v_r(t) + \delta v(t) \\ \omega(t) = \omega_r(t) + \delta\omega(t) \end{cases}$$

où les variables d'écart $\delta x(t)$, $\delta y(t)$, $\delta\theta(t)$, $\delta v(t)$ et $\delta\omega(t)$ sont supposées petites.

Q6/ Vérifier que la trajectoire de référence est bien solution des équations de la dynamique.

On vérifie bien :

$$\begin{cases} \dfrac{d}{dt}x_r(t) = a = \underbrace{v_r(t)}_{=a}.\underbrace{cos\big(\theta_r(t)\big)}_{=1} \\ \dfrac{d}{dt}y_r(t) = 0 = \underbrace{v_r(t)}_{=a}.\underbrace{sin\big(\theta_r(t)\big)}_{=0} \\ \dfrac{d}{dt}\theta_r(t) = 0 = \omega_r(t) \end{cases}$$

Q7/ Montrer qu'au premier ordre, les variables d'écart satisfont la dynamique linéaire suivante :

$$\begin{cases} \dfrac{\boldsymbol{d}}{\boldsymbol{dt}}\boldsymbol{\delta x(t)} = \boldsymbol{\delta v(t)} \\ \dfrac{\boldsymbol{d}}{\boldsymbol{dt}}\boldsymbol{\delta y(t)} = \boldsymbol{a}.\boldsymbol{\delta\theta(t)} \\ \dfrac{\boldsymbol{d}}{\boldsymbol{dt}}\boldsymbol{\delta\theta(t)} = \boldsymbol{\delta\omega(t)} \end{cases}$$

Il suffit de faire le développement limité à l'ordre 1 selon les variables d'écart des équations de la dynamique :

$$\begin{cases} \frac{d}{dt}x(t) = v(t).\cos\big(\theta(t)\big) \\ \frac{d}{dt}y(t) = v(t).\sin\big(\theta(t)\big) \\ \frac{d}{dt}\theta(t) = \omega(t) \end{cases} \Rightarrow \begin{cases} a + \frac{d}{dt}\delta x(t) = \big(a + \delta v(t)\big).\underbrace{\cos\big(\delta\theta(t)\big)}_{\approx 1} \approx a + \delta v(t) \\ \frac{d}{dt}\delta y(t) = \big(a + \delta v(t)\big).\underbrace{\sin\big(\delta\theta(t)\big)}_{\approx \delta\theta(t)} \approx a.\delta\theta(t) \\ \frac{d}{dt}\delta\theta(t) = \delta\omega(t) \end{cases}$$

On constate qu'au premier ordre, les équations forment deux systèmes indépendants.

Q8/ Proposer un bouclage d'état qui stabilise la dynamique longitudinale :

$$\boldsymbol{\frac{d}{dt}\delta x(t) = \delta v(t)}$$

Pour faire un bouclage d'état, on cherche une commande $\delta v(t)$ qui dépend linéairement de l'état $\delta x(t)$ et qui garantit que $\delta x(t) \underset{t\to\infty}{\longrightarrow} 0$.

$\delta v(t)$ s'écrit donc nécessairement de la forme :

$$\delta v(t) = -k_x.\delta x(t)$$

où k_x est une constante à choisir judicieusement.

La dynamique longitudinale s'écrit avec cette commande :

$$\frac{d}{dt}\delta x(t) = -k_x.\delta x(t)$$

En choisissant $k_x > 0$, on a $\delta x(t) = \delta x(0).e^{-k_x.t} \underset{t\to\infty}{\longrightarrow} 0$ et on obtient un bouclage d'état stabilisant la dynamique longitudinale.

Q9/ Proposer un bouclage d'état qui stabilise la dynamique latérale :

$$\begin{cases} \boldsymbol{\frac{d}{dt}\delta y(t) = a.\delta\theta(t)} \\ \boldsymbol{\frac{d}{dt}\delta\theta(t) = \delta\omega(t)} \end{cases}$$

Pour faire un bouclage d'état, on cherche une commande $\delta\omega(t)$ qui dépend linéairement des états $\delta y(t)$ et $\delta\theta(t)$ et qui garantit que $\delta y(t) \underset{t\to\infty}{\longrightarrow} 0$ et $\delta\theta(t) \underset{t\to\infty}{\longrightarrow} 0$.

$\delta\omega(t)$ s'écrit donc nécessairement de la forme :

$$\delta\omega(t) = -k_y.\delta y(t) - k_\theta.\delta\theta(t)$$

où k_y et k_θ sont des constantes à choisir judicieusement.

La dynamique latérale s'écrit avec cette commande :

$$\begin{cases} \frac{d}{dt}\delta y(t) = a.\delta\theta(t) \\ \frac{d}{dt}\delta\theta(t) = -k_y.\delta y(t) - k_\theta.\delta\theta(t) \end{cases}$$

Mettons ce système sous forme d'état :

$$\frac{d}{dt}\underbrace{\begin{pmatrix}\delta y(t)\\ \delta\theta(t)\end{pmatrix}}_{X(t)} = \underbrace{\begin{pmatrix}0 & a\\ -k_y & -k_\theta\end{pmatrix}}_{A_{BF}}.\underbrace{\begin{pmatrix}\delta y(t)\\ \delta\theta(t)\end{pmatrix}}_{X(t)}$$

On sait que si les valeurs propres de A_{BF} sont à partie réelle strictement négative, le système est asymptotiquement stable et $X(t) \underset{t\to\infty}{\longrightarrow} 0$.

On utilise la propriété suivante :

Soit A une matrice carrée de dimension 2.

Les valeurs propres de A sont à partie réelle strictement négative $\Leftrightarrow \begin{cases} \mathrm{Tr}(A) < 0 \\ \det(A) > 0 \end{cases}$

Ici on a :

$$\begin{cases} \mathrm{Tr}(A_{BF}) = -k_\theta < 0 \\ \det(A_{BF}) = k_y . a > 0 \end{cases} \Leftrightarrow \begin{cases} k_y > 0 \\ k_\theta > 0 \end{cases}$$

En choisissant $k_y > 0$ et $k_\theta > 0$, on obtient un bouclage d'état stabilisant la dynamique latérale.

Q10/ Implémenter dans la simulation un bouclage d'état qui stabilise les dynamiques longitudinales et latérales. Vérifier que le système en boucle fermé converge bien vers la trajectoire de référence avec $a = 1$ pour des conditions initiale faiblement en écart avec la trajectoire de référence.

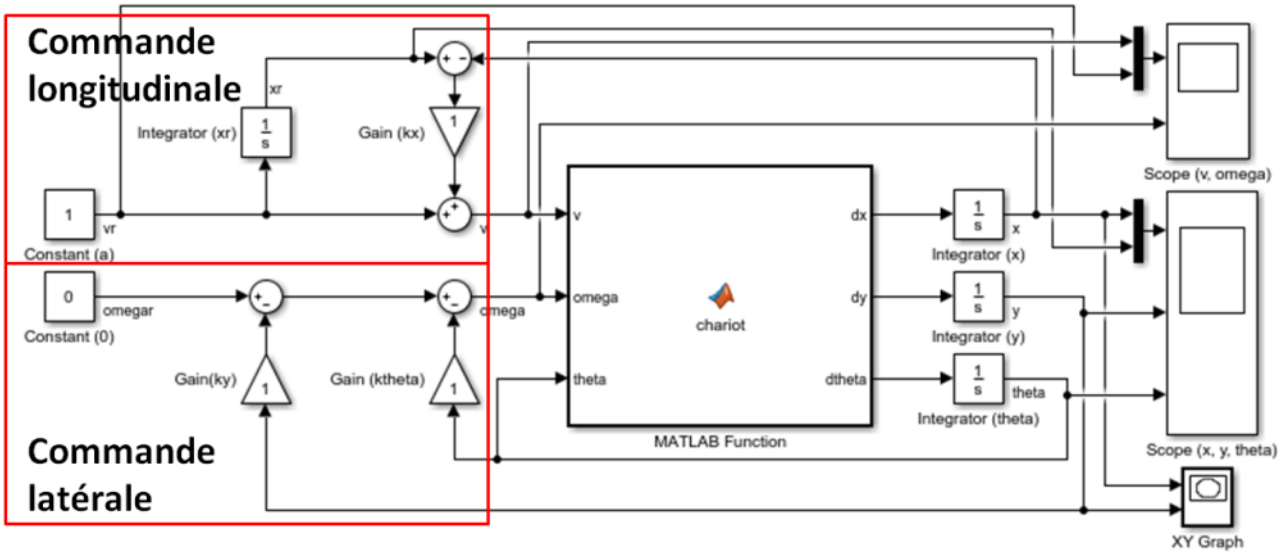
La commande en boucle fermée s'écrit :

