Techniques d'intelligence artificielle pour les équations différentielles autonomes

Soutenance de Stage - Lundi 28 Juin 2021

Maxime BOUCHEREAU
Stage encadré par Philippe CHARTIER, Mohammed LEMOU et Florian MEHATS
Université Rennes 1

Plan de la présentation

Partie 1 : Introduction et outils

Position du problème

Soient $f : \mathbb{R}^d \to \mathbb{R}^d$ un champ de vecteurs continu et localement lipschitzien.
Soient $h > 0, N \in \mathbb{N}^*$ et $0 = t_0 < t_1 < \cdots < t_N = h$.

- **Problème de Cauchy considéré**:

$$\begin{cases} \dot{y} = f(y) \\ y(0) = y_0 \end{cases}$$

- **Données:** pour tout $j \in [\![0, N]\!]$, $y_{\frac{t_j}{h}} = \phi_{t_j}(y_0)$

- **Objectif:** retrouver le champ de vecteurs f à partir des données y_j de la solution à différents temps.

- **Plusieurs trajectoires étudiées**:

  - K données initiales.
  - Pour tous $k \in [\![0, K-1]\!], j \in [\![1, N]\!]$, $y^{(k)}_{\frac{t_j}{h}} = \phi_{t_j}(y_0^{(k)})$

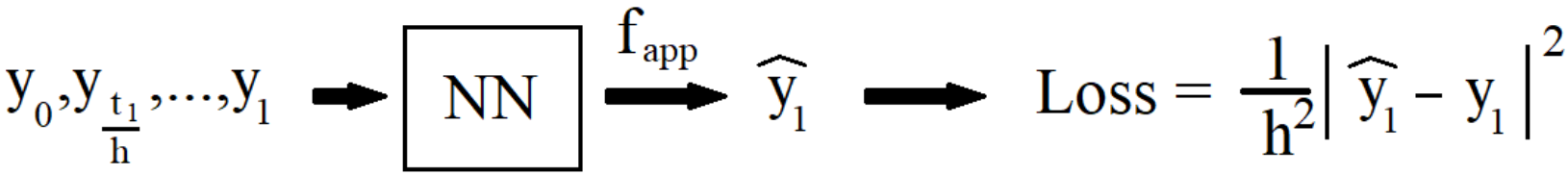


Figure: Illustration de la technique d'IA

- f_{app} s'écrit à l'aide des paramètres du réseau de neurones
- $\hat{y_1}$: valeur prédite de y_1 par un certain schéma numérique, en faisant intervenir $y_0, y_{\frac{t_1}{h}}, \cdots, y_1$ et f_{app}
- **Objectif**: Minimiser la fonction *Loss*

Objectif: Minimiser la fonction *Loss*

- **1^{re} étape - Entrainement (ou Apprentissage)**: Minimiser la fonction

$$Loss_{Training} = \frac{1}{K_0} \sum_{k=0}^{K_0-1} \frac{1}{h^2} \left| \hat{y_1}^{(k)} - y_1^{(k)} \right|^2$$

- **2^{me} étape - Test**: Tester la fonction

$$Loss_{Test} = \frac{1}{K - K_0} \sum_{k=K_0}^{K-1} \frac{1}{h^2} \left| \hat{y_1}^{(k)} - y_1^{(k)} \right|^2$$

$K_0 = \lfloor pK \rfloor$, $p \in]0, 1[$

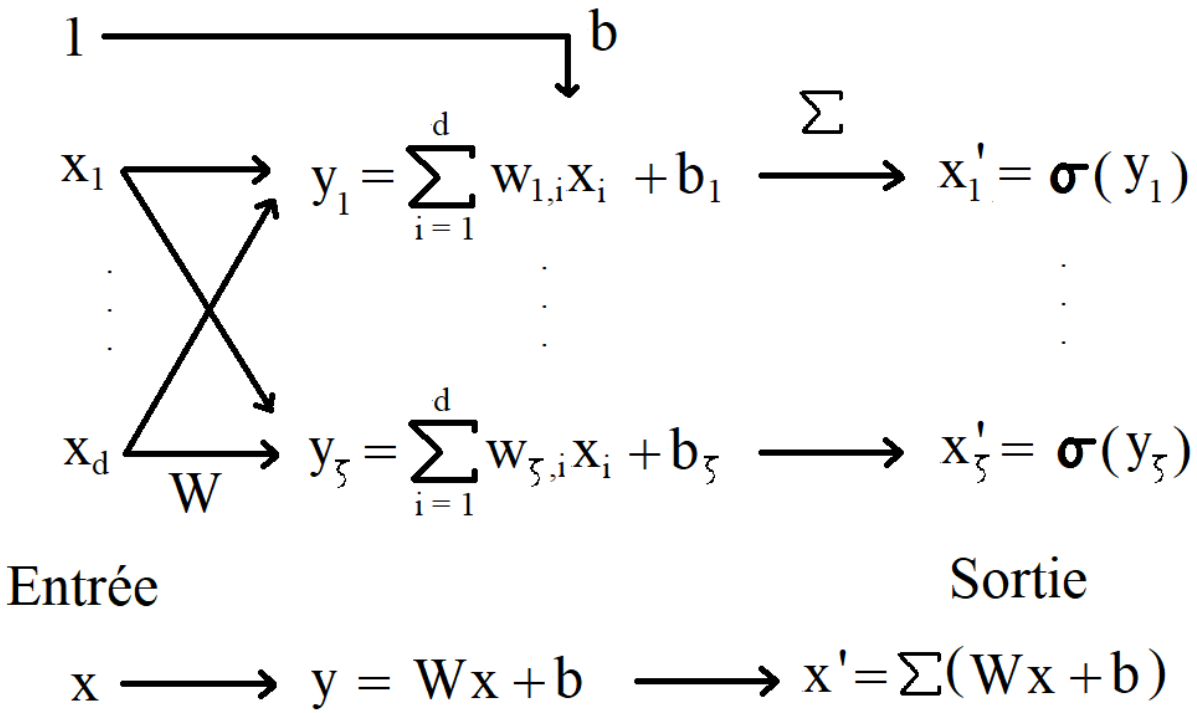


Figure: Illustration du fonctionnement de la couche de ζ neurones

- **Paramètres**: Matrice poids $W \in \mathcal{M}_{\zeta,d}(\mathbb{R})$, biais 1 de poids $b \in \mathbb{R}^\zeta$, fonction d'activation $\sigma : \mathbb{R} \to \mathbb{R}$
- $x \in \mathbb{R}^d$, $y, x' \in \mathbb{R}^\zeta$

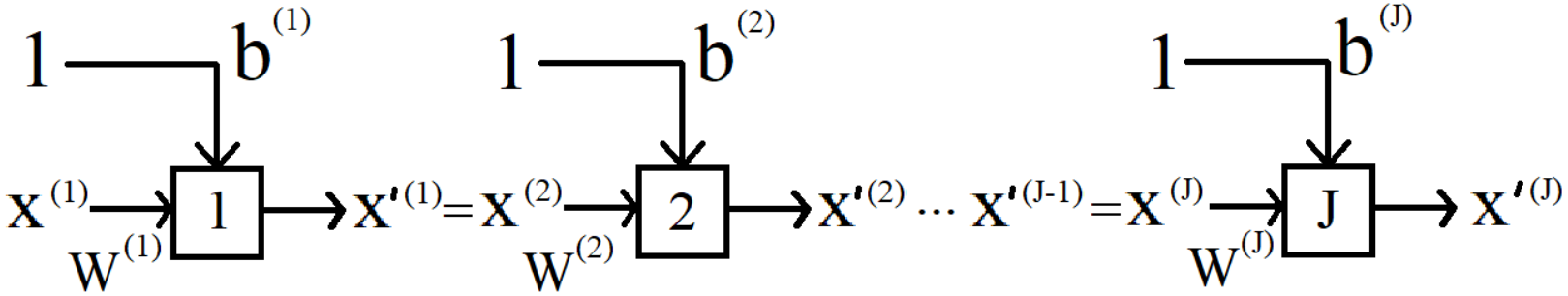


Figure: Illustration du fonctionnement d'un PMC comportant J couches de neurones, l'entrée, la sortie et le poids du biais associés à la couche j sont respectivement notés $x^{(j)}$, $x'^{(j)}$ et $b^{(j)}$

- **PMC**: succession de couches dans lesquelles la sortie de la j-ème couche correspond à l'entrée de la $(j+1)$-ème couche
- Approximation du champ de vecteurs f en f_{app} via: $x'^{(J)} = f_{app}(x^{(1)})$.
- Fonction Loss qui dépend des paramètres $W^{(1)},...,W^{(J)},b^{(1)},...,b^{(J)}$. **Objectif**: optimiser en ces paramètres.

- **Choix du pas de la méthode numérique** $h > 0$
- **Création des données**:
  - Choix au hasard de K données initiales $y_0^{(k)}$ pour $k \in [\![0, K-1]\!]$
  - Calcul de $\phi_{t_j}(y_0^{(k)})$ pour $k \in [\![0, K-1]\!], j \in [\![1, N]\!]$
- **Création des données prédites**: Calcul de $\hat{y_1^{(k)}}$ pour $k \in [\![0, K-1]\!]$ à partir de $y_0^{(k)}, y_{\frac{t_1}{h}}^{(k)}, \cdots, y_1^{(k)}$ et f_{app}
- **Introduction des fonctions *Loss***:

$$L(W, k) \quad = \quad \frac{1}{h^2} \left| \hat{y_1^{(k)}} - y_1^{(k)} \right|^2$$

  W: Paramètres du réseau de neurones
- **Calcul des différentielles** des fonctions *Loss* en les paramètres par différence finie centrée.

