République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

**UNIVERSITÉ MOULOUD MAMMERI DE TIZI-OUZOU**

**FACULTÉ DU GÉNIE ÉLECTRIQUE ET D'INFORMATIQUE**
**DÉPARTEMENT D'AUTOMATIQUE**

## Mémoire de Fin d'Études
## de MASTER ACADEMIQUE

Domaine : Sciences et Technologies
Filière : Automatique
Spécialité : Automatique et Informatique Industrielle

Présenté par

**Rayane AKKOUCHE**

**Thème :**

# Contrôleur PID d'ordre fractionnaire réglé par réseau de neurones

*Mémoire soutenu le .../.../2026 devant le jury composé de :*

| | | |
|---|---|---|
| M. Prénom NOM | Grade, Lieu d'exercice | Président |
| Mme Karima AMOURA | Grade, UMMTO | Encadrante |
| M. Prénom NOM | Grade, Lieu d'exercice | Examinateur |
| M. Prénom NOM | Grade, Lieu d'exercice | Examinateur |

# Remerciements

# Dédicace

# Résumé

Ce mémoire porte sur la synthèse d'un contrôleur PID d'ordre fractionnaire (FOPID) dont les paramètres sont réglés à l'aide d'un réseau de neurones artificiels, appliqué à la commande de la vitesse d'un véhicule électrique hybride. L'objectif est d'imposer à la boucle fermée un comportement robuste d'iso-amortissement par l'intermédiaire d'un modèle de référence fractionnaire.

Dans un premier temps, les notions fondamentales des systèmes d'ordre fractionnaire sont présentées, notamment le dérivateur généralisé et son approximation par la méthode d'Oustaloup. Le modèle de référence fractionnaire d'iso-amortissement est ensuite introduit et analysé.

Le deuxième chapitre traite du contrôleur PID classique et du contrôleur FOPID, leurs structures, leurs types et leurs méthodes de réglage, incluant une application numérique. Le troisième chapitre est consacré aux réseaux de neurones artificiels : architecture, types, apprentissage, régularisation et critères de performance.

Le quatrième chapitre présente le modèle du véhicule électrique hybride, sa modélisation et son analyse, ainsi que la conception complète du contrôleur Neuro-FOPID et les résultats de simulation. Le Neuro-FOPID réduit le dépassement à 0.29%, l'ISE d'un facteur 7 par rapport au PID classique, et maintient ces performances sur $\pm 30\%$ de variation de gain.

**Mots-clés :** Contrôleur FOPID, Réseau de neurones, Modèle de référence fractionnaire, Systèmes d'ordre fractionnaire, Véhicule électrique, Commande robuste, Iso-amortissement.

# Table des matières

# Table des figures

# Liste des tableaux

# Liste des Symboles

## Symboles Généraux et Mathématiques

| | |
|---|---|
| $s$ | Variable de Laplace (opérateur de dérivation) |
| $t$ | Temps (s) |
| $j$ | Unité imaginaire ($j^2 = -1$) |
| $\omega$ | Pulsation (rad/s) |
| $\omega_c$ | Fréquence de coupure ou de référence (rad/s) |
| $\Gamma(\cdot)$ | Fonction Gamma d'Euler |
| $e(t)$ | Signal d'erreur ($e(t) = r(t) - y(t)$) |
| $u(t)$ | Signal de commande (sortie du contrôleur) |
| $r(t)$ | Consigne (référence) |
| $y(t)$ | Sortie du système |

## Chapitre I : Systèmes d'Ordre Fractionnaire

| | |
|---|---|
| $D^\alpha$ | Opérateur de dérivée/intégrale fractionnaire d'ordre $\alpha$ |
| $\alpha, \lambda, \nu$ | Ordres fractionnaires (exposants réels) |
| $n$ | Ordre entier (dans les définitions Riemann-Liouville et Caputo) |
| $\omega_l$ | Fréquence basse de la bande d'approximation (rad/s) |
| $\omega_h$ | Fréquence haute de la bande d'approximation (rad/s) |
| $\tau_c$ | Constante de temps du modèle de référence fractionnaire |
| $\varphi_m$ | Marge de phase (rad ou deg) |
| $M_p$ | Dépassement (Overshoot) (%) |
| $T_s$ | Temps de stabilisation (s) |

## Chapitre II : Contrôleurs PID et FOPID

| | |
|---|---|
| $K_p$ | Gain proportionnel |
| $K_i$ | Gain intégral |
| $K_d$ | Gain dérivé |
| $T_i$ | Constante de temps intégrale (s) |
| $T_d$ | Constante de temps dérivée (s) |
| $N_f$ | Facteur de filtrage du terme dérivé |
| $\lambda$ | Ordre de l'intégration fractionnaire |
| $\mu$ | Ordre de la dérivation fractionnaire |
| IAE | Integral of Absolute Error |
| ITAE | Integral of Time-weighted Absolute Error |
| ISE | Integral of Squared Error |
| ITSE | Integral of Time-weighted Squared Error |

## Chapitre III : Réseaux de Neurones Artificiels

| | |
|---|---|
| $w_{ij}$ | Poids synaptique entre le neurone $j$ et le neurone $i$ |
| $b_i$ | Biais du neurone $i$ |
| $\eta$ | Taux d'apprentissage |
| $E$ | Fonction de coût (erreur d'entraînement) |

## Chapitre IV : Véhicule Électrique et Simulation

| | |
|---|---|
| $v$ | Vitesse du véhicule (m/s) |
| $\theta$ | Position du papillon des gaz (commande) |
| $F_e$ | Force générée par le moteur (N) |
| $m$ | Masse du véhicule (kg) |
| $\alpha_t$ | Coefficient de traînée aérodynamique (N/(m/s)$^2$) |
| $\tau_e$ | Constante de temps du moteur (s) |
| $G_v(s)$ | Fonction de transfert du véhicule ($V(s)/\Theta(s)$) |
| $T_{ref}(s)$ | Modèle de référence fractionnaire d'iso-amortissement |

# Introduction Générale

Depuis l'avènement de l'automatisme industriel, la commande des systèmes dynamiques n'a cessé d'évoluer pour répondre à des exigences toujours plus strictes en termes de précision, de rapidité et de robustesse. Au cœur de cette évolution se trouve le régulateur PID (Proportionnel, Intégral, Dérivé), dont la simplicité structurelle et l'efficacité pratique en ont fait l'outil de régulation dominant. Desborough et Miller rapportent que plus de 97% des contrôleurs industriels sont des régulateurs PID [16]. Cette omniprésence s'explique par sa capacité à bien contrôler une large gamme de processus, sa robustesse de mise en œuvre et son rapport qualité-prix avantageux. Cependant, face à des systèmes dynamiques de plus en plus complexes, non linéaires et soumis à des incertitudes paramétriques importantes, le PID classique trouve ses limites et devient incapable de satisfaire tous les besoins de performance [17].

Depuis les années soixante, l'extension des outils du calcul différentiel aux ordres non entiers a ouvert une nouvelle voie dans la modélisation et la commande des systèmes. La commande CRONE développée par Oustaloup [7] et le contrôleur $\mathrm{PI}^{\lambda}\mathrm{D}^{\mu}$ proposé par Podlubny [8] ont démontré la supériorité des contrôleurs fractionnaires sur leurs homologues entiers, notamment en termes de robustesse vis-à-vis des variations des paramètres du processus. Le PID fractionnaire (FOPID) dispose en effet de deux degrés de liberté supplémentaires — les ordres fractionnaires $\lambda$ et $\mu$ — qui permettent un réglage plus fin et un découplage entre rapidité et amortissement, difficilement atteignable avec un PID classique.

Malgré ses avantages, le réglage optimal d'un contrôleur FOPID à cinq paramètres ($K_p$, $K_i$, $K_d$, $\lambda$, $\mu$) demeure une tâche délicate. Les méthodes d'optimisation classiques — programmation quadratique séquentielle (SQP), algorithme du simplexe de Nelder-Mead — sont efficaces mais restent sensibles au point d'initialisation et peuvent converger vers des minima locaux. L'intelligence artificielle, et en particulier les réseaux de neurones artificiels, offre une alternative prometteuse : une fois entraîné hors ligne sur une base de données d'exemples optimaux, un réseau de neurones est capable de fournir instantanément les paramètres optimaux du FOPID pour n'importe quelle condition.

La robustesse constitue sans doute la propriété la plus recherchée dans un système de commande industriel. L'intégrateur fractionnaire d'ordre non entier proposé par Bode dès 1945 [9] comme modèle de boucle ouverte garantit une propriété remarquable

d'iso-amortissement : le dépassement en boucle fermée est insensible aux variations du gain de la chaîne d'action. Ce modèle de référence fractionnaire constitue aujourd'hui un cadre de référence reconnu pour la synthèse de régulateurs robustes [14].

L'application à la commande de vitesse des véhicules électriques illustre parfaitement les enjeux de cette problématique. Ce domaine, en pleine expansion dans le contexte de la transition énergétique mondiale, requiert des stratégies de commande capables de concilier précision, réactivité et robustesse face aux variations de la charge et des conditions routières.

**L'objectif de ce travail** est de concevoir un contrôleur FOPID dont les cinq paramètres sont calculés automatiquement par un réseau de neurones artificiel entraîné à minimiser un critère d'erreur composite, en imposant à la boucle fermée un comportement fractionnaire robuste d'iso-amortissement. La stratégie de commande est validée par simulation sur le modèle de vitesse d'un véhicule électrique hybride et comparée au PID classique et au FOPID réglé par optimisation numérique directe.

Ce mémoire s'articule en quatre chapitres :

Le **Chapitre I** est consacré aux généralités sur les systèmes d'ordre fractionnaire. Nous y présentons les opérateurs différentiels selon Riemann-Liouville et Caputo, le dérivateur généralisé, les méthodes d'approximation de Charef et d'Oustaloup, ainsi que le modèle de référence fractionnaire d'iso-amortissement et ses propriétés.

Dans le **Chapitre II**, nous traitons les contrôleurs PID classique et FOPID. Après une description des structures du PID, nous présentons le FOPID, ses différents types, les méthodes de réglage par optimisation et un exemple d'application numérique.

Le **Chapitre III** est entièrement consacré aux réseaux de neurones artificiels. Nous y détaillons le neurone formel, les différents types de réseaux (MLP, CNN, RNN...), les fonctions d'activation, l'algorithme de rétropropagation et les critères d'erreur.

Le **Chapitre IV** présente la modélisation du véhicule électrique hybride ainsi que la conception complète et les résultats de simulation du contrôleur Neuro-FOPID, incluant la comparaison avec les contrôleurs de référence, les tests de robustesse et le rejet de perturbation.

Enfin, une **Conclusion Générale** résume les principaux résultats et propose des perspectives de travaux futurs.

Chapitre 1

# Simulation des systèmes d'ordre fractionnaires

## 1.1 Introduction

Le calcul d'ordre fractionnaire est un domaine qui traite des dérivées et des intégrales d'ordre non entier. Avec l'avancée de la science et le développement de l'outil informatique, l'intérêt de la dérivation et de l'intégration d'ordre non entier ne cesse de croître, notamment dans le domaine de l'automatique pour la modélisation, l'identification et la commande des systèmes.

Dans le domaine du contrôle, les systèmes d'ordre fractionnaire apportent un réel avantage. Un bon exemple est l'utilisation de la fonction idéale de Bode [9] comme modèle de référence robuste : en raison de ses caractéristiques d'iso-amortissement, elle sert souvent de référence pour ajuster les contrôleurs PID fractionnaires [5]. Ce type de contrôleur offre une meilleure robustesse et de meilleures performances lorsque les conditions changent ou que le système est complexe.

