# Contrôle garanti par réseaux de neurones pour des robots mobiles

Pôle Systèmes Cyber-Physiques

Tarek OMRAN Ali RAMLAOUI

20 avril 2022

Encadré par Adnane SAOUD

# Mise en contexte

Système dynamique

$$\begin{aligned} F : \mathbb{R}^n \times \mathbb{R}^m &\to \mathbb{R}^n \\ (x(k), u(k)) &\mapsto F(x(k), u(k)) = x(k+1) \end{aligned}$$

muni d'un contrôleur prenant des décisions sur l'entrée en fonction de l'état $x(k)$

# Mise en contexte

Système dynamique

$$F : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$$
$$(x(k), u(k)) \mapsto F(x(k), u(k)) = x(k+1)$$

muni d'un contrôleur prenant des décisions sur l'entrée en fonction de l'état $x(k)$

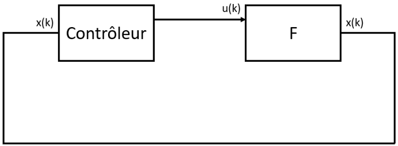



Schéma bloc du système

# Mise en contexte

Système dynamique

$$F : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$$
$$(x(k), u(k)) \mapsto F(x(k), u(k)) = x(k+1)$$

muni d'un contrôleur prenant des décisions sur l'entrée en fonction de l'état $x(k)$

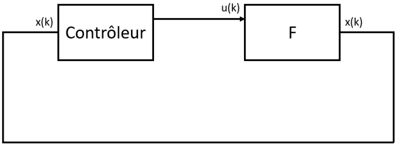



Schéma bloc du système

- Contrôleur : réseau de neurones
- Analyse d'atteignabilité par intervalles
- Images d'intervalles par la fonction $F$

# Notion d'intervalle

**Définition (Intervalle de dimension n)**

*Considérons l'espace* $\mathbb{R}^n$, *avec* $n \in \mathbb{N}^*$. *Alors un intervalle de dimension n est un ensemble pouvant s'écrire* $\left[\underline{a_1}, \overline{a_1}\right] \times \ldots \times \left[\underline{a_n}, \overline{a_n}\right]$, *où* $(\underline{a_i} \leq \overline{a_i}) \in \mathbb{R}^2$, $\forall i \in [\![1, n]\!]$. *On notera l'intervalle* $[a]$.

# Notion d'intervalle

**Définition (Intervalle de dimension n)**

*Considérons l'espace* $\mathbb{R}^n$, *avec* $n \in \mathbb{N}^*$. *Alors un intervalle de dimension n est un ensemble pouvant s'écrire* $\left[\underline{a_1}, \overline{a_1}\right] \times \ldots \times \left[\underline{a_n}, \overline{a_n}\right]$, *où* $(\underline{a_i} \leq \overline{a_i}) \in \mathbb{R}^2, \forall i \in [\![1, n]\!]$. *On notera l'intervalle* $[a]$.

**Definition (Sur-approximation par un intervalle)**

Pour tout ensemble $\mathcal{H} \subset \mathrm{R}^n$, on appelle $\mathcal{I}_{\mathcal{H}}$, le plus petit intervalle de $\mathbb{R}^n$ tel que $\mathcal{H} \subset \mathcal{I}_{\mathcal{H}}$.

# Notion d'intervalle

**Définition (Intervalle de dimension n)**

*Considérons l'espace* $\mathbb{R}^n$*, avec* $n \in \mathbb{N}^*$*. Alors un intervalle de dimension n est un ensemble pouvant s'écrire* $\left[\underline{a_1}, \overline{a_1}\right] \times \ldots \times \left[\underline{a_n}, \overline{a_n}\right]$*, où* $(\underline{a_i} \leq \overline{a_i}) \in \mathbb{R}^2, \forall i \in [\![1, n]\!]$*. On notera l'intervalle* $[a]$*.*

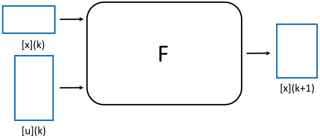
**Definition (Sur-approximation par un intervalle)**

Pour tout ensemble $\mathcal{H} \subset \mathrm{R}^n$, on appelle $\mathcal{I}_{\mathcal{H}}$, le plus petit intervalle de $\mathbb{R}^n$ tel que $\mathcal{H} \subset \mathcal{I}_{\mathcal{H}}$.

