

RÉSOLUTION D'ÉQUATIONS AUX DÉRIVÉES PARTIELLES PAR L'USAGE DE RÉSEAUX DE NEURONES.

PROBLÉMATIQUE

Comment utiliser les réseaux de neurones afin de mettre en œuvre une méthode de résolution des équations aux dérivées partielles ?

Comment peut-on mettre en pratique une telle méthode dans le cadre de la résolution du modèle LWR ?

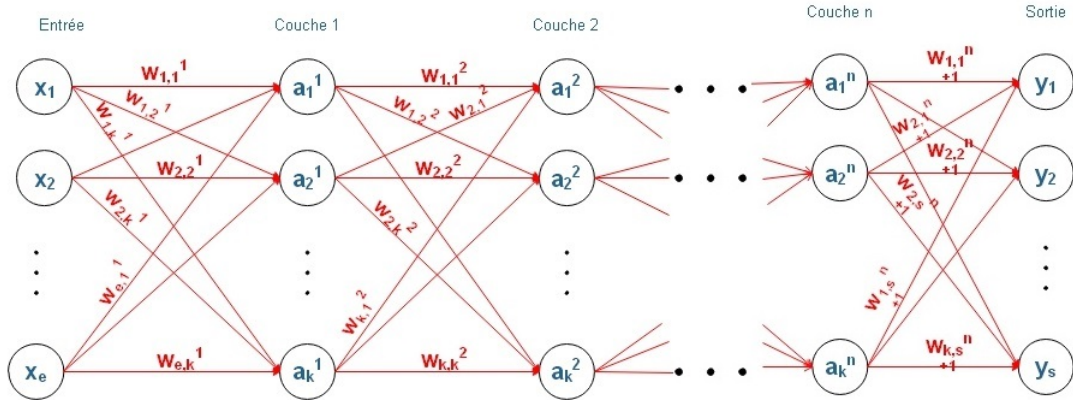
1. PRÉSENTATION DE LA MÉTHODE DE RÉSOLUTION ET DU MODÈLE LWR

1.1. Les réseaux de neurones.

Definition 1. Un réseau de neurones est une fonction non linéaire de ses entrées.

On appelle NN :
$$\begin{cases} \mathbb{R}^e \rightarrow \mathbb{R}^s \\ \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_e \end{bmatrix} \mapsto \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_s \end{bmatrix} \end{cases} \quad \text{une fonction définie par un réseau de neurones.}$$

Cette fonction est définie par le graphe suivant :



Notation :

- n est le nombre de couches du réseau.
- k est le nombre de neurones par couche.
- $\forall i, j \in [1, k] \times [1, n]$ a_i^j est le neurone i de la couche j
- $\forall i, j, l \in [1, k] \times [1, k] \times [1, n]$ $w_{i,j}^l$ est le poids de a_i^{l-1} à a_j^l
- $\forall i, j \in [1, k] \times [1, n]$ b_i^j est le biais attaché à a_i^j (non représenté sur le graphe)
- $\sigma : \mathbb{R} \rightarrow \mathbb{R}$ est la fonction d'activation.

Chaque neurone est l'image par σ d'une combinaison linéaire des neurones de la couche précédente : $\forall i, j \in [1, k] \times [1, n]$ $a_i^j = \sigma(\sum_{m=1}^k w_{m,i}^j x_m + b_i^j)$.

Ainsi on possède une expression littérale de la fonction NN :

$$\forall i \in [1, s] \quad y_i = \sum_{m=1}^k w_{m,i}^n \sigma(\sum_{n=1}^k w_{n,m}^j (\dots (\sigma(\sum_{m=1}^k w_{m,p}^1 x_m + b_p^1) + b_i^j) + b_i^j).$$

Le principe de la méthode de résolution est de modifier les poids et les biais de façon à ce que la fonction NN satisfasse une équation aux dérivées partielles. Ce principe repose sur le théorème d'approximation universelle.

Theorem 2. *Théorème d'approximation universelle (Hornik) : un réseau de neurones contenant un nombre fini de neurones peut approximer des fonctions continues sur des sous-ensembles compacts et ceci à n'importe quel degré de précision.*

1.2. Le modèle LWR.

Introduction et hypothèses. Le modèle LWR décrit l'évolution macroscopique du trafic sur une route unidimensionnelle, sans intersection, en considérant le trafic comme un fluide incompressible et en faisant l'hypothèse d'un milieu continu. On peut alors définir une densité de voitures $\rho(x, t)$ et une vitesse moyenne $v(x, t)$.