Ce chapitre a pour but de se familiariser avec l'univers des systèmes d'ordre fractionnaire. On y présente certains fondements clés ainsi que les différentes méthodes d'approximation des dérivateurs, notamment la méthode de Charef [2] et celle d'Oustaloup [7], largement utilisées pour la simulation des systèmes d'ordre fractionnaire.

## 1.2 Systèmes d'ordre fractionnaire

Un système d'ordre fractionnaire est décrit par une équation différentielle impliquant des dérivées fractionnaires :

$$a_n D^{\alpha_n} y(t) + \cdots + a_1 D^{\alpha_1} y(t) = b_m D^{\beta_m} u(t) + \cdots + b_0 D^{\beta_0} u(t), \tag{1.1}$$

où $D^\alpha$ représente l'opérateur de dérivée d'ordre $\alpha \in \mathbb{R}^+$, défini au sens de Riemann-Liouville ou Caputo [1, 8]. Ces formalismes capturent des dynamiques non exponentielles, comme les relaxations en loi de puissance.

### 1.2.1 Opérateurs différentiels fractionnaires

Un opérateur fractionnaire généralise les dérivées et intégrales classiques à des ordres réels. Pour $\alpha > 0$, l'opérateur dérive d'ordre $\alpha$ ; pour $\alpha < 0$, il intègre d'ordre $|\alpha|$.

**Dérivée de Riemann-Liouville [8] :**

$$D^\alpha f(t) = \frac{1}{\Gamma(n-\alpha)} \frac{d^n}{dt^n} \int_a^t \frac{f(\tau)}{(t-\tau)^{\alpha-n+1}}\, d\tau, \quad n-1 < \alpha < n, \tag{1.2}$$

où $n$ est l'entier tel que $n - 1 < \alpha < n$.

**Dérivée de Caputo [1] :**

$$D^\alpha f(t) = \frac{1}{\Gamma(n-\alpha)} \int_a^t \frac{f^{(n)}(\tau)}{(t-\tau)^{\alpha-n+1}}\, d\tau, \quad n-1 < \alpha < n. \tag{1.3}$$

La formulation de Caputo est préférable en commande car elle permet d'utiliser des conditions initiales exprimées en termes de dérivées entières, plus accessibles physiquement. La fonction $\Gamma$ est la généralisation de la factorielle aux réels :

$$\Gamma(\nu) = \int_0^{+\infty} e^{-x}\, x^{\nu-1}\, dx. \tag{1.4}$$

### 1.2.2 Dérivateur généralisé

Le dérivateur généralisé est un opérateur linéaire noté $s^\alpha$ ($\alpha \in \mathbb{R}$), généralisant la dérivation et l'intégration. Une variante courante inclut une fréquence de normalisation $\omega_c$ [8] :

$$D_{\text{gen}}(s) = \left(\frac{s}{\omega_c}\right)^\alpha, \quad \alpha \in \mathbb{R}. \tag{1.5}$$

Dans le domaine fréquentiel :

$$D_{\text{gen}}(j\omega) = \left(\frac{j\omega}{\omega_c}\right)^\alpha = \left|\frac{\omega}{\omega_c}\right|^\alpha e^{j\alpha\pi/2}. \tag{1.6}$$

Cette forme préserve un gain en $(\omega/\omega_c)^\alpha$ et une phase constante de $\alpha\pi/2$. Cette phase constante sur une large plage de fréquences rend sa réalisation physique complexe et nécessite des approximations à bande limitée.

### 1.2.3 Méthodes d'approximation

Le dérivateur généralisé $s^\alpha$ constitue l'élément principal pour la synthèse des systèmes fractionnaires. Plusieurs méthodes d'approximation sont proposées dans la littérature ; les plus utilisées dans le domaine continu sont la méthode de Charef et la méthode d'Oustaloup [8].

### 1.2.4 Méthode de Charef

La méthode de Charef utilise une cascade de pôles et zéros pour reproduire la pente $\alpha$ sur une plage de fréquences limitée [2] :

$$H_{\text{Charef}}(s) = K \prod_{k=1}^{N} \frac{1 + s/\omega_{z_k}}{1 + s/\omega_{p_k}}, \tag{1.7}$$

où les pôles et zéros sont distribués récursivement :

$$a = 10^{\varepsilon/[10(1-\alpha)]}, \quad b = 10^{\varepsilon/(10\alpha)}, \qquad \omega_{p_k} = \omega_{p_0}(ab)^k, \quad \omega_{z_k} = a \cdot \omega_{p_0}(ab)^k, \tag{1.8}$$

avec $N = \lceil \log(\omega_{\max}/\omega_{p_0}) / \log(ab) \rceil + 1$.

### 1.2.5 Méthode d'Oustaloup

La méthode d'Oustaloup approxime $s^\alpha$ sur une bande $[\omega_l, \omega_h]$ avec $2N+1$ pôles et zéros répartis symétriquement [7] :

$$H_{\text{Oustaloup}}(s) = \omega_h^\alpha \prod_{k=-N}^{N} \frac{1 + s/\omega_{z_k}}{1 + s/\omega_{p_k}}, \tag{1.9}$$

avec :

$$\omega_{z_k} = \omega_l \left(\frac{\omega_h}{\omega_l}\right)^{\frac{k+N+1-\alpha/2}{2N+1}}, \quad \omega_{p_k} = \omega_l \left(\frac{\omega_h}{\omega_l}\right)^{\frac{k+N+1+\alpha/2}{2N+1}}. \tag{1.10}$$

Un choix typique est $N = 5$ pour une erreur inférieure à 2 dB [5].

TABLE 1.1 – Comparaison entre les méthodes d'approximation de Charef et d'Oustaloup.

| **Méthode** | **Précision** | **Bande** | **Complexité** | **Usage** |
|---|---|---|---|---|
| Charef | Moyenne à bonne | Limitée | Faible | Approximations rapides |
| Oustaloup | Très bonne | Large | Élevée | Commande robuste |

Dans ce travail, la méthode d'Oustaloup est retenue en raison de sa précision et de son intégration dans la FOMCON Toolbox MATLAB.

## 1.3 Modèle de référence fractionnaire d'iso-amortissement

L'iso-amortissement est une propriété très souhaitable dans un système asservi : le dépassement en boucle fermée est insensible aux variations du gain de la boucle ouverte, conférant une robustesse structurelle au régulateur. Bode a proposé dès 1945 [9] un intégrateur d'ordre fractionnaire comme fonction de transfert en boucle ouverte garantissant cette propriété. Une phase constante sur une large plage de fréquences évite les variations brutales de marge de phase, stabilisant le système malgré les incertitudes de gain. Cette fonction est appelée *fonction idéale de Bode* :

$$L(s) = \frac{1}{\tau_c s^{\lambda+1}}, \tag{1.11}$$

où $\tau_c = 1/\omega_c^{\lambda+1}$, $\lambda = \alpha - 1 \in (0,1)$, et $\omega_c$ est la fréquence de coupure. Son diagramme de Bode présente un gain de $-20\alpha$ dB/décade et une phase constante $-\alpha\pi/2$.

### 1.3.1 Modèle de référence en boucle fermée

La fonction de transfert en boucle fermée correspondante est :

$$F(s) = \frac{1}{\tau_c s^{\lambda+1} + 1}, \quad 0 < \lambda < 1. \tag{1.12}$$

En boucle fermée, le système se comporte similairement à un système du second ordre amorti, avec deux paramètres ajustables indépendamment.

### 1.3.2 Analyse des paramètres du modèle de référence en boucle fermée

En fixant $\tau_c$ et en faisant varier $\lambda$, le dépassement augmente avec $\lambda$ tandis que la rapidité reste inchangée. En fixant $\lambda$ et en faisant varier $\tau_c$, la rapidité change mais le dépassement reste constant — c'est la **propriété d'iso-amortissement**. Les formules de correspondance avec les systèmes entiers sont [5, 14] :

$$M_p\,(\%) = 0.8\lambda(\lambda + 0.25), \quad T_s(5\%) \approx \frac{3}{\cos\!\left(\pi - \frac{\pi}{\lambda+1}\right)}\tau_c, \quad \phi_\alpha = \pi - (\lambda+1)\frac{\pi}{2}. \tag{1.13}$$

### 1.3.3 Robustesse du modèle de référence fractionnaire

La marge de phase est invariante avec la fréquence ($\approx -\alpha\pi/2$), ce qui confère une stabilité robuste. Cette propriété d'iso-amortissement maintient les performances malgré les variations de gain, faisant de la fonction idéale de Bode un modèle de référence par excellence [14].

## 1.4 Conclusion

Dans ce chapitre, nous avons présenté les fondements des systèmes d'ordre fractionnaire : le dérivateur généralisé, les opérateurs de Riemann-Liouville et Caputo, et les méthodes d'approximation de Charef et d'Oustaloup. La méthode d'Oustaloup est retenue pour sa précision et son intégration logicielle. Le modèle de référence fractionnaire d'iso-amortissement a été étudié en boucle fermée : ses deux paramètres $\lambda$ et $\tau_c$ permettent d'ajuster indépendamment l'amortissement et la rapidité, conférant une robustesse structurelle intrinsèque.

## Chapitre 2

# Contrôleurs PID Classique et FOPID

## 2.1 Introduction

La régulation est l'âme technique de l'automatisme. Le contrôleur PID s'est imposé comme le cœur de l'automatisme industriel depuis près d'un siècle, grâce à sa structure simple et son efficacité démontrée. Cependant, face aux systèmes complexes, le PID classique présente des limitations qui ont motivé le développement de techniques plus avancées.

Le PID fractionnaire (FOPID), ou $\mathrm{PI}^{\lambda}\mathrm{D}^{\mu}$, se présente comme une extension puissante intégrant des ordres fractionnaires ($\lambda$ et $\mu$) aux actions intégrale et dérivée. Ses cinq paramètres ($K_p$, $K_i$, $K_d$, $\lambda$, $\mu$) offrent une flexibilité de réglage supérieure, mais au prix d'une complexité accrue.

Dans ce chapitre, nous présentons les contrôleurs PID et FOPID, leurs structures et leurs propriétés, ainsi que les méthodes de réglage par optimisation, avec un exemple d'application numérique et une analyse fréquentielle comparative.

## 2.2 PID Classique

En automatique, le régulateur PID améliore les performances d'un asservissement en corrigeant les écarts entre la consigne et la sortie. Il possède trois actions fondamentales [6] :

- **Action proportionnelle (P) :** $u_p(t) = K_p e(t)$. Réagit immédiatement à l'écart mais laisse subsister une erreur statique résiduelle.
- **Action intégrale (I) :** $u_i(t) = K_i \int_0^t e(\tau)\, d\tau$. Supprime l'erreur statique mais peut provoquer des oscillations si $K_i$ est trop élevé.
- **Action dérivée (D) :** $u_d(t) = K_d \frac{d}{dt} e(t)$. Anticipe les changements futurs et limite les dépassements, mais est sensible au bruit.

La commande globale est :

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)\, d\tau + K_d \frac{d}{dt} e(t), \tag{2.1}$$

soit en Laplace : $C(s) = K_p + K_i/s + K_d s$.

### 2.2.1 Différentes structures du PID

Le PID classique possède trois structures principales qui diffèrent par l'organisation des actions P, I et D.

