

Résolution d'équations différentielles par réseaux de neurones

Matthieu Carreau
Telecom Paris, Institut Polytechnique de Paris
F-91120, Palaiseau, France
`matthieu.carreau@telecom-paris.fr`

Supervisors:
Stam Nicolis
Institut Denis Poisson
Université de Tours, Université d'Orléans, CNRS (UMR7013)
Parc de Grandmont, F-37200, Tours, France
`stam.nicolis@lmpt.univ-tours.fr`

Pascal Thibaudeau
CEA Le Ripault
BP 16, F-37260, Monts, France
`pascal.thibaudeau@cea.fr`

Juillet 2022

Résumé

Faire un résumé de ce qu'il y a dans ce document

Table des matières

1	Introduction	2
2	Equation différentielle d'ordre 1	3
2.1	Solutions en séries de Fourier	3
2.1.1	Première méthode : inversion d'un système linéaire . .	4
2.1.2	Seconde méthode : descente de gradients	5
2.2	Solutions par réseau de neurones	6
2.2.1	Résultats obtenus	8
3	Mouvement de précession	10
3.1	Solutions en séries de Fourier	10
3.1.1	Résultats obtenus	11
4	Conclusion et perspectives	12

Chapitre 1

Introduction

Cette partie positionne le travail dans un contexte. Décrire le contexte. Dire ici ce que l'on doit faire et proposer un petit résumé des documents lus de façon à montrer en quoi ils sont pertinents pour le problème posé. Par exemple : Bidulle et Machin dans la référence [1] ont montré que ... tandis que Truc et Chmuc dans la référence [2] ont prouvé que... Dans la section 2, je montrerai que... Dans la section 3, je montrerai... Enfin dans la section 4, je discuterai des résultats obtenus et proposerai quelques perspectives.

Chapitre 2

Equation différentielle d'ordre 1

Soit Ψ une fonction à une variable dont la solution satisfait l'équation différentielle suivante où A désigne ?

$$\begin{cases} \frac{d\Psi(x)}{dx} + \cos(2\pi x) = 0 \\ \Psi(0) = A \end{cases} \quad (2.1)$$

On cherche à tester les méthodes présentées dans la section 1 sur l'équation (2.1), pour tout $x \in [0, 1]$. L'équation (2.1) et sa condition initiale donnée en $x = 0$, admet une solution analytique unique qui s'écrit

$$\Psi(x) = A - \frac{1}{2\pi} \sin(2\pi x) \quad (2.2)$$

Qu'est-ce-ça permet de savoir pour la suite ?

2.1 Solutions en séries de Fourier

On cherche des solutions numériques approchées de l'équation (2.1) sous la forme de séries de Fourier tronquées avec M harmoniques :

$$\begin{cases} \tilde{\Psi}(x) = A + \mathcal{N}(x, P) \\ \mathcal{N}(x, P) = \sum_{m=1}^M A_m \sin(2\pi m x) \end{cases} \quad (2.3)$$

Définir ce que sont toutes ces grandeurs

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

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

$$E = \frac{1}{2} \sum_{i=1}^N \left(\sum_{m=1}^M 2\pi m A_m \cos(2\pi m x_i) + \cos(2\pi x_i) \right)^2 \quad (2.4)$$

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

$$\frac{\partial E}{\partial A_l} = \sum_{i=1}^N \left(\sum_{m=1}^M 2\pi m A_m \cos(2\pi m x_i) + \cos(2\pi x_i) \right) 2\pi l \cos(2\pi l x_i) \quad (2.5)$$

Pourquoi faut-il faire cela ?

Les deux méthodes suivantes ont pour objectif de trouver les coefficients $(A_m)_{m \in \llbracket 1, M \rrbracket}$ qui minimisent E .

2.1.1 Première méthode : inversion d'un système linéaire

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

$$\mathcal{M} = (r_{m,l})_{(m,l) \in \llbracket 1, M \rrbracket^2}, \mathbf{A} = \begin{pmatrix} A_1 \\ A_2 \\ \vdots \\ A_M \end{pmatrix}, \mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_M \end{pmatrix} \quad (2.6)$$

$$\forall (m, l) \in \llbracket 1, N \rrbracket^2, \begin{cases} r_{m,l} = 2\pi m l \sum_{i=1}^N \cos(2\pi m x_i) \cos(2\pi l x_i) \\ b_l = -l \sum_{i=1}^N \cos(2\pi x_i) \cos(2\pi l x_i) \end{cases} \quad (2.7)$$

On résout le système linéaire associé en écrivant l'équation matricielle le représentant, soit

$$\mathcal{M}\mathbf{A} = \mathbf{b} \Leftrightarrow \mathbf{A} = \mathcal{M}^{-1}\mathbf{b} \quad (2.8)$$

Comment s'y prend-on en pratique pour faire cela ? Qu'as-tu dû faire ?