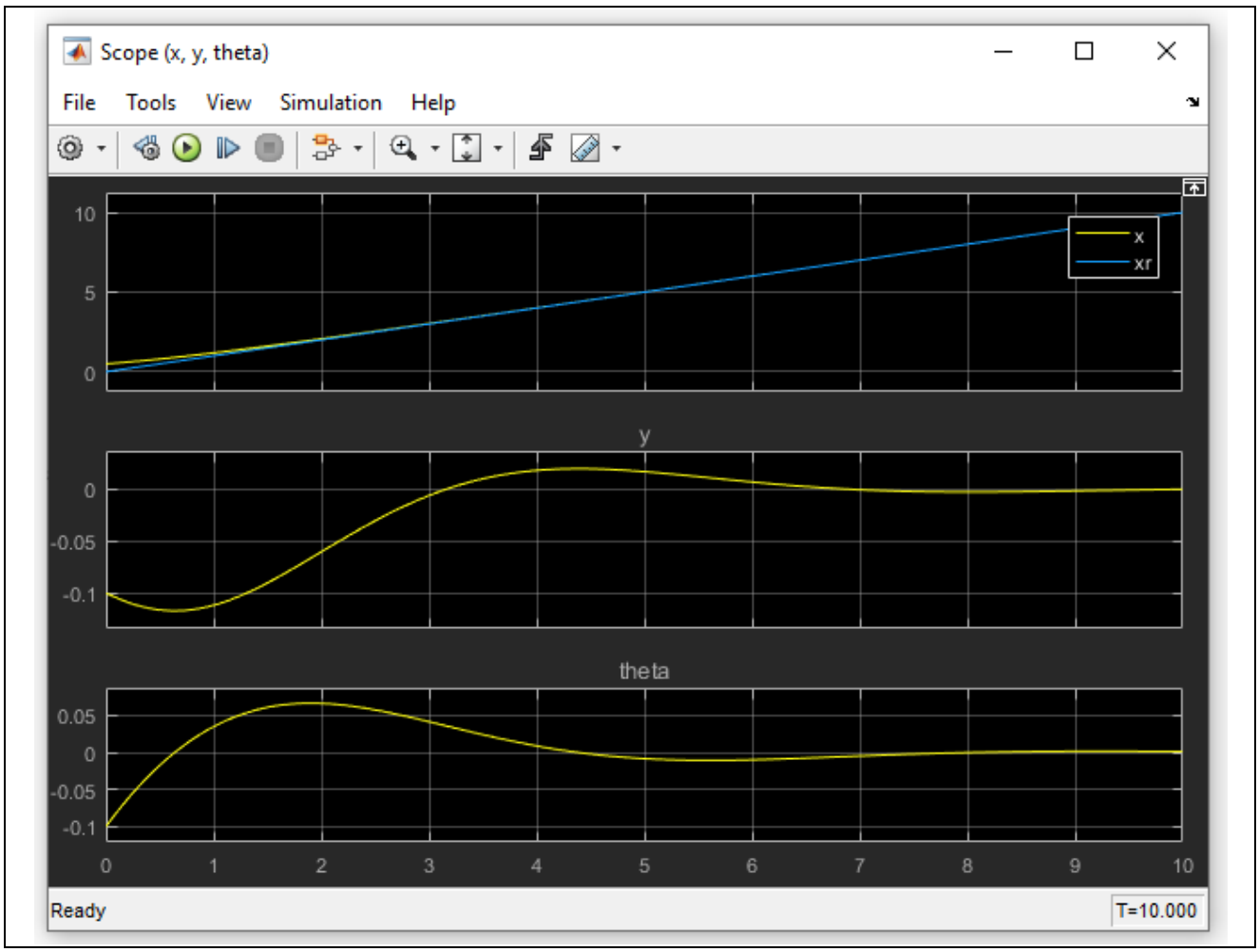
$$\begin{cases} v(t) = v_r + \delta v(t) = a - k_x . \delta x(t) = a + k_x . \big(a.t - x(t)\big) \\ \omega(t) = \omega_r + \delta\omega(t) = -k_y . \delta y(t) - k_\theta . \delta\theta(t) = -k_y . y(t) - k_\theta . \theta(t) \end{cases}$$

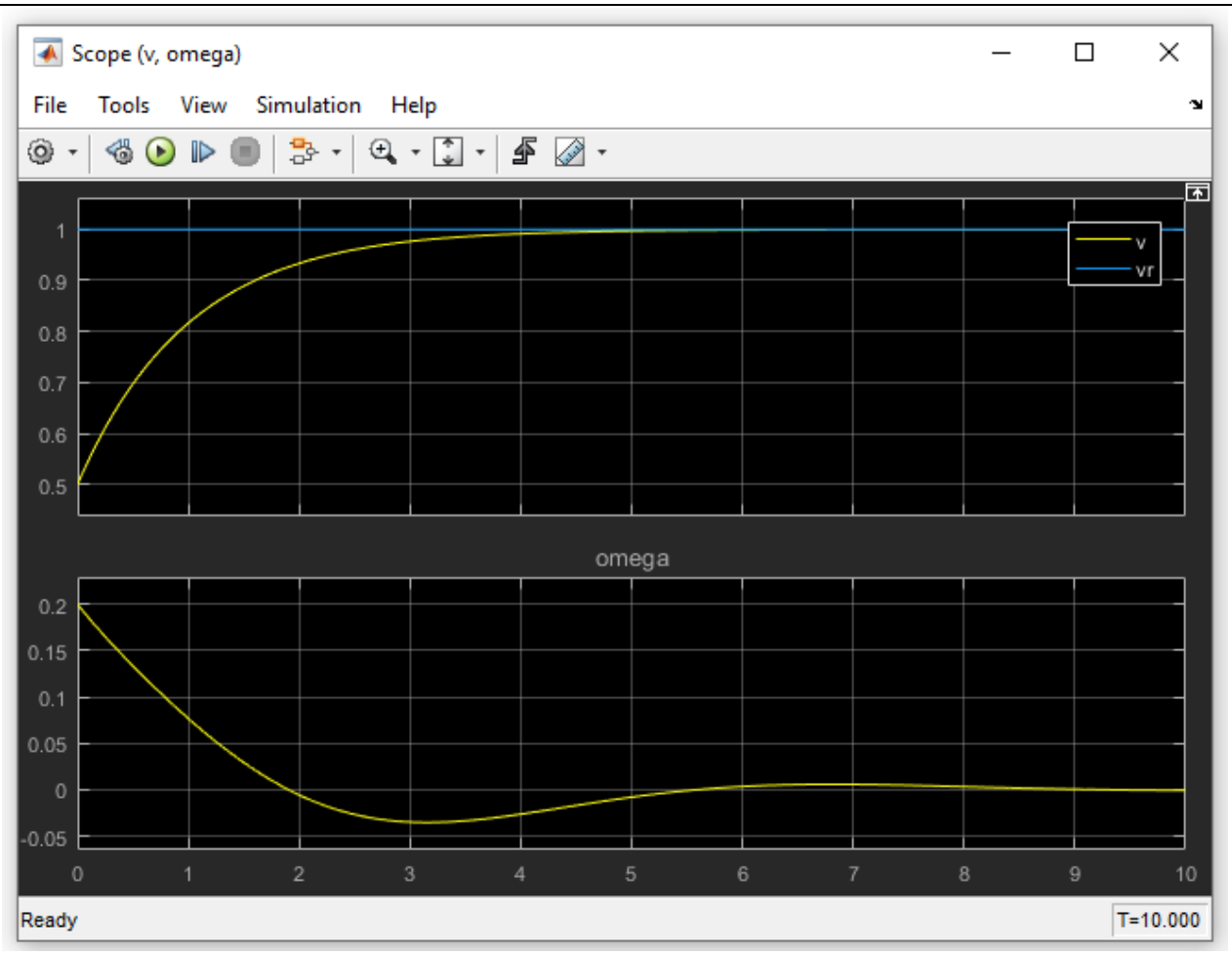
On choisit (arbitrairement) le paramétrage suivant :

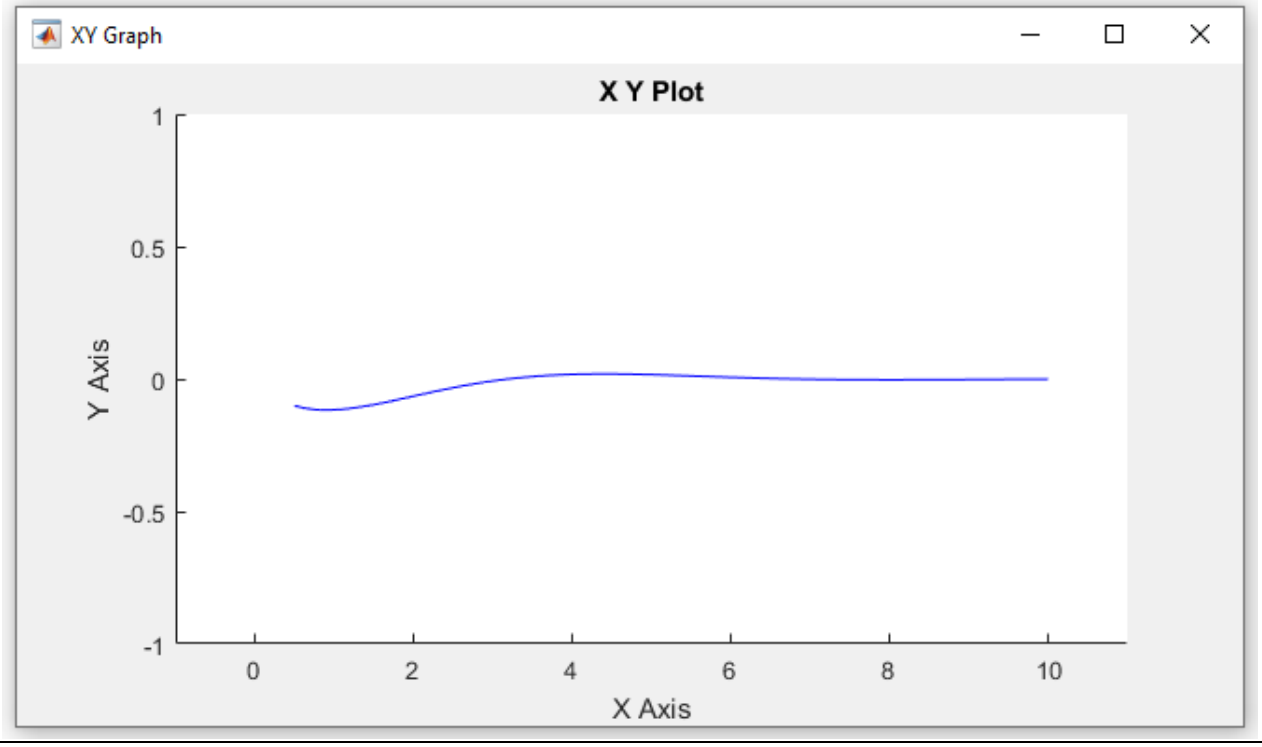
$$\begin{cases} k_x = 1 \\ k_y = 1 \\ k_\theta = 1 \end{cases} \quad \begin{cases} x(0) = 0{,}5 \\ y(0) = -0{,}1 \\ \theta(0) = -0{,}1 \end{cases}$$