Ce modèle est complété par une relation empirique entre ces deux grandeurs. On choisit le schéma de Newell simplifié: $v(\rho) = (1 - \exp(-(1 - \frac{1}{\rho})))$

On définit le flux de voitures $f(\rho) = \rho v = \rho(1 - \exp(-(1 - \frac{1}{\rho})))$.

1.2.1. Loi de conservation.

Theorem 3.

- *Loi de conservation :* $\frac{d}{dt} \int_{x_1}^{x_2} \rho(x, t) dx = f(x_1, t) - f(x_2, t)$
- *Forme locale :* $\frac{\partial \rho}{\partial t} + \frac{\partial f \circ \rho}{\partial x} = 0$

Proof.

La route étant sans intersection la variation de la quantité de voitures entre x_1 et x_2 : $\frac{d}{dt} \int_{x_1}^{x_2} \rho(x, t) dx$ s'écrit aussi comme le flux de voitures entrant en x_1 moins le flux de voitures sortant en x_2 .

D'où $\frac{d}{dt} \int_{x_1}^{x_2} \rho(x, t) dx = f \circ \rho(x_1, t) - f \circ \rho(x_2, t)$.

Le segment $[x_1, x_2]$ étant fixe, on a

$$\begin{aligned} \int_{x_1}^{x_2} \frac{\partial \rho(x, t)}{\partial t} dx &= - \int_{x_1}^{x_2} \frac{\partial f \circ \rho(x, t)}{\partial x} dx. \Rightarrow \int_{x_1}^{x_2} \left(\frac{\partial \rho(x, t)}{\partial t} + \frac{\partial f \circ \rho(x, t)}{\partial x} \right) dx = 0 \\ &\Rightarrow \frac{\partial \rho}{\partial t} + \frac{\partial f \circ \rho}{\partial x} = 0 \text{ (car l'égalité précédente est valable pour tout segment)} \end{aligned}$$

□

$\frac{\partial \rho}{\partial t} + \frac{\partial f \circ \rho}{\partial x} = 0$ est l'équation aux dérivées partielles, de fonction inconnue ρ , que on cherche à résoudre.

1.2.2. Situation initiale étudiée.

On se place sur le carré $[0, 1]^2$ et on étudie une situation initiale où on arrive dans un embouteillage :

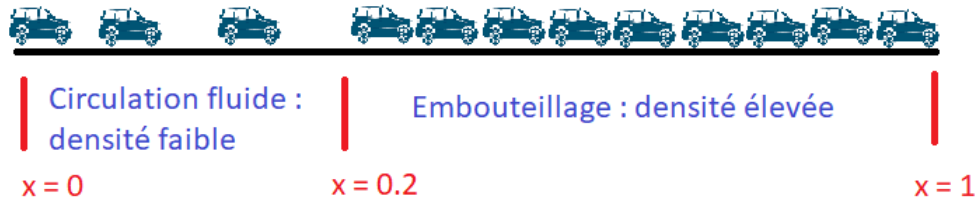


FIGURE 1.1. Schéma de la situation initiale

Cette situation se traduit mathématiquement par des conditions initiales pour notre équation aux dérivées partielles au temps $t = 0$:

$$\rho(x, 0) = g(x) \text{ avec } g : \begin{cases} [0, 1] \rightarrow [0, 1] \\ x \mapsto \begin{cases} 0.65 \text{ si } x \leq 0.2 \\ 0.9 \text{ sinon} \end{cases} \end{cases}$$

1.3. Détail de la méthode de résolution.

La méthode de résolution consiste à définir par une fonction de coût une mesure de l'écart entre la fonction NN réalisée par le réseau de neurones et une solution du problème, puis à utiliser un algorithme d'optimisation pour réduire cet écart.

1.3.1. Implémentation de la fonction de coût.

L'équation étant $\frac{\partial \rho}{\partial t} + \frac{\partial f \circ \rho}{\partial x} = 0$ on va chercher à minimiser $\frac{\partial NN(x, t)}{\partial t} + \frac{\partial f \circ NN(x, t)}{\partial x}$ pour tous points (x, t) de l'espace.

Ainsi on définit la fonction de coût $\varphi = \sum_{x, t \in I} \left(\frac{\partial NN(x, t)}{\partial t} + \frac{\partial f \circ NN(x, t)}{\partial x} \right)^2$ pour un sous-ensemble I de notre espace.