**Structure parallèle :** Chaque terme agit directement sur l'erreur. C'est la structure la plus utilisée, qui facilite le réglage indépendant de chaque paramètre :

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)\, d\tau + K_d \frac{de(t)}{dt}. \tag{2.2}$$

**Structure série :** Le régulateur est disposé comme une chaîne de blocs successifs, avec une interaction naturelle entre paramètres :

$$C(s) = K \left(1 + \frac{1}{T_i s}\right)(1 + T_d s). \tag{2.3}$$

**Structure mixte :** La structure mixte combine les actions de manière optimisée pour réduire la sensibilité au bruit et améliorer la stabilité :

$$u(t) = K_p \left[ e(t) + \frac{1}{T_i} \int e(t)\, dt - T_d \frac{d}{dt} y(t) \right]. \tag{2.4}$$

## 2.3 PID d'ordre fractionnaire (FOPID)

### 2.3.1 Définition et représentation dans l'espace des paramètres

Le FOPID est une extension du PID classique où les actions intégrale et dérivée sont d'ordre fractionnaire [8] :

$$C(s) = K_p + \frac{K_i}{s^\lambda} + K_d s^\mu, \tag{2.5}$$

où $\lambda > 0$ est l'ordre de l'intégration et $\mu > 0$ est l'ordre de la dérivation. Pour $\lambda = \mu = 1$ on retrouve le PID classique. La figure 2.1 illustre l'espace des paramètres $(\lambda, \mu)$.

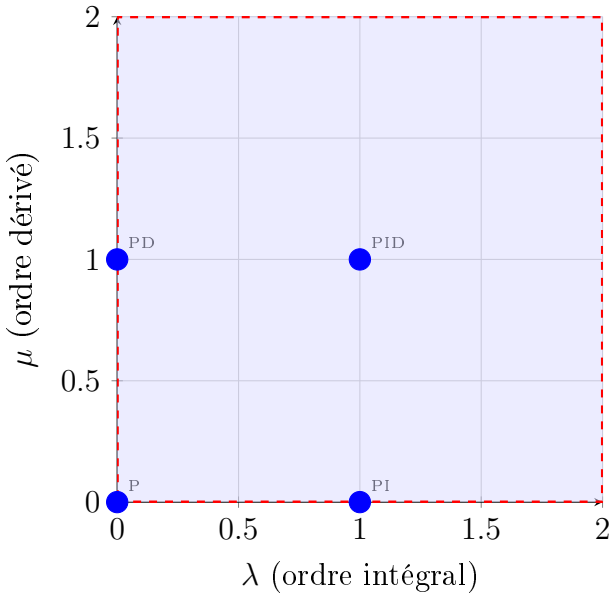


FIGURE 2.1 – Espace des paramètres $(\lambda, \mu)$ du contrôleur $PI^{\lambda}D^{\mu}$ [8].

### 2.3.2 Différents types du PID fractionnaire

Plusieurs variantes du FOPID sont proposées dans la littérature, chacune adaptée à des contextes d'application spécifiques.

**$PI^{\lambda}D^{\mu}$ basique :** La forme standard $C(s) = K_p + K_i s^{-\lambda} + K_d s^{\mu}$ offre cinq paramètres libres pour un réglage flexible.

**FOPID avec filtre :** Pour rendre $s^{\mu}$ physiquement réalisable et réduire la sensibilité au bruit :

$$C(s) = K_p + K_i s^{-\lambda} + \frac{K_d s^{\mu}}{1 + \tau s}. \tag{2.6}$$

**FOF-PID [19] :** Un PID classique en série avec un filtre fractionnaire issu de la combinaison IMC-CRONE :

$$C(s) = F(s)\left[K_c\left(\frac{1}{\tau_i s} + \tau_d s + 1\right)\right], \quad F(s) = \frac{1}{1 + \tau_f s^{\gamma}}. \tag{2.7}$$

## 2.4 Méthodes de réglage du PID et FOPID

### 2.4.1 Méthodes de réglage classiques du PID

Il existe plusieurs méthodes pour calculer les paramètres du PID classique [20] : la méthode de Ziegler-Nichols (basée sur le gain critique), les méthodes par modèle interne

(IMC), et les méthodes d'optimisation numérique. Ces dernières ont été étendues au réglage du FOPID.

### 2.4.2 Méthodes d'optimisation pour le réglage du FOPID

Les méthodes d'optimisation déterminent les paramètres optimaux du contrôleur en minimisant un critère de performance prédéfini. Les critères d'erreur intégraux couramment utilisés sont [11] :

$$\text{IAE} = \int_0^T |e(t)|\,dt, \quad \text{ITAE} = \int_0^T t|e(t)|\,dt, \quad \text{ISE} = \int_0^T e^2(t)\,dt. \tag{2.8}$$

**Méthode SQP [15] :** La programmation quadratique séquentielle résout séquentiellement des sous-problèmes quadratiques approximant le problème original :

$$\min_d \ \nabla f(x_k)^T d + \tfrac{1}{2} d^T H_k d, \quad \text{s.c. } c(x_k) + \nabla c(x_k)^T d \leq 0. \tag{2.9}$$

La convergence est super-linéaire sous des hypothèses de régularité classiques.

**Méthode de Nelder-Mead [13] :** Un algorithme sans gradient utilisant un simplexe de $n+1$ points qui évolue par réflexion, expansion et contraction. Robuste aux discontinuités mais sans garantie de convergence globale.

**Limitation commune :** Ces méthodes dépendent du point d'initialisation et peuvent converger vers des minima locaux — ce qui motive l'approche par réseau de neurones de la section 4.6.

### 2.4.3 Exemple d'application : réglage d'un FOPID par `fmincon`

Considérons le système de troisième ordre de Barbosa et al. [14] : $G(s) = 1/(s+1)^3$, avec le modèle de référence $F(s) = 1/(1+\tau_c s^{\lambda+1})$ ($\tau_c = 1$, $\lambda = 0.25$).

TABLE 2.1 – Paramètres des contrôleurs obtenus par optimisation (système de Barbosa).

| **Paramètre** | **FOPID** | **PID Classique** |
|---|---|---|
| $K_c$ | 4.0355 | 2.5891 |
| $T_i$ | 1.2308 | 1.6625 |
| $T_d$ | 2.5781 | 2.7760 |
| $\lambda$ | 1.1906 | N/A |
| $\mu$ | 1.3616 | N/A |

La figure 2.2 compare les réponses indicielles des trois systèmes. Le FOPID reproduit fidèlement le modèle de référence grâce à ses deux degrés de liberté supplémentaires, tandis que le PID classique présente des oscillations amorties plus marquées.

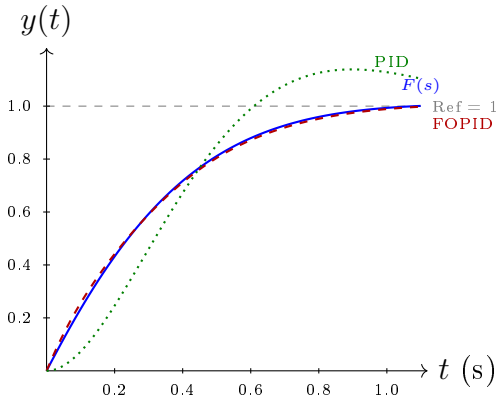


FIGURE 2.2 – Réponses indicielles : PID, FOPID et modèle de référence $F(s)$ (système de Barbosa, $\tau_c = 1$, $\lambda = 0.25$).

Le tableau 2.2 quantifie les performances.

TABLE 2.2 – Comparaison des performances temporelles : PID vs FOPID vs modèle de référence.

| **Critère** | **Modèle** $F(s)$ | **FOPID** | **PID** |
|---|---|---|---|
| Dépassement $M_p$ (%) | 0 | $< 5$ | $\sim 15$ |
| Suivi du modèle de réf. | — | excellent | approximatif |
| $\lambda$ / $\mu$ | — | 1.19 / 1.36 | N/A |

Ces résultats confirment la supériorité paramétrique du FOPID. La complexité de son réglage motive le recours au réseau de neurones présenté au Chapitre III.

### 2.4.4 Analyse fréquentielle comparative : PID vs FOPID

La comparaison des réponses fréquentielles en boucle ouverte met en évidence la différence structurelle fondamentale entre les deux régulateurs appliqués au système du véhicule $G_v(s)$.

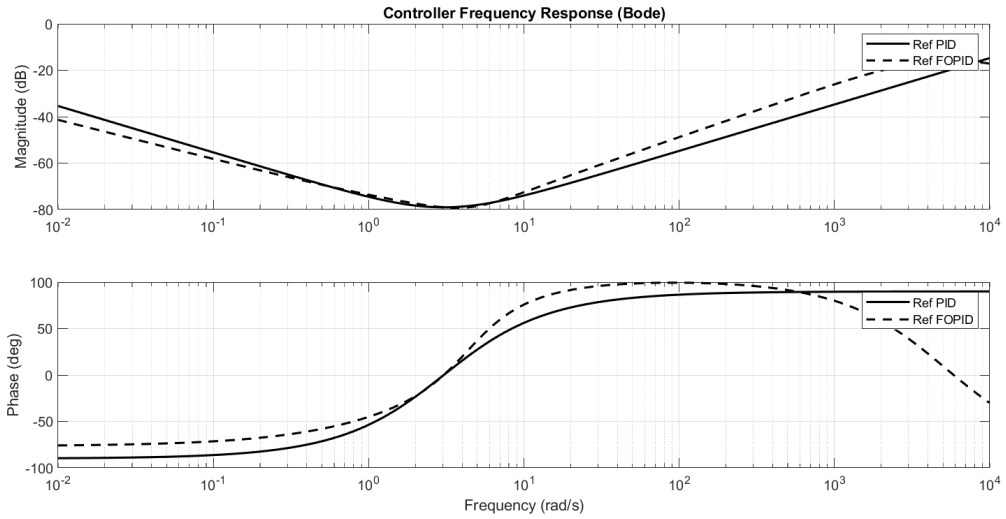


FIGURE 2.3 – Diagrammes de Bode en boucle ouverte pour le PID-Réf et le FOPID-Réf appliqués au système du véhicule $G_v(s)$.

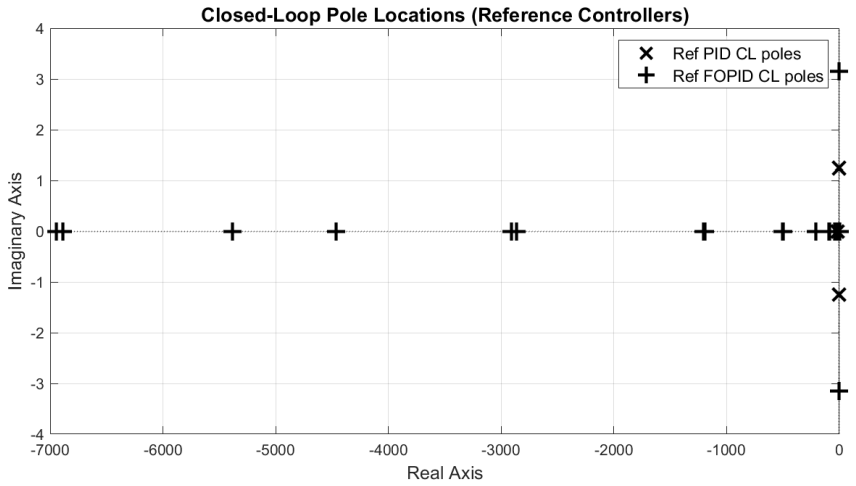


FIGURE 2.4 – Emplacements des pôles en boucle fermée pour le PID-Réf et le FOPID-Réf.

Le PID entier dispose d'un intégrateur $s^{-1}$ qui contribue $-90°$ de déphasage et d'une dérivée $s^1$ qui contribue $+90°$. Le FOPID, avec $s^{-\lambda}$ ($\lambda = 0.75$), ne contribue que $-67.5°$, tandis que $s^{\mu}$ ($\mu = 1.25$) apporte $+112.5°$ de phase d'avance. Il en résulte une réponse de phase **plus plate sur une plus large bande de fréquences** — c'est la propriété d'iso-amortissement qui rend le FOPID structurellement plus robuste aux variations de gain.