On ajoute des blocs **Sum** et **Gain** pour implémenter le bouclage par retour d'état. On ajoute également des blocs **Mux** pour créer des vecteurs avec, pour chaque variable d'état ou de commande, la valeur de référence et la valeur courante, afin de pouvoir afficher deux courbes sur chaque graphique.




Scope (x, y, theta)
File
Tools
View
Simulation
Help
10
5
0
x
xr
y
0
-0.05
-0.1
theta
0.05
0
-0.05
-0.1
0
1
2
3
4
5
6
7
8
9
10
Ready
T=10.000



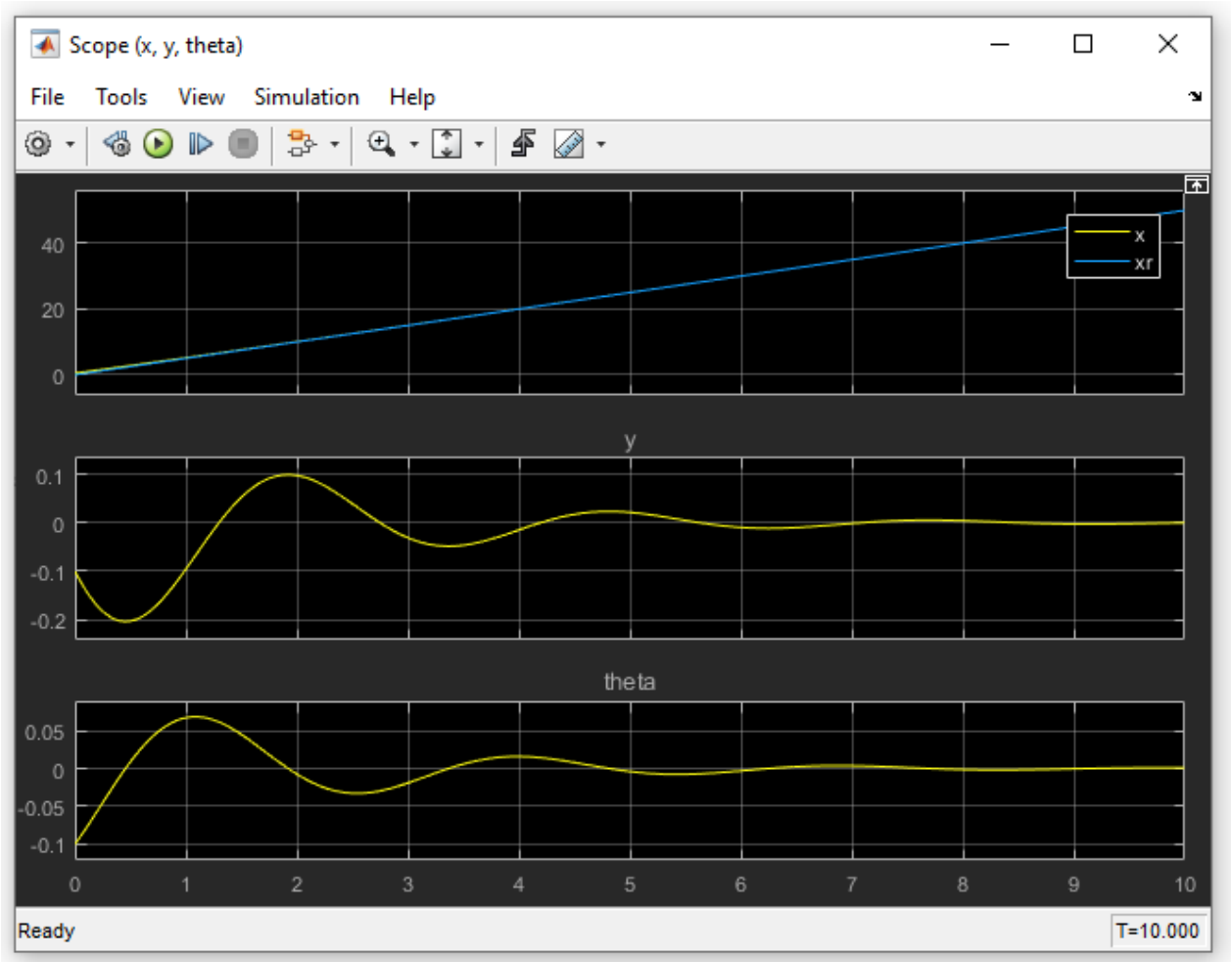


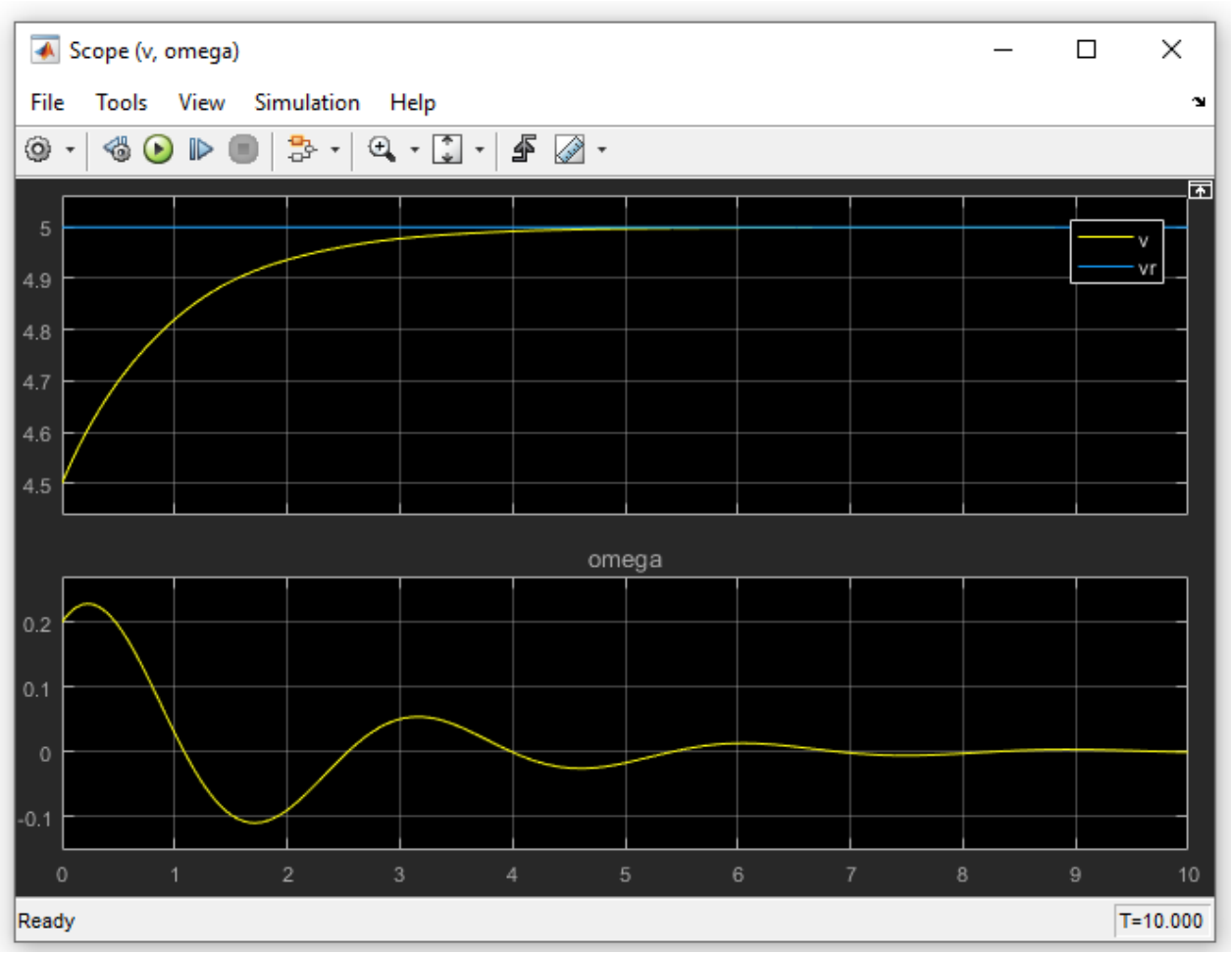
Q11/ Comment se comporte le système bouclé lorsque la vitesse de référence a augmente ? diminue ? Que peut-on proposer pour que le comportement du système bouclé soit similaire quelle que soit la valeur de a ?