I est constitué d'un nombre fixé de points pris au hasard.

Remarque : Pour accéder au coût on a besoin de calculer les dérivées partielles de NN par rapport aux variables de l'espace. Cette étape est facile et sans approximation car on connaît l'expression littérale de NN.

1.3.2. Implémentation des conditions initiales.

On cherche aussi à ce que notre fonction NN satisfasse les conditions initiales. On propose deux implémentations :

a) On satisfait les conditions initiales par construction.

Cette implémentation est la plus communes dans la littérature. Elle consiste à construire une fonction NN* satisfaisant les conditions initiales :

$$\forall x, t \in [0, 1]^2 \quad NN^* = A(x) + t \cdot NN(x, t) \text{ avec } A : \begin{cases} \mathbb{R} \rightarrow \mathbb{R} \\ x \mapsto \begin{cases} 0.65 \text{ si } x \leq 0.2 \\ 0.9 \text{ sinon} \end{cases} \end{cases}$$

Ainsi $\forall x \in [0, 1] \quad NN^*(x, 0) = A(x)$ et A satisfait les conditions initiales. Il suffit alors de remplacer NN par NN* dans l'expression du coût.

On a alors $\varphi = \sum_{x, t \in I} \left(\frac{\partial NN^*(x, t)}{\partial t} + \frac{\partial f \circ NN^*(x, t)}{\partial x} \right)^2$.

Néanmoins cette méthode n'a pas été concluante pour notre équation, ainsi on propose une seconde implémentation.

b) On ajoute les conditions initiales dans le coût.

C'est à dire qu'on ajoute un terme lors de l'évaluation du coût local en un point (x, t), celui est nul si $t \neq 0$ et rend compte de l'écart aux conditions initiales sinon :

$$\varphi = \sum_{x, t \in I} \left[\left(\frac{\partial NN(x, t)}{\partial t} + \frac{\partial f \circ NN(x, t)}{\partial x} \right)^2 + ci \cdot B(x, t)^2 \right] \text{ et } B : \begin{cases} \mathbb{R}^2 \rightarrow \mathbb{R} \\ x, t \mapsto \begin{cases} 0 \text{ si } t \neq 0 \\ NN(x, t) - 0.65 & \text{si } t = 0 \text{ et } x \leq 0.2 \\ NN(x, t) - 0.9 & \text{si } t = 0 \text{ et } x \geq 0.2 \end{cases} \end{cases}$$

$ci \in \mathbb{R}$ est un coefficient contrôlant le poids des conditions initiales dans le coût final.

Cette seconde méthode possède un inconvénient : lorsqu'on définit I en prenant des points aux hasard, on doit s'assurer qu'un nombre suffisant de points à $t=0$ appartiennent à I, afin de bien rendre compte de l'écart aux conditions initiales.

1.3.3. Algorithme du gradient.

Maintenant qu'on a défini notre coût, on cherche à le minimiser. On fait le choix de l'algorithme du gradient comme algorithme d'optimisation.

L'algorithme du gradient se déroule en deux étapes :

- Calcul de $\nabla\varphi(w_{1,1}^1, w_{1,2}^1, \dots, w_{k,k}^{n+1}, b_1^1, \dots, b_k^{n+1})$; i.e $\forall i, j, l \in [[1, k]] \times [[1, k]] \times [[1, n]] \frac{\partial\varphi}{\partial w_{i,j}^l}$ et $\frac{\partial\varphi}{\partial b_i^l}$;
- "Descente du gradient" $\forall i, j, l \in [[1, k]] \times [[1, k]] \times [[1, n]] \begin{cases} w_{i,j}^l \leftarrow w_{i,j}^l - v \cdot \frac{\partial\varphi}{\partial w_{i,j}^l} \\ b_i^l \leftarrow b_i^l - v \cdot \frac{\partial\varphi}{\partial b_i^l} \end{cases}$; $v \in \mathbb{R}^+$ est la vitesse d'apprentissage

Remarque :

L'algorithme du gradient permet d'arriver à un minimum local mais rien ne prouve qu'il s'agit d'un minimum global.