## 2.5 Conclusion

Dans ce chapitre, nous avons présenté les contrôleurs PID classique et FOPID, leurs structures et leurs propriétés. Les deux degrés de liberté supplémentaires ($\lambda$ et

$\mu$) du FOPID offrent une flexibilité accrue pour le découplage précision-robustesse. L'analyse fréquentielle a confirmé que la phase plate du FOPID sur une large bande est le mécanisme sous-jacent de sa robustesse structurelle. Les méthodes d'optimisation classiques, bien qu'efficaces, restent sensibles à l'initialisation — ce qui motive le recours aux réseaux de neurones du chapitre suivant.

Chapitre 3

# Réseaux de Neurones Artificiels

## 3.1 Introduction

Les **réseaux de neurones artificiels** (RNA) constituent aujourd'hui l'un des outils les plus puissants de l'apprentissage automatique. Inspirés du fonctionnement du cerveau biologique, ils sont capables d'apprendre des représentations complexes à partir de données et d'approximer des fonctions non linéaires arbitrairement complexes [22]. Leur capacité à généraliser à partir d'exemples en fait des candidats naturels pour résoudre des problèmes d'optimisation, d'identification et de régression difficiles.

Ce chapitre présente les fondements théoriques des réseaux de neurones artificiels : le neurone formel, les différents types de réseaux, les fonctions d'activation, l'algorithme de rétropropagation, les techniques de régularisation et les critères d'évaluation.

## 3.2 Le Neurone Formel

### 3.2.1 Modèle de McCulloch-Pitts

Le neurone formel, introduit par McCulloch et Pitts en 1943, est l'unité de calcul élémentaire d'un réseau de neurones [21]. Il réalise une somme pondérée de ses entrées, augmentée d'un biais, puis applique une fonction d'activation non linéaire :

$$y_i = f\left(\sum_{j=1}^{n} w_{ij}\, x_j + b_i\right) = f(z_i), \tag{3.1}$$

où $w_{ij}$ est le poids synaptique, $b_i$ le biais, $z_i$ la préactivation, et $f(\cdot)$ la fonction d'activation. En notation matricielle : $\mathbf{y} = f(\mathbf{W}\mathbf{x} + \mathbf{b})$.

## 3.3 Types de Réseaux de Neurones

Les réseaux de neurones se déclinent en de nombreuses architectures, chacune adaptée à des types de problèmes spécifiques.

### 3.3.1 Perceptron Simple

Le perceptron simple est le modèle le plus élémentaire : un neurone unique à seuil qui ne peut résoudre que des problèmes linéairement séparables [21].

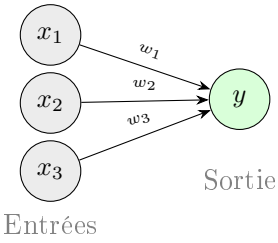


FIGURE 3.1 – Architecture du perceptron simple.

### 3.3.2 Perceptron Multi-Couche (MLP)

Le Perceptron Multi-Couche (MLP) est l'architecture la plus répandue pour la régression non linéaire, organisée en couche d'entrée, couches cachées et couche de sortie [21]. Le théorème d'approximation universelle [22] garantit qu'un MLP avec une seule couche cachée suffisamment large peut approximer n'importe quelle fonction continue.

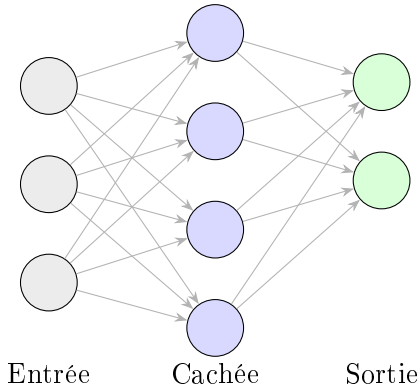


FIGURE 3.2 – Architecture d'un Perceptron Multi-Couche (MLP) à une couche cachée.

### 3.3.3 Réseaux de Neurones Récurrents (RNN)

Les RNN introduisent des connexions récurrentes permettant de traiter des séquences temporelles [21]. L'état caché à l'instant $t$ est calculé à partir de l'entrée cou-

rante et de l'état précédent :

$$\mathbf{h}_t = f(\mathbf{W}_h \mathbf{h}_{t-1} + \mathbf{W}_x \mathbf{x}_t + \mathbf{b}). \tag{3.2}$$

Les réseaux LSTM résolvent le problème du gradient évanescent des RNN classiques.

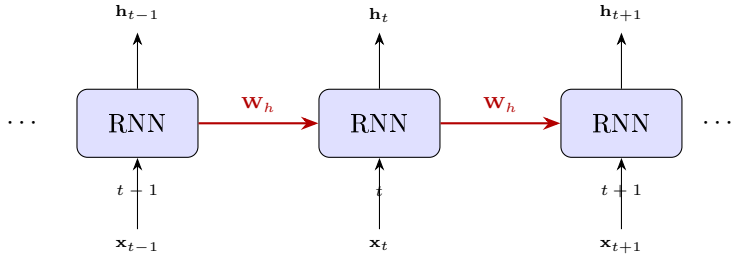


FIGURE 3.3 – Réseau de neurones récurrent déplié dans le temps — les connexions rouges transmettent l'état caché.

### 3.3.4 Réseaux de Neurones Convolutifs (CNN)

Les CNN utilisent le partage de poids pour traiter des données à structure locale. Une couche convolutive calcule, pour le filtre $k$ :

$$Z_{ij}^{(k)} = \sum_{m,n} W_{mn}^{(k)} X_{(i+m)(j+n)} + b^{(k)}, \tag{3.3}$$

suivie d'une activation non linéaire et d'une opération de pooling.

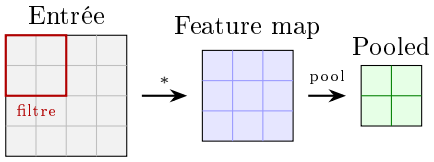


FIGURE 3.4 – Principe d'une couche convolutive : convolution + pooling.

### 3.3.5 Autoencodeurs et réseaux RBF

Un autoencodeur apprend à reconstruire son entrée via un code latent comprimé, utilisé pour la réduction de dimensionnalité et la détection d'anomalies. Les réseaux RBF utilisent des fonctions gaussiennes $\phi(\|\mathbf{x} - \mathbf{c}_k\|)$ et présentent une approximation locale rapide.

## 3.4 Fonctions d'Activation

Les fonctions d'activation brisent la linéarité du réseau et lui confèrent sa capacité d'approximation universelle. Leur choix influence directement la vitesse de convergence

et les performances du réseau.

### 3.4.1 Sigmoïde logistique

$\sigma(x) = 1/(1+e^{-x}) \in (0,1)$, avec dérivée $\sigma'(x) = \sigma(x)(1-\sigma(x))$. La sigmoïde souffre du problème de gradient évanescent pour $|x| \gg 1$.

### 3.4.2 Tangente hyperbolique

$\tanh(x) = (e^x - e^{-x})/(e^x + e^{-x}) \in (-1,1)$, centrée en zéro ce qui accélère la convergence. Sa dérivée est $\tanh'(x) = 1 - \tanh^2(x)$.

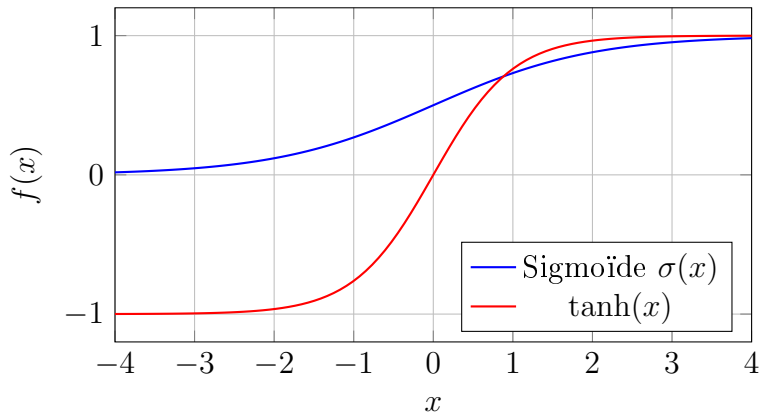


FIGURE 3.5 – Fonctions d'activation sigmoïde et tangente hyperbolique.

### 3.4.3 Rectified Linear Unit (ReLU)

$\mathrm{ReLU}(x) = \max(0, x)$, fonction standard pour les couches cachées des réseaux profonds. Elle ne sature pas pour $x > 0$, atténuant le gradient évanescent. La Leaky ReLU améliore encore ceci en ajoutant une pente $\alpha_{\mathrm{lr}} x$ pour $x \leq 0$.

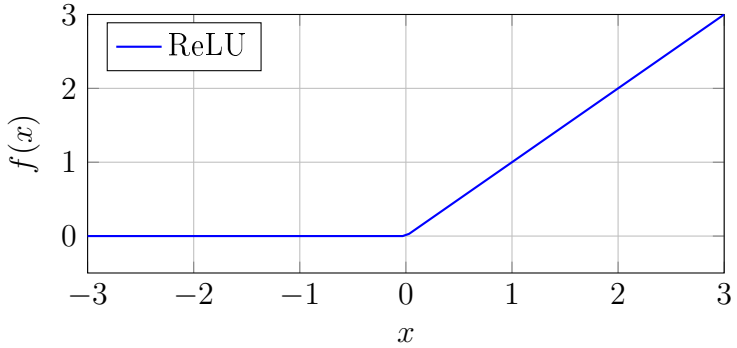


FIGURE 3.6 – Fonction d'activation ReLU.

TABLE 3.1 – Comparaison des principales fonctions d'activation.

| **Fonction** | **Plage** | **Centrage** | **Grad. évanescent** | **Calcul** |
|---|---|---|---|---|
| Sigmoïde | $(0, 1)$ | Non | Oui (fort) | Moyen |
| tanh | $(-1, 1)$ | Oui | Oui (modéré) | Moyen |
| **ReLU** | $[0, +\infty)$ | Non | **Non** (pour $x > 0$) | **Faible** |
| Leaky ReLU | $(-\infty, +\infty)$ | Non | Non | Faible |

## 3.5 Apprentissage par Rétropropagation du Gradient

### 3.5.1 Fonction de coût

L'apprentissage consiste à minimiser la MSE (erreur quadratique moyenne) sur un ensemble de $P$ exemples :

$$\mathcal{L}(\mathbf{W}, \mathbf{b}) = \frac{1}{P} \sum_{k=1}^{P} \|\hat{\mathbf{y}}_k - \mathbf{y}_k\|^2 . \tag{3.4}$$

### 3.5.2 Algorithme de rétropropagation

L'algorithme de rétropropagation [23] calcule le gradient par la règle de dérivation en chaîne en deux passes. La *forward pass* calcule les activations couche par couche ; la *backward pass* propage les erreurs en sens inverse :

$$\boldsymbol{\delta}^{(L)} = \frac{\partial \mathcal{L}}{\partial \mathbf{z}^{(L)}}, \qquad \boldsymbol{\delta}^{(l)} = \left(\mathbf{W}^{(l+1)}\right)^T \boldsymbol{\delta}^{(l+1)} \odot f'^{(l)}\left(\mathbf{z}^{(l)}\right) . \tag{3.5}$$

### 3.5.3 Règles de mise à jour des poids

La descente de gradient classique (SGD) met à jour les poids par $\mathbf{W} \leftarrow \mathbf{W} - \eta\, \partial\mathcal{L}/\partial\mathbf{W}$. L'algorithme Adam [24] améliore SGD par une estimation adaptative des moments :

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t, \quad v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2, \tag{3.6}$$

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t, \tag{3.7}$$

avec $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$. L'algorithme de Levenberg-Marquardt (`trainlm`) exploite le second ordre via la Jacobienne : $\Delta\mathbf{w} = -(\mathbf{J}^T\mathbf{J} + \mu_{\text{LM}}\mathbf{I})^{-1}\mathbf{J}^T\mathbf{e}$, efficace sur les petits réseaux mais coûteux pour les grandes bases.