Lorsque a augmente :

- la dynamique longitudinale converge toujours avec le même temps de réponse ;
- la dynamique latérale converge toujours mais devient de plus en plus oscillante.

Pour l'exemple qui suit, on à choisit $a = 5$.

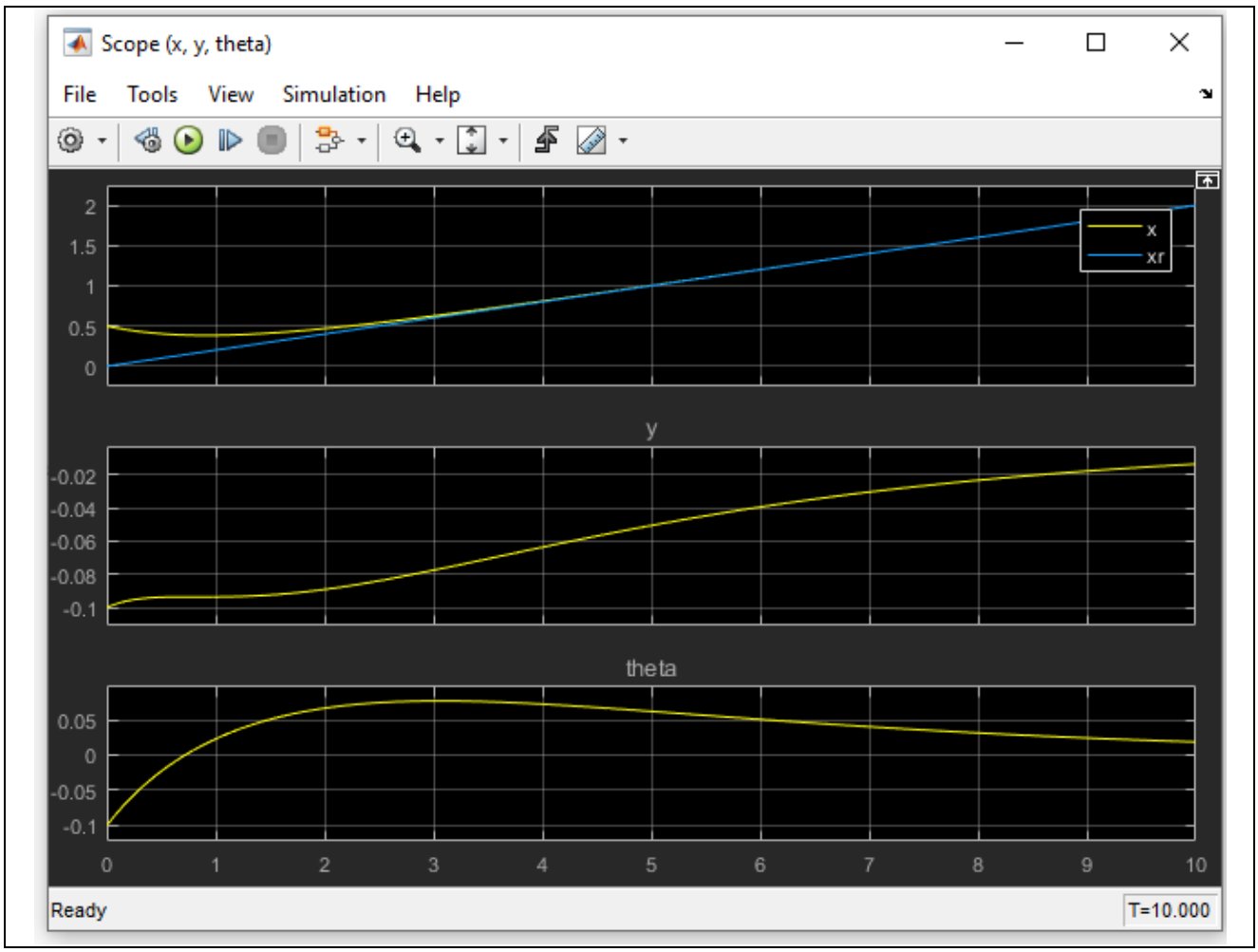


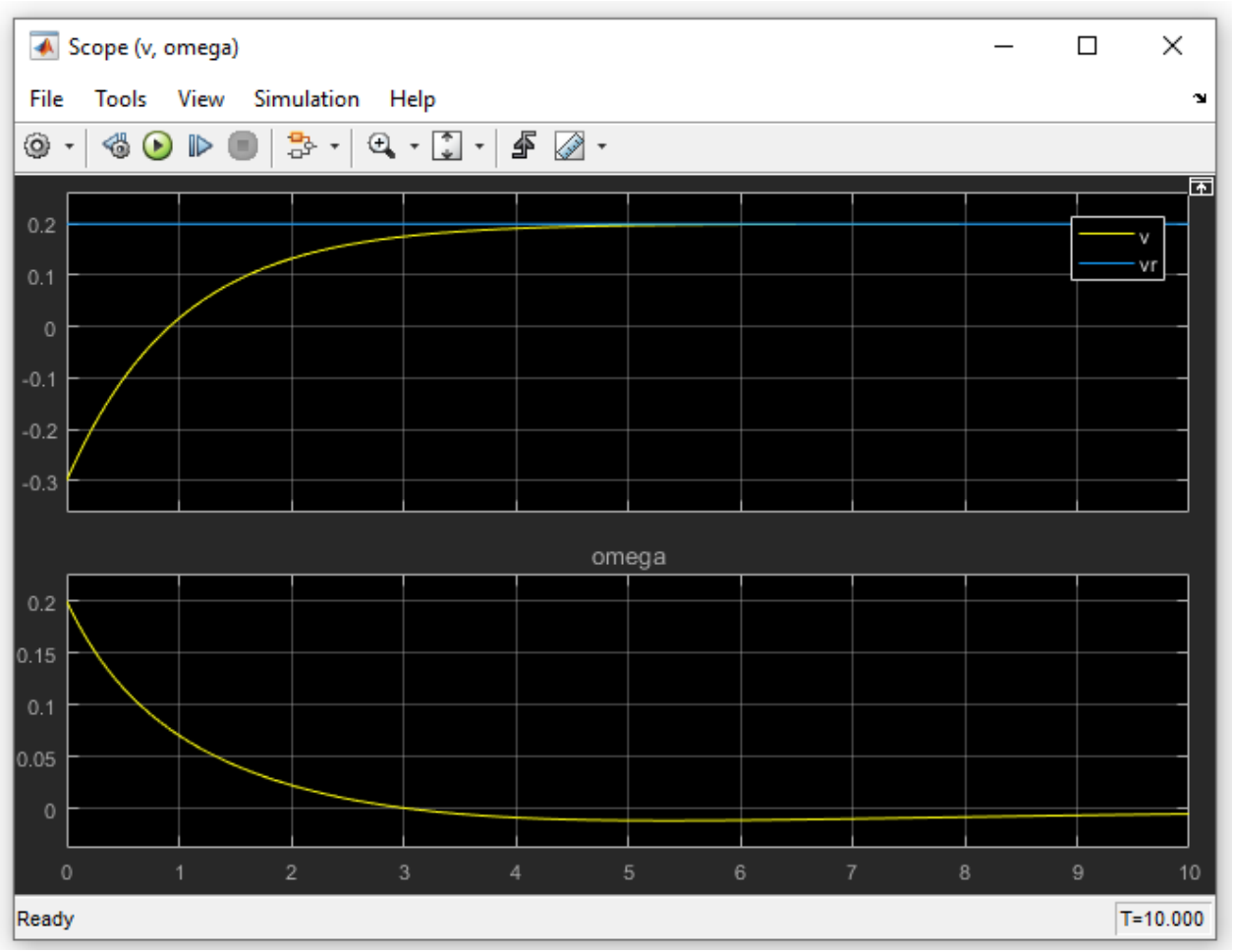


Lorsque a diminue :

- la dynamique longitudinale converge toujours avec le même temps de réponse ;
- la dynamique latérale du converge de moins en moins rapidement.

Pour l'exemple qui suit, on à choisit $a = 0{,}2$.


Scope (x, y, theta)
File Tools View Simulation Help
x
xr
y
theta
Ready
T=10.000



Pour que le comportement du système soit le même quelque soit la valeur de a, on peut proposer un paramétrage du gain k_y qui varie en fonction de a de sorte que le polynôme caractéristique de A_{BF} (et donc ses valeurs propres) ne dépende pas de a :

$$\chi_{A_{BF}}(s) = s^2 - \mathrm{Tr}(A_{BF}).s + \det(A_{BF}) = s^2 + k_\theta.s + k_y.a$$

Si le polynôme caractéristique désiré en boucle fermé est $s^2 + b_1.s + b_0$, on doit alors choisir :

$$\begin{cases} k_y = \dfrac{b_0}{a} \\ k_\theta = b_1 \end{cases}$$

Pour avoir un comportement similaire à celui de la question précédente, on doit donc choisir :

$$\begin{cases} k_y = \dfrac{1}{a} \\ k_\theta = 1 \end{cases}$$

On vérifie bien en simulation qu'avec ce paramétrage, le comportement du système bouclé est similaire quelle que soit la valeur de a.