- **Choix d'une proportion de données pour l'entraînement** $p \in]0,1[$ et calcul de $K_0 = \lfloor pK \rfloor$.
- **Introduction des fonctions**:

$$Loss_{Training}(W) = \frac{1}{K_0} \sum_{k=0}^{K_0-1} L(W,k)$$

$$Loss_{Test}(W) = \frac{1}{K-K_0} \sum_{k=K_0}^{K-1} L(W,k)$$

- **Choix d'un nombre d'itérations** N_{iter}
- **Minimisation** de la fonction $Loss_{Training}$ via un algorithme d'optimisation, que l'on arrête au bout de N_{iter} itérations.
- **Test** de la fonction $Loss_{Test}$ (on s'assure qu'elle décroît comme $Loss_{Training}$)
- **Comparaison**: Mesure de l'écart entre f et f_{app}

Algorithme du gradient stochastique

- **Choix des paramètres** $\alpha > 0$ (Taux d'apprentissage) et N_{iter} (Nombre d'itérations de l'algorithme)
- **Choix au hasard** de $(W)_0$, paramètres du réseau de neurones initiaux.
- Pour $t \in [\![0, N_{iter} - 1]\!]$:
  - $i_t \sim \mathcal{U}([\![0, K_0 - 1]\!])$
  - $(W)_{t+1} = (W)_t - \alpha \cdot nablaL((W)_t, i_t)$

$nablaL(\cdot, k)$: Approximation de la différentielle de $L(\cdot, k)$ en les paramètres W par différence finie centrée.

Convergence: l'algorithme converge lorsque la fonction *Loss* est (strictement) convexe vers un minimum (global)

Partie 2 : Cas linéaire: Première formulation

Position du problème

Soient $A \in \mathcal{M}_d(\mathbb{R})$ et $h > 0$.

- **Problème de Cauchy considéré**:

$$\begin{cases} \dot{y} = Ay \\ y(0) = y_0 \end{cases}$$

- **Données:** $y_1 = e^{hA} y_0$.

- **Objectif:** retrouver la matrice A à partir des données y_0 et y_1 de la solution aux temps 0 et h.

- **Plusieurs trajectoires étudiées**:

  - K données initiales.
  - Pour tout $k \in [\![0, K-1]\!]$, $y_1^{(k)} = e^{hA} y_0^{(k)}$

Position du problème

- **Structure du réseau de neurones**: Une couche de d neurones, fonction d'activation: $\sigma = Id_{\mathbb{R}}$
- **Paramètres**: Une matrice poids $W \in \mathcal{M}_d(\mathbb{R})$ donnant ainsi, pour tout $k \in [\![0, K-1]\!]$, $\hat{y_1}^{(k)} = A_h y_0^{(k)}$ (A_h est la matrice d'itération propre au schéma numérique, dépend de W. $f_{app}(y_0) = Wy_0$)
- **Forme de la fonction *Loss***:

$$L(W, k) \quad = \quad \frac{1}{h^2} \left| \left(A_h - e^{hA} \right) y_0^{(k)} \right|^2$$

- **Minimiseur**: Matrice $W_{Min} \in \mathcal{M}_d(\mathbb{R})$ telle que $A_h = e^{hA}$, sous réserve d'avoir **au moins d données linéairement indépendantes**.

Matrice d'itération: $A_h = I_d + hW$

Proposition (Fonction *Loss* - Méthode d'Euler Explicite)

Soit $k \in [\![0, K-1]\!]$. *La fonction*

$$L(\cdot, k) : W \mapsto \frac{1}{h^2} \left| \left(I_d + hW - e^{hA} \right) y_0^{(k)} \right|^2$$

est convexe sur $\mathcal{M}_d(\mathbb{R})$ *et est presque partout strictement convexe sur* $\mathcal{M}_d(\mathbb{R})$.

Preuve (idée):

- **Convexité**: Utilisation de la différentielle via la relation:

$$L(W_1, k) + dL(W_1, k) \cdot (W_2 - W_1) \quad \leqslant \quad L(W_2, k)$$

- **Convexité stricte**: Utilisation du fait que, pour presques tous $W \in \mathcal{M}_d(\mathbb{R})$, $y_0 \in \mathbb{R}^d$, $|Wy_0| > 0$.

Proposition (Méthode d'Euler Explicite - Ordre de convergence)

L'apprentissage pour la méthode d'Euler Explicite est d'ordre 1:

$$W_{Min} \underset{h \to 0}{=} A + \mathcal{O}(h)$$

Preuve (idée):

- **Minimiseur**:

$$W_{Min} = \frac{e^{hA} - I_d}{h}$$

- **Ordre de convergence**: Développement limité en $h \to 0$.

- **Matrice d'itération**: $A_h = (I_d - hW)^{-1}$
- **Fonction *Loss***:

$$L(W, k) = \frac{1}{h^2} \left| \left((I_d - hW)^{-1} - e^{hA} \right) y_0^{(k)} \right|^2$$

Proposition (Fonction *Loss* - Méthode d'Euler Implicite en dimension 1)

Soit $R > 0$. *Soient* $k \in [\![0, K-1]\!]$ *et* $y_0^{(k)} \neq 0$. *La fonction*

$$L(\cdot, k) : W \mapsto \frac{1}{h^2} \left(\frac{1}{1 - hW} - e^{hA} \right)^2 y_0^{(k)^2}$$

est strictement convexe sur $[-R, R]$ *dès que* h *est assez petit.*

Preuve (idée):
Stricte convexité: Stricte positivité de la dérivée seconde

∎

Proposition (Méthode d'Euler Implicite - Ordre de convergence)

L'apprentissage pour la méthode d'Euler Implicite est d'ordre 1:

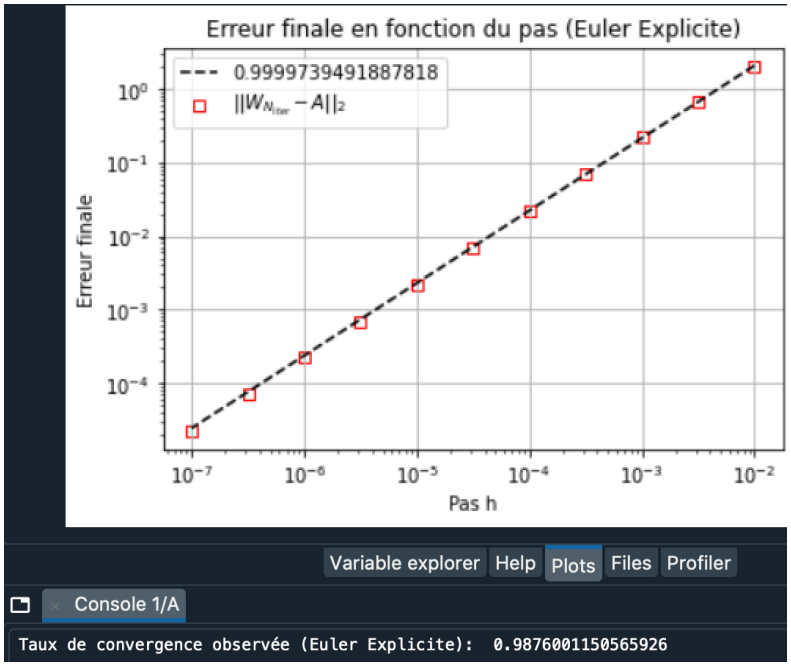
$$W_{Min} \underset{h \to 0}{=} A + \mathcal{O}(h)$$

Preuve (idée):

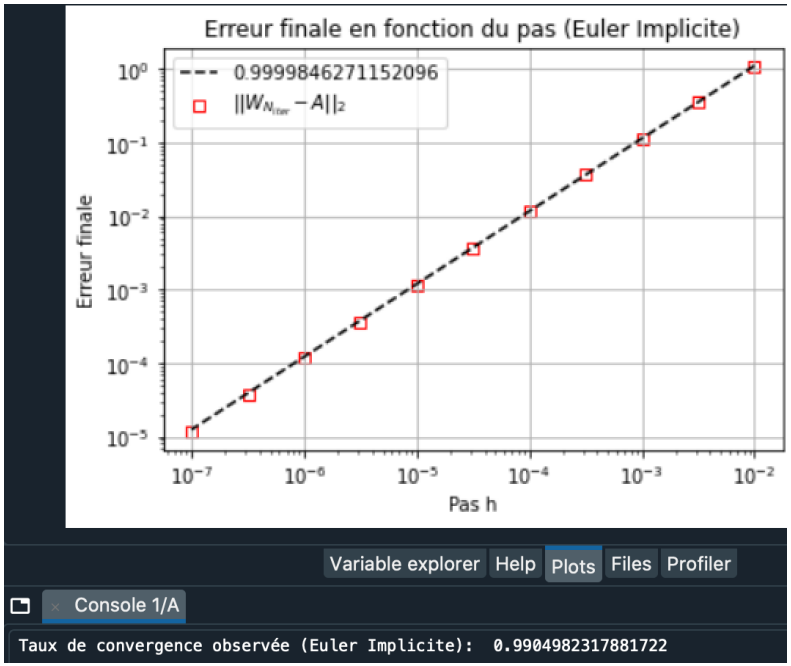
- **Minimiseur**:

$$W_{Min} = \frac{I_d - e^{-hA}}{h}$$

- **Ordre de convergence**: Développement limité en $h \to 0$.