## 3.6 Régularisation et Généralisation

### 3.6.1 Sur-apprentissage (*Overfitting*)

Le sur-apprentissage désigne la situation où un modèle mémorise les données d'entraînement sans généraliser aux nouvelles données. Il se manifeste par une divergence croissante entre la perte d'entraînement et la perte de validation (figure 3.7).

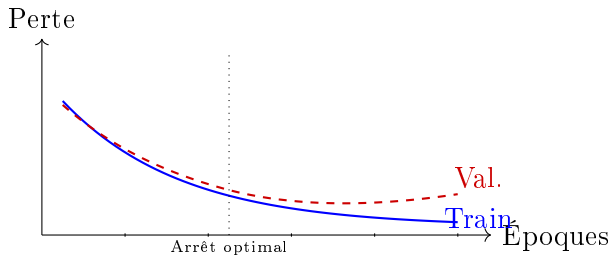


FIGURE 3.7 – Diagnostic du sur-apprentissage : divergence entre pertes d'entraînement et de validation.

**Causes principales :** capacité excessive du modèle, base insuffisante, entraînement trop prolongé, absence de régularisation.

**Stratégies de mitigation :** quatre approches sont employées, dont trois dans ce travail (augmentation des données, régularisation $L_2$, arrêt prématuré) :

1. **Augmentation des données :** les 463 conditions couvrent $\pm 50\%$ de variation paramétrique, diversifiant la base.
2. **Régularisation $L_2$ :** pénalité sur la norme des poids (§3.6.2).
3. **Arrêt prématuré :** stop au minimum de la perte de validation (§3.6.3).
4. **Dropout :** désactivation aléatoire de neurones (§3.6.4).

### 3.6.2 Régularisation $L_2$ (Weight Decay)

La régularisation de Tikhonov ajoute une pénalité sur la norme des poids :

$$\mathcal{L}_{\text{reg}} = \mathcal{L} + \frac{\lambda_{\text{reg}}}{2}\|\mathbf{w}\|^2. \tag{3.8}$$

### 3.6.3 Arrêt prématuré (*Early Stopping*)

L'entraînement est stoppé dès que l'erreur de validation cesse de décroître. Les données sont divisées en ensemble d'entraînement (mise à jour des poids), de validation (surveillance de la généralisation), et de test (évaluation finale).

### 3.6.4 Dropout

Le dropout consiste à désactiver aléatoirement une fraction $p$ des neurones à chaque itération, forçant le réseau à apprendre des représentations redondantes et robustes.

## 3.7 Critères d'évaluation des performances

### 3.7.1 Critères d'erreur intégraux

Les quatre critères intégraux classiques sont : IAE $= \int |e|\, dt$ (pénalise uniformément), ITAE $= \int t|e|\, dt$ (pénalise l'erreur persistante), ISE $= \int e^2\, dt$ (pénalise les grandes erreurs), ITSE $= \int te^2\, dt$ (combine les deux).

### 3.7.2 Métriques de régression

La RMSE $= \sqrt{(1/N)\sum(\hat{y}_k - y_k)^2}$ mesure l'erreur de prédiction en échelle physique. Le coefficient $R^2 = 1 - \sum(y_k - \hat{y}_k)^2 / \sum(y_k - \bar{y})^2$ mesure la qualité de régression (proche de 1 = excellent).

## 3.8 Conclusion

Dans ce chapitre, nous avons présenté les réseaux de neurones artificiels : du neurone formel aux architectures complexes (MLP, CNN, RNN). Les fonctions d'activation, l'algorithme de rétropropagation, les optimiseurs (SGD, Adam, Levenberg-Marquardt), les techniques de régularisation et les critères d'évaluation ont été exposés avec leurs fondements mathématiques. Ces éléments servent de cadre pour la conception du réseau Neuro-FOPID décrit dans la section 4.7.

Chapitre 4

# Contrôle de vitesse des véhicules électriques

## 4.1 Introduction

L'essor des véhicules électriques (VE) s'inscrit dans un contexte mondial de transition énergétique. Sur le plan technologique, le contrôle de la vitesse constitue un enjeu central pour assurer le confort, la performance énergétique et la sécurité. Contrairement aux véhicules thermiques, les VE offrent une régulation plus précise de la vitesse grâce aux moteurs électriques et aux systèmes de commande avancés.

Ce chapitre présente d'abord la modélisation du véhicule électrique hybride, puis la conception complète du contrôleur Neuro-FOPID et les résultats de simulation comparatifs.

## 4.2 Définition d'un véhicule électrique

Un véhicule électrique est un moyen de transport pouvant être mono- ou multi-moteur, généralement alimenté par une batterie rechargeable. Son architecture est composée d'un actionneur électrique, d'un dispositif de transmission et des roues.

## 4.3 Mode de fonctionnement d'un VE

Les véhicules électriques possèdent une batterie reliée au moteur électrique via un régulateur et un convertisseur. Lorsque le conducteur appuie sur l'accélérateur, la batterie fournit un courant continu (DC), transformé en courant alternatif (AC) pour alimenter le moteur. Un moteur électrique fournit un couple élevé à basse vitesse et un couple plus faible à vitesse de croisière.

### 4.3.1 Types de véhicules électriques

**Véhicule tout électrique (BEV) :** Alimenté uniquement par une batterie rechargeable. Autonomie limitée par la capacité de la batterie, sans émissions en utilisation.

**Véhicule à pile à combustible (FCEV) :** L'énergie est produite en continu à partir de l'hydrogène, contrairement aux BEV.

**Véhicule hybride :** Combine moteur thermique et moteur électrique. Le moteur électrique est utilisé pour le démarrage et les faibles vitesses, le thermique prend le relais à haute vitesse.

## 4.4 Spécifications fonctionnelles du système de commande de vitesse

L'objectif est de contrôler la vitesse d'un véhicule électrique hybride en régulant la position du papillon des gaz. Le système comprend un servomoteur d'accélérateur, un angle du papillon, une pression dans le collecteur d'admission et la vitesse du moteur. Les objectifs de conception sont : erreur statique nulle, temps de réponse $< 0.5$ s, et rejet de perturbations $> 90\%$.

## 4.5 Modélisation du véhicule hybride

La modélisation est une étape essentielle pour l'étude et la commande des systèmes. Le modèle non linéaire du système est donné par [19] :

$$m\frac{dv}{dt} = F_e(\theta) - \alpha_t v^2 - F_g, \tag{4.1}$$

$$\tau_e\frac{dF_e}{dt} = -F_e(\theta) + F_{e1}(\theta), \quad F_{e1}(\theta) = F_1 + \gamma_m\sqrt{\theta}, \tag{4.2}$$

avec $m = 1000$ kg, $\alpha_t = 4$ N/(m/s)$^2$, $F_1 = 6400$ N, $\gamma_m = 12500$ N, $\tau_e = 0.2$ s.

### 4.5.1 Linéarisation autour du point de fonctionnement

Au point d'équilibre $v_0 = 0$, la linéarisation conduit aux matrices d'état :

$$A = \begin{pmatrix} 0 & 0.001 \\ 0 & -5 \end{pmatrix}, \quad B = \begin{pmatrix} 0 \\ 8.29 \times 10^8 \end{pmatrix}, \quad C = \begin{pmatrix} 1 & 0 \end{pmatrix}, \quad D = [0]. \tag{4.3}$$

La fonction de transfert du véhicule est :

$$G_v(s) = \frac{V(s)}{\Theta(s)} = \frac{8.29 \times 10^5}{s(s+5)}. \tag{4.4}$$

### 4.5.2 Analyse de la commandabilité et de l'observabilité

La matrice de commandabilité $\mathcal{C} = [B\ AB]$ a $\det(\mathcal{C}) = -6.872 \times 10^{14} \neq 0$ : le système est **commandable**. La matrice d'observabilité $\mathcal{O} = [C^T\ (CA)^T]^T$ a $\det(\mathcal{O}) = 0.001 \neq 0$ : le système est **observable**.

La fonction de transfert $G_v(s) = 8.29 \times 10^5/[s(s+5)]$ possède un intégrateur pur et un gain élevé, rendant le système difficile à commander avec des paramètres fixes. Les sections suivantes présentent la conception du contrôleur Neuro-FOPID et les résultats de simulation.

## 4.6 Simulation et Résultats

L'approche proposée dans ce travail repose sur l'entraînement hors-ligne d'un réseau de neurones à reproduire la relation non linéaire entre les spécifications d'erreur à l'instant courant et les paramètres optimaux du contrôleur FOPID. Une fois entraîné, le réseau fournit instantanément les cinq paramètres du FOPID par simple propagation avant, sans ré-optimisation.

### 4.6.1 Type de réseau choisi et justification

L'architecture retenue est un **Perceptron Multi-Couche** (MLP). Le problème de réglage est un problème de **régression** (mapper $\mathbb{R}^3 \to \mathbb{R}^5$) ; le théorème d'approximation universelle [22] garantit qu'un MLP peut approximer la relation non linéaire recherchée. Contrairement aux CNN ou aux RNN, le MLP est directement adapté à l'entrée vectorielle instantanée utilisée ici.

### 4.6.2 Entrées, sorties, couches et neurones

Le réseau opère en temps réel à chaque pas de simulation. Les entrées sont les trois signaux d'erreur :

$$\mathbf{x}(t) = \left[e(t),\ \dot{e}(t),\ \int_0^t e(\tau)\, d\tau\right]^T \in \mathbb{R}^3, \tag{4.5}$$

et les sorties sont les cinq paramètres du FOPID :

$$\hat{\mathbf{y}} = [K_p,\ K_i,\ K_d,\ \mu,\ \lambda]^T \in \mathbb{R}^5. \tag{4.6}$$

L'architecture $3 \to 64 \to 64 \to 32 \to 5$ (ReLU dans les couches cachées, linéaire en sortie) totalise **6 661 paramètres entraînables**. La figure 4.1 présente l'architecture complète avec les labels d'entrée et de sortie.

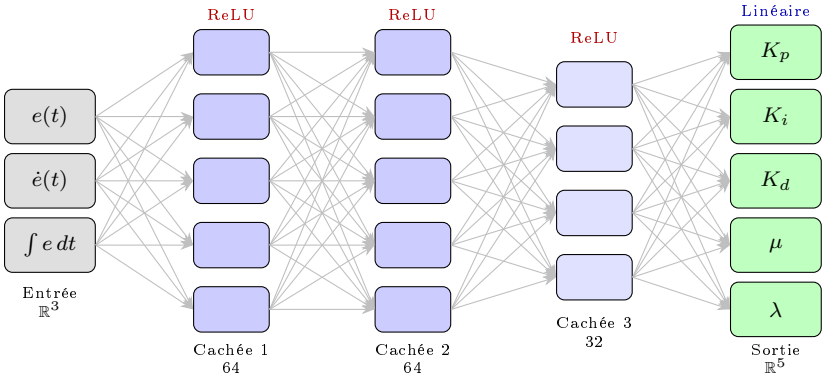


FIGURE 4.1 – Architecture du réseau Neuro-FOPID ($3 \to 64 \to 64 \to 32 \to 5$, 6 661 paramètres). Entrées : signaux d'erreur. Sorties : paramètres FOPID. Activations ReLU (cachées), linéaire (sortie).

