Résolution d'équations différentielles à l'aide de réseaux de neurones

Matthieu Carreau, encadré par Stam Nicolis et Pascal Thibaudeau

Juillet 2022

1 Présentations de différentes approches pour la résolution d'une équation différentielle d'ordre 1

On cherche à tester nos méthodes sur l'équation différentielle suivante, pour $x \in [0, 1]$:

$$\begin{cases} \frac{d\Psi}{dx} + cos(2\pi x) = 0 \\ \Psi(0) = A \end{cases} \tag{1}$$

Cette équation avec condition initiale admet une unique solution analytique :

$$\Psi(x) = A - \frac{1}{2\pi} sin(2\pi x) \tag{2}$$

1.1 Recherche de solutions en série de Fourier

On cherche des solutions numériques approchées sous la forme de séries de Fourier tronquées avec M harmoniques :

$$\begin{cases} \tilde{\Psi}(x) = A + \mathcal{N}(x, P) \\ \mathcal{N}(x, P) = \sum_{m=1}^{M} A_m sin(2\pi m x) \end{cases} \tag{3}$$

P représente les coefficients $(A_m)_{m \in [\![1,M]\!]}$ qui sont les paramètres à ajuster. On cherche à obtenir la solution analytique, i.e $\forall m \in [\![1, M]\!], A_m = -\frac{1}{2\pi}\delta_1^m$.

On définit une fonction d'erreur pour ces solutions potentielles, en s'interressant aux N points suivants : $\forall i \in [\![1, N]\!], x_i = \frac{i}{N-1}$

$$E = \frac{1}{2} \sum_{i=1}^{N} (\sum_{m=1}^{M} 2\pi m A_m cos(2\pi m x_i) + cos(2\pi x_i))^2 \tag{4}$$

On calcule alors la dérivée partielle de cette erreur par rapport à chaque paramètre A_l :

$$\frac{\partial E}{\partial A_l} = \sum_{i=1}^{N}(\sum_{m=1}^{M} 2\pi m A_m cos(2\pi m x_i) + cos(2\pi x_i))2\pi l cos(2\pi l x_i) \tag{5}$$

Les deux méthodes suivantes ont pour objectif de trouver les coefficient $(A_m)_{m\in[\![1,M]\!]}$ qui minimisent E.

1.1.1 Première méthode : inversion d'un système linéaire

On cherche à résoudre le système linéaire donné par : $\forall m \in [\![1, M]\!]$, $\frac{\partial E}{\partial A_l} = 0$, on définit pour cela les matrices suivantes :

$$\mathcal{M} = (r_{m,l})_{(m,l)\in[\![1,M]\!]^2}, \vec{A} = \begin{pmatrix} A_1 \\ A_2 \\ \vdots \\ A_M \end{pmatrix}, \vec{b} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_M \end{pmatrix} \tag{6}$$

$$\forall (m,l) \in [\![1, N]\!]^2, \begin{cases} r_{m,l} = 2\pi m l \sum_{i=1}^{N} cos(2\pi m x_i) cos(2\pi l x_i) \\ b_l = -l \sum_{i=1}^{N} cos(2\pi x_i) cos(2\pi l x_i) \end{cases} \tag{7}$$

On résoud le système en écrivant l'équation matricielle le représentant :

$$\mathcal{M}\vec{A} = \vec{b} \Leftrightarrow \vec{A} = \mathcal{M}^{-1}\vec{b} \tag{8}$$

(Attention pour certains paramètres comme $(M = 5, N = 10)$, $\mathcal{M}$ n'est pas inversible).

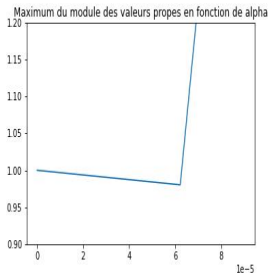
On initialise l'algorithme avec les paramètres suivants : $(M = 10, N = 100)$
On obtient les résultats suivants :
$A = [-1.59154943e-01, 4.74338450e-19, 1.08420217e-19, 2.71050543e-20, 2.74438675e-19, 1.05032085e-19, 9.82558219e-20, 5.92923063e-20, 5.42101086e-20, 5.42101086e-20]$
On constate comme attendu que le coefficient A_1 est très proche de $-\frac{1}{2\pi}$ (erreur relative de l'ordre de 10^{-16}), et que les autres coefficients ont une valeur absolue maximale de 1.1510^{-17}. On peut donc valider notre modèle.