(a) **Méthode d'Euler Explicite**:
$\hat{y_1} = (I_d + hW)y_0$



(b) **Méthode d'Euler Implicite**:
$\hat{y_1} = (I_d - hW)^{-1}y_0$

- **Matrice d'itération**:

$$A_h = \left(I_d - \frac{h}{2} W\right)^{-1} \left(I_d + \frac{h}{2} W\right)$$

- **Fonction *Loss***:

$$L(W, k) = \frac{1}{h^2} \left| \left(\left(I_d - \frac{h}{2} W\right)^{-1} \left(I_d + \frac{h}{2} W\right) - e^{hA} \right) y_0^{(k)} \right|^2$$

Proposition (Fonction *Loss* - Méthode du Point Milieu en dimension 1)

Soit $R > 0$. *Soient* $k \in [\![0, K-1]\!]$ *et* $y_0^{(k)} \neq 0$. *La fonction*

$$L(\cdot, k) : W \mapsto \frac{1}{h^2} \left(\frac{1 + \frac{h}{2} W}{1 - \frac{h}{2} W} - e^{hA} \right)^2 {y_0^{(k)}}^2$$

est strictement convexe sur $[-R, R]$ *dès que* h *est assez petit.*

Preuve (idée):
Stricte convexité: Stricte positivité de la dérivée seconde

Proposition (Méthode du Point Milieu - Ordre de convergence)

L'apprentissage pour la méthode du Point Milieu est d'ordre 2:

$$W_{Min} \underset{h \to 0}{=} A + \mathcal{O}(h^2)$$

Preuve (idée):

- **Minimiseur**:

$$\left(I_d - \frac{h}{2} W_{Min}\right)^{-1} \left(I_d + \frac{h}{2} W_{Min}\right) = e^{hA}$$

- **Ordre de convergence**: Développement limité en $h \to 0$ afin de montrer que $A \underset{h \to 0}{=} W_{Min} + \mathcal{O}(h^2)$ en utilisant le logarithme matriciel

- **Matrice d'itération**: $A_h = I_d + hW + \frac{h^2}{2} W^2$
- **Fonction *Loss***:

$$L(W, k) = \frac{1}{h^2} \left| \left(I_d + hW + \frac{h^2}{2} W^2 - e^{hA} \right) y_0^{(k)} \right|^2$$

Proposition (Fonction *Loss* - Méthode de Runge-Kutta 2 en dimension 1)

Soit $R > 0$. *Soient* $k \in [\![0, K-1]\!]$ *et* $y_0^{(k)} \neq 0$. *Si on a:*

$$h < \min \left\{ \frac{1}{R} \left(1 - \frac{1}{\sqrt{2}} \right), \frac{1}{|A|} \log \left(\frac{5}{4} \right) \right\}$$

alors la fonction:

$$L(\cdot, k) : W \mapsto \frac{1}{h^2} \left(1 + hW + \frac{h^2}{2} W^2 - e^{hA} \right)^2 {y_0^{(k)}}^2$$

est strictement convexe sur $[-R, +\infty[$.

Preuve (idée):
Stricte convexité: Stricte positivité de la dérivée seconde

Proposition (Fonction *Loss* - Méthode de Runge-Kutta 2 en dimension supérieure)

Soit $R > 0$. *Soient* $k \in [\![0, K-1]\!]$ *et* $y_0^{(k)} \neq 0$. *La fonction*

$$L(\cdot, k) : W \mapsto \frac{1}{h^2} \left| I_d + hW + \frac{h^2}{2} W^2 - e^{hA} \right|^2 {y_0^{(k)}}^2$$

est convexe sur $\mathbb{B}_{||\cdot||}(0, R)$ *où* $||\cdot||$ *est une norme matricielle quelconque.*

Preuve (idée):

Convexité: Faire un développement limité en $h \to 0$ pour montrer que la fonction *Loss* ressemble à une forme quadratique en $W \to 0_d$:

$$L(W, k) \underset{h \to 0}{=} \frac{1}{h^2} \left| (W - A) y_0^{(k)} \right|^2 + \mathcal{O}(h)$$

■

Proposition (Méthode de Runge-Kutta 2 - Ordre de convergence)

L'apprentissage pour la méthode de Runge-Kutta 2 est d'ordre 2:

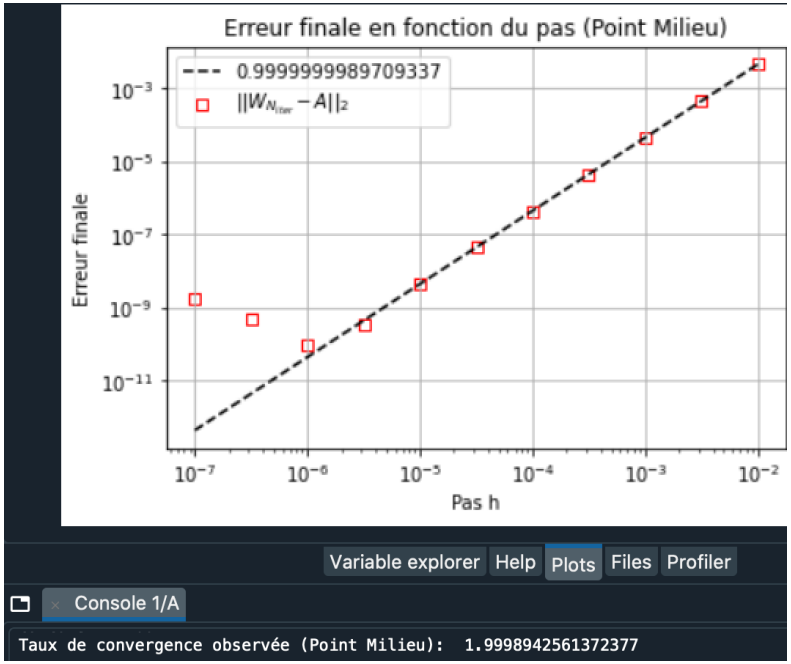
$$W_{Min} \underset{h \to 0}{=} A + \mathcal{O}(h^2)$$

Preuve (idée):

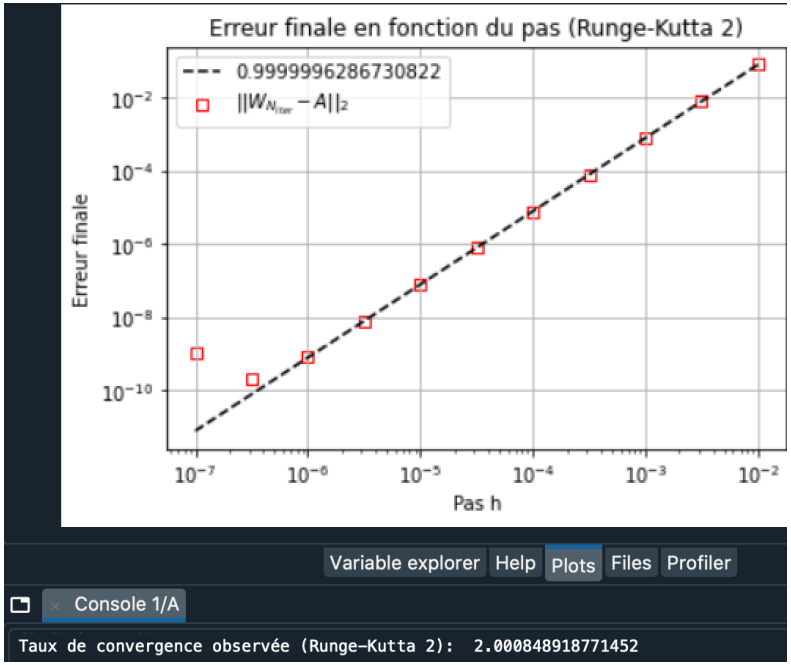
- **Minimiseur**:

$$I_d + hW_{Min} + \frac{h^2}{2} W_{Min}^2 = e^{hA}$$

- **Ordre de convergence**: Développement limité en $h \to 0$ afin de montrer que $A \underset{h \to 0}{=} W_{Min} + \mathcal{O}(h^2)$ en utilisant le logarithme matriciel



(a) **Méthode du Point Milieu:**
$\hat{y_1} =$
$\left(I_d - \frac{h}{2} W\right)^{-1} \left(I_d + \frac{h}{2} W\right) y_0$

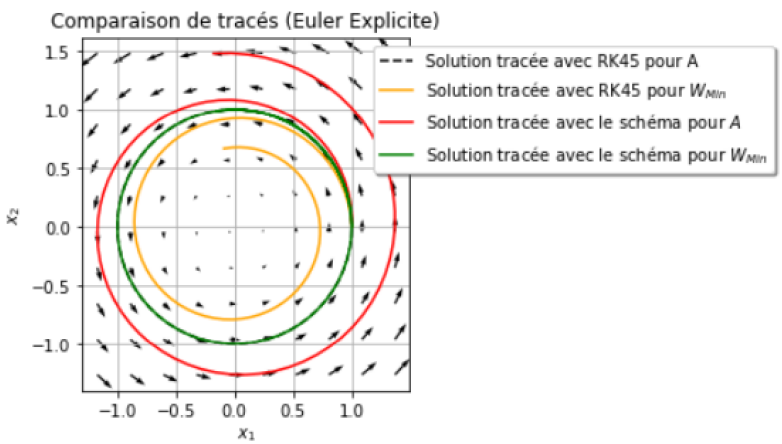


(b) **Méthode de Runge-Kutta 2:**
$\hat{y_1} =$
$\left(I_d + hW + \frac{h^2}{2} W^2\right) y_0$