## 4.7 Entraînement du Réseau

### 4.7.1 Schéma de la boucle de simulation

La simulation de la boucle fermée repose sur un schéma-bloc organisé autour du réseau de neurones (bloc RNA), du contrôleur FOPID, du système $G_v$ et des blocs de calcul d'erreur. La figure 4.2 présente l'architecture de cette boucle.

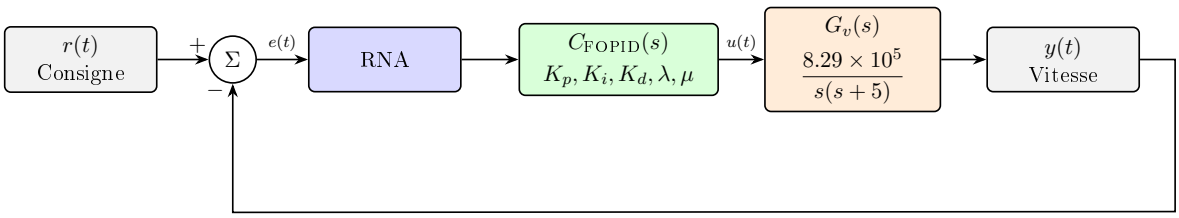


FIGURE 4.2 – Schéma-bloc de la boucle fermée Neuro-FOPID. Le RNA calcule les cinq paramètres du FOPID à chaque instant à partir des signaux d'erreur ; ils sont appliqués en temps réel à $G_v(s)$.

### 4.7.2 Génération de la base de données

La base de données d'entraînement est construite selon la procédure suivante. Pour chaque variante, 500 perturbations aléatoires du gain ($\alpha_g \sim \mathcal{U}(0.5, 1.5)$) et de l'amortissement ($\alpha_d \sim \mathcal{U}(0.5, 1.5)$) sont générées, couvrant $\pm 50\%$ de variation. Les paramètres optimaux sont calculés avec `fmincon` (point intérieur) :

$$J = \text{ISE} + 20\,\delta_{OS}^2 + 50\,\varepsilon_{ss}^2. \tag{4.7}$$

Chaque condition est simulée sur $T = 1.5$ s ($\Delta t = 0.005$ s). Les entrées/sorties sont normalisées par z-score. La base finale comporte **139 363 échantillons** issus de **463 conditions acceptées** sur 500. Le calcul parallèle (`parfor`, 6 workers) a nécessité **44 minutes**.

TABLE 4.1 – Statistiques des paramètres FOPID dans la base d'entraînement.

| **Paramètre** | **Min** | **Moyenne** | **Max** | **Écart-type** |
|---|---|---|---|---|
| $K_p$ | $3.38 \times 10^{-5}$ | $4.65 \times 10^{-4}$ | $5.00 \times 10^{-4}$ | $7.82 \times 10^{-5}$ |
| $K_i$ | $1.03 \times 10^{-7}$ | $6.71 \times 10^{-6}$ | $1.90 \times 10^{-4}$ | $1.90 \times 10^{-5}$ |
| $K_d$ | $5.84 \times 10^{-5}$ | $9.73 \times 10^{-5}$ | $1.00 \times 10^{-4}$ | $5.74 \times 10^{-6}$ |
| $\mu$ | 1.000 | 1.284 | 1.399 | 0.043 |
| $\lambda$ | 0.603 | 0.982 | 1.389 | 0.124 |

### 4.7.3 Paramètres d'entraînement

Les hyperparamètres de l'entraînement sont récapitulés dans le tableau 4.2. L'entraînement s'est arrêté à l'époque 119 sur 200, après 27.5 minutes sur CPU.

TABLE 4.2 – Hyperparamètres de l'entraînement du réseau Neuro-FOPID.

| **Hyperparamètre** | **Valeur** |
|---|---|
| Algorithme | Adam |
| Architecture | $3 \to 64 \to 64 \to 32 \to 5$ |
| Paramètres entraînables | 6 661 |
| Activation (couches cachées / sortie) | ReLU / Linéaire |
| Taux d'apprentissage $\eta$ | $10^{-3}$ |
| Taille du mini-lot | 256 |
| Époques maximum | 200 |
| Répartition train / val. | 85% / 15% |
| Normalisation | Z-score |

## 4.8 Résultats de simulation

### 4.8.1 Performances du réseau entraîné

Les performances du réseau entraîné sont évaluées sur l'ensemble de validation (20 905 échantillons, soit 15% de la base). Le tableau 4.3 présente le RMSE par paramètre, en échelle physique après dénormalisation.

TABLE 4.3 – RMSE de validation sur les paramètres FOPID (20 905 échantillons de validation).

| Paramètre | RMSE | Moyenne prédite | Moyenne réelle |
|---|---|---|---|
| $K_p$ | $3.35 \times 10^{-5}$ | $4.66 \times 10^{-4}$ | $4.65 \times 10^{-4}$ |
| $K_i$ | $8.44 \times 10^{-6}$ | $6.23 \times 10^{-6}$ | $6.71 \times 10^{-6}$ |
| $K_d$ | $3.03 \times 10^{-6}$ | $9.76 \times 10^{-5}$ | $9.73 \times 10^{-5}$ |
| $\mu$ | $2.71 \times 10^{-2}$ | 1.286 | 1.284 |
| $\lambda$ | $1.15 \times 10^{-1}$ | 0.981 | 0.980 |

## 4.9 Comparaison avec le PID et le FOPID de référence

### 4.9.1 Paramètres des contrôleurs de référence

Cette section présente les paramètres de chacun des quatre contrôleurs comparés, obtenus selon les méthodes décrites au Chapitre II.

Le contrôleur PID classique est réglé par `pidtune` (marge de phase 60°) :

$$C_{\text{PID}}(s) = K_p + \frac{K_i}{s} + \frac{K_d s}{1 + K_d/N_f \cdot s}, \tag{4.8}$$

avec $K_p = 6.361 \times 10^{-5}$, $K_i = 8.127 \times 10^{-5}$, $K_d = 1.106 \times 10^{-5}$, $N_f = 100$. Le FOPID classique reprend ces gains (mis à l'échelle) avec les ordres fractionnaires issus de la littérature [5] ($\lambda = 0.75$, $\mu = 1.25$) : $K_p = 1.272 \times 10^{-4}$, $K_i = 6.501 \times 10^{-5}$, $K_d = 2.213 \times 10^{-5}$.

Le tableau 4.4 compare les paramètres moyens des deux contrôleurs FOPID.

TABLE 4.4 – Paramètres FOPID classique vs Neuro-FOPID (valeurs moyennes).

| Paramètre | FOPID-Réf | Neuro-FOPID | Écart |
|---|---|---|---|
| $K_p$ | $1.272 \times 10^{-4}$ | $4.736 \times 10^{-4}$ | +272% |
| $K_i$ | $6.501 \times 10^{-5}$ | $2.584 \times 10^{-6}$ | −96% |
| $K_d$ | $2.213 \times 10^{-5}$ | $9.879 \times 10^{-5}$ | +346% |
| $\lambda$ | 0.750 | 1.290 | +72% |
| $\mu$ | 1.250 | 0.981 | −21% |
| Temps de calcul | $< 1$ ms | ∼44 min (base) / ∼1 ms (inférence) | — |

L'écart marqué illustre la richesse de la politique apprise : $K_p$ ×3.7 en début de transitoire, $K_i$ ÷25 pour éviter le dépassement, et $\lambda > 1$ pour une phase d'avance supplémentaire absente dans le FOPID classique.

### 4.9.2 Comparaison globale des performances

Le tableau 4.5 récapitule les performances mesurées en simulation continue (`lsim`) pour les quatre contrôleurs dans des conditions nominales identiques.

TABLE 4.5 – Comparaison des performances temporelles et intégrales des quatre contrôleurs.

| Critère | PID-Réf | Neuro-PID | FOPID-Réf | Neuro-FOPID |
|---|---|---|---|---|
| $M_p$ (%) | 10.267 | 8.082 | 3.282 | **0.288** |
| $t_r$ (s) | 0.1900 | 0.1280 | 0.1540 | **0.0270** |
| $t_s$ (s) | 1.5830 | 1.1950 | 1.2000 | **0.0460** |
| ISE | 0.058670 | 0.041126 | 0.016122 | **0.008171** |
| IAE | 0.182228 | 0.129095 | 0.091164 | **0.015525** |
| ITAE | 0.088336 | 0.072704 | 0.096097 | **0.006297** |
| ITSE | 0.007278 | 0.002705 | 0.001245 | **0.000050** |

Les quatre contrôleurs forment une progression monotone claire. La figure 4.3 montre les réponses indicielles complètes, la figure 4.4 zoome sur le transitoire, et la figure 4.5 montre l'erreur de suivi.

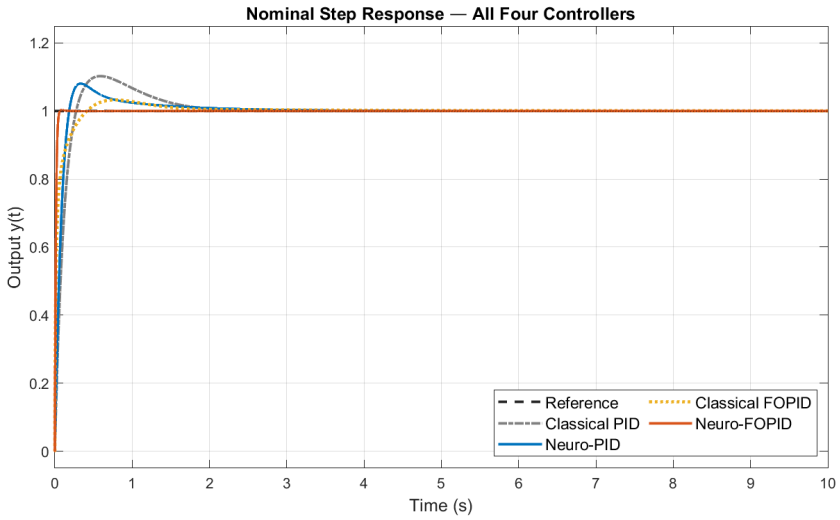


FIGURE 4.3 – Réponses indicielles nominales des quatre contrôleurs (0–10 s).

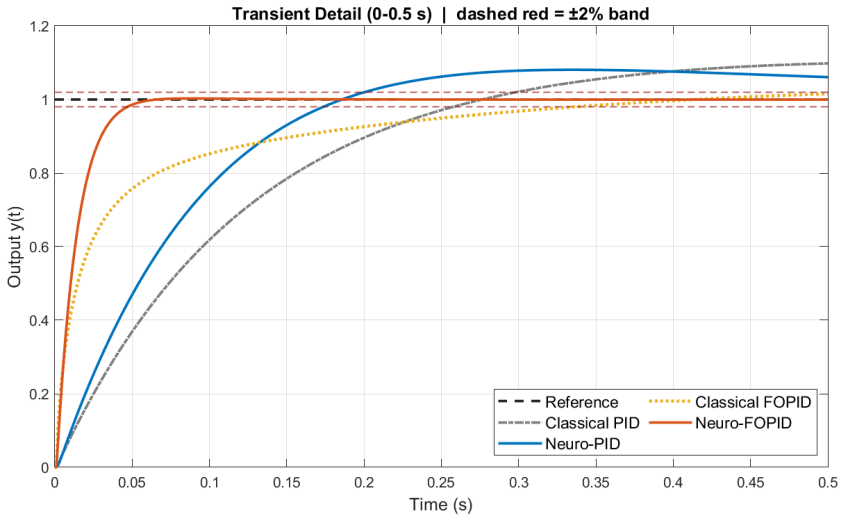


FIGURE 4.4 – Zoom sur le régime transitoire (0–0.5 s).

