M2 MMS : Réseaux de neurones pour la modélisation

July 12, 2025

Contents

Introduction

Objectifs

- Connaître les bases des machines et réseaux de neurones
- Savoir les utiliser dans le contexte de modèles basées sur les équations différentielles ordinaires (EDO)
- Savoir utiliser `pytorch`

Chaque chapitre correspond à 1-3 cours/TP.

1 Machines et réseaux de neurones

Définition 1.1. *Soit $m, n, n_w \in \mathbb{N}$. Une **machine** est une application*

$$\Phi : \mathbb{R}^n \times \mathbb{R}^{n_w} \to \mathbb{R}^m, \qquad (x, \tilde{w}) \to y = \Phi(x, \tilde{w}).$$

***L'apprentissage** consiste à fixer $w = w_*$ de sorte que l'application $x \mapsto \Phi(x, w_*)$ permet de mieux représenter des données ou un modèle physique.*

*On appelle Φ une **machine vectoriel**, si*

$$\Phi(x, \tilde{w}) = \sum_{i=1}^{N} c_i \phi_i(x, w).$$

Dans ce cas nous avons $\tilde{w} = (x, w)$ et $X_\Phi := \operatorname{Im} \Phi = \operatorname{vect} \{\phi_i(x, w) \mid 1 \leq i \leq N\}$.

Exemple 1.2. *Voici quelques exemples de machines vectoriels.*

- *L'interpolation de Lagrange d'une fonction univariée continue $f : [0; 1] \to \mathbb{R}$ avec les points d'interpolation $0 = t_0 < \cdots < t_{i-1} < t_i < t_N = 1$.*

$$I_n f(t) = \sum_{i=0}^{N} c_i \phi_i(t), \quad \phi_i(t) = \prod_{\substack{j=1 \\ j \neq i}}^{N} \frac{t - t_j}{t_i - t_j}.$$

  Ici, nous avons "$w = \emptyset$".

- *Même chose, mais avec les points d'interpolation variables, donc "$w = (t_i)$".*

- *Espace $P^1([0; 1])$. Similaire au deux précédents.*

- *Espace des séries de Fourier tronquées. Similaire au précédents.*

Définition 1.3. *Pour $f : \mathbb{R} \to \mathbb{R}$ on définit $f : \mathbb{R}^n \to \mathbb{R}^n$ (sans distinction de notation !) composante par composante, $(f(x))_i = f(x_i), 1 \leq i \leq n$.*

*Un **réseau à une couche** $(W, b, \sigma), W \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$ est l'application*

$$\Phi(x, W, b, \sigma) : \mathbb{R}^n \to \mathbb{R}^m, \qquad x \mapsto \sigma(Wx + b). \tag{1}$$

*Un **réseau multi-couches (MLP)** est la composition de réseau à une couche $\Phi_i = \Phi_i(W_i, b_i, \sigma_i)$, $1 \leq i \leq L$ avec*

$$\Phi(x, \tilde{w}) : \mathbb{R}^n \to \mathbb{R}^m \qquad \Phi(x, \tilde{w}) = \Phi_L \circ \cdots \circ \Phi_1. \tag{2}$$

Évidemment nous avons $\tilde{w} = (W_i, b_i)_{1 \leq i \leq L}$ et

$$W_i \in \mathbb{R}^{n_i \times n_{i-1}}, \quad b_i \in \mathbb{R}^{n_i}, \quad n_0 = n, \quad n_L = m.$$

Remarque 1.4. *Un MLP est une machine vectoriel, si $b_L = 0$ et $\sigma_L = \mathrm{id}$. Nous avons $N = n_{L-1}$.*

2 Fonction de perte

Définition 2.1. *On appelle* ***données*** *(data) un ensemble* $\mathcal{D} = (x_i, y_i)_{1 \leq i \leq d}$, $x_i \in \mathbb{R}^n$, $y \in \mathbb{R}^m$.

Définition 2.2. *On appelle* ***fonction de perte*** *(loss) une fonction* $l : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$ *et l'****apprentissage*** *(learning) est le problème de minimisation*

$$\min \{l(w, \mathcal{D}) \mid w \in \mathbb{R}^p\}, \quad l(w, \mathcal{D}) = \sum_{i=1}^{d} l(\Phi(x_i, w), y_i). \tag{3}$$

Exemple 2.3.

- *(moindre carrés,* l^2*)* $l(z, y) = \frac{1}{2}\|y - z\|^2$.
- *(*l^1*)* $l(z, y) = \|y - z\|_{l^1(\mathbb{R}^m)}$.