1.3.4. Pseudo-code.

Algorithmic 1 pseudo-code

```

1: Mise en place du réseau
2: Initialisation des poids et biais
3: N = nombre de périodes d'apprentissage
4: Définition de la fonction coût
5:
6: for  $j = 1$  to  $N$  do
7:   I = Ensemble des points d'apprentissage pris au hasard
8:    $\varphi = 0$ 
9:
10:  for  $i$  in  $I$  do
11:     $\varphi += \text{coût}(i)$ 
12:  end for
13:  Calcul des dérivées partielles de  $\varphi$ 
14:  Descente du gradient
15: end for

```

2. ANALYSE DES RÉSULTATS ET MÉTHODE DES CARACTÉRISTIQUES

Avant d'évaluer la méthode de résolution il est nécessaire de s'assurer de la cohérence des résultats. On utilise des outils mathématiques comme la méthode des caractéristiques et la condition de Rankine-Hugoniot pour vérifier la pertinence des résultats ainsi que la condition d'entropie pour distinguer les solutions physiquement acceptables.

2.1. Méthode des caractéristiques.

La méthode des caractéristiques est une méthode de résolution partielle de notre équation aux dérivées partielles qui consiste à chercher des courbes (appelées caractéristiques) le long desquelles la solution à notre problème est constante.

Proposition 4. $\frac{\partial\rho(x,t)}{\partial t} + \frac{\partial f \circ \rho(x,t)}{\partial x} = 0 \Leftrightarrow \frac{\partial\rho(x,t)}{\partial t} + f'(\rho) \cdot \frac{\partial\rho(x,t)}{\partial x} = 0$

Theorem 5. ρ solution de l'équation est constante le long de la courbe $X : t \mapsto (x(t), t) \Rightarrow \forall t \in [0, 1] \begin{cases} \frac{d\rho \circ X}{dt}(t) = 0 \\ x'(t) = f'(\rho(x(t), t)) \end{cases}$

Proof.

$$\begin{aligned} \text{Soit } \rho \text{ solution constante le long de la courbe} &\Rightarrow \forall t \in [0, 1] \frac{d\rho \circ X}{dt}(t) = 0 \\ &\Rightarrow \frac{\partial \rho}{\partial x}(x(t), t) \cdot x'(t) + \frac{\partial \rho}{\partial t}(x(t), t) = 0 \end{aligned}$$

$$\text{Puis } \rho \text{ solution de l'équation} \Rightarrow \frac{\partial \rho}{\partial x}(x(t), t) \cdot f'(\rho) + \frac{\partial \rho}{\partial t}(x(t), t) = 0$$

$$\text{Donc } \forall t \in [0, 1] \frac{\partial \rho}{\partial x}(x(t), t) \cdot (x'(t) - f'(\rho)) = 0$$

$$\text{D'où en supposant } \frac{\partial \rho}{\partial x}(x(t), t) \neq 0, \text{ on a } x'(t) = f'(\rho(x(t), t)).$$

□

Corollary 6. La caractéristique partant de x_0 a pour équation $x(t) = f'(\rho(x_0, 0)) \cdot t + x_0$

Proof.

$$\begin{aligned} \begin{cases} x'(t) = f'(\rho(x(t), t)) \\ \rho(x(t), t) \text{ constante} \end{cases} &\Rightarrow x' \text{ constante} \\ &\Rightarrow \text{la courbe } (x(t), t) \text{ est une droite} \\ &\Rightarrow x(t) = f'(\rho(x_0, 0)) \cdot t + x_0 \end{aligned}$$

□

Ainsi les caractéristiques dépendent uniquement des conditions initiales. On peut donc l'appliquer à notre problème :

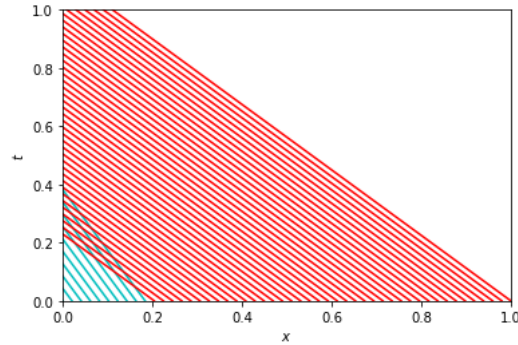


FIGURE 2.1. Résultat de la méthode des caractéristiques

On observe deux types de caractéristiques (en rouge et en bleu) issues des deux cas des conditions initiales : $x \leq 0.2$ et $x > 0.2$.