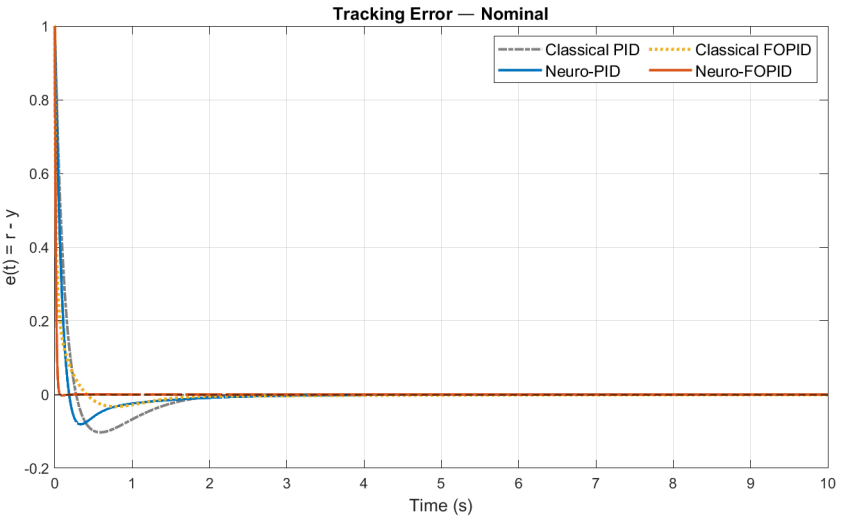


FIGURE 4.5 – Évolution de l'erreur de suivi $e(t) = r(t) - y(t)$ pour les quatre contrôleurs.

Les figures 4.6 et 4.7 comparent respectivement les indices intégraux et les métriques temporelles clés.

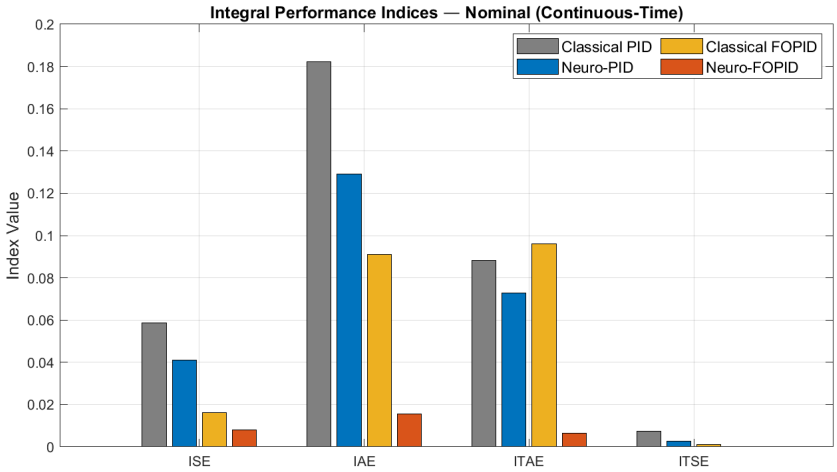


FIGURE 4.6 – Indices intégraux de performance (ISE, IAE, ITAE, ITSE) pour les quatre contrôleurs.

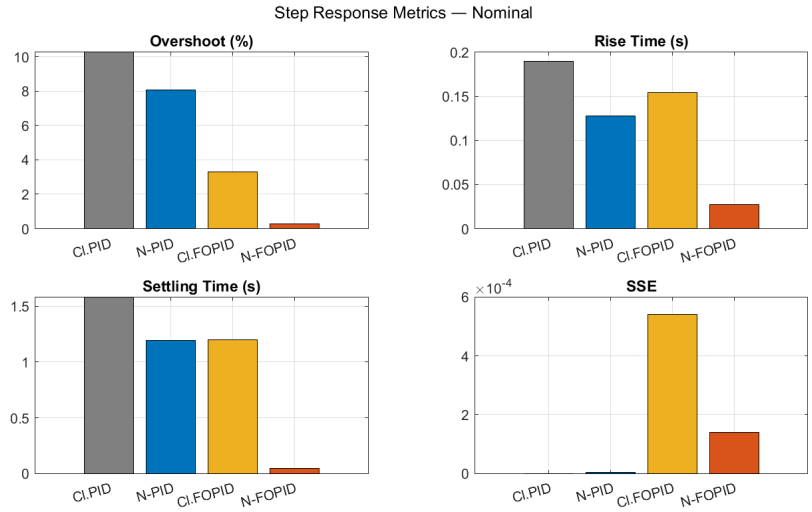


FIGURE 4.7 – Métriques temporelles comparatives : dépassement, temps de montée et stabilisation.

Le Neuro-FOPID domine tous les critères intégraux simultanément. Deux leviers se cumulent : (1) PID → FOPID réduit le dépassement de 8.1% à 3.28% grâce à la phase d'avance fractionnaire ; (2) classique → neural réduit l'ISE de 49% par planification adaptative.

### 4.9.3 Test de robustesse aux variations paramétriques

Un contrôleur performant en régime nominal peut se dégrader lorsque les paramètres du système varient. Pour évaluer la robustesse des quatre contrôleurs, le gain de $G_v(s)$ est perturbé de $\delta \in \{-30\%, \ldots, +30\%\}$, les autres paramètres restant nominaux.

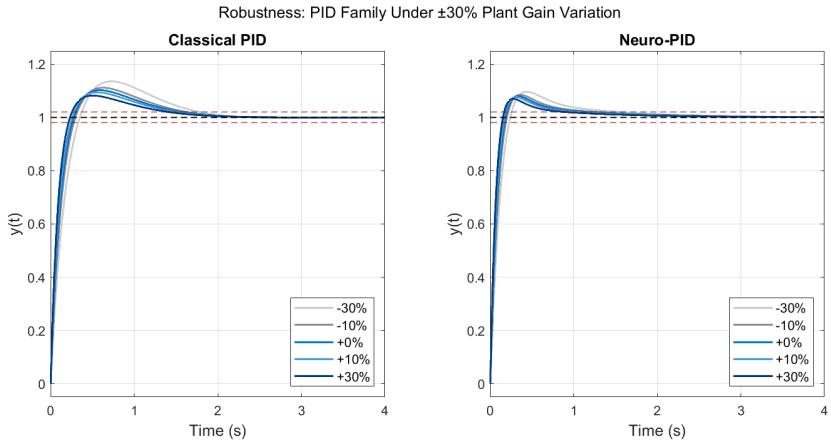


FIGURE 4.8 – Famille PID : réponses en boucle fermée pour $\delta = -30\%$ à $+30\%$. La dispersion traduit la sensibilité paramétrique.

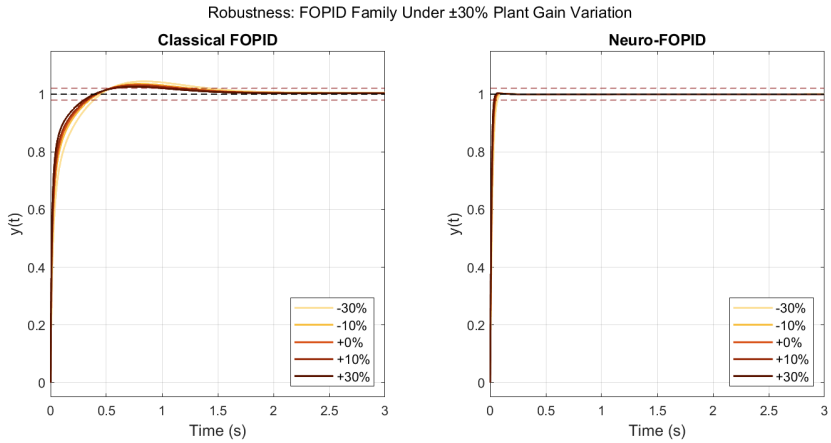


FIGURE 4.9 – Famille FOPID : réponses pour les mêmes perturbations. Le faisceau est visuellement beaucoup plus serré.

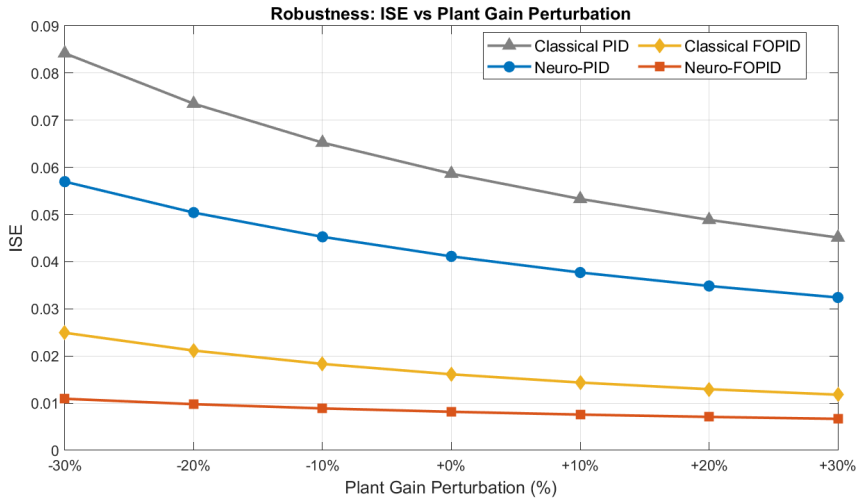


FIGURE 4.10 – ISE en fonction de $\delta$ pour les quatre contrôleurs. Le Neuro-FOPID maintient le plus faible ISE sur toute la plage.

Le tableau 4.6 récapitule le dépassement à chaque niveau de perturbation.

TABLE 4.6 – Dépassement $M_p$ (%) en fonction de la perturbation du gain ($\delta$).

| $\delta$ | **PID-Réf** | **Neuro-PID** | **FOPID-Réf** | **Neuro-FOPID** |
|---|---|---|---|---|
| $-30\%$ | 13.585 | 9.696 | 4.462 | **0.227** |
| $-20\%$ | 12.269 | 9.068 | 3.987 | **0.253** |
| $-10\%$ | 11.181 | 8.538 | 3.601 | **0.273** |
| $\pm 0\%$ | 10.267 | 8.082 | 3.282 | **0.288** |
| $+10\%$ | 9.487 | 7.683 | 3.015 | **0.300** |
| $+20\%$ | 8.814 | 7.330 | 2.787 | **0.310** |
| $+30\%$ | 8.227 | 7.014 | 2.591 | **0.318** |
| **Étendue** | 5.358 | 2.682 | 1.871 | **0.091** |

La famille PID montre une étendue de 5.36 points sur $\pm 30\%$ de variation de gain. Les deux contrôleurs FOPID maintiennent un faisceau serré : 1.87 points pour le FOPID-Réf, seulement 0.09 point pour le Neuro-FOPID. Cet avantage découle de la propriété d'iso-amortissement (voir section 2.4.4) : la phase reste quasi-constante sur une large bande, rendant les marges de stabilité insensibles aux variations de gain.

### 4.9.4 Rejet de perturbation

Le rejet de perturbation est testé en appliquant un échelon de gain $+10\%$ à $t = 5$ s, après que tous les contrôleurs ont atteint leur régime permanent nominal.

