ÉCOLE CENTRALE LYON

PE-01 Apprentissage automatique pour la discrétisation des lois de conservation

Rapport de synthèse lot 1

Auteurs :
M. Vicente CITTOLIN
M. Sebastián FLORES
M. Jonathan GUÉRIN
Mme. Clémence MARTIN
M. Abderrahmane RACHADI
M. Hugo TOSELLO

Tuteurs :
M. Alexis GIAUQUE
M. Cristophe CORRE

Version du
14 décembre 2020

1 Description des tâches réalisées

Tout d'abord, on a transcrit le code SciLab pour la discrétisation de l'équation de diffusion en 1D dans le langage Python. Étant une tâche initiale de prise en main, aucun problème notable n'a été rencontré.

L'objectif de ce lot est de réaliser un réseau de neurones capable de donner un pas de temps dt maximal pour une paire de paramètres α et dx donnée. De ce fait, on s'est mis à créer (i) une base de données de simulation du problème et (ii) un algorithme de création et entraînement d'un réseau de neurones.

Pour la création de la base de données, on a écrit un algorithme Python pour la générer, et on la sauvegarde en format SQL (extension *.db*). Le premier problème rencontré est le temps de calcul pris par l'algorithme, ce qui demande de diviser l'exécution du programme en deux ordinateurs et la tâche postérieur de consolider les deux bases générées dans un seul fichier *.db*.

Pour le réseau de neurones, et après avoir résolu les problèmes d'installation du module TensorFlow sur les ordinateurs de toute l'équipe, le codage de l'algorithme d'entraînement est réalisé sans problèmes majeurs. La fonction d'activation est d'abord ReLU et on expérimente avec différentes nombres de couches et de neurones par couche. A priori, les résultats d'entraînement sont satisfaisants, c'est-à-dire, on obtient des paramètres de correlation et de R^2 bien proches de 1.

Suite à la rencontre de plusieurs valeurs *NaN* dans la base de données, on a trouvé des fautes dans l'algorithme qui la génère. Après la correction de ces erreurs, une nouvelle base de données est créée, cette fois-ci, avec plus de points. Le problème du temps de calcul est accentué, prenant plusieurs jours pour finir une partie.

Avant de l'entraînement du réseau, on ajoute un filtrage des données : on enlève les points qui ne sont pas stables et on prend les points avec la plus grande valeur de dt. Ce processus-ci laisse la base de données d'une taille considérablement plus petite que à l'origine (environ 90% des données sont enlevées).

Pour diminuer le temps de calcul, le code de génération de base de données est traduit en C++, ce qui en effet rend l'exécution considérablement plus rapide, générant le même volume de données en quelques minutes.

Avec la nouvelle base de données, le réseau est entraîné de nouveau, et un script est créé pour tester la performance par rapport aux attentes théoriques. Ce test est fait par la génération de valeurs aléatoires de α et dx, puis leur insertion au réseau comme *input*, et après la comparaison du *output* avec la relation théorique $dt \leq \frac{dx^2}{\alpha}$.

Une difficulté rencontrée à ce stade est la question de la normalisation des données d'entrée. Comme le réseau est entraîné avec une base de données normalisée, c'est-à-dire, de moyenne $\mu = 0$ et écart-type $\sigma = 1$, les données d'entrée doivent s'adapter à la façon dont le réseau est entraîné. La question principale est si les données d'entrée doivent être elles mêmes normalisés, ou si elles doivent être transformées par la moyenne et l'écart-type de la base de données d'entraînement avant normalisation. La décision a favorisé la deuxième option.

Postérieurement à la comparaison des résultats du réseau et ceux de la théorie, on a remarqué que plusieurs prédictions du réseau sont au-dessus de la courbe théorique, et donc sont instables. Ce problème est dû principalement à deux faits principales : d'un côté, il y a des données dans la base qui sont marquées "stables" tandis qu'elles ne vérifient pas la relation théorique, et d'autre côté, il n'y a aucun moyen de pénaliser le réseau s'il donne des prédictions instables.