On remarque aussi une zone où les caractéristiques se chevauchent. Sa présence implique qu'il n'y a pas unicité pour la solution de notre problème. De plus, elle indique généralement la présence d'une discontinuité.

2.2. Solutions faibles.

L'étude précédente avec la méthode des caractéristiques met en avant des solutions discontinues. De telles solutions ne peuvent être des solutions classiques de notre problème car ρ solution de $\frac{\partial \rho}{\partial t} + \frac{\partial f \circ \rho}{\partial x} = 0$ est nécessairement continue car différentiable. Il faut donc introduire l'ensemble des solutions faibles.

Definition 7. Une fonction ρ intégrable sur $\mathbb{R} \times \mathbb{R}^+$ est une solution faible si et seulement si pour toute fonction $\varphi \in C^1(\mathbb{R} \times \mathbb{R}^+, \mathbb{R})$ à support compact, elle respecte l'identité :

$$\int_{\mathbb{R}^+} \int_{\mathbb{R}} (\rho \frac{\partial \varphi}{\partial t} + f(\rho) \frac{\partial \varphi}{\partial x}) dx dt + \int_{\mathbb{R}} g(x) \varphi(x, 0) dx = 0$$

Les conditions de régularité ne sont plus portées par la solution ρ mais par la fonction $\varphi \in C^1(\mathbb{R} \times \mathbb{R}^+, \mathbb{R})$.

Proposition 8. *L'ensemble des solutions faibles contient les solutions classiques.*

Soit ρ une solution classique de $\frac{\partial \rho}{\partial t} + \frac{\partial f(\rho)}{\partial x} = 0$

$$\Rightarrow \forall \varphi \in C^1(\mathbb{R} \times \mathbb{R}^+, \mathbb{R}) \text{ à support compact, } \int_{\mathbb{R}^+} \int_{\mathbb{R}} (\frac{\partial \rho}{\partial t} + \frac{\partial f(\rho)}{\partial x}) \varphi dx dt = 0.$$

$$\Rightarrow [\int_{\mathbb{R}} \rho(x, t) \varphi(x, t) dx]_0^{+\infty} - \int_{\mathbb{R}^+} \int_{\mathbb{R}} (\frac{\partial \varphi}{\partial t}) \rho dx dt + [\int_{\mathbb{R}^+} f(\rho(x, t)) \varphi(x, t) dt]_{-\infty}^{+\infty} - \int_{\mathbb{R}^+} \int_{\mathbb{R}} \frac{\partial \varphi}{\partial x} f(\rho) dx dt = 0$$

(Intégration par partie)

$$\Rightarrow \int_{\mathbb{R}^+} \int_{\mathbb{R}} (\rho \frac{\partial \varphi}{\partial t} + f(\rho) \frac{\partial \varphi}{\partial x}) dx dt + \int_{\mathbb{R}} g(x) \varphi(x, 0) dx = 0 \quad (\varphi \text{ à support compact})$$

2.3. Condition de Rankine Hugoniot.

Soit ρ une solution faible admettant une discontinuité. On note :

- $(X(t), t)$ la courbe sur laquelle se situe la discontinuité, $(\lambda(t), 1)$ sa dérivée.
- $\Omega = [0, 1]^2$; $\Omega^+ = [0, 1] \times [X(t), 1]$; $\Omega^- = [0, 1] \times [0, X(t)]$
- La frontière $\partial\Omega = \{(x(t), t) | t \in [0, 1]\}$
- $\rho|_{\Omega^+} = \rho^+$; $\rho|_{\Omega^-} = \rho^-$

Theorem 9. *Condition de Rankine-Hugoniot : $\lambda = \frac{f(\rho^+) - f(\rho^-)}{\rho^+ - \rho^-}$*

Proof.

On admet les théorèmes suivants :

□

- (1) $\text{div}(\varphi \vec{v}) = \varphi \text{div}(\vec{v}) + \vec{v} \cdot \overrightarrow{\text{grad}}(\varphi)$
- (2) $\iiint_{\Omega} \text{div}(\vec{v}) dV = \iint_{\partial\Omega} \vec{v} \cdot \vec{ds}$ avec \vec{ds} un vecteur unitaire normal à la frontière dirigé vers l'extérieur.