**Definition (Longueur d'un intervalle)**

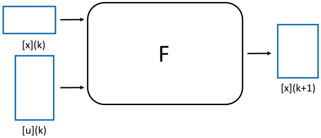
$$\rho(\mathcal{I}) = \max_{i \in [\![1,n]\!]} (\underline{a_i} - \overline{a_i})$$

# Atteignabilité

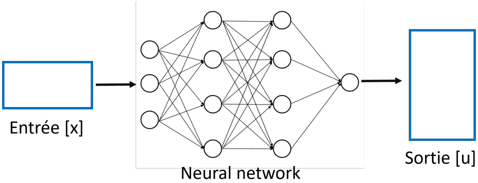



Atteignabilité Système

# Atteignabilité

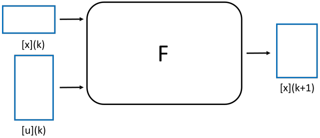



Atteignabilité Système

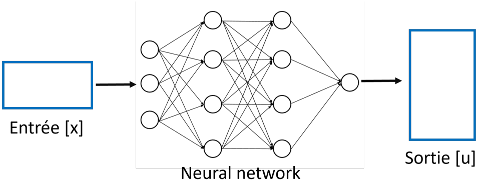



Atteignabilité contrôleur

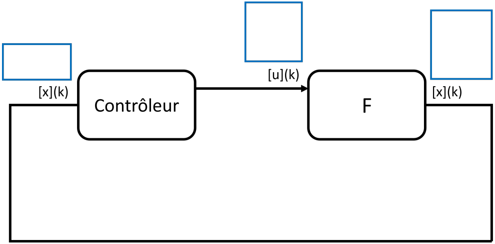
# Atteignabilité




Atteignabilité Système

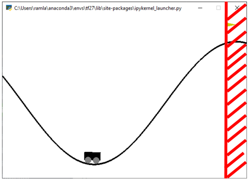



Atteignabilité contrôleur



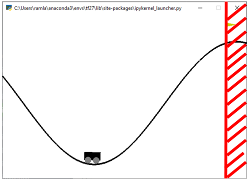

Atteignabilité boucle fermée

# Systèmes considérés




Mountain car

# Systèmes considérés

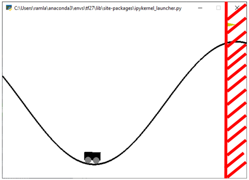



Mountain car

$$F : \begin{cases} x(k+1) & = x(k) + v(k+1) \\ v(k+1) & = u(k)P - 0.0025\cos(3x(k)) \end{cases}$$

où $P$, constante, $u \in [-1, 1]$, $x \in [-1.2, 0.6]$, position horizontale, et vitesse, $v \in [-0.07, 0.07]$

## Systèmes considérés




Mountain car

$$F : \begin{cases} x(k+1) & = x(k) + v(k+1) \\ v(k+1) & = u(k)P - 0.0025\cos(3x(k)) \end{cases}$$

où $P$, constante, $u \in [-1, 1]$, $x \in [-1.2, 0.6]$, position horizontale, et vitesse, $v \in [-0.07, 0.07]$

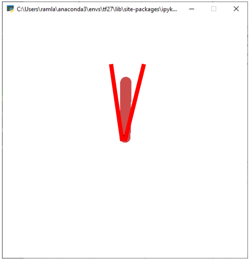
Reward function : Critiquer l'action du contrôleur

$$q(k) = \begin{cases} 100 & \text{si } x(k) \geq 0.6 \\ q(k-1) - 0.1u(k)^2 & \text{sinon} \end{cases}$$

**Objectif d'atteignabilité**

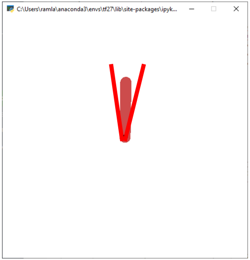
Exemple classique en Reinforcement Learning

# Systèmes considérés




Pendule

# Systèmes considérés



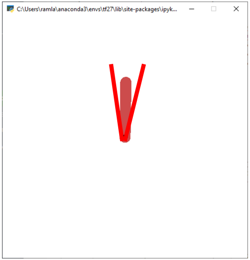

Pendule

$$F : \begin{cases} \theta(k+1) & = \theta(k) + \delta_t \dot{\theta}(k) \\ \dot{\theta}(k+1) & = \dot{\theta}(k) + \frac{3g\delta_t}{2l} \sin(\theta(k)) + \frac{3}{ml^2} u(k) \delta_t \end{cases}$$

où $\delta_t, g, l, m$, constantes, $u(k) \in [-2, 2]$ et

$$X(k) = \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix}, X(k) \in [0, 2\pi] \times [-8, 8]$$