Pour régler le premier problème, on a appliqué un deuxième filtrage des données en utilisant la relation théorique, ce qui n'est pas idéal si on prend en compte qu'avoir une relation aussi explicite pour évaluer la pertinence d'un point de la base est assez rare (même pour l'advection 2D cela ne sera pas le cas). On présentera les résultats avec ces deux filtrages différents. Pour le deuxième problème, on n'a pas trouvé une solution.

2 Présentation des résultats

Description de la base de données

La base de données générée contient 7500 points, pour lesquels on stocke la valeur de α, de dx, de dt, si le schéma est stable avec ces valeurs, le nombre d'itérations réalisées N_{iter}, et l'erreur en moyenne quadratique par rapport à la solution exacte. Pour toutes ces données, on fixe un temps maximal de $dt \cdot N_{iter} = 500$.

α	dx	dt	stable	N_{iter}	erreur
$2,35 \cdot 10^{-4}$	0,010582	0,233290	1	2143	$8,19 \cdot 10^{-4}$
$2,35 \cdot 10^{-4}$	0,010582	0,242621	0	2060	$4,87 \cdot 10^{26}$
$2,35 \cdot 10^{-4}$	0,010582	0,251953	0	1984	$1,99 \cdot 10^{88}$
$2,35 \cdot 10^{-4}$	0,010582	0,261284	0	1913	$2,40 \cdot 10^{141}$
$2,35 \cdot 10^{-4}$	0,010582	0,270616	0	1847	$1,00 \cdot 10^{7}$

TABLE 1 – Tail de la base de données avant filtrage

On voit dans le tableau 1 que pour chaque paire (α, dx) on obtient plusieurs valeurs de dt.

Dans la figure 1 on trace la valeur de dt en fonction de la valeur théorique de dt_{max}, $\frac{dx^2}{\alpha}$. La ligne rouge représente la fonction identité.

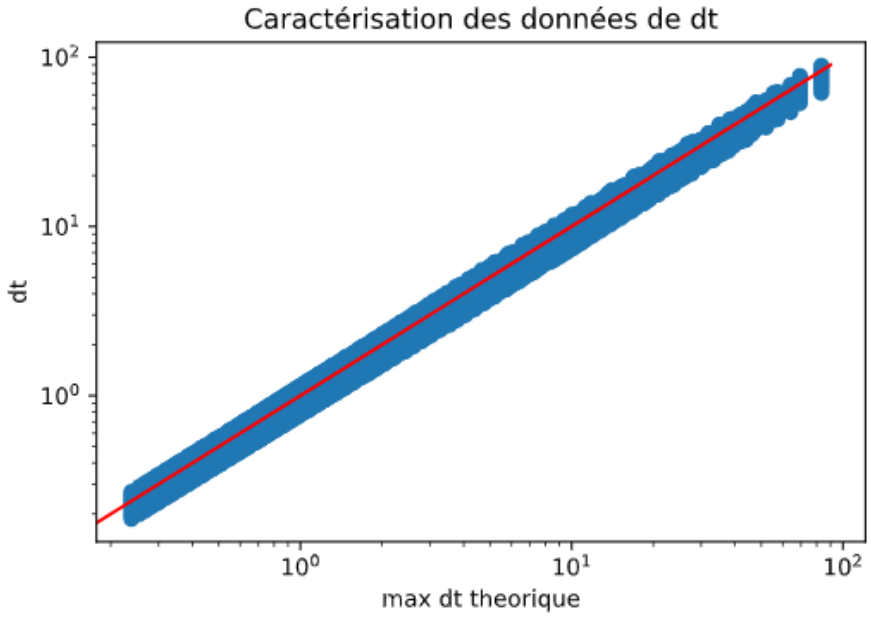

FIGURE 1 – Graphique résumant la base de données avant filtrage en échelle log-log.

Pour la base de données filtrée, on obtient, dans les deux cas (filtrage avec et sans la relation théorique), une base de taille 750. Dans la figure 2 on trace la même relation que dans la figure 1, mais avec les deux bases filtrés. On peut bien voir que dans la base avec filtrage sans la relation théorique il existe un

dépassement de la ligne de l'identité croissant à droite, ce qui veut dire que les données sont nommés stables par la base mais ne vérifient pas la relation $dt \leq \frac{dx^2}{\alpha}$.