Soit $\varphi \in C^1(\mathbb{R} \times \mathbb{R}^+, \mathbb{R})$ à support compact telle que $\forall x \in \mathbb{R} \varphi(x, 0) = 0$,

$$D'après la définition d'une solution faible : \int \int_{\Omega} (\rho \frac{\partial \varphi}{\partial t} + f(\rho) \frac{\partial \varphi}{\partial x}) dx dt = 0$$

$$\Rightarrow \int \int_{\Omega^+} (\rho \frac{\partial \varphi}{\partial t} + f(\rho) \frac{\partial \varphi}{\partial x}) dx dt + \int \int_{\Omega^-} (\rho \frac{\partial \varphi}{\partial t} + f(\rho) \frac{\partial \varphi}{\partial x}) dx dt = 0$$

$$\begin{aligned}
\text{Puis } \int \int_{\Omega^-} \rho \frac{\partial \varphi}{\partial t} dx dt &= \int \int_{\Omega^-} \rho \operatorname{div}(0, \varphi) dx dt \\
&= \int \int_{\Omega^-} \operatorname{div}(\rho(0, \varphi)) dx dt - \int \int_{\Omega^-} (0, \varphi) \cdot \overrightarrow{\operatorname{grad}}(\varphi) dx dt \quad (d'après (1)) \\
&= \int_{\partial \Omega^-} \rho(0, \varphi) \cdot \overrightarrow{dv} - \int \int_{\Omega^-} (0, \varphi) \cdot \overrightarrow{\operatorname{grad}}(\varphi) dx dt \quad (d'après (2)) \\
&\text{avec } \overrightarrow{dv} = \alpha(1, -\lambda) dv \text{ et } \alpha = (1 + \lambda^2)^{-\frac{1}{2}} \\
&= - \int_{\partial \Omega^-} \alpha \rho \varphi \lambda(t) dv - \int \int_{\Omega^-} \frac{\partial \varphi}{\partial t} dx dt
\end{aligned}$$

$$\begin{aligned}
\text{Ainsi } \int \int_{\Omega^-} (\rho \frac{\partial \varphi}{\partial t} + f(\rho) \frac{\partial \varphi}{\partial x}) dx dt &= - \int \int_{\Omega^-} \varphi (\frac{\partial \rho^-}{\partial t} + \frac{\partial f \rho^-}{\partial x}) dx dt + \int_{\partial \Omega^-} (-\rho^- \varphi \lambda(t) + f(\rho^-) \varphi) \alpha dv \\
\text{De même, } \int \int_{\Omega^+} (\rho \frac{\partial \varphi}{\partial t} + f(\rho) \frac{\partial \varphi}{\partial x}) dx dt &= - \int \int_{\Omega^+} \varphi (\frac{\partial \rho^+}{\partial t} + \frac{\partial f \rho^+}{\partial x}) dx dt + \int_{\partial \Omega^-} (-\rho^+ \varphi \lambda(t) + f(\rho^+) \varphi) \alpha dv
\end{aligned}$$

$$\begin{aligned}
&\Rightarrow - \int_{\partial \Omega^-} (-\rho^+ \lambda(t) + f(\rho^+)) \varphi \alpha dv + \int_{\partial \Omega^-} (-\rho^- \lambda(t) + f(\rho^-)) \varphi \alpha dv = 0 \\
&\Rightarrow \int_{\partial \Omega^-} ((\rho^+ - \rho^-) \lambda(t) + (f(\rho^-) - f(\rho^+))) \varphi \alpha dv = 0 \\
&\Rightarrow (\lambda(t)(\rho^+ - \rho^-) + (f(\rho^-) - f(\rho^+))) = 0
\end{aligned}$$

2.4. Présentation des résultats.

1ère implémentation des conditions initiales.

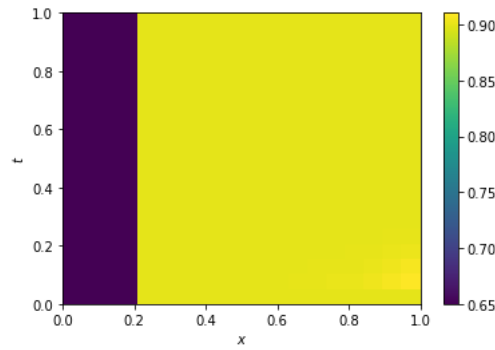


FIGURE 2.2. Solution 1

Avec cette implémentation des conditions initiales le résultat est décevant, on obtient seulement une solution “triviale” où les dérivées partielles de la fonction sont nulles de part et d’autre d’une ligne de discontinuité. Ce résultat ne correspond pas au résultat de la méthode des caractéristiques.