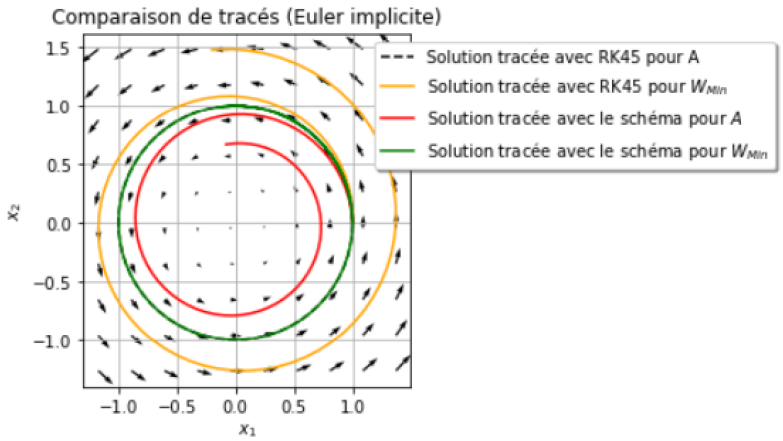
- **Système différentiel:**

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

- **Pas de la méthode numérique:** $h = 0.1$



(a) **Méthode d'Euler Explicite:**
$\hat{y_1} = (I_d + hW)y_0$

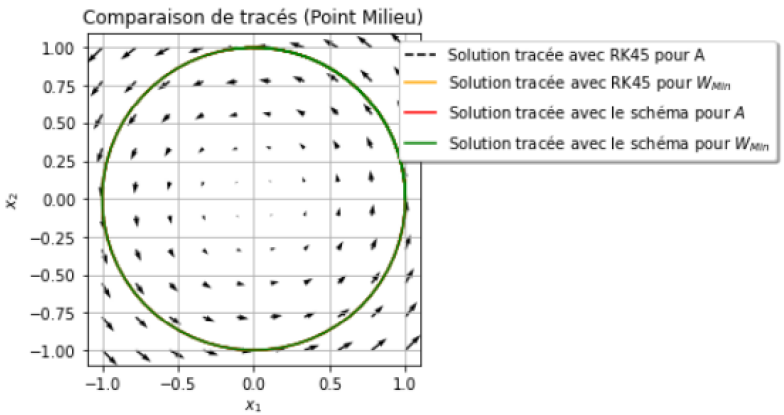


(b) **Méthode d'Euler Implicite:**
$\hat{y_1} = (I_d - hW)^{-1}y_0$

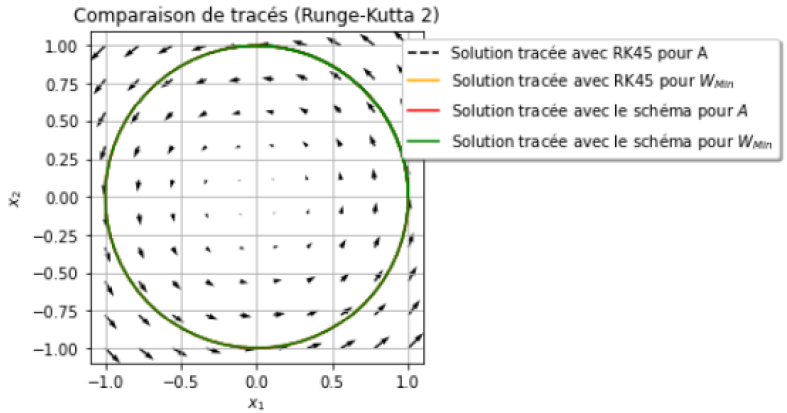
- **Système différentiel:**

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

- **Pas de la méthode numérique:** $h = 0.1$



(a) **Méthode du Point Milieu:**
$\hat{y_1} =$
$\left(I_d - \frac{h}{2} W\right)^{-1} \left(I_d + \frac{h}{2} W\right) y_0$



(b) **Méthode de Runge-Kutta 2:**
$\hat{y_1} =$
$\left(I_d + hW + \frac{h^2}{2} W^2\right) y_0$

Partie 3 : Cas linéaire: Nouvelle formulation et méthodes supplémentaires

Position du problème

Soient $A \in \mathcal{M}_d(\mathbb{R}), h > 0$ et $0 = t_0 < t_1 < \cdots < t_N = h$

- **Problème de Cauchy considéré**:

$$\begin{cases} \dot{y} = Ay \\ y(0) = y_0 \end{cases}$$

- **Données:** pour tout $j \in [\![0, N]\!]$, $y_{\frac{t_j}{h}} = e^{t_j A} y_0$.

- **Objectif:** retrouver la matrice A à partir des données $y_{\frac{t_j}{h}}$ de la solution aux temps t_j.

- **Plusieurs trajectoires étudiées**:

  - K données initiales.
  - Pour tous $k \in [\![0, K-1]\!], j \in [\![1, N]\!]$, $y^{(k)}_{\frac{t_j}{h}} = e^{t_j A} y_0^{(k)}$

- **Structure du réseau de neurones**: Une couche de d neurones, fonction d'activation: $\sigma = Id_{\mathbb{R}}$

- **Paramètres**: Une matrice poids $W \in \mathcal{M}_d(\mathbb{R})$ donnant ainsi, pour tout $k \in [\![0, K-1]\!]$, $\hat{y_1^{(k)}}$ sous cette forme:

$$\hat{y_1^{(k)}} = \sum_{j=0}^{N} \alpha_j \, y^{(k)}_{\frac{t_j}{h}} + W \sum_{j=0}^{N} \beta_j \, h_j \, y^{(k)}_{\frac{t_j}{h}}$$

- **Forme de la fonction *Loss***:

$$L(W, k) = \frac{1}{h^2} \left| \sum_{j=0}^{N} \alpha_j \, y^{(k)}_{\frac{t_j}{h}} - y_1^{(k)} + W \sum_{j=0}^{N} \beta_j \, h_j \, y^{(k)}_{\frac{t_j}{h}} \right|^2$$

- **Minimiseur**: Matrice $W_{Min} \in \mathcal{M}_d(\mathbb{R})$ telle que $A_h = e^{hA}$, sous réserve d'avoir **au moins d données linéairement indépendantes**.

- **Forme des fonctions *Loss***:

$$\begin{aligned} L(W,k) &= \frac{1}{h^2}\left|u^{(k)} + Wv^{(k)}\right|^2 \\ &= \frac{1}{h^2}\left|By_0^{(k)} + WCy_0^{(k)}\right|^2 \end{aligned}$$

- **Minimiseur**: Matrice $W_{min} = -BC^{-1}$, sous réserve d'avoir **au moins** d **données linéairement indépendantes**.

Proposition (Fonction *Loss* - Forme quadratique en W)

Soit $k \in [\![0, K-1]\!]$. *La fonction*

$$L(\cdot,k) : W \mapsto \frac{1}{h^2}\left|u^{(k)} + Wv^{(k)}\right|^2$$

est convexe sur $\mathcal{M}_d(\mathbb{R})$ *et est presque partout strictement convexe sur* $\mathcal{M}_d(\mathbb{R})$.

Preuve:
Comme pour Euler Explicite

■

Méthodes à deux points

- **Données**: Seulement y_0 et y_1.
- **Trois méthodes**:
  - **Méthode d'Euler Explicite**: $\hat{y_1} = (I_d + hW)\, y_0$
  - **Méthode d'Euler Implicite**: $\hat{y_1} = y_0 + hWy_1$
  - **Méthode du Point Milieu**: $\hat{y_1} = y_0 + \frac{h}{2} W(y_0 + y_1)$

Proposition (Méthodes d'Euler et du Point Milieu - Ordres de convergence)

- *L'apprentissage pour les méthodes d'Euler Explicite et Implicite sont d'ordre 1:*

$$W_{Min} \underset{h \to 0}{=} A + \mathcal{O}(h)$$

- *L'apprentissage pour la méthode du Point Milieu est d'ordre 2:*

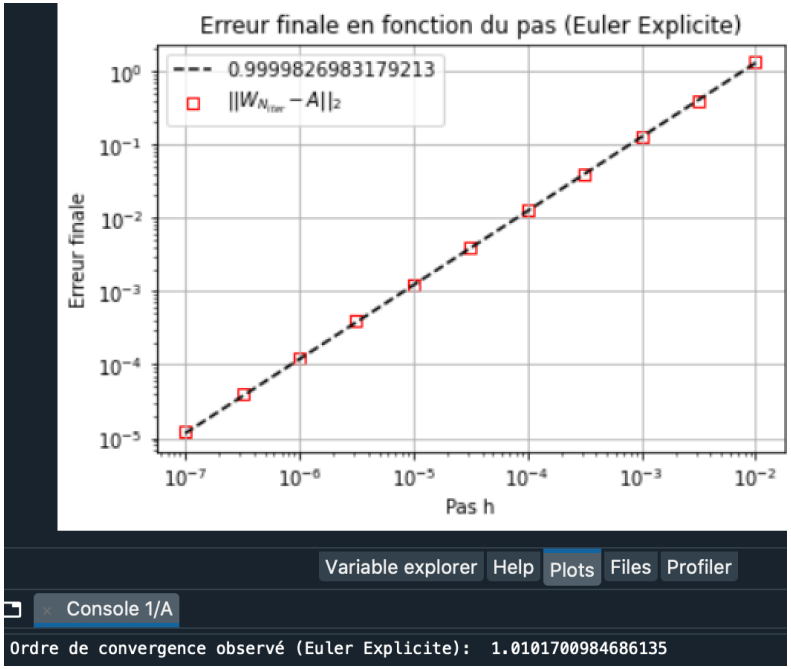
$$W_{Min} \underset{h \to 0}{=} A + \mathcal{O}(h^2)$$

Preuve (idée):

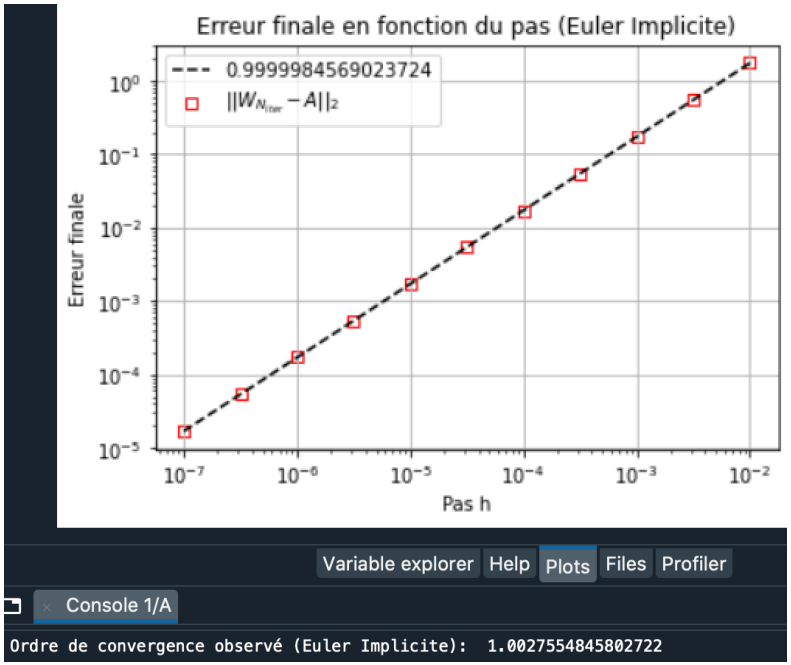
- **Méthodes d'Euler**: Mêmes minimiseurs qu'en partie 1, puis développement limité en $h \to 0$.
- **Méthode du Point Milieu**: Forme du minimiseur:

$$W_{Min} = \frac{2}{h}(I_d + e^{hA})^{-1}(e^{hA} - I_d)$$

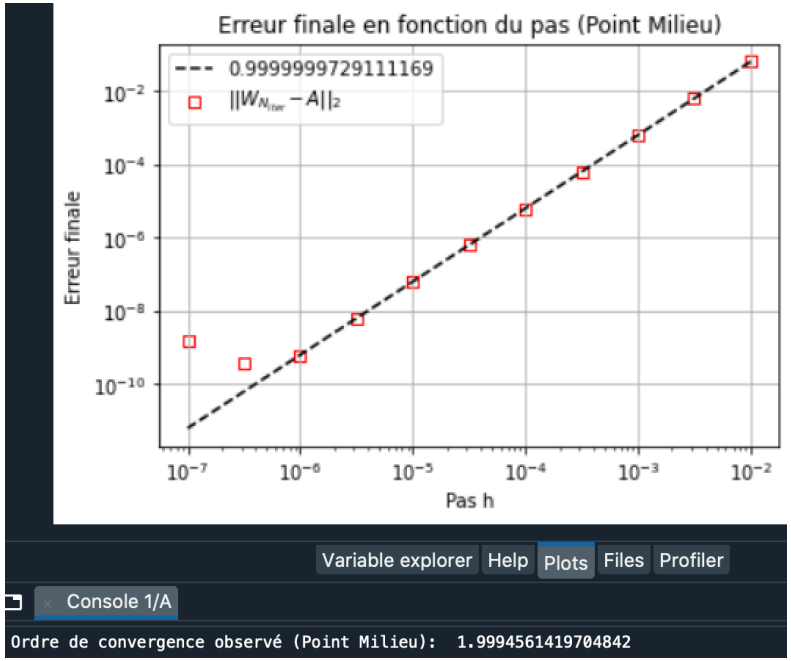
puis développement limité en $h \to 0$.