(Attention pour certains paramètres comme $(M = 5, N = 10)$, \mathcal{M} n'est pas inversible). Que ce passe-t-il à ce moment là et pourquoi ?

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.

2.1.2 Seconde méthode : descente de gradients

On définit les paramètres suivants :

$$\alpha > 0, \vec{A}^{(0)} = \begin{pmatrix} A_1^{(0)} \\ A_2^{(0)} \\ \vdots \\ A_M^{(0)} \end{pmatrix}, \vec{g}^{(0)} = \begin{pmatrix} \frac{\partial E^{(0)}}{\partial A_1^{(0)}} \\ \frac{\partial E^{(0)}}{\partial A_2^{(0)}} \\ \vdots \\ \frac{\partial E^{(0)}}{\partial A_M^{(0)}} \end{pmatrix}, \quad (2.9)$$

Puis on calcule itérativement :

$$\vec{A}^{(k+1)} = \vec{A}^{(k)} - \alpha \vec{g}^{(k)} \quad (2.10)$$

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 convergence 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 :

$$\vec{g}^{(k)} = \mathcal{M}\vec{A}^{(k)} - \vec{b} \quad (2.11)$$

Ainsi, l'équation de récurrence (2.10) se réécrit comme une suite arithmético-géométrique de vecteurs :

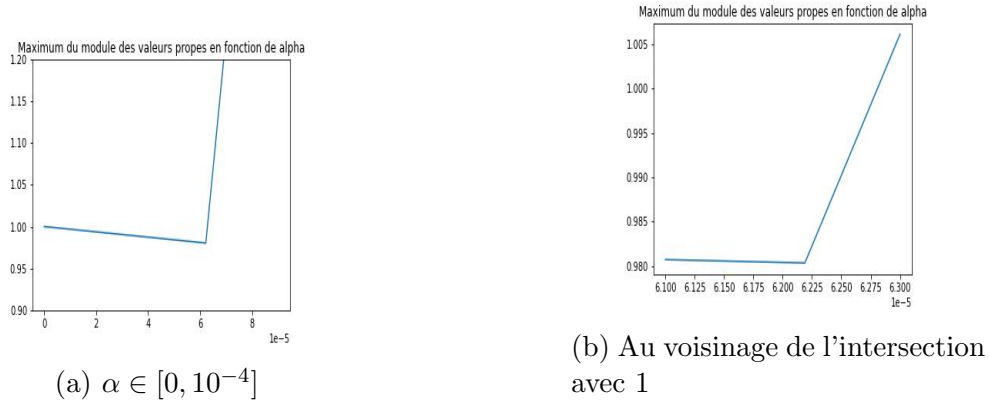


FIGURE 2.1 – Maximum des valeurs propres de \mathcal{R}_α en fonction de α

$$\vec{A}^{(k+1)} = (\mathcal{I}_M - \alpha \mathcal{M}) \vec{A}^{(k)} + \alpha \vec{b} \quad (2.12)$$

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 2.1. On en déduit la valeur critique $\alpha_c = 6.25 \cdot 10^{-5}$.

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.

2.2 Solutions par 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 :

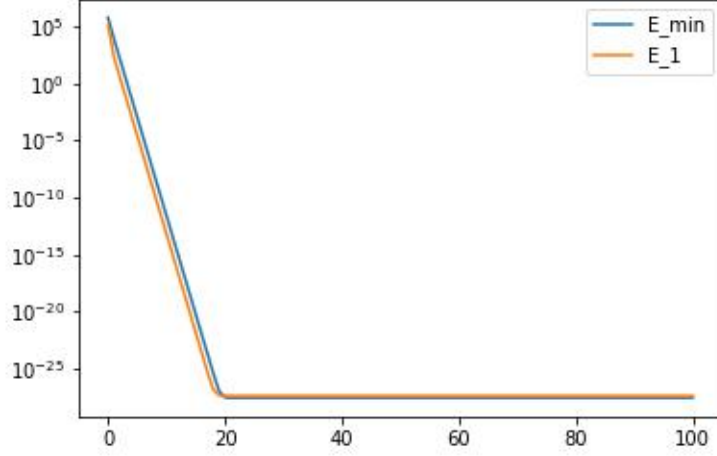


FIGURE 2.2 – Erreurs avec α_{min} et α_1

$$\begin{cases} \tilde{\Psi}(x) = A + \mathcal{N}(x, P) \\ \mathcal{N}(x, P) = \sum_{j=1}^H v_j \sigma(w_j x + b_j) \end{cases} \quad (2.13)$$

$\mathcal{N}(x, P)$ correspond donc à la sortie d'un réseau de neurones dont l'architecture est présentée en figure 2.3, contenant une couche cachée intermédiaire, 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 $(w_j)_{j \in \llbracket 1, H \rrbracket}$, $(b_j)_{j \in \llbracket 1, H \rrbracket}$ et $(v_j)_{j \in \llbracket 1, H \rrbracket}$.