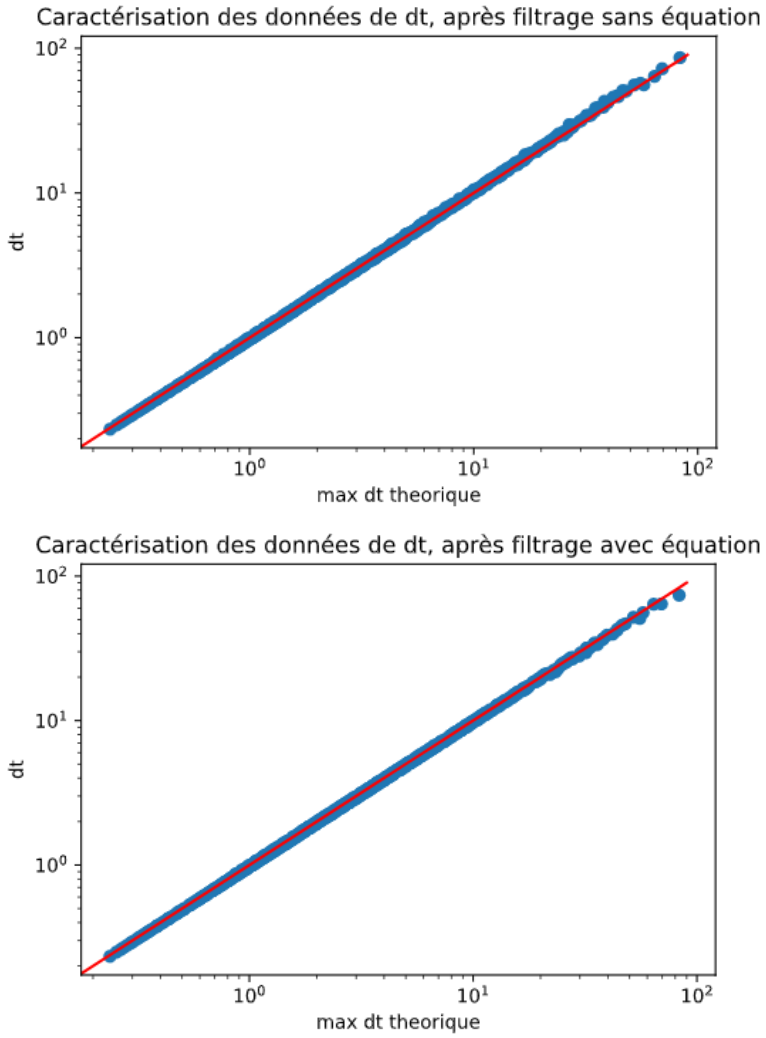

FIGURE 2 – Graphique résumant la base après filtrage sans équation (en haut), et avec l'équation (en bas).

Dans la table 2, on peut voir les différences entre les paramètres des bases de données après filtrage. On note que dans le cas du filtrage avec équation, la moyenne et la valeur maximale de dt sont plus basses.

Sans équation	α	dx	dt
Moyenne	$7,37 \cdot 10^{-5}$	$1,98 \cdot 10^{-2}$	$6,03$
Ecart-type	$5,61 \cdot 10^{-5}$	$8,71 \cdot 10^{-3}$	$9,26$
Maximum	$2,35 \cdot 10^{-4}$	$4,08 \cdot 10^{-2}$	$8,61 \cdot 10$
Minimum	$1,00 \cdot 10^{-5}$	$1,06 \cdot 10^{-2}$	$2,33 \cdot 10^{-1}$
Avec équation	α	dx	dt
Moyenne	$7,37 \cdot 10^{-5}$	$1,98 \cdot 10^{-2}$	$5,74$
Ecart-type	$5,61 \cdot 10^{-5}$	$8,71 \cdot 10^{-3}$	$9,26$
Maximum	$2,35 \cdot 10^{-4}$	$4,08 \cdot 10^{-2}$	$7,38 \cdot 10$
Minimum	$1,00 \cdot 10^{-5}$	$1,06 \cdot 10^{-2}$	$2,33 \cdot 10^{-1}$