# Systèmes considérés




Pendule

$$F : \begin{cases} \theta(k+1) & = \theta(k) + \delta_t \dot{\theta}(k) \\ \dot{\theta}(k+1) & = \dot{\theta}(k) + \frac{3g\delta_t}{2l} \sin(\theta(k)) + \frac{3}{ml^2} u(k)\delta_t \end{cases}$$

où $\delta_t, g, l, m$, constantes, $u(k) \in [-2, 2]$ et

$$X(k) = \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix}, \ X(k) \in [0, 2\pi] \times [-8, 8]$$

Reward function : Critiquer l'action du contrôleur

$$q(k+1) = m(\theta(k))^2 + 0.01\dot{\theta}(k)^2 + 0.001(u(k))^2, \text{ où } \theta(k) \in [-\pi, \pi]$$

où m est la mesure principale de l'angle

**Objectif de stabilité**

# Principe du Deep Reinforcement Learning

- Objectif : maximiser la fonction récompense sur l'ensemble des décisions prises
- Toute itération est représentée par un état $s_t$, une action $a_t$, une récompense $r_t$ et un nouvel état $s_{t+1}$
- Critique d'une action : Récompense à l'instant $t$ + "récompense potentiel" à partir du nouvel état (valeur Q) $q_t = r_t + \gamma q_{t+1}, 0 < \gamma < 1$

## Principe du Deep Reinforcement Learning

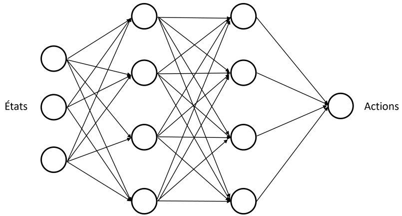
- Objectif : maximiser la fonction récompense sur l'ensemble des décisions prises
- Toute itération est représentée par un état $s_t$, une action $a_t$, une récompense $r_t$ et un nouvel état $s_{t+1}$
- Critique d'une action : Récompense à l'instant $t$ + "récompense potentiel" à partir du nouvel état (valeur Q) $q_t = r_t + \gamma q_{t+1}, 0 < \gamma < 1$

Il faut introduire un décalage entre le réseau qui renvoie $q_t$ et $q_{t+1}$ pour maintenir la stabilité numérique.

- Introduire des réseaux "target" qui marquent ce décalage (copie décalée des réseaux principaux)
- Implémentation sur TensorFlow et Gym (OpenAI) + parallélisation de la boucle d'entraînement

# Deep Deterministic Policy Gradient (DDPG)

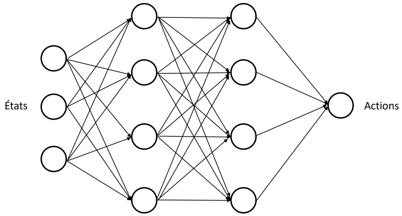
**Réseau acteur**




- C'est le contrôleur
- Couches 1 et 2 : 16 neurones, ReLU
- Couche 3 : 1 neurone, tanh
- Sortie ramenée à l'échelle des actions

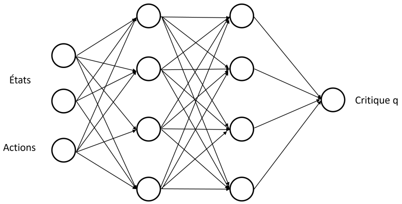
# Deep Deterministic Policy Gradient (DDPG)

## Réseau acteur



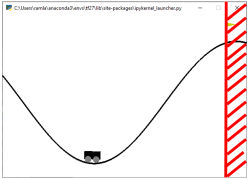

- C'est le contrôleur
- Couches 1 et 2 : 16 neurones, ReLU
- Couche 3 : 1 neurone, tanh
- Sortie ramenée à l'échelle des actions

## Réseau critique



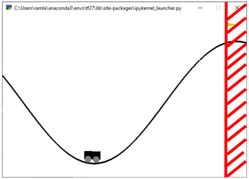

- Concaténation des états et des actions
- Couches 1 et 2 : 16 neurones, ReLU
- Couche 3 : 1 neurone