EXERCICE 2 – Oscillateur de Van der Pol

On cherche à simuler un oscillateur de Van der Pol. Physiquement, l'oscillateur de Van der Pol est un circuit électrique oscillant avec une résistance variable pouvant être négative, qui peut être modélisé par l'équation suivante :

$$\frac{d^2}{dt^2}x(t) + \rho.(x(t)^2 - 1).\frac{d}{dt}x(t) + x(t) = u(t)$$

Le terme d'amortissement varie ainsi non-linéairement.

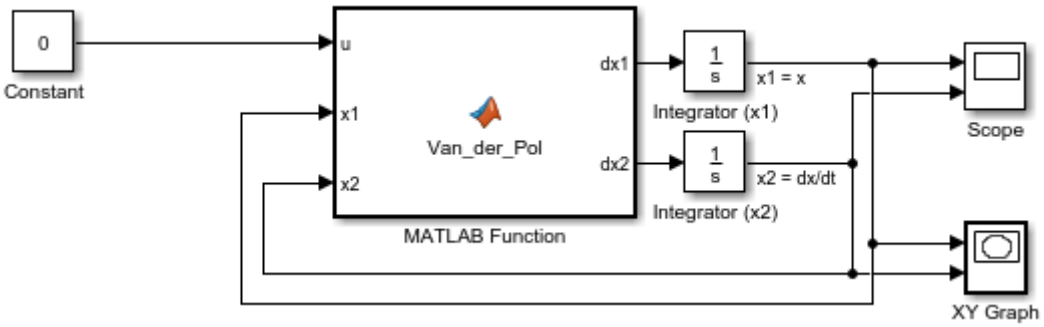
On étudie le comportement de l'oscillateur de Van der Pol sans excitation externe. Le terme $u(t)$ est donc supposé nul.

Q1/ Modéliser dans le module Simulink de MATLAB l'oscillateur de Van der Pol. Utiliser les conditions initiales $x(0) = 1$ et $\frac{d}{dt}x(0) = 0,4$ et le paramètre $\rho = 1$.

On met le système dynamique sous forme d'état :

$$\begin{cases} x_1(t) = x(t) \\ x_2(t) = \frac{d}{dt}x(t) \end{cases} \quad \begin{cases} \frac{d}{dt}x_1(t) = x_2(t) \\ \frac{d}{dt}x_2(t) = -x_1(t) - \rho.(x_1(t)^2 - 1).x_2(t) + u(t) \end{cases}$$

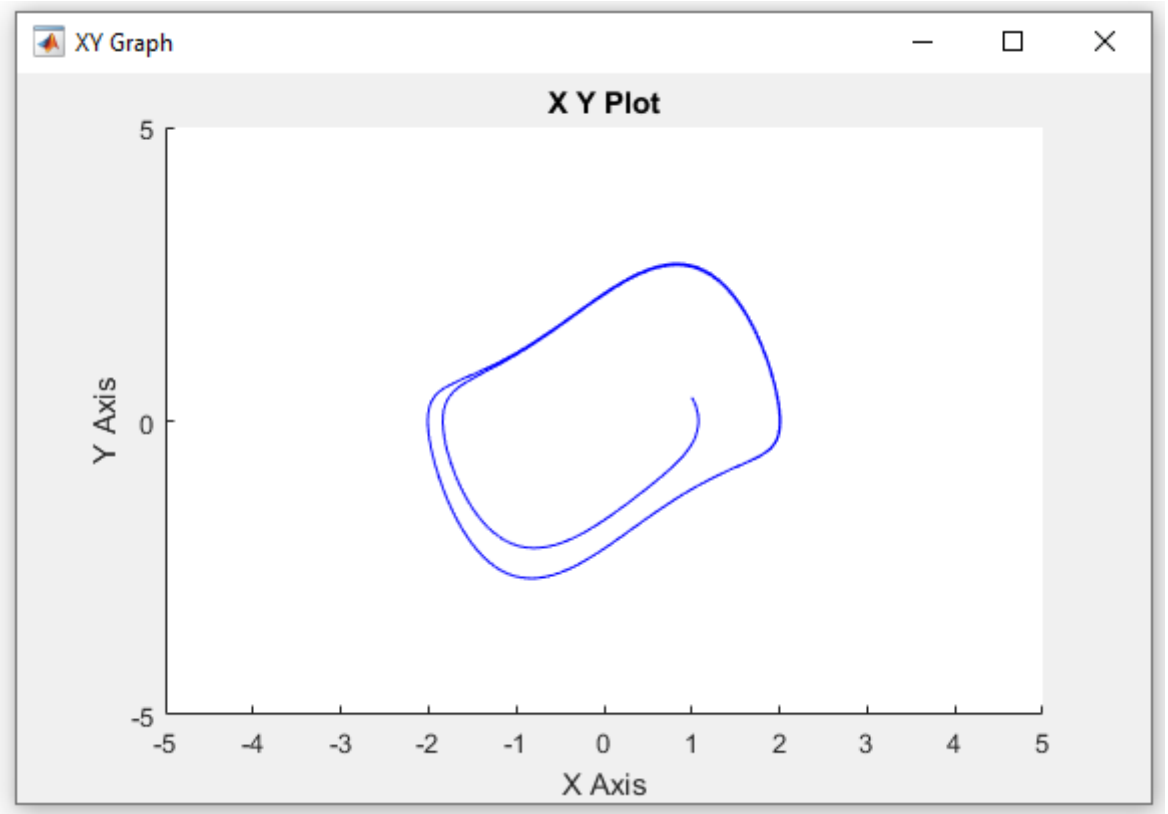
En utilisant un bloc **MATLAB Function** et un bloc **Integrator** pour chaque état, on obtient le schéma bloc suivant :



La fonction d'état est codée dans le bloc **MATLAB Function** :

```
function [dx1, dx2] = Van_der_Pol(u, x1, x2)

rho = 1;

dx1 = x2;
dx2 = -x1 - rho*(x1^2-1)*x2 + u;
```

Q2/ Représenter la trajectoire issue de la condition initiale précédente. On pourra utiliser un temps de simulation de 100 secondes.

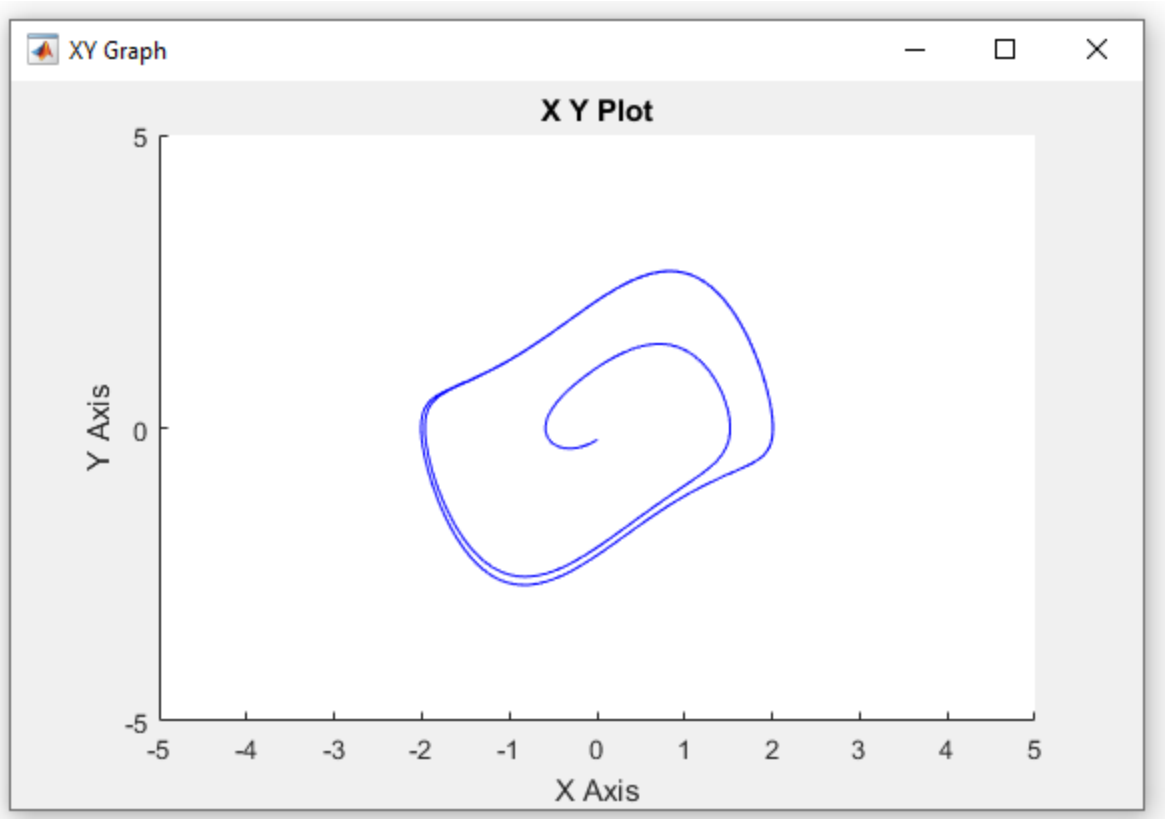


Q3/ Vers quoi l'état du système converge-t-il ? Quel est le terme dans l'équation de l'oscillateur de Van der Pol qui engendre ce phénomène ?