1.1.2 Seconde méthode : descente de gradient

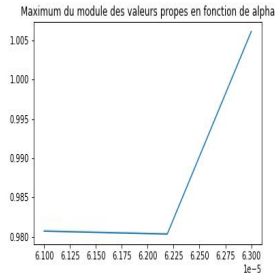
On définit les paramètres suivants :

$$\alpha > 0, \vec{A^{(0)}} = \begin{pmatrix} A_1^{(0)} \\ A_2^{(0)} \\ \vdots \\ A_M^{(0)} \end{pmatrix}, \vec{g}^{(0)} = \begin{pmatrix} \frac{\partial E^{(0)}}{\partial A_1^{(0)}} \\ \frac{\partial E^{(0)}}{\partial A_2^{(0)}} \\ \vdots \\ \frac{\partial E^{(0)}}{\partial A_M^{(0)}} \end{pmatrix}, \tag{9}$$

Puis on calcule itérativement :



(a) $\alpha \in [0, 10^{-4}]$



(b) Au voisinage de l'intersection avec 1

FIGURE 1 – Maximum des valeurs propres de $\mathcal{R}_\alpha$ en fonction de α

$$\vec{A}^{(k+1)} = \vec{A}^{(k)} - \alpha\vec{g}^{(k)} \tag{10}$$

On cherche à trouver le coefficient α optimal qui assure la convergence tout en maximisant la vitesse de convergence, c'est-à-dire le α le plus élevé possible qui permet la convegrence de la suite.

On exprime tout d'abord le gradient en fonction de la matrice $\mathcal{M}$ et du vecteur $\vec{b}$ définis précédemment qui sont indépendants de $\vec{A}$ et de k :

$$\vec{g}^{(k)} = \mathcal{M}\vec{A}^{(k)} - \vec{b} \tag{11}$$

Ainsi, l'équation de récurrence (10) se réécrit comme une suite arithmético-géométrique de vecteurs :

$$\vec{A}^{(k+1)} = (\mathcal{I}_M - \alpha\mathcal{M})\vec{A}^{(k)} + \alpha\vec{b} \tag{12}$$

On en déduit que la suite converge si et seulement si le maximum du module des valeurs propres de la matrice $\mathcal{R}_\alpha = \mathcal{I}_M - \alpha\mathcal{M}$ est strictement inférieur à 1. On trace donc ce maximum en fonction de α en figure 1. On en déduit la valeur critique $\alpha_c = 6.25.10^{-5}$.

On initialise l'algorithme avec les paramètres suivants : $(M = 10, N = 100, V_0 = 1, \alpha = 10^{-5})$ On obtient au bout de 10000 itérations les résultats suivants :
$A = [-1.59154943e-01, 1.15066541e-17, 7.99406786e-18, 5.64963221e-18, 4.88181580e-18, 3.85811144e-18, 3.46024465e-18, 2.88560910e-18, 2.81846501e-18, 2.43172156e-18]$
On constate comme attendu que le coefficient A_1 est très proche de $-\frac{1}{2\pi}$ (erreur relative de l'ordre de 10^{-15}), et que les autres coefficients ont une valeur absolue maximale de 1.1510^{-17}. On peut donc valider notre modèle.