# Mountain Car - Contrôleur DDPG

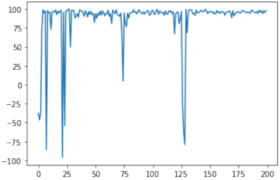



Mountain car

# Mountain Car - Contrôleur DDPG

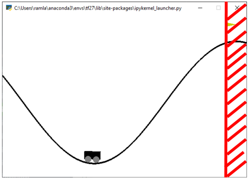



Mountain car

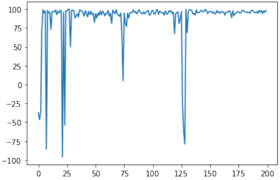



Reward sur 200 épisodes

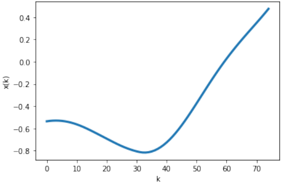
# Mountain Car - Contrôleur DDPG



Mountain car

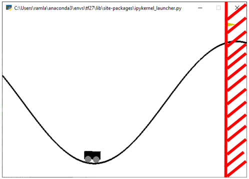



Reward sur 200 épisodes

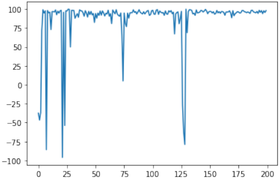



Position de la voiture

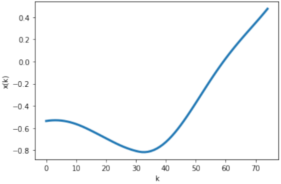
# Mountain Car - Contrôleur DDPG



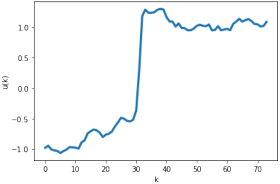

Mountain car




Reward sur 200 épisodes

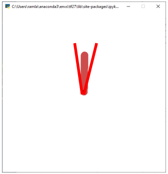



Position de la voiture



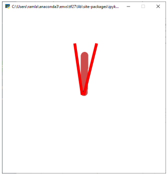

Actions du contrôleur

# Pendule - Contrôleur DDPG

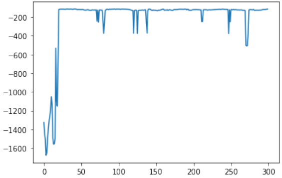



Pendule

# Pendule - Contrôleur DDPG

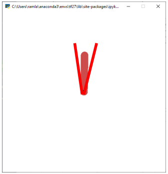


Pendule

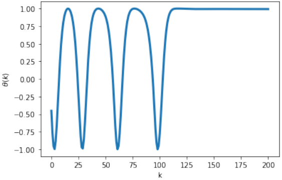



Reward sur 200 épisodes

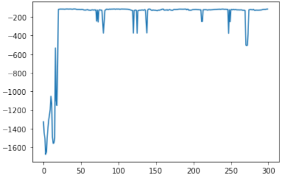
# Pendule - Contrôleur DDPG




Pendule

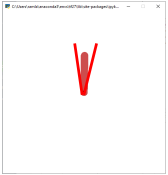



Position du pendule

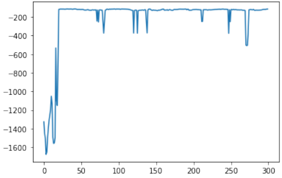



Reward sur 200 épisodes

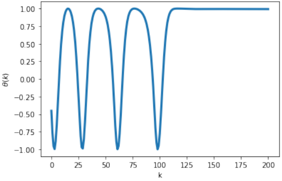
# Pendule - Contrôleur DDPG



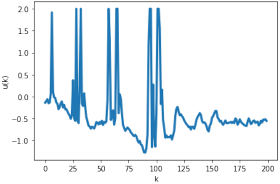
Pendule




Reward sur 200 épisodes



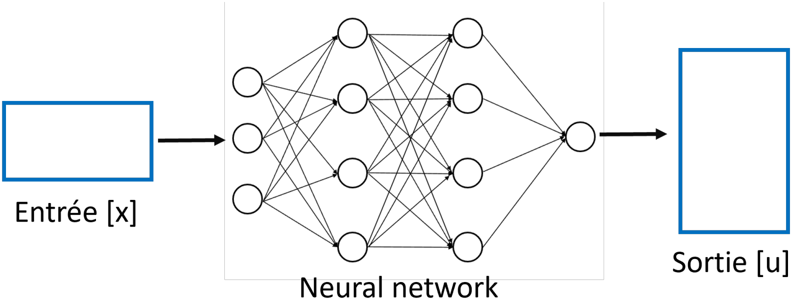

Position du pendule




Actions du contrôleur

# Atteignabilité du réseau de neurones

- Fonctions d'activation croissantes
- Réseau de neurone $\Phi$ à couches denses et $L$ couches intermédiaires
- Paramètres $(n_0, n_1, \ldots, n_L) \in \mathbb{N}^{L+1}$, $W^l \in \mathbb{R}^{n_l \times n_{l-1}}$, $b^l \in \mathbb{R}^{n_l}$, $\sigma_l : \mathbb{R}^{n_l} \to \mathbb{R}^{n_l}$
- Objectif : Image d'un intervalle $[x]$ par le réseau de neurones ?