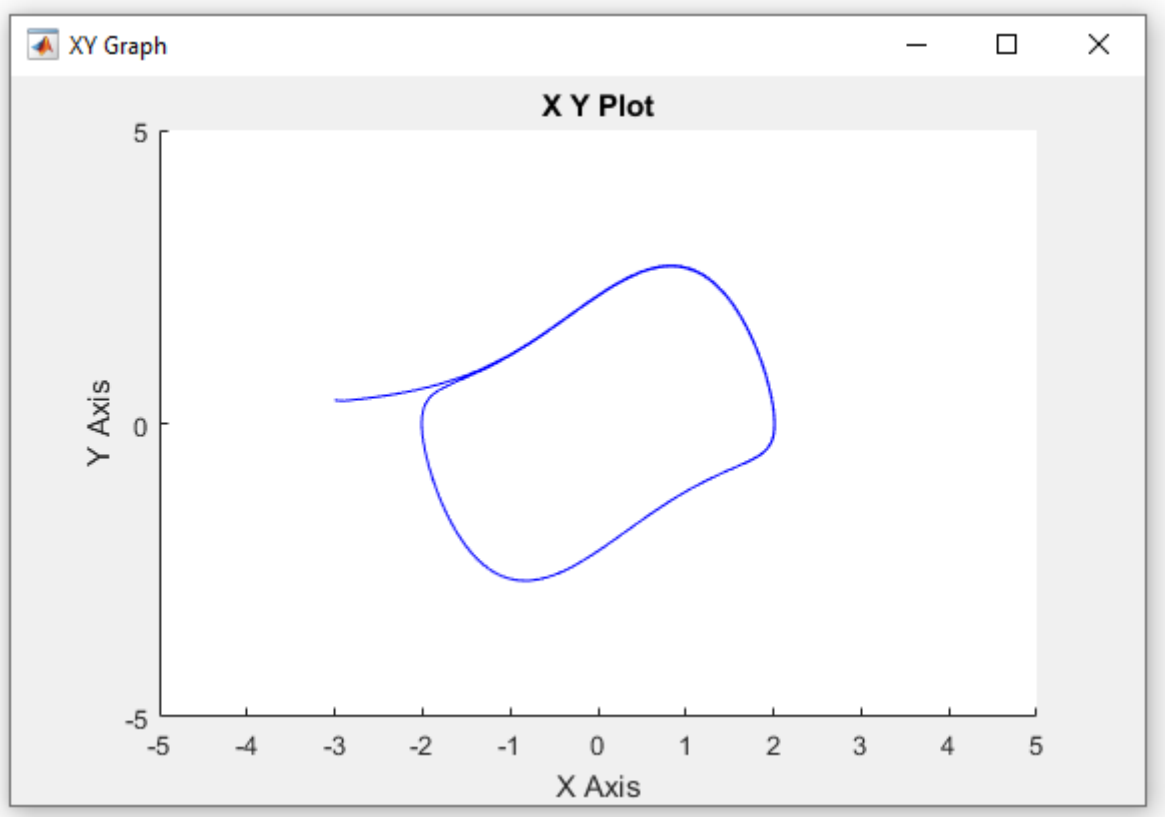
L'état du système converge vers un cycle limite (trajectoire périodique). Ce comportement est dû au terme d'amortissement non linéaire et plus précisément la résistance variable pouvant devenant négative lorsque $|x(t)| < 1$, rendant le point d'équilibre $x = \frac{d}{dt}x = 0$ instable.

Q4/ Faire varier les conditions initiales. Que constate-t-on ? Aurait-on obtenu la même propriété avec un oscillateur linéaire ?

Avec $x(0) = 0$ et $\frac{d}{dt}x(0) = -0{,}2$ (conditions initiales à l'intérieur du cycle limite), on obtient le comportement suivant :



Avec $x(0) = -3$ et $\frac{d}{dt}x(0) = 0{,}4$ (conditions initiales à l'extérieure du cycle limite), on obtient le comportement suivant :



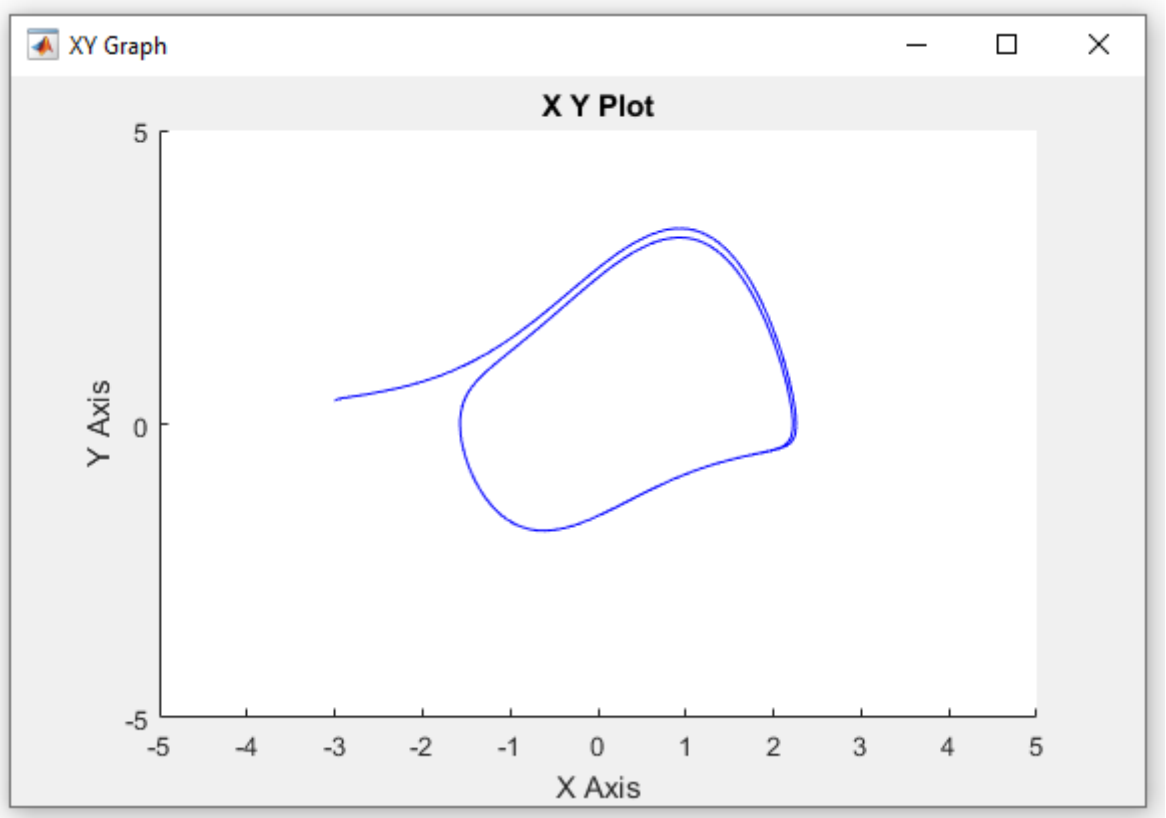
Quelles que soient les conditions initiales (hormis le point d'équilibre instable), l'état du système converge vers le cycle limite.

On étudie maintenant le comportement de l'oscillateur de Van der Pol avec excitation externe. Le terme $u(t)$ n'est donc plus supposé nul.

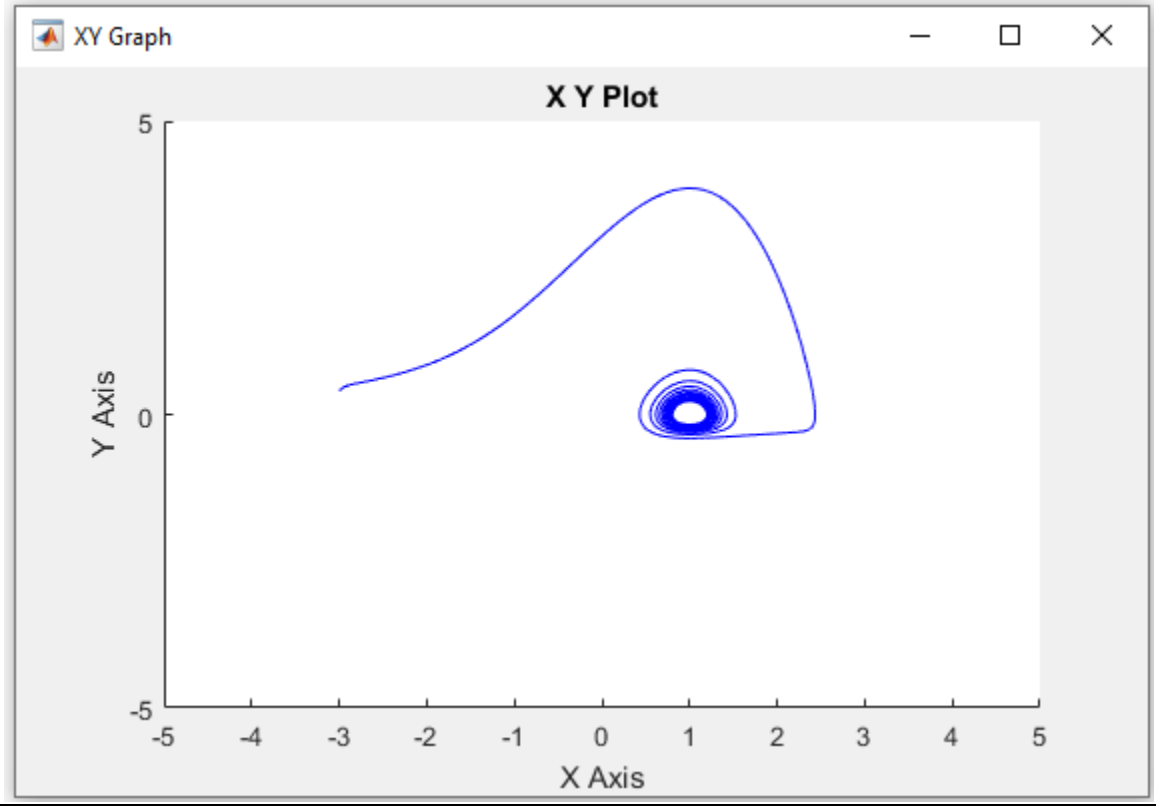
Q5/ Ajouter une perturbation $u(t)$. Que constate-t-on ? Quel genre de terme peut-on rajouter sans risquer de modifier trop la période de l'oscillateur ? Proposer un cas problématique.