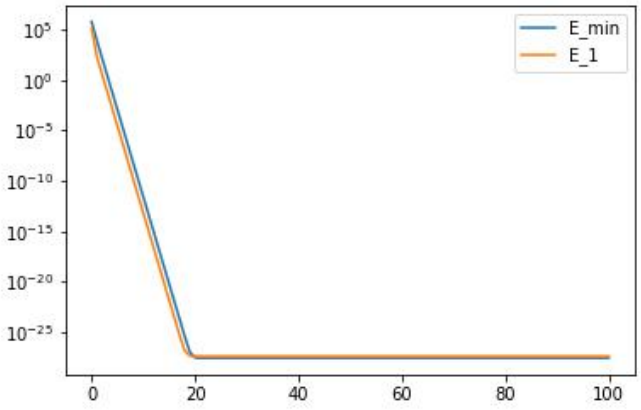


FIGURE 2 – Erreurs avec α_{min} et α_1

1.2 Recherche de solutions à l'aide d'un réseau de neurones

On cherche à présent à utiliser un réseau de neurones pour approcher la solution de l'équation différentielle. On cherche désormais des solutions approchées sous la forme suivante :

$$\begin{cases} \tilde{\Psi}(x) = A + \mathcal{N}(x, P) \\ \mathcal{N}(x, P) = \sum_{j=1}^{H} v_j \sigma(w_j x + b_j) \end{cases} \tag{13}$$

$\mathcal{N}(x, P)$ correspond donc à la sortie d'un réseau de neurones dont l'architecture est présentée en figure 3, contenant une couche cachée intermèdiaire, qui réalise en sortie une somme pondérée de sigmoïdes, la fonction utilisée est $\forall x \in \mathbf{R}, \sigma(x) = \frac{1}{1+e^{-x}}$. Les paramètres P à ajuster sont désormais les coefficients $(w_j)_{j\in[\![1,H]\!]}$, $(b_j)_{j\in[\![1,H]\!]}$ et $(v_j)_{j\in[\![1,H]\!]}$.

On définit une nouvelle fonction d'erreur, calculée à partir des mêmes N points que précédemment : $\forall i \in [\![1, N]\!], x_i = \frac{i}{N-1}$

$$E(P) = \sum_{i=1}^{N} (\frac{d\tilde{\Psi}}{dx}(x_i) + cos(2\pi x))^2 \tag{14}$$

On calcule ensuite les expressions analytiques des dérivées partielles de $E(P)$ par rapport à chaque paramètre ajustable, puis on cherche à minimiser cette erreur à l'aide de l'algorithme de descente de gradients.

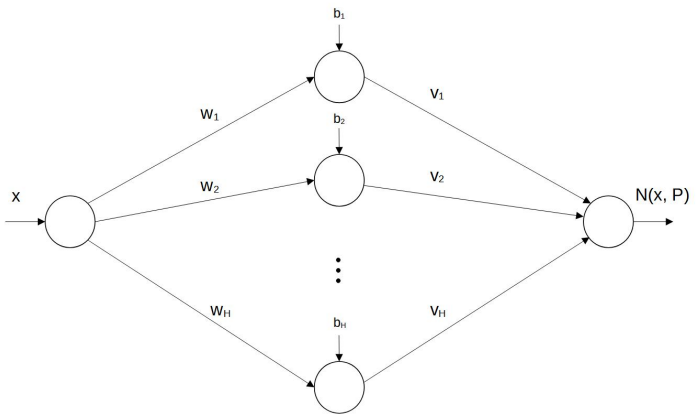


FIGURE 3 – Réseau de neurones

1.2.1 Résultats obtenus

On initialise l'algorithme avec les paramètres suivants : $(H = 4, N = 20)$ On obtient une erreur de $1,2.10^{-2}$ et une estimation visible en figure 4. Cela permet de valider notre modèle sur l'étude à une dimension.