- Fonctions d'activation croissantes
- Réseau de neurone $\Phi$ à couches denses et $L$ couches intermédiaires
- Paramètres $(n_0, n_1, \ldots, n_L) \in \mathbb{N}^{L+1}$, $W^l \in \mathbb{R}^{n_l \times n_{l-1}}$, $b^l \in \mathbb{R}^{n_l}$, $\sigma_l : \mathbb{R}^{n_l} \to \mathbb{R}^{n_l}$
- Objectif : Image d'un intervalle $[x]$ par le réseau de neurones ?

$$\underline{x}^l = \sigma_l(\sum_{j=1}^{n_{l-1}} \underline{p}_{i,j} + b_i^l) \qquad \underline{p}_{i,j} = \begin{cases} w_{i,j}^l \underline{x}_j \text{ si } w_{i,j}^l \geq 0 \\ w_{i,j}^l \overline{x}_j \text{ si } w_{i,j}^l < 0 \end{cases}$$

avec

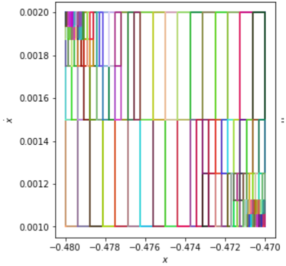
$$\overline{x}^l = \sigma_l(\sum_{j=1}^{n_{l-1}} \overline{p}_{i,j} + b_i^l) \qquad \overline{p}_{i,j} = \begin{cases} w_{i,j}^l \overline{x}_j \text{ si } w_{i,j}^l \geq 0 \\ w_{i,j}^l \underline{x}_j \text{ si } w_{i,j}^l < 0 \end{cases}$$

# Principe de l'algorithme d'atteignabilité

- Sur-approximation de l'image réelle en utilisant les formules
- L'erreur de sur-approximation diminue avec la taille des intervalles considérés
- Idée : Partitionner l'intervalle de départ en sous-intervalles permettant de garantir une erreur inférieure à $\delta$ fixé

# Principe de l'algorithme d'atteignabilité

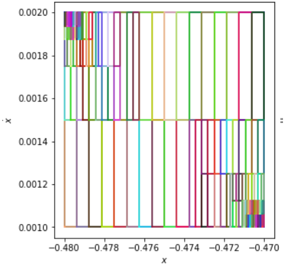
- Sur-approximation de l'image réelle en utilisant les formules
- L'erreur de sur-approximation diminue avec la taille des intervalles considérés
- Idée : Partitionner l'intervalle de départ en sous-intervalles permettant de garantir une erreur inférieure à $\delta$ fixé



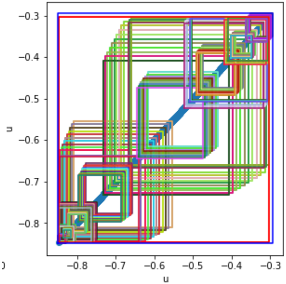

Intervalle de départ avec $\epsilon = 0.0001$

# Principe de l'algorithme d'atteignabilité

- Sur-approximation de l'image réelle en utilisant les formules
- L'erreur de sur-approximation diminue avec la taille des intervalles considérés
- Idée : Partitionner l'intervalle de départ en sous-intervalles permettant de garantir une erreur inférieure à $\delta$ fixé