(a) **Méthode d'Euler Explicite**:
$\hat{y_1} = (I_d + hW)y_0$



(b) **Méthode d'Euler Implicite**:
$\hat{y_1} = y_0 + hWy_1$



(a) **Méthode du Point Milieu:**
$\hat{y_1} = y_0 + \frac{h}{2} W(y_0 + y_1)$

- **Données**: Solutions $y_{\frac{j}{N}} = e^{t_j A} y_0$ en les temps $t_j = \frac{jh}{N}$, $j \in [\![0, N-1]\!]$
- **Méthode**:

$$\hat{y_1} = \sum_{i=0}^{N-1} \alpha_i y_{1-\frac{i+1}{N}} + \frac{h}{N} W \sum_{i=0}^{N-1} \beta_i y_{1-\frac{i+1}{N}}$$

Proposition (Méthode Multi-pas à pas constant - Ordre de convergence)

Si on a:

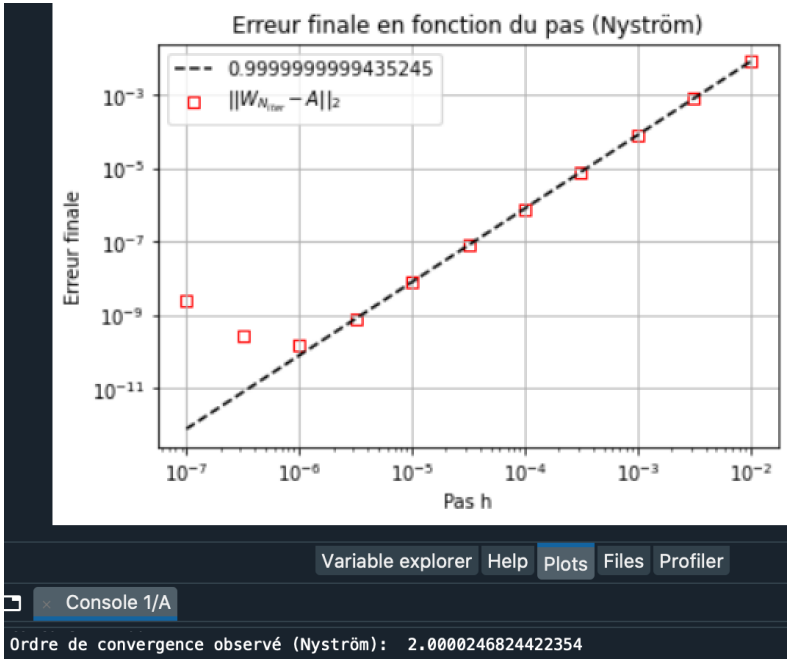
- $$\sum_{i=0}^{N-1} \alpha_i = 1$$
- *Pour tout* $q \in [\![1, p]\!]$*:*

$$\sum_{i=0}^{N-1} i^q \alpha_i - q i^{q-1} \beta_i \quad = \quad (-1)^q$$

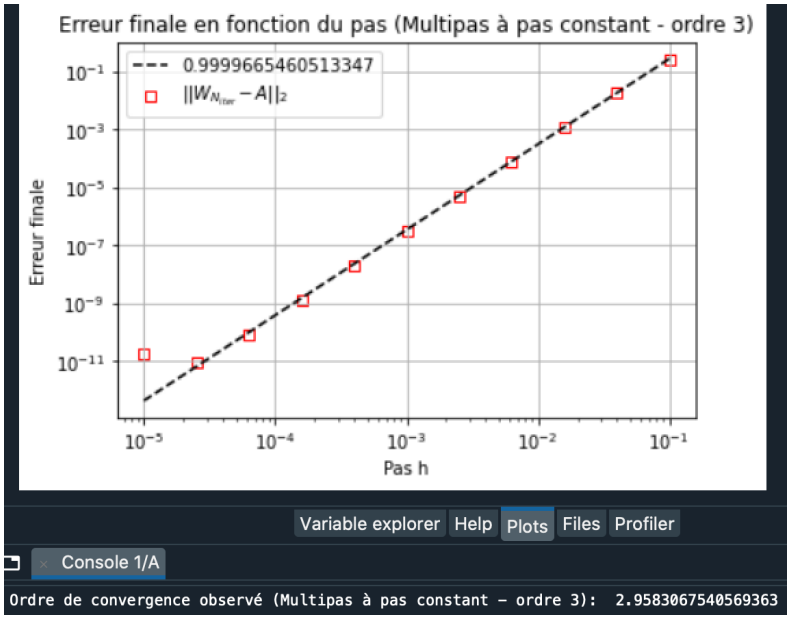
Alors:

$$W_{Min} \underset{h \to 0}{=} A + \mathcal{O}(h^p)$$

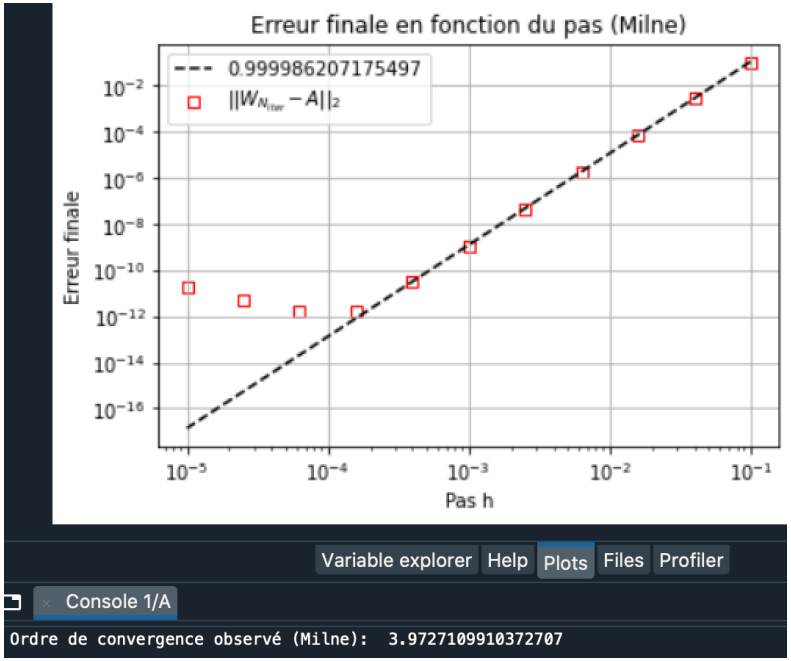
Preuve (idée): Utiliser le développement limité de l'exponentielle de matrice, puis les conditions en hypothèses.



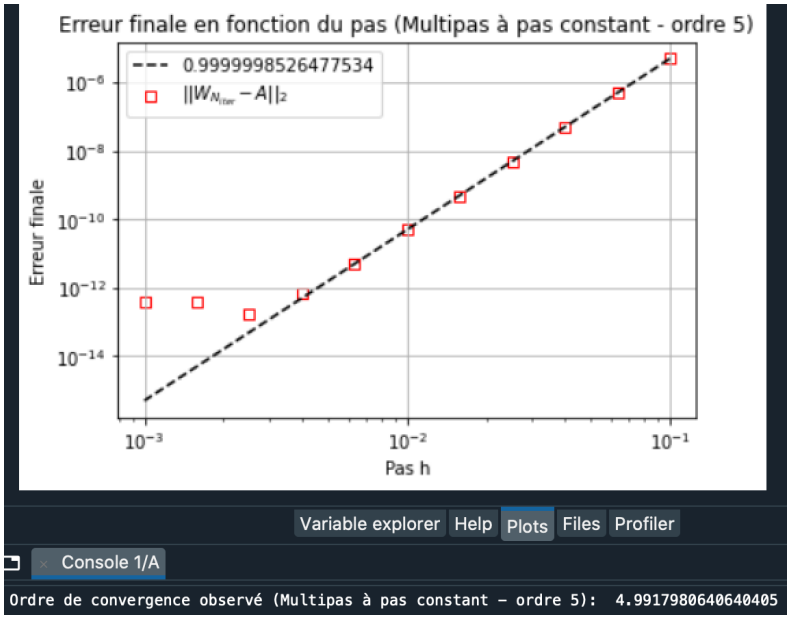
(a) **Méthode de Nyström:**
$N = 2$
$(\alpha_0, \alpha_1, \beta_0, \beta_1) = (0, 1, 2, 0)$
$\hat{y_1} = y_0 + \frac{h}{2} W(2y_{\frac{1}{2}})$



(b) **Méthode d'ordre 3:** $N = 2$
$(\alpha_0, \alpha_1, \beta_0, \beta_1) =$
$(-4, 5, 4, 2)$
$\hat{y_1} =$
$-4y_{\frac{1}{2}} + 5y_0 + \frac{h}{2} W(4y_{\frac{1}{2}} + 2y_0)$