2 Etude du cas de deux équations d'ordre 1 couplées, représentant un mouvement de précession

On s'intéresse désormais au problème de la précession d'un moment magnétique dans un champ magnétique constant. On le modélise par les équations suivantes pour $t \in [0, 1]$:

$$\begin{cases} \frac{dv_x}{dt} = \omega v_y \\ \frac{dv_y}{dt} = -\omega v_x \end{cases} \quad \begin{cases} v_x(0) = V_0 \\ v_y(0) = 0 \end{cases} \tag{15}$$

Ce problème admet une unique solution analytique :

$$\begin{cases} v_x(t) = V_0 cos(2\omega t) \\ v_y(t) = -V_0 sin(2\omega t) \end{cases} \tag{16}$$

2.1 Recherche des solutions en séries de Fourier

On cherche des solutions numériques approchées sous la forme de séries de Fourier tronquées avec M harmoniques, en posant la forme suivante :

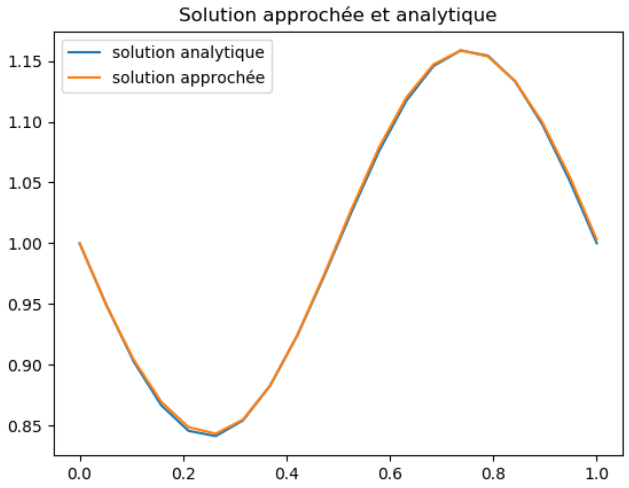


FIGURE 4 – estimation de la solution par un réseau de neurones

$$\begin{cases} \tilde{v}_x(t) = V_0 + \sum_{m=1}^{M} A_m(cos(\omega mt) - 1) + B_m sin(\omega mt) \\ \tilde{v}_y(t) = \sum_{m=1}^{M} -A_m sin(\omega mt) + B_m(cos(\omega mt) - 1) \end{cases} \tag{17}$$

Les coefficients $(A_m)_{m\in[\![1,M]\!]}$ et $(B_m)_{m\in[\![0,M]\!]}$ sont les paramètres à ajuster. On cherche à obtenir la solution analytique, i.e $\forall m \in [\![0, M]\!], A_m = \delta_1^m$ et $\forall m \in [\![1, M]\!], B_m = 0$. On remarque que le coefficient A_0 n'a aucune influence.

On définit une fonction d'erreur pour ces solutions potentielles, en s'interressant aux N points suivants : $\forall i \in [\![1, N]\!], t_i = \frac{i}{N-1}$:

$$E(P) = \sum_{i=1}^{N} (\frac{d\tilde{v}_x}{dt}(t_i) - \omega\tilde{v}_y(t_i))^2 + (\frac{d\tilde{v}_y}{dt}(t_i) + \omega\tilde{v}_x(t_i))^2 \tag{18}$$

On utilise ensuite la méthode de descente de gradients définie précédemment, en calculant les dérivées partielles suivantes : $(\frac{\partial E}{\partial A_l}, \frac{\partial E}{\partial B_l})_{l\in[\![1,M]\!]}$

2.1.1 Résultats obtenus

On initialise l'algorithme avec les paramètres suivants : $(M = 10, N = 100, V_0 = 1, \omega = 2\pi, \alpha = 10^{-6})$ On obtient au bout de 10000 itérations les résultats suivants :
$A = [1.00000255e+00, -1.23919994e-06, -2.20679520e-07, -9.12537244e-08, -4.93048945e-08, -3.05670278e-08, -2.06219718e-08, -1.47337251e-08, -1.09721848e-08, -8.43076573e-09]$,
$B = [-6.59235880e-07, 3.20274560e-07, 5.70352161e-08, 2.35847707e-08, 1.27429827e-08, 7.90013061e-09, 5.32980412e-09, 3.80797092e-09, 2.83579070e-$

$09, 2.17895410e-09]$
On constate comme attendu que le coefficient A_0 est très proche de 1 (erreur relative inférieure de 2.5510^{-6}), et que les autres coefficients ont une valeur absolue maximale de 1.2410^{-6}. On peut donc valider notre modèle.