2ème implémentation des conditions initiales.

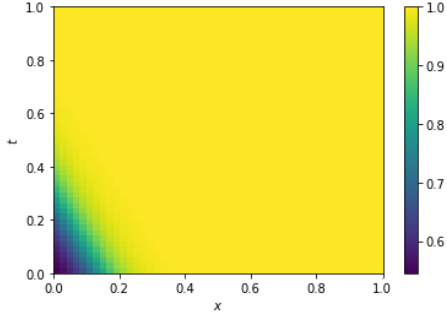


FIGURE 2.3. Solution 2

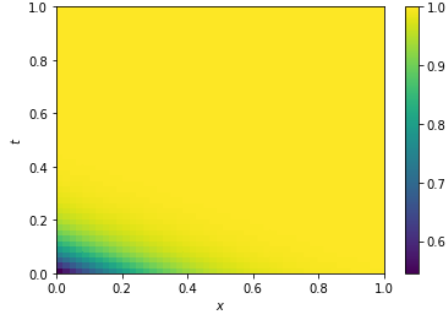


FIGURE 2.4. Solution 3

Pour cette seconde implémentation on obtient un ensemble de solutions en accord avec la méthode des caractéristiques.

On peut distinguer les solutions les plus acceptables grâce à la condition de Rankine-Hugoniot : $\lambda = \frac{f(\rho^+) - f(\rho^-)}{\rho^+ - \rho^-}$.

Application numérique : $f'(0.65) = 0.27$ et $f'(0.9) = 0.095 \Rightarrow \lambda = -0.70$

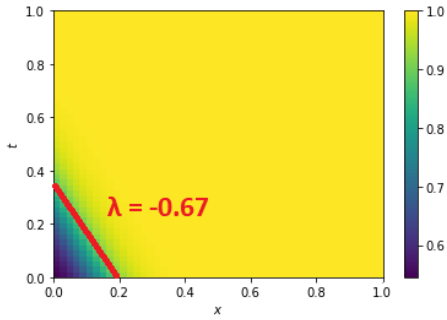


FIGURE 2.5. Solution 2

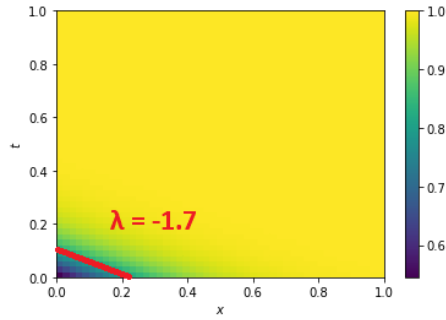


FIGURE 2.6. Solution 3

On en conclut que la solution 2 est la plus acceptable.

3. BILAN ET ÉVALUATION DE LA MÉTHODE

3.1. Intérêt de la méthode.

3.1.1. Principe de généralisation.

Le principale intérêt des réseau de neurones réside dans leur capacité de généralisation. C'est à dire qu'il est possible de les entrainer uniquement sur un sous ensemble et de les utiliser sur l'ensemble entier. Cette capacité offre ici deux avantages :

- Il permet de limiter l'ensemble de points pour l'apprentissage et donc d'assurer un faible temps de calcul.
- Le résultat de la méthode de résolution n'est pas seulement un graphe de la solution mais véritablement une fonction dont on connaît l'expression littérale. Le résultat est donc particulièrement simple à utiliser.

Exemple de généralisation :

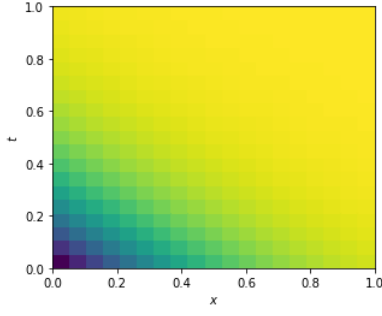


FIGURE 3.1. Entrainement sur un quadrillage de $[0, 1]^2$ avec 20*20 points

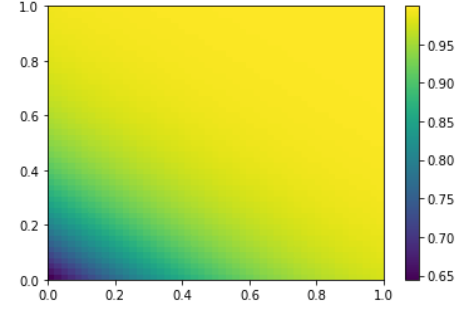


FIGURE 3.2. Résultat sur un quadrillage de $[0, 1]^2$ avec 50*50 points

On observe qu'en entrainant le réseau de neurones sur des sous-ensembles plus petits on retrouve la forme de la solution mais la frontière de discontinuité est particulièrement floue. Ce défaut était prévisible car le théorème de König assure seulement la capacité d'approximation des réseaux de neurones pour des fonctions continues.