On teste d'abord des valeurs constantes pour $u(t)$.

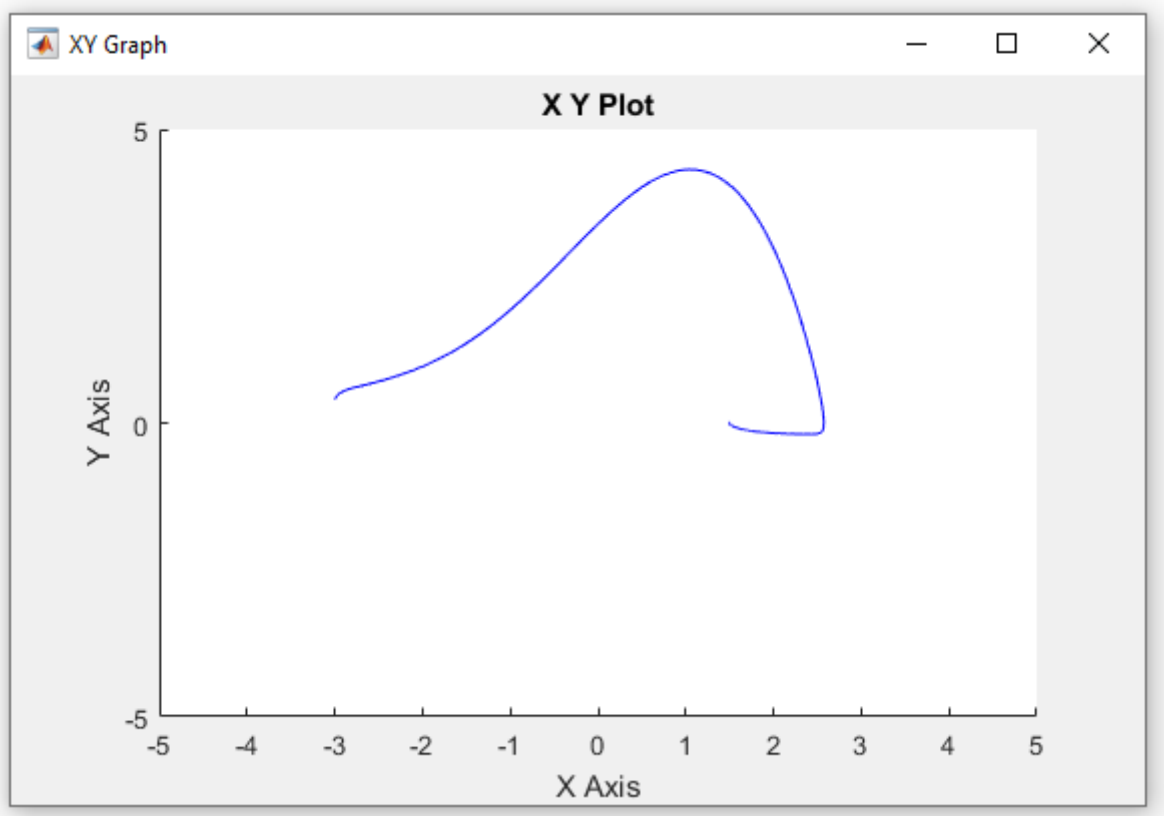
Avec $u(t) = 0{,}5$, le système converge vers un cycle limite « déformé ».



Avec $u(t) = 1$, le système semble converger de plus en plus lentement vers un point d'équilibre.

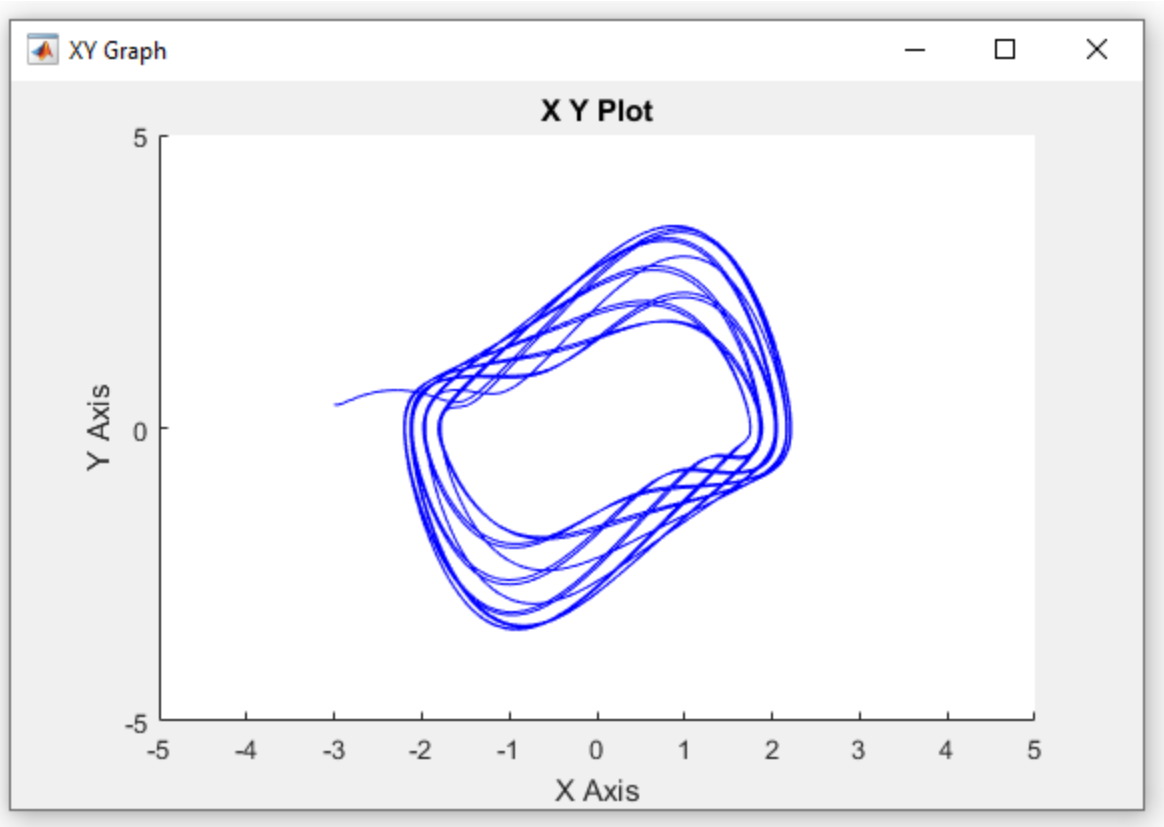


Avec $u(t) = 1{,}5$, le système converge vers un point d'équilibre.