(a) **Méthode de Milne:** $N = 4$
$(\alpha_0, \alpha_1, \alpha_2, \alpha_3, \beta_0, \beta_1, \beta_2, \beta_3) = (0, 0, 0, 1, \frac{8}{3}, -\frac{4}{3}, \frac{8}{3}, 0)$
$\hat{y_1} = y_0 + \frac{h}{4} W \left(\frac{8}{3} y_{\frac{3}{4}} - \frac{4}{3} y_{\frac{1}{2}} + \frac{8}{3} y_{\frac{1}{4}} \right)$



(b) **Méthode d'ordre 5:** $N = 3$
$(\alpha_0, \alpha_1, \alpha_2, \beta_0, \beta_1, \beta_2) = (-18, 9, 10, 9, 18, 3)$
$\hat{y_1} = -18 y_{\frac{2}{3}} + 9 y_{\frac{1}{3}} + 10 y_0 + \frac{h}{3} W (9 y_{\frac{2}{3}} + 18 y_{\frac{1}{3}} + 3 y_0)$

- **Données**: Solutions $y_{\frac{t_j}{h}} = e^{t_j A} y_0$ en les temps $0 = t_0 < t_1 < \cdots < t_N = h$
- **Méthode**:

$$\begin{aligned} \hat{y_1} &= y_0 + W \sum_{j=0}^{N} \left(\int_0^h L_{j,N}(t) \mathrm{d}t \right) y_{\frac{t_j}{h}} \\ &= y_0 + W \sum_{j=0}^{N} \left(\int_0^h L_{j,N}(t) \mathrm{d}t \right) e^{t_j A} y_0 \end{aligned}$$

$(L_{j,N})_{0 \leqslant j \leqslant N}$: Base de polynômes interpolateurs de Lagrange de $\mathbb{R}_N[X]$, interpolation de $t \mapsto We^{tA} y_0$ en $t_0, \cdots, t_N$.

Proposition (Méthode Multi-pas à pas non constant - Ordre de convergence)

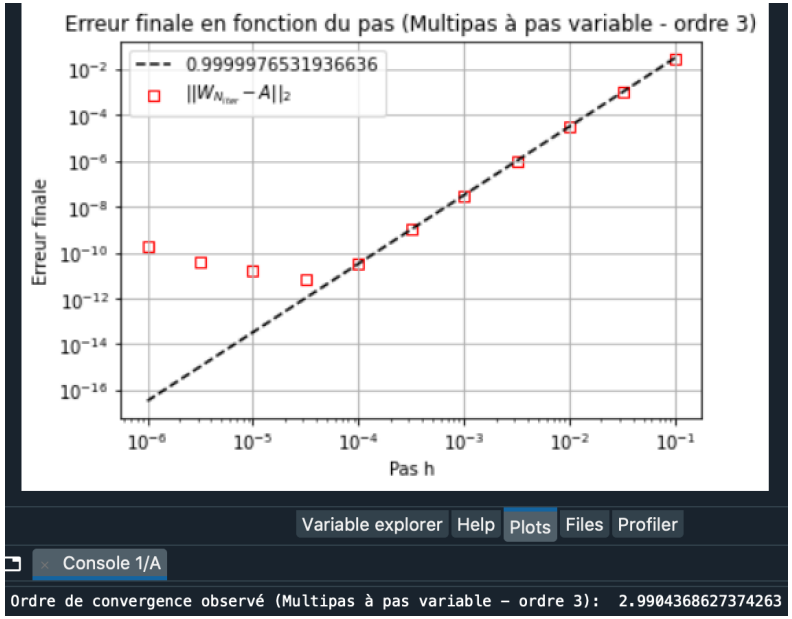
Si $A \in GL_d(\mathbb{R})$ (ce qui est le cas presque partout dans $\mathcal{M}_d(\mathbb{R})$), l'apprentissage pour cette méthode est d'ordre $N+1$:

$$W_{Min} \underset{h\to 0}{=} A + \mathcal{O}(h^{N+1})$$

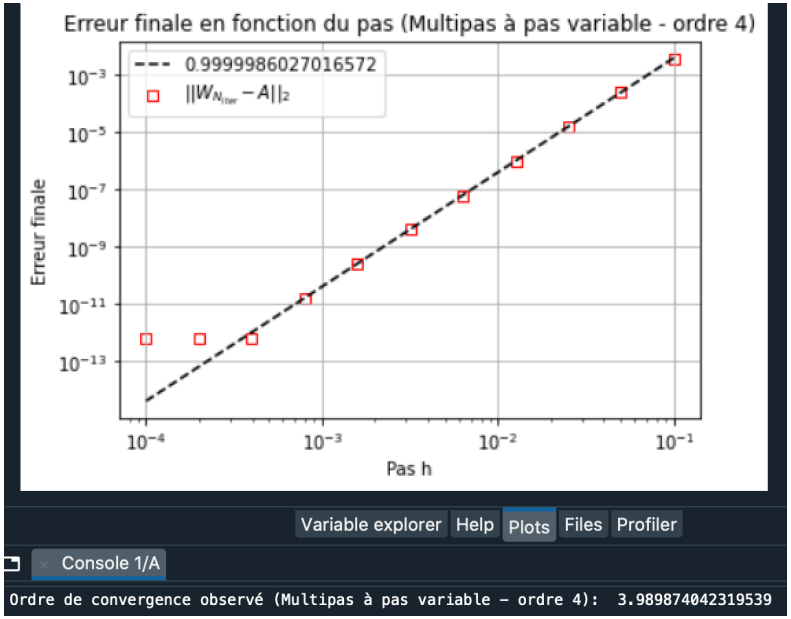
Preuve (idée): Étudier le minimiseur:

$$\begin{aligned} W_{Min} &= \left[\sum_{j=0}^{N}\left(\int_0^h L_{j,N}(t)\mathrm{d}t\right)e^{t_j A}\right]^{-1}\left(e^{hA}-I_d\right) \\ &= \left[\int_0^h e^{tA}\mathrm{d}t - \int_0^h e^{tA} - \left(\sum_{j=0}^{N} L_{j,N}(t)e^{t_j A}\right)\mathrm{d}t\right]^{-1}\left(e^{hA}-I_d\right) \end{aligned}$$

en notant que le polynôme interpolateur approche $t \mapsto e^{tA}$



(a) **Méthode d'ordre 3**: $N = 2$
Solutions données en 0, $\frac{h}{3}$ et h
Intégration numérique des $L_{j,2}(t)$ sur $[0, h]$



(b) **Méthode d'ordre 4**: $N = 3$
Solutions données en 0, $\frac{h}{4}$, $\frac{h}{2}$ et h
Intégration numérique des $L_{j,3}(t)$ sur $[0, h]$

- Solutions $y_{\frac{t_j}{h}} = e^{t_j A} y_0$ en $0 = t_0 < \cdots < t_N = h$ de $\dot{y} = Ay$.
- **Méthode**:
  - **Vecteur prédit par le réseau de neurones**:

$$\hat{y_1} = \sum_{j=0}^{N} \alpha_j \, y_{\frac{t_j}{h}} + W \sum_{j=0}^{N} \beta_j \, h_j \, y_{\frac{t_j}{h}}$$

  - **Vecteur prédit par la méthode numérique**:

$$\widetilde{y_1} = \sum_{j=0}^{N} \alpha_j \, y_{\frac{t_j}{h}} + A \sum_{j=0}^{N} \beta_j \, h_j \, y_{\frac{t_j}{h}}$$

Théorème (Ordre de convergence - Cas linéaire)

Si la méthode est d'ordre p, i.e.

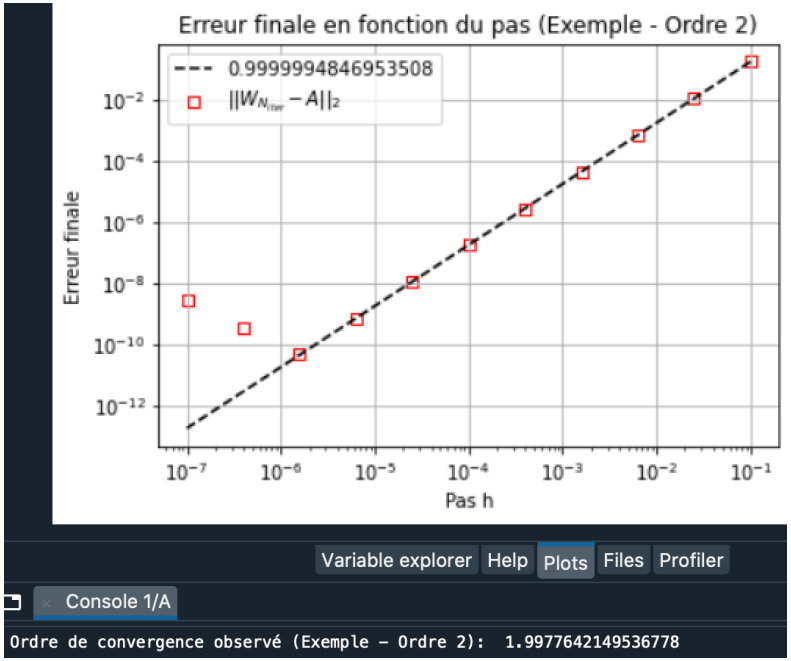
$$\widetilde{y_1} - y_1 \underset{h \to 0}{=} \mathcal{O}(h^{p+1})$$

et si $\frac{1}{h}\left(\beta_0 h_0 + \cdots + \beta_N h_N\right) \neq 0$, *alors l'apprentissage est d'ordre p:*