3.1.2. Complexité temporelle.

On note :

- N le nombre de périodes d'apprentissage
- k la somme du nombre de poids et du nombre de biais
- i le cardinal de l'ensemble I contenant les points utilisés pour une période d'apprentissage

Theorem 10. *La complexité temporelle asymptotique de l'algorithme est en $O(Nki)$*

Proof.

La partie couteuse de l'algorithme est le calcul des dérivées partielles par rapport à chaque biais et poids, pour chaque point de l'ensemble d'apprentissage I à chaque période d'apprentissage. La complexité d'une dérivée partielle se fait en temps constant car on connaît son expression littérale. D'où une complexité asymptotique en $O(Nki)$. \square

De plus, un avantage des réseaux de neurones est la facilité à mettre en place une parallélisation des calculs. En effet, les calculs pour chaque point de l'ensemble I d'apprentissage sont indépendants les uns des autres. On peut donc diviser I en sous-ensembles, chacun traité par un processeur différent.

Algorithmic 2 pseudocode après parallélisation

```

1: Mise en place du réseau
2: Initialisation des poids et biais
3:  $N$  = nombre de périodes d'apprentissage
4: Définition de la fonction coût
5:  $H$  = nombre de processeurs utilisés
6: for  $j = 1$  to  $N$  do
7:   for  $h = 1$  to  $H$  do
8:      $I_h$  = Ensemble des points d'apprentissage pour le processeur  $h$ 
9:   end for
10:   $\varphi = 0$ 
11:  for  $h = 1$  to  $H$  do {en parallèle}
12:    for  $i$  in  $I_h$  do
13:       $\varphi += \text{coût}(i)$ 
14:    end for
15:    Calcul des dérivées partielles de  $\varphi$ 
16:    Descente du gradient
17:  end for
18: end for

```

En notant H le nombre de processeurs utilisés.
 La nouvelle complexité temporelle est en $O(\frac{Nik}{H})$.

3.2. Limite de la méthode.

3.2.1. Dépendance aux hyper-paramètres.

On appelle hyper-paramètre les différents paramètres de notre réseau de neurones dont il est difficile de prévoir leur valeur optimal. Il est ainsi nécessaire de procéder à un recalibrage des hyper-paramètres à chaque fois que l'on change d'équation. Ainsi l'aléatoire de la réussite des apprentissages dépend aussi de ces hyper-paramètres.

Exemples d'hyper-paramètres : n nombre de couches du réseau, k nombre de neurones par couche, N nombre de périodes d'apprentissage, v vitesse d'apprentissage, initialisation des poids etc...

CONCLUSION

La méthode de résolution a été testée avec succès. Elle possède un faible temps de calcul et renvoie des résultats particulièrement simples à utiliser mais l'étalonnage laborieux de nombreux hyper-paramètres fait que cette méthode est peu commode à appliquer en pratique. Cependant les réseaux de neurones constituant un champ de recherche vaste et très actif de nombreuses améliorations de la méthode sont possibles.

Bibliographie

- K. Hornik : Multilayer Feedforward Networks are Universal Approximators : Neural network Volume 2, Pages 359-366 (1989) ; 10.1016/0893-6080(89)90020-8
- I. E. Lagaris, A. Likas, D. I. Fotiadis : Artificial Neural Networks for Solving Ordinary and Partial Differential Equations. Université d'Ioannina (1997) ; 10.1109/72.712178
- W. Jin : Traffic Flow Models and Their Numerical Solutions : <https://arxiv.org/abs/math/0406214v1> (1998)
- G. Newell : Nonlinear effects in the dynamics of car following : Operations Research Volume 9 Pages 209-229 (1961) ; 10.1287/opre.9.2.209.
- M.L.Delle Monache : Traffic flow modeling by conservation laws. Université Nice Sophia Antipolis (2014) ; NNT : 2014NICE4056