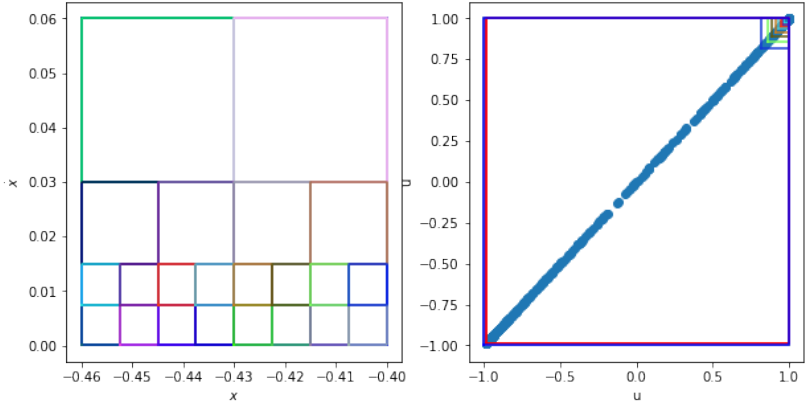



Intervalle de départ avec $\epsilon = 0.0001$



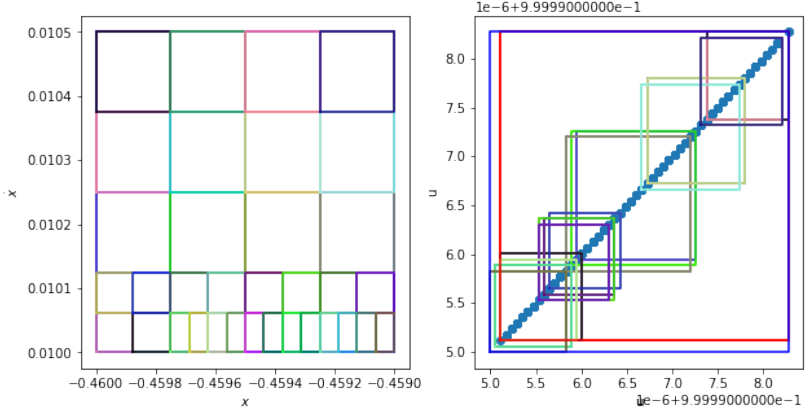

Contrôle du découpage

# Résultats obtenus



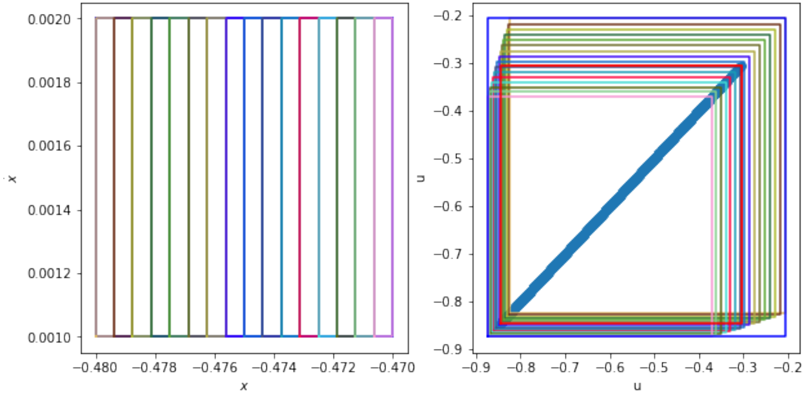

Mountain car - Intervalle large

# Résultats obtenus



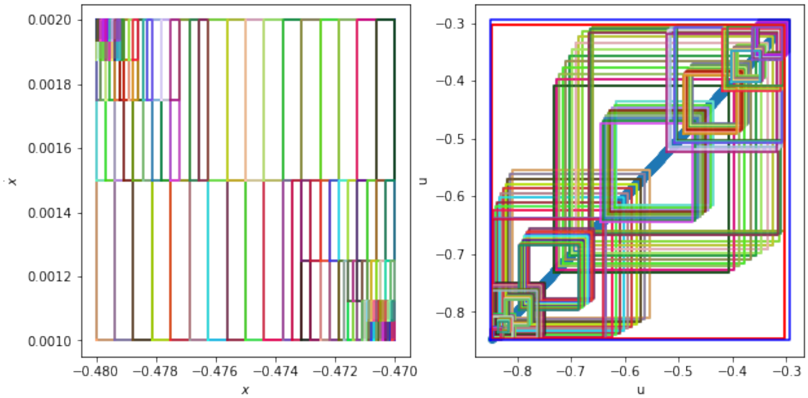

Mountain car - Petit Intervalle

# Résultats obtenus



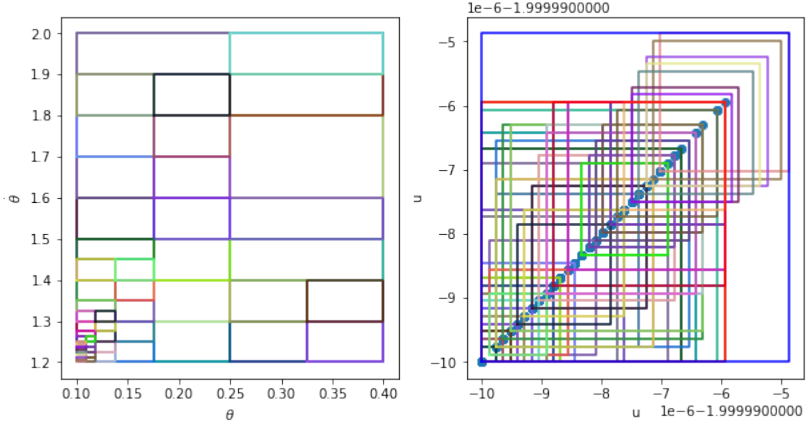

Mountain car - $\epsilon = 0.001$

# Résultats obtenus




Mountain car - $\epsilon = 0.0001$

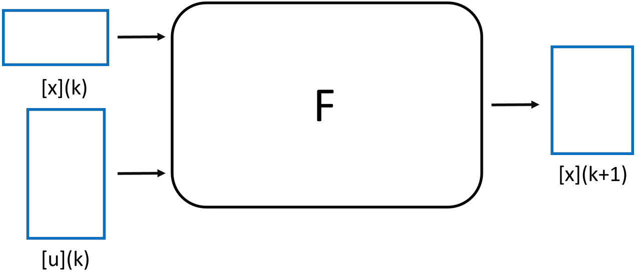
# Résultats obtenus




Atteignabilité du contrôleur du pendule

## Atteignabilité du système dynamique

- Faire de l'arithmétique d'intervalles
- Remplacer les opérations de la dynamique $F$ par des opérations sur des intervalles



# Atteignabilité du système dynamique

- Faire de l'arithmétique d'intervalles
- Remplacer les opérations de la dynamique $F$ par des opérations sur des intervalles

**Exemples**

$$[\underline{x}, \overline{x}]^{-1} = \begin{cases} [-\infty, \infty] & \text{si } \underline{x} < 0 \text{ et } \overline{x} > 0 \\ \left[-\infty, \frac{1}{\underline{x}}\right] & \text{si } \overline{x} = 0 \\ \left[\frac{1}{\overline{x}}, \infty\right] & \text{si } \underline{x} = 0 \\ \left[\frac{1}{\overline{x}}, \frac{1}{\underline{x}}\right] & \text{si } \underline{x}\overline{x} > 0 \\ \emptyset & \text{si } \underline{x} = \overline{x} = 0 \end{cases}$$