Peux-tu donner une référence pour quelqu'un qui cherche ce que tout ceci veut dire ?

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

$$E(P) = \sum_{i=1}^N \left(\frac{d\tilde{\Psi}}{dx}(x_i) + \cos(2\pi x) \right)^2 \quad (2.14)$$

L'équation (2.14) est-elle correcte ?

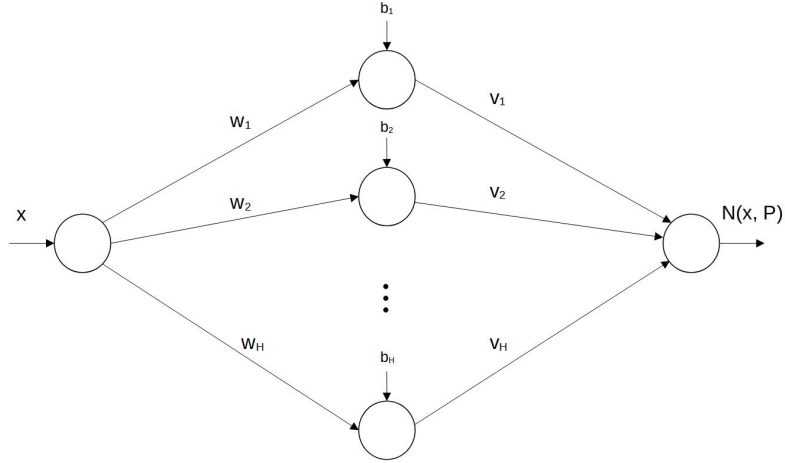


FIGURE 2.3 – Réseau de neurones

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.

2.2.1 Résultats obtenus

On initialise l'algorithme avec les paramètres suivants : ($H = 4, N = 20$) On obtient une erreur de $1,2 \cdot 10^{-2}$ et une estimation visible en figure 2.4. Cela permet de valider notre modèle sur l'étude à une dimension.

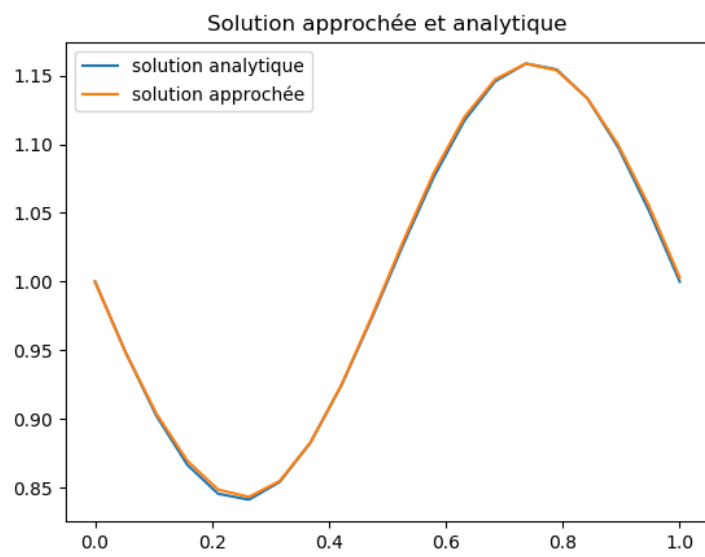


FIGURE 2.4 – estimation de la solution par un réseau de neurones

Chapitre 3

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 $t \in [0, 1]$:

$$\begin{cases} \frac{dv_x}{dt} = \omega v_y \\ \frac{dv_y}{dt} = -\omega v_x \end{cases} \quad (3.1)$$

avec les conditions initiales

$$\begin{cases} v_x(0) = V_0 \\ v_y(0) = 0 \end{cases} \quad (3.2)$$

dont la solution analytique vaut

$$\begin{cases} v_x(t) = V_0 \cos(2\omega t) \\ v_y(t) = -V_0 \sin(2\omega t) \end{cases} \quad (3.3)$$

3.1 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 :

$$\begin{cases} \tilde{v}_x(t) = V_0 + \sum_{m=1}^M A_m(\cos(m\omega t) - 1) + B_m \sin(m\omega t) \\ \tilde{v}_y(t) = \sum_{m=1}^M -A_m \sin(m\omega t) + B_m(\cos(m\omega t) - 1) \end{cases} \quad (3.4)$$

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

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

$$E(P) = \sum_{i=1}^N \left(\frac{d\tilde{v}_x}{dt}(t_i) - \omega \tilde{v}_y(t_i) \right)^2 + \left(\frac{d\tilde{v}_y}{dt}(t_i) + \omega \tilde{v}_x(t_i) \right)^2 \quad (3.5)$$

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

3.1.1 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.

Chapitre 4

Conclusion et perspectives

Bibliographie

- [1] C.Bidule and A.Machin, Journal of Computer Power **12** 123 (2020)
- [2] C.Truc and T.Chmuc, (2020)