2.1 Classification binaire

$$y_i \in \{-1, +1\}. \tag{4}$$

2.2 Classification

$$y_i \in [\![1, \cdots, n_c]\!]. \tag{5}$$

2.3 Moindre carré

$$l(z, y) = \frac{1}{2}\|y - z\|^2. \tag{6}$$

3 Apprentissage

3.1 Méthode de gradient

3.2 Différentation automatique

3.3 Apprentissage par étape

4 EDO et méthode de collocation

On considère $U \subset \mathbb{R}^n$ ouvert, $u_0 \in U$ et une fonction $f \in C^1(U, \mathbb{R}^n)$. L'équation différentielle ordinaire (EDO) d'ordre un autonome

$$\begin{cases} \dfrac{du}{dt} = f(u) \\ u(0) = u_0. \end{cases} \tag{7}$$

On rappelle le théorème de Cauchy-Lipschitz. S'il existe $r > 0$ et $L > 0$ tel que

$$\|f(u) - f(v)\| \leq L\|u - v\| \quad \forall u, v \in U \cap B_r(u_0), \tag{8}$$

alors il existe $T > 0$ une solution unique de (7) sur $[0, T]$. $f \in C^1(U, \mathbb{R}^n)$ est suffisant pour (8).

4.1 Méthode de collocation

Une méthode de collocation sur un intervalle $I = [0; T]$ consiste à choisir un espace $X_n \subset C^1(I, \mathbb{R}^n)$ de dimension $N + 1$ et des points (maillage)

$$0 = t_0 < t_1 < \cdots < t_N = T \tag{9}$$

et imposer les équations à une fonction $u_N \in X_N$

$$u_N(0) = u_0, \quad \frac{du_N}{dt}(t_i) = f(u_N(t_i)) \quad 1 \leq i \leq N. \tag{10}$$

Soit $(\phi_j)_{0 \leq j \leq N}$ une base de X_N. (10) est alors converti en un système algébrique pour les coefficient $c \in \mathbb{R}^{N+1}$

$$u_N(t) = \sum_{j=0}^{N} c_j \phi_j(t), \quad \sum_{j=0}^{N} c_j \phi_j(0) = u_0, \quad \sum_{j=0}^{N} c_j \frac{d\phi_j}{dt}(t_i) = f(\sum_{j=0}^{N} c_j \phi_j(t)).$$

Exemple 4.1. *Pour $X_N = B^1(I)$, l'espace des spline quadratique de classe C^1 sur le **même** maillage $(t_i)_{0 \leq i \leq N}$ on obtient un schéma de type Crank-Nicoloson.*

4.2 Fonction loss

Une première idée est d'utiliser un espace $\tilde{X} = X_\Phi$ généré par une machine vectorielles Φ est d'utiliser la fonction perte

$$\frac{1}{2}\|\tilde{u}(0) - u_0\|^2 + \sum_{i=0}^{N} \frac{1}{2}\|\frac{d\tilde{u}}{dt}(t_i) - f(\tilde{u}(t_i)\|^2$$

Il est alors facile de rajouter un term avec des données (PINN) $\mathcal{D} = (\tilde{t}_k, \tilde{w}_k)$ (mesures expermientales)

$$l_{\text{PINN}}(\tilde{u}) = \frac{1}{2}\|\tilde{u}(0) - u_0\|^2 + \sum_{i=0}^{N} \frac{1}{2}\|\frac{d\tilde{u}}{dt}(t_i) - f(\tilde{u}(t_i)\|^2 + \sum_{k=1}^{d} \frac{1}{2}\|\tilde{u}(\tilde{t}_k) - \tilde{w}_k\|^2$$

Remarque 4.2. *Dans le cas sans données ($d = 0$), si le MLP produit $B^1(I)$, la machine produit la solution de Crank-Nicolson. Une difficulté est le conditionnement du problème.*

4.3 EDO avec paramètres

Dans la pratique, les modèles mathématiques contiennent des paramètres (physique ou non). Soit $n_p \in \mathbb{N}$ et $p = (p_0, p_1) \in \mathbb{R}^{n_p}$ et

$$\begin{cases} \dfrac{du}{dt} = f(u, p_1) \\ u(0) = u_0(p_0). \end{cases} \tag{11}$$

Nous avons alors une application

$$S : \mathbb{R}^{n_p} \to C^1(I, \mathbb{R}^n) \qquad p \to u(p). \tag{12}$$

Des questions typiques sont

- Déterminer des paramètres à partir de mesures.
- Plus modestement : déterminer la sensibilité des solution par rapport aux paramètre.
- Plus ambitieux : déterminer les mesures les plus importantes.
- Dans un autre registre : trouver des valeurs critiques des paramètre. Les valeurs critiques sont celles quand la solution $p \to u(p)$ change de comportement, par exemple des points de bifurcation.

La clé à toutes ces questions est l'étude de l'application S définie en (12). Si elle est différentiable nous avons

$$\begin{cases} u := S(p), \quad \delta u := S'(p)(\delta p) \\ \begin{cases} \dfrac{d\delta u}{dt} = f'_u(u, p_1)\delta u + f'_p(u, p_1)\delta p_1 \\ \delta u(0) = u'_0(p_0)(\delta p_0). \end{cases} \end{cases} \tag{13}$$

5 Réseaux pour les EDO

6 Calcul de dérivées