# Atteignabilité du système dynamique

- Faire de l'arithmétique d'intervalles
- Remplacer les opérations de la dynamique $F$ par des opérations sur des intervalles

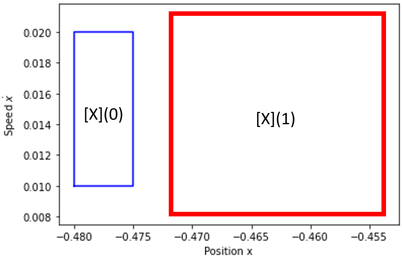
**Exemples**

$$[\underline{x}, \overline{x}]^{-1} = \begin{cases} [-\infty, \infty] & \text{si } \underline{x} < 0 \text{ et } \overline{x} > 0 \\ \left[-\infty, \frac{1}{\underline{x}}\right] & \text{si } \overline{x} = 0 \\ \left[\frac{1}{\overline{x}}, \infty\right] & \text{si } \underline{x} = 0 \\ \left[\frac{1}{\overline{x}}, \frac{1}{\underline{x}}\right] & \text{si } \underline{x}\overline{x} > 0 \\ \emptyset & \text{si } \underline{x} = \overline{x} = 0 \end{cases}$$

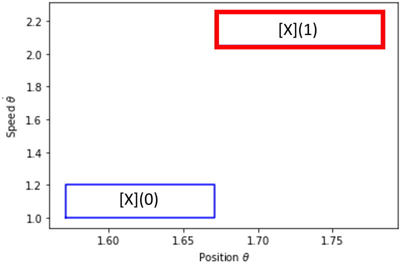
$$\cos([\underline{x}, \overline{x}]) = [\underline{a}, \overline{a}] \text{ , où } \underline{a} = \begin{cases} -1 & \text{si } \exists k \in \mathbb{Z} | (2k\pi + \pi) \in [\underline{x}, \overline{x}] \\ \min(\cos(\underline{a}), \cos(\overline{a})) & \text{sinon} \end{cases}$$

$$\overline{a} = \begin{cases} 1 & \text{si } \exists k \in \mathbb{Z} | (2k\pi) \in [\underline{x}, \overline{x}] \\ \max(\cos(\underline{a}), \cos(\overline{a})) & \text{sinon} \end{cases}$$