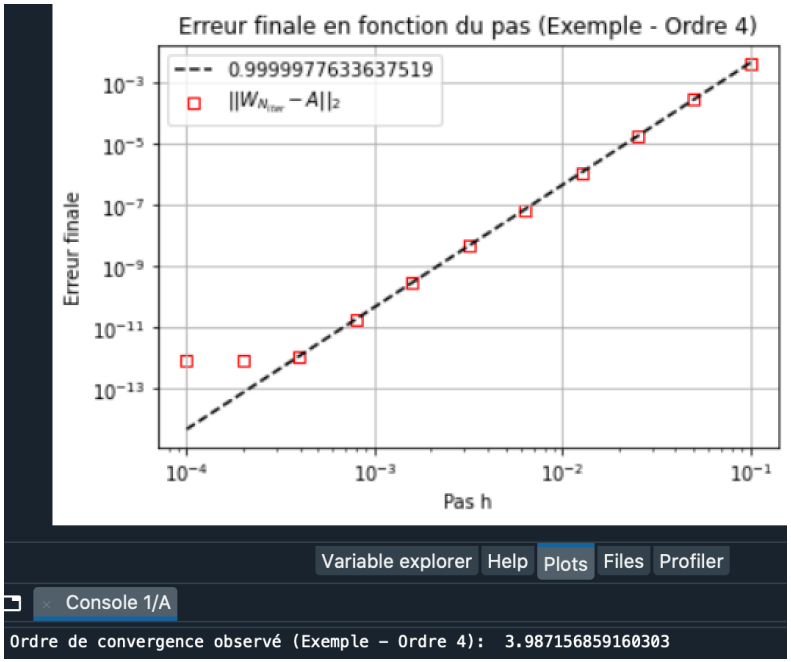
$$W_{Min} \underset{h \to 0}{=} A + \mathcal{O}(h^p)$$

Preuve (idée): Étudier le minimiseur en y_0:

$$W_{Min}\, y_0 = A y_0 + \left(\sum_{j=0}^{N} \beta_j \frac{h_j}{h} e^{t_j A}\right)^{-1} \frac{1}{h}\left(y_1 - \widetilde{y_1}\right)$$



(a) **Méthode d'ordre 2**:
Solutions données en 0, $\frac{h}{2}$ et h
$\hat{y_1} = y_0 + \frac{h}{4} W \left(y_0 + 2y_{\frac{1}{2}} + y_1 \right)$

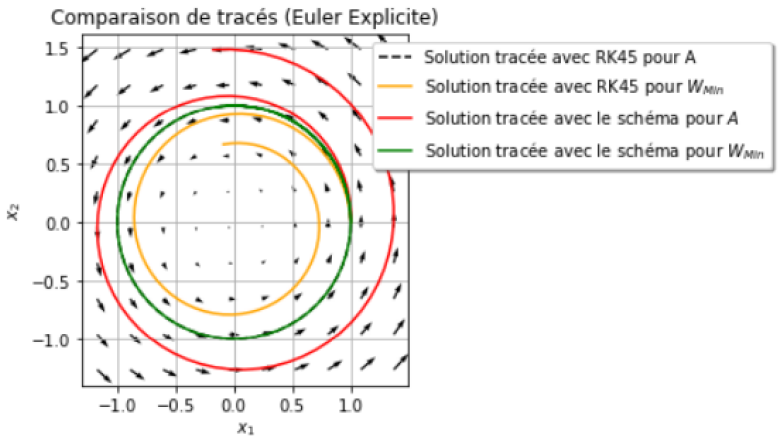


(b) **Méthode d'ordre 4**:
Solutions données en 0, $\frac{h}{3}$, $\frac{2h}{3}$ et h
$\hat{y_1} = y_0 + \frac{h}{8} W \left(y_0 + 3y_{\frac{1}{3}} + 3y_{\frac{2}{3}} + y_1 \right)$

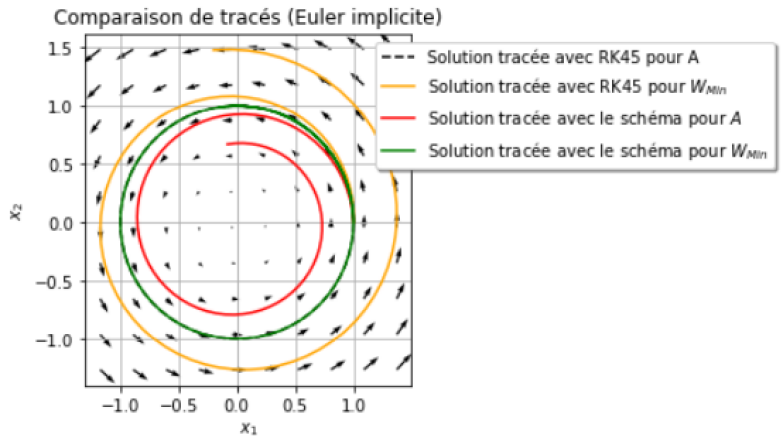
- **Système différentiel:**

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

- **Pas de la méthode numérique:** $h = 0.1$



(a) **Méthode d'Euler Explicite:**
$\hat{y_1} = (I_d + hW)y_0$

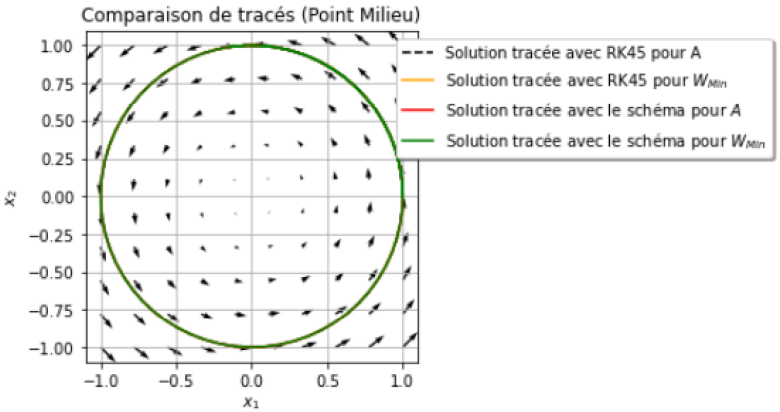


(b) **Méthode d'Euler Implicite:**
$\hat{y_1} = y_0 + hWy_1$

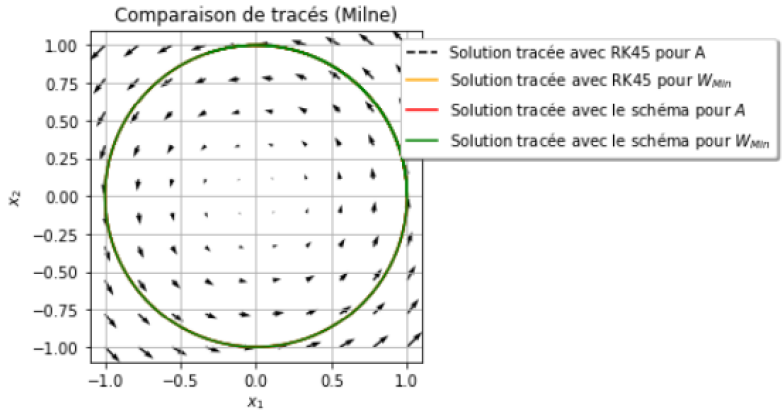
- **Système différentiel:**

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

- **Pas de la méthode numérique:** $h = 0.1$



(a) **Méthode du Point Milieu:**
$\hat{y_1} = y_0 + \frac{h}{2} W(y_0 + y_1)$



(b) **Méthode de Milne:**
$\hat{y_1} = y_0 +$
$\frac{h}{4} W \left(\frac{8}{3} y_{\frac{3}{4}} - \frac{4}{3} y_{\frac{1}{2}} + \frac{8}{3} y_{\frac{1}{4}} \right)$

Partie 4 : Cas non-linéaire

- **Équation**: $\dot{y} = f(y)$, $f \in \mathcal{C}^0(\mathbb{R}^d, \mathbb{R}^d)$, localement lipschitzien
- **Données**: Solutions $y_{\frac{t_j}{h}} = \Phi_{t_j}(y_0)$ (Φ: flot associé à l'EDO) en les temps $0 = t_0 < \cdots < t_N = h$
- K données initiales, donnant K séries de données: pour tout $k \in [\![0, K-1]\!]$ $y^{(k)}_{\frac{t_j}{h}} = \Phi_{t_j}\left(y_0^{(k)}\right)$

Théorème (Théorème d'universalité)

Soit $g \in \mathcal{C}(\mathbb{R}^d, \mathbb{R})$, *soit* $\sigma \in \mathcal{C}^0(\mathbb{R}, \mathbb{R}) \backslash \mathbb{R}[X]$. *Alors, pour tout* $\varepsilon > 0$, *il existe* $g_{app} \in \mathcal{M}_\sigma$ *telle que, pour tout* $\Omega \subset \mathbb{R}^d$ *compact:*

$$||g - g_{app}||_{L^\infty(\Omega)} \leqslant \varepsilon$$

avec:

- $$\mathcal{M}_\sigma = Vect\left\{x \mapsto \sigma(w \cdot x + b), w \in \mathbb{R}^d, b \in \mathbb{R}\right\}$$
- $$\mathbb{R}[X] \simeq \{Fonctions\ polynomiales\}$$

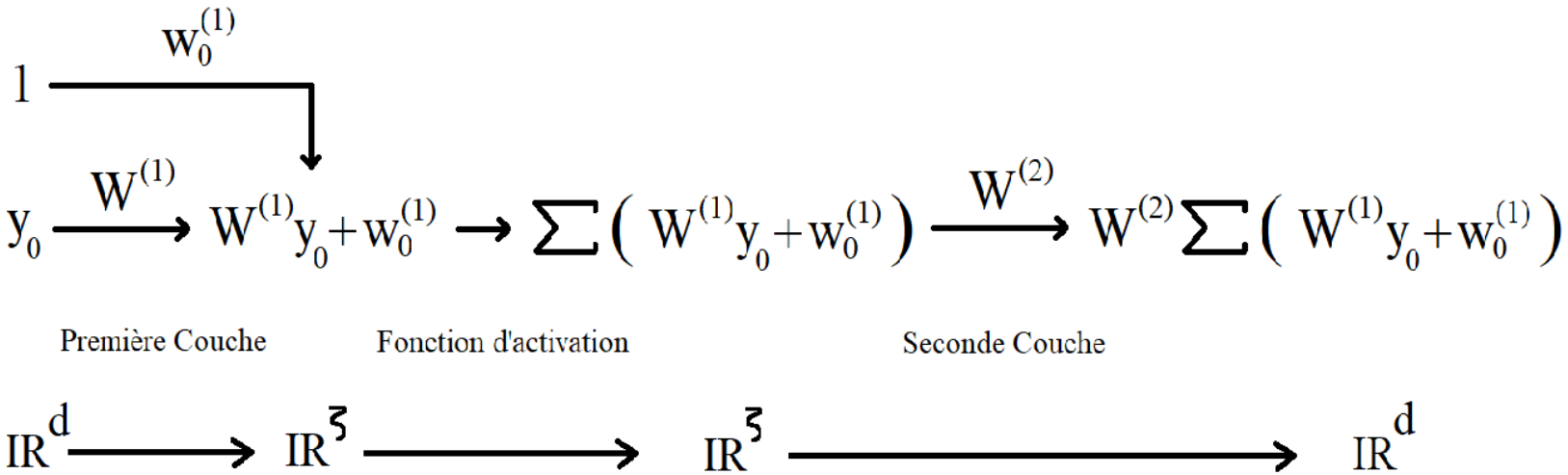


Figure: Structure du réseau de neurones pour le problème non-linéaire

$$f_{app}(y_0) \quad = \quad W^{(2)}\Sigma\left(W^{(1)}y_0 + w_0^{(1)}\right)$$

Première formulation:

- **Vecteur prédit par le réseau de neurones**:

$$\hat{y_1} = \sum_{j=0}^{N} \alpha_j \, y_{\frac{t_j}{h}} + h \sum_{j=0}^{N} \beta_j \, \frac{h_j}{h} f_{app} \left(y_{\frac{t_j}{h}} \right)$$

- **Vecteur prédit par la méthode numérique**:

$$\widetilde{y_1} = \sum_{j=0}^{N} \alpha_j \, y_{\frac{t_j}{h}} + h \sum_{j=0}^{N} \beta_j \, \frac{h_j}{h} f \left(y_{\frac{t_j}{h}} \right)$$

- **Minimiseur de la fonction *Loss***: $f_{app}^{*} \in \mathcal{C}^0(\mathbb{R}^d, \mathbb{R}^d)$ tel que $L(f_{app}^{*}) = 0$

Théorème (Ordre de convergence dans le cas général - Première formulation)

Si la méthode est d'ordre p, i.e.

$$\widetilde{y_1} - y_1 \underset{h \to 0}{=} \mathcal{O}(h^{p+1})$$

et si $\frac{1}{h}\left(\beta_0 h_0 + \cdots + \beta_N h_N\right) \neq 0$, *alors l'apprentissage est d'ordre p:*

$$f^*_{app}(y_0) \underset{h \to 0}{=} f(y_0) + \mathcal{O}(h^p)$$

Preuve (idée): Étudier le minimiseur en y_0 en montrant que:

$$f^*_{app}(y_0) \underset{h \to 0}{=} f(y_0) + \frac{1}{h \sum_{j=0}^{N} \beta_j \frac{h_j}{h}} \left(y_1 - \widetilde{y_1}\right)$$

Nouvelle formulation:

- **Vecteur prédit par le réseau de neurones:**

$$\hat{y_1} = \sum_{j=0}^{N} \alpha_j \, y_{\frac{t_j}{h}} + h f_{app} \left(\sum_{j=0}^{N} \beta_j \frac{h_j}{h} \, y_{\frac{t_j}{h}} \right)$$

- **Vecteur prédit par la méthode numérique:**

$$\widetilde{y_1} = \sum_{j=0}^{N} \alpha_j \, y_{\frac{t_j}{h}} + h f \left(\sum_{j=0}^{N} \beta_j \frac{h_j}{h} \, y_{\frac{t_j}{h}} \right)$$

- **Minimiseur de la fonction *Loss*:** $f^*_{app} \in \mathcal{C}^0(\mathbb{R}^d, \mathbb{R}^d)$ tel que $L(f^*_{app}) = 0$

Théorème (Ordre de convergence dans le cas général - Nouvelle formulation)

Si la méthode est d'ordre p, i.e.

$$\widetilde{y_1} - y_1 \underset{h \to 0}{=} \mathcal{O}(h^{p+1})$$

et si $\frac{1}{h}\left(\beta_0 h_0 + \cdots + \beta_N h_N\right) \neq 0$, *alors l'apprentissage est d'ordre* p*:*

$$f^*_{app}(y_0) \underset{h \to 0}{=} f(y_0) + \mathcal{O}(h^p)$$

Preuve (idée): Étudier le minimiseur en y_0 en montrant que:

$$\left[f^*_{app} - f\right]\left(\sum_{j=0}^{N} \beta_j \frac{h_j}{h} y_{\frac{t_j}{h}}\right) = \frac{1}{h}\left(y_1 - \widetilde{y_1}\right)$$

■

- **Système différentiel:**

$$\begin{cases} \dot{p} &= q \\ \dot{q} &= -\sin(p) \end{cases}$$

- **Pas de la méthode numérique:** $h = 0.1$

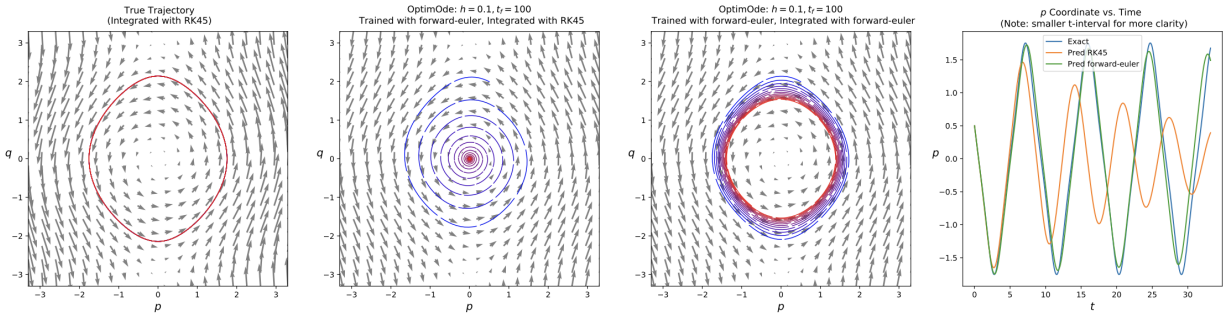


Figure: **Méthode d'Euler Explicite**, $\zeta = 20$ neurones

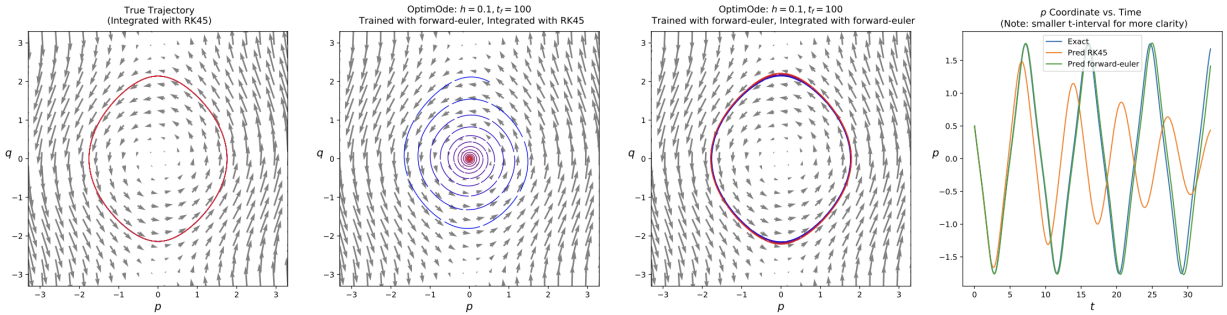


Figure: **Méthode d'Euler Explicite**, $\zeta = 200$ neurones

- **Système différentiel**:

$$\begin{cases} \dot{p} &= q \\ \dot{q} &= -\sin(p) \end{cases}$$

- **Pas de la méthode numérique**: $h = 0.1$

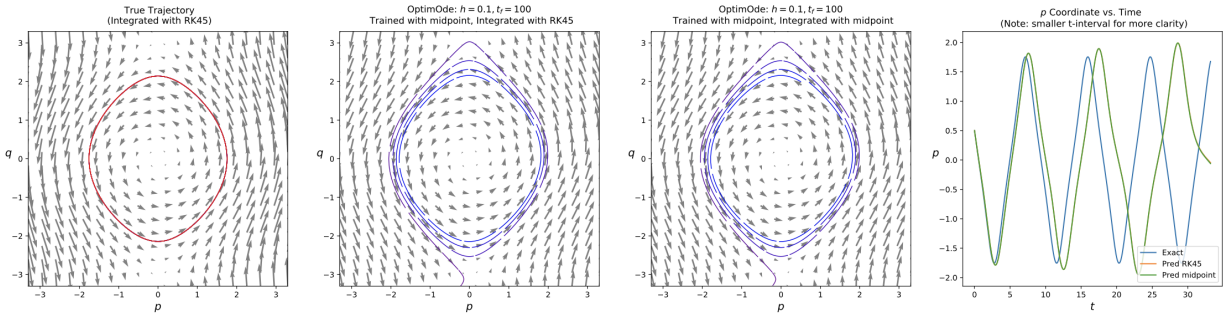


Figure: **Méthode du Point Milieu**, $\zeta = 20$ neurones

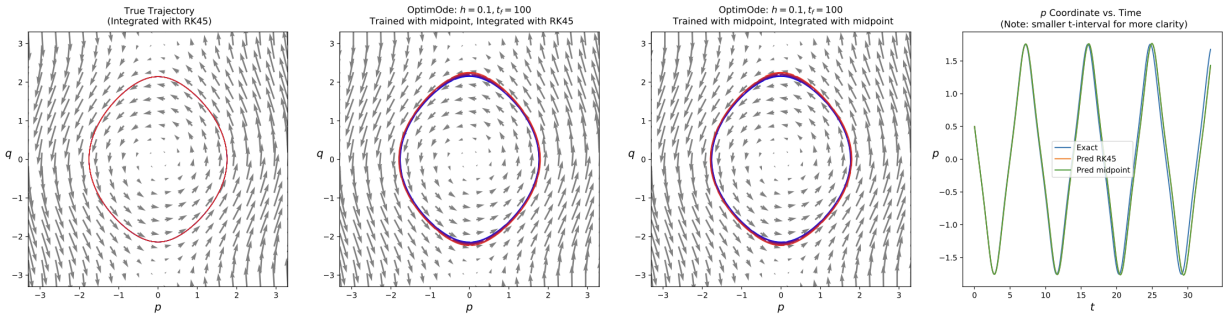


Figure: **Méthode du Point Milieu**, $\zeta = 200$ neurones

Merci pour votre attention !