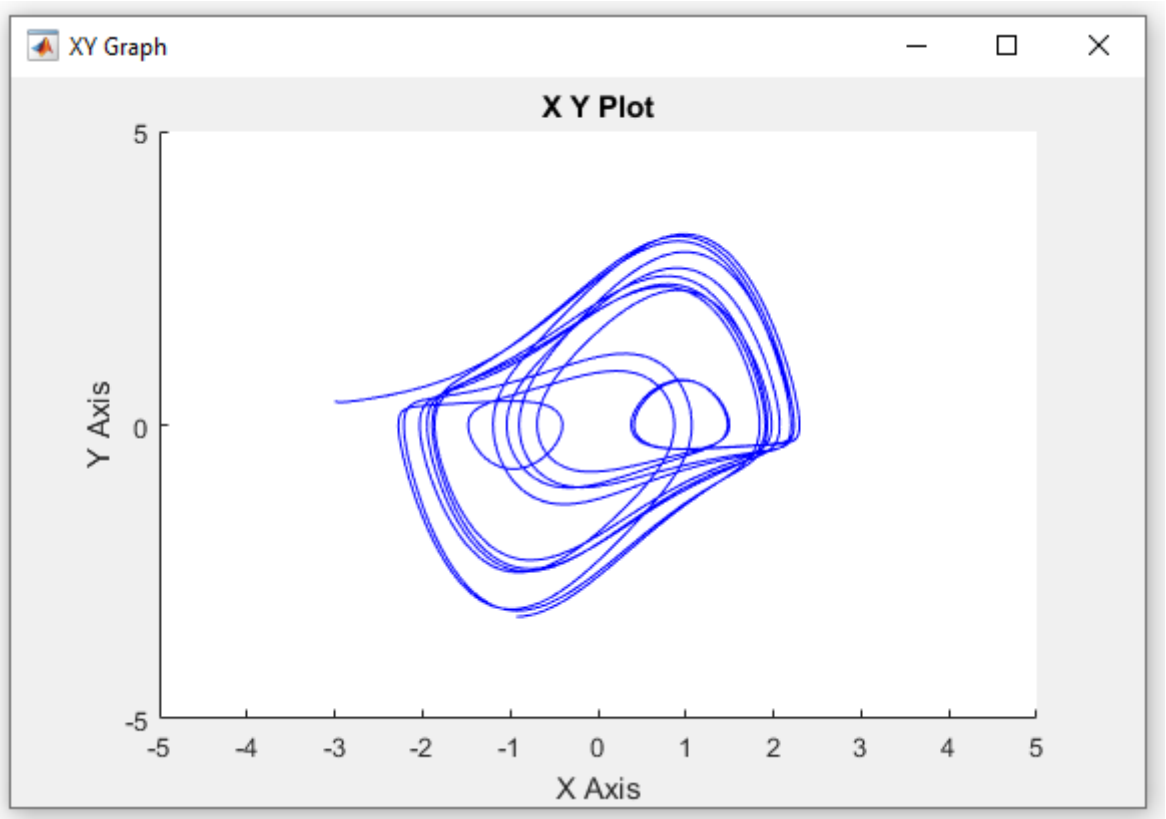


Testons maintenant des signaux sinusoïdaux $u(t) = \sin(\omega . t)$.

Avec $\omega = 2$, le système oscille autour du cycle limite.



Avec $\omega = 0{,}1$, le système oscille entre deux points d'attraction.

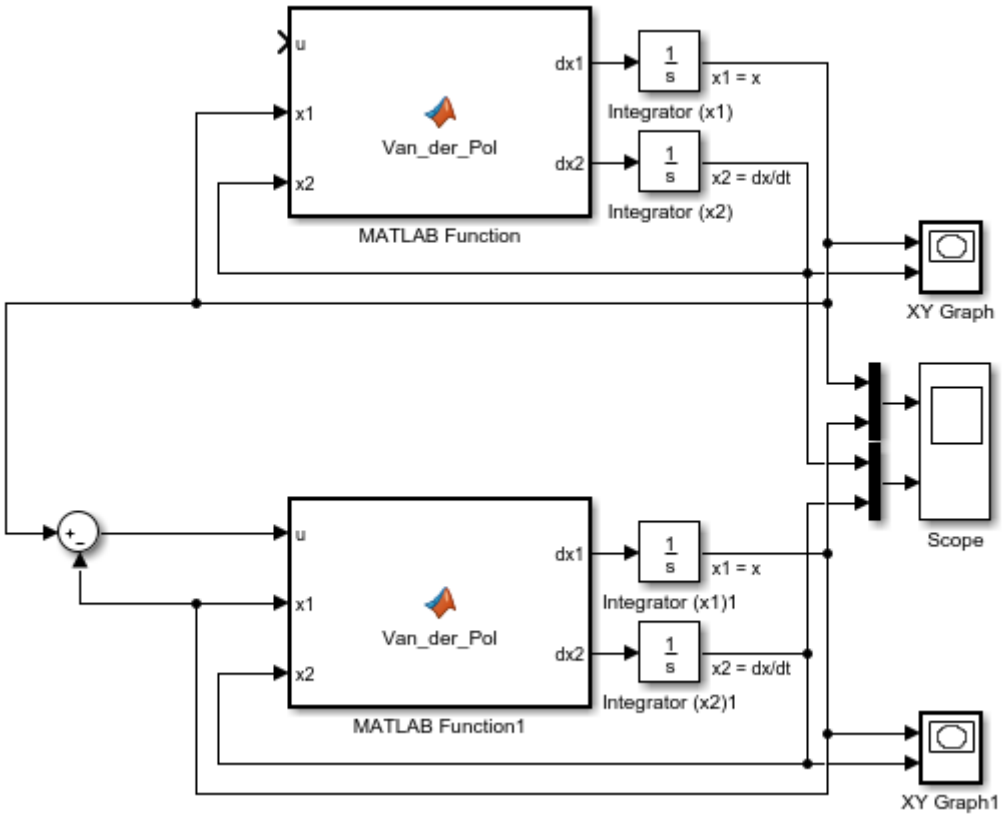


On observe des comportements très différents en fonction des perturbations. Le comportement de l'oscillateur de Van der Pol perturbé est chaotique.

Q6/ Dupliquer l'oscillateur de Van der Pol. Proposer un schéma pour synchroniser les deux oscillateurs. On pourra par exemple chercher à ralentir ou accélérer l'un des deux systèmes en fonction de mesures. La méthode obtenue est-elle robuste à des perturbations dans les équations des oscillateurs ?

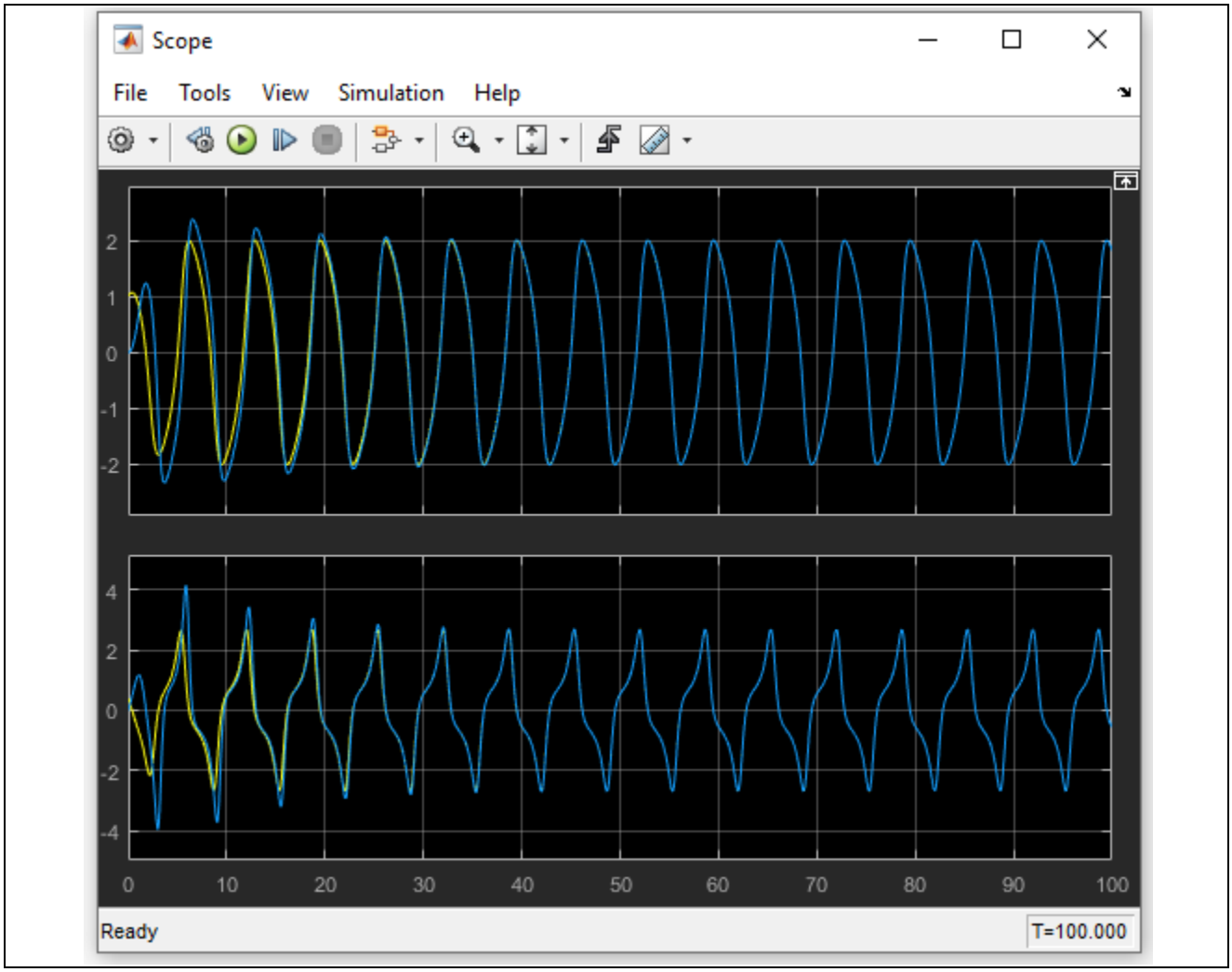
Pour synchroniser le second oscillateur de Van der Pol sur le premier, on applique au second une commande $u(t) = x_{VdP1}(t) - x_{VdP2}(t)$. Plus les deux oscillateurs sont désynchronisés, plus la commande modifiera le comportement du second oscillateur en créant un terme de rappel.

On obtient le schéma bloc suivant :



On choisit d'initialiser le second oscillateur de Van der Pol sur la position d'équilibre instable $x = \frac{d}{dt}x = 0$.

On observe que la commande proposée permet bien de faire converger l'état du second oscillateur de Van der Pol vers celui du premier : les deux oscillateurs sont synchronisés.


Scope
File
Tools
View
Simulation
Help
2
1
0
-1
-2
4
2
0
-2
-4
0
10
20
30
40
50
60
70
80
90
100
Ready
T=100.000