# Résultats obtenus




Mountain car sur une itération

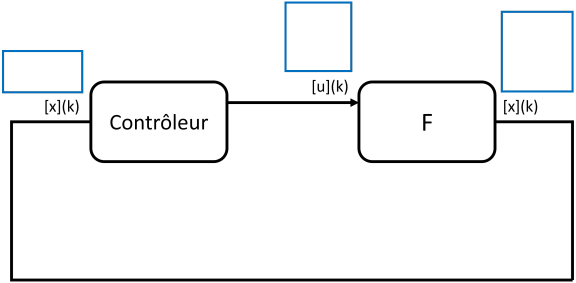



Pendule sur une itération

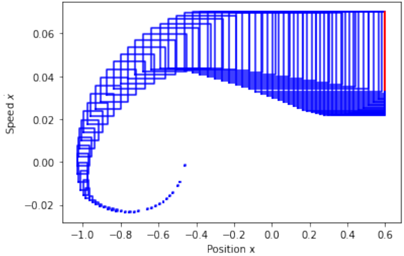
# Contrôle garanti

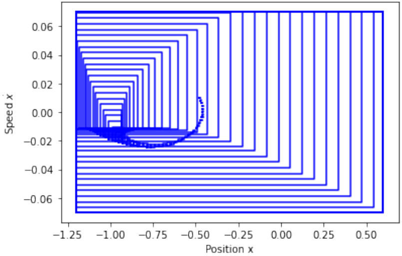
- Objectif : Montrer que le système vérifie une spécification $S$

- Itérer les algorithmes précédents pour estimer les ensembles atteignables au bout de $k$ itérations et simuler le système en boucle fermée



## Mountain Car

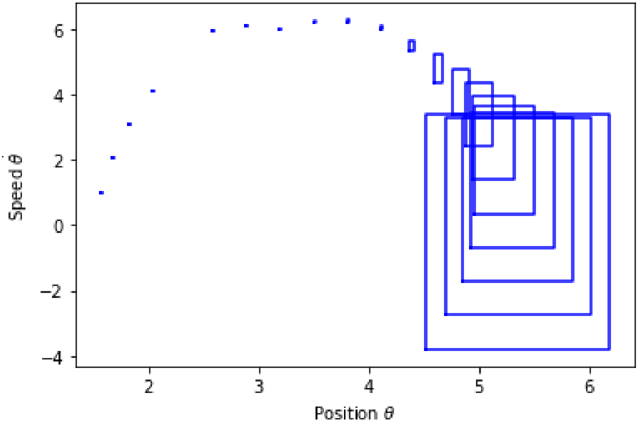



$x(0) \in [-0.48, -0.4795]\,, \dot{x}(0) \in [0.01, 0.0101]$

$x(0) \in [-0.48, -0.475]\,, \dot{x}(0) \in [0.01, 0.0101]$

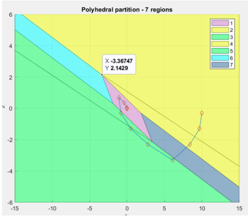
## Pendule




Intervalle très petit

# Conclusion et Perspectives

- Généraliser la preuve de mountain car a tout l'intervalle de départ
- Utiliser de nouvelles méthodes d'atteignabilité de systèmes contrôlés par des réseaux de neurones (approche ReLUVal, conditions de continuité de l'acteur DDPG...)
- Approcher un Model Predictive Control (MPC) par un réseau de neurones pour le contrôle du système




MPC - Principe

# Références


Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra.
**Continuous control with deep reinforcement learning, 2015.**

Weiming Xiang, Hoang-Dung Tran, Xiaodong Yang, and Taylor T. Johnson.
**Reachable set estimation for neural network control systems : A simulation-guided approach, 2020.**

P.J. Meyer, A. Devonport, and M. Arcak.
**Interval Reachability Analysis : Bounding Trajectories of Uncertain Systems with Boxes for Control and Verification.**
SpringerBriefs in Electrical and Computer Engineering. 2021.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba.
**Openai gym, 2016.**

Thierry Lecomte, Thierry Servat, and Guilhem Pouzancre.
**Formal methods in safety-critical railway systems.**
08 2007.

Anthony Corso and Mykel J. Kochenderfer.
**Interpretable safety validation for autonomous vehicles.**
*CoRR*, 2020.