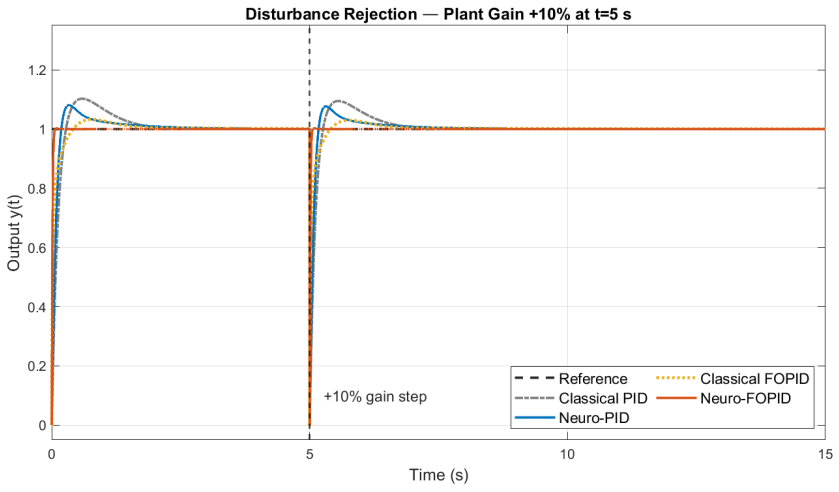


FIGURE 4.11 – Réponse de rejet de perturbation : échelon de gain $+10\%$ à $t = 5$ s.

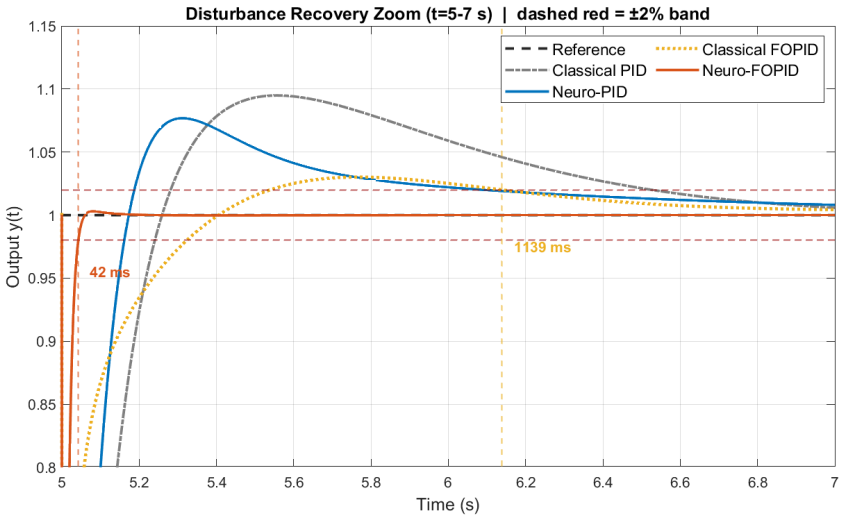


FIGURE 4.12 – Zoom sur la phase de récupération après perturbation ($t = 5$–$7$ s).

Le tableau 4.7 quantifie les performances de rejet.

TABLE 4.7 – Performances de rejet d'une perturbation en échelon (+10% à $t = 5$ s).

| **Métrique** | **PID-Réf** | **Neuro-PID** | **FOPID-Réf** | **Neuro-FOPID** |
|---|---|---|---|---|
| $T_{\text{récupération}}$ ($\pm 2\%$) | 1.526 s | 1.092 s | 1.139 s | **0.042 s** |
| Dépassement | Oui | Oui | Non | Non |

Les deux contrôleurs FOPID récupèrent sans dépassement grâce à la dérivée fractionnaire qui fournit une phase d'avance lors du saut d'erreur. Le Neuro-FOPID récupère en 42 ms — plus court que son propre temps de stabilisation nominal (46 ms) — car le RNA applique instantanément une action dérivative maximale en réponse au saut.

## 4.10 Conclusion

Ce chapitre a présenté le véhicule électrique hybride, sa modélisation et l'ensemble des résultats de simulation. La fonction de transfert $G_v(s) = 8.29 \times 10^5/[s(s+5)]$ a servi de base de test pour quatre contrôleurs. Le Neuro-FOPID domine avec un dépassement de 0.29% (contre 10.27%), un temps de stabilisation de 46 ms (34× plus court), une ISE 7× inférieure. En robustesse, son étendue de dépassement sur ±30% est de 0.09 point (contre 5.36 pour le PID-Réf). En rejet de perturbation, sa récupération en 42 ms sans dépassement confirme l'avantage structurel de la combinaison fractionnaire-neuronale.

# Conclusion Générale

Ce mémoire a abordé la problématique du réglage automatique d'un contrôleur PID d'ordre fractionnaire par réseau de neurones artificiels, appliqué à la commande de la vitesse d'un véhicule électrique hybride. L'objectif principal était de combiner les avantages de la commande fractionnaire — notamment la propriété d'iso-amortissement et la robustesse structurelle du FOPID — avec la capacité d'adaptation et d'optimisation instantanée des réseaux de neurones.

Dans le premier chapitre, nous avons établi les fondements théoriques des systèmes d'ordre fractionnaire. Les opérateurs de Riemann-Liouville et Caputo ont été présentés, ainsi que les méthodes d'approximation de Charef et d'Oustaloup. Cette dernière a été retenue pour sa précision et son intégration dans la FOMCON Toolbox de MATLAB. Le modèle de référence fractionnaire d'iso-amortissement a été étudié en boucle fermée : son comportement est similaire à celui d'un système de second ordre amorti dont le dépassement reste constant malgré les variations de gain, ce qui en fait un modèle de référence robuste par excellence.

Le deuxième chapitre a permis de comparer les contrôleurs PID classique et FOPID. Les trois structures du PID ont été présentées, ainsi que les différents types de FOPID. Un exemple numérique sur un système de troisième ordre a démontré la supériorité du FOPID vis-à-vis du PID classique en termes de précision de suivi. Les méthodes d'optimisation SQP et Nelder-Mead ont été décrites et leur limitation principale — la sensibilité à l'initialisation — a été identifiée comme la motivation principale pour l'approche par réseau de neurones.

Le troisième chapitre a fourni les fondements théoriques des réseaux de neurones artificiels. Les différentes architectures (MLP, CNN, RNN), les fonctions d'activation, l'algorithme de rétropropagation et les techniques de régularisation ont été présentés. Ces éléments ont servi de cadre pour la conception du réseau Neuro-FOPID.

Le quatrième chapitre a rassemblé la modélisation du véhicule électrique hybride ainsi que l'ensemble des résultats de simulation. Le modèle non linéaire a été linéarisé au point d'équilibre, conduisant à $G_v(s) = 8.29 \times 10^5/[s(s+5)]$. Le réseau MLP à architecture $3 \to 64 \to 64 \to 32 \to 5$, entraîné avec l'algorithme Adam sur 139 363 échantillons générés hors-ligne, a démontré d'excellentes performances de régression. En simulation en boucle fermée, le Neuro-FOPID surpasse le PID classique et le FOPID

de référence sur tous les critères : dépassement 0.29% contre 10.27%, ISE 7× inférieure, temps de stabilisation 34× plus court. Les tests de robustesse (±30% de gain) et de rejet de perturbation confirment la supériorité structurelle de l'approche fractionnaire-neuronale.

**Pour l'avenir**, plusieurs perspectives sont envisageables :

— Utiliser des architectures RNA plus avancées (LSTM) pour une meilleure généralisation sur des dynamiques complexes,

— Combiner les RNA avec des algorithmes évolutionnaires pour la génération de la base d'entraînement,

— Valider expérimentalement les résultats sur une plateforme embarquée,

— Étendre l'approche aux modèles non linéaires complets du véhicule (frottement, dynamique batterie).

En définitive, ce mémoire démontre que les approches fractionnaires combinées aux réseaux de neurones constituent une avancée significative pour les systèmes de commande modernes.

# Bibliographie


[1] Caputo, M. (1967). Linear models of dissipation whose Q is almost frequency independent—II. *Geophysical Journal International*, 13(5), 529–539.

[2] Charef, A. (1992). Analogue realisation of fractional-order integrator. *IEEE Transactions on Circuits and Systems*, 39(9), 789–792.

[3] Charef, A. (2006). Analogue realisation of fractional-order integrator, differentiator and fractional PID controller. *IET Control Theory & Applications*, 153(6), 714–722.

[4] Kilbas, A. A., Srivastava, H. M., & Trujillo, J. J. (2006). *Theory and Applications of Fractional Differential Equations*. Elsevier.

[5] Monje, C. A., Chen, Y., Vinagre, B. M., Xue, D., & Feliu, V. (2010). *Fractional-order Systems and Controls : Fundamentals and Applications*. Springer.

[6] Ogata, K. (2010). *Modern Control Engineering* (5th ed.). Prentice Hall.

[7] Oustaloup, A., Mathieu, B., & Lanusse, P. (1995). The CRONE control of resonant plants. *European Journal of Control*, 1(2), 113–121.

[8] Podlubny, I. (1999). *Fractional Differential Equations*. Academic Press.

[9] Bode, H. W. (1945). *Network Analysis and Feedback Amplifier Design*. Van Nostrand.

[10] Chen, Y., & Moore, K. L. (2003). Discretization schemes for fractional-order differentiators and integrators. *IEEE Transactions on Circuits and Systems*, 49(3), 363–367.

[11] Définition et usages de ITAE. *Ijee.ie*. https://www.ijee.ie/articles/Vol21-5/Ijee1673.pdf

[12] MathWorks. *fmincon*. https://www.mathworks.com/help/optim/ug/fmincon.html

[13] Nelder, J. A., & Mead, R. (1965). A simplex method for function minimization. *Computer Journal*, 7(4), 308–313.

[14] Barbosa, R. S., Machado, J. A. T., & Ferreira, I. M. (2004). Tuning of PID Controllers Based on Bode's Ideal Transfer Function. *Nonlinear Dynamics*, 38, 305–321.

[15] Boggs, P. T., & Tolle, J. W. (1995). Sequential Quadratic Programming. *Acta Numerica*, 4, 1–51.

[16] Desborough, L., & Miller, R. (2002). Increasing customer value of industrial control performance monitoring. *AIChE Symposium Series*, pp. 169–189.

[17] Liu, L., Wang, Z., & Zhang, H. (2017). A multi-objective optimization method for fractional order PID controller. *ISA Transactions*, 72, 256–267.

[18] Zhao, C., Xue, D., & Chen, Y. Q. (2005). A Fractional Order PID Tuning Algorithm. *Int. Conf. on Mechatronics and Automation*, pp. 216–221.

[19] AMOURA, Karima. *Contribution à la synthèse de contrôleurs fractionnaires d'ordre réduit*. Thèse de doctorat, UMMTO, 2018.

[20] Ziegler, J. G., & Nichols, N. B. (1942). Optimum settings for automatic controllers. *Transactions of the ASME*, 64, 759–768.

[21] Haykin, S. (1999). *Neural Networks : A Comprehensive Foundation* (2nd ed.). Prentice Hall.

[22] Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5), 359–366.

[23] Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323, 533–536.

[24] Kingma, D. P., & Ba, J. (2015). Adam : A method for stochastic optimization. *ICLR 2015*, San Diego, CA.

[25] Valério, D., & Sá da Costa, J. (2013). *An Introduction to Fractional Control*. IET.

[26] Petráš, I. (2011). *Fractional-Order Nonlinear Systems*. Springer.

[27] Garcia, C. E., & Morari, M. (1982). Internal model control. *Ind. Eng. Chem. Process Des. Dev.*, 21(2), 308–323.