TABLE 2 – Tableaux avec un résumé des deux bases de données après filtrage

Type de couche	Output shape	N° paramètres
Dense	(None, 5)	15
LeakyReLU	(None, 5)	0
Dense	(None, 20)	120
LeakyReLU	(None, 20)	0
Dense	(None, 20)	420
LeakyReLU	(None, 20)	0
Dense	(None, 1)	21

TABLE 3 – Structure du réseau de neurones

Résultats du réseau entraîné

Pour chacune des bases, nous avons entraîné le réseau avec les mêmes caractéristiques initiales, affichées dans la table 3. Le nombre de degrés de liberté est de 576.

Après environ 800 époques, l'entraînement atteint un état stationnaire (tolérance de 10^{-17}, patience de 10 époques), et les résultats en terme de correlation sont présentés dans la figure 4. Pour le cas du filtre basé en la simulation (sans équation), la corrélation est de $0,9978$ et le coefficient $R2$ de $0,9937$, tandis que pour le filtre basé en l'équation la corrélation et le $R2$ sont de $0,9989$ et de $0,9975$. On voit donc que les différences en termes d'entraînement sont négligeables.

Dans la figure **??** on montre les résultats de prédiction du réseau pour un ensemble aléatoire de valeurs de α et dx, dans les deux cas d'entraînement.

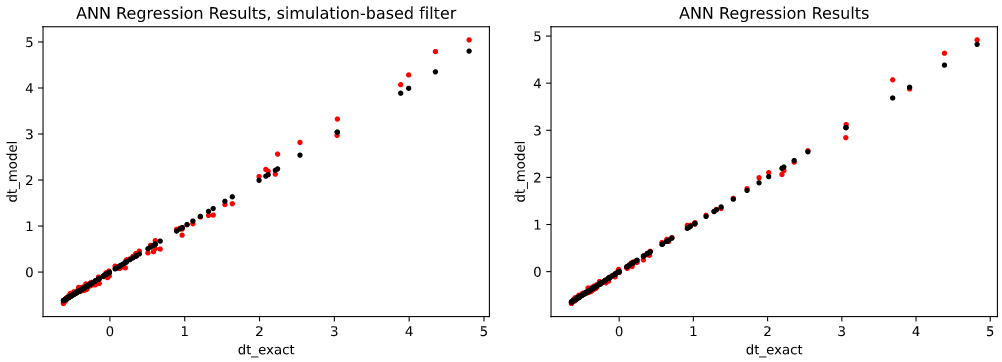

FIGURE 3 – Résultats de correlation dans le cas sans équation (gauche), et dans le cas avec équation (droite).

On peut voir que pour les deux cas il existe des zones où le réseau donne des résultats au-dessus de la valeur théorique, ce qui n'est pas souhaitable, car ces valeurs ne donnent pas des schémas de discrétisation stables.

3 Conclusion

Dans le développement on a appris à approcher un problème d'apprentissage automatique lié à la discrétisation d'une EDP. Même si les résultats n'ont pas été idéales, on a pu conclure des points importants pour la suite.

D'un côté, on a vu que la maîtrise de la base données est un problème primordial pour un entraînement pertinent et précis. Cela veut dire qu'il faut (i) que le code qui la génère soit claire et pertinent, (ii) vérifier que les données générées sont correctes par rapport au problème en question, et pour cela caractériser statistiquement la base, et (iii) que la base ait une taille suffisante pour obtenir des résultats précis. Une faute dans cet aspect peut entraîner des difficultés en étapes postérieures ou des résultats erronés .

D'autre côté, il faut vérifier que l'entraînement du réseau soit ajusté aux contraintes du problème. On parle ici, par exemple et dans ce contexte-ci, de bornes sur les prédictions. Bien que la solution technique à ce problème reste comme tâche à réaliser, on a vu l'importance de cet aspect dans les résultats du réseau entraîné, car bien que les données de la base étaient toutes stables, le réseau résultant donne quand même des résultats instables. De ce fait, on peut conclure que ce type de conditions doivent être impliquées dans l'entraînement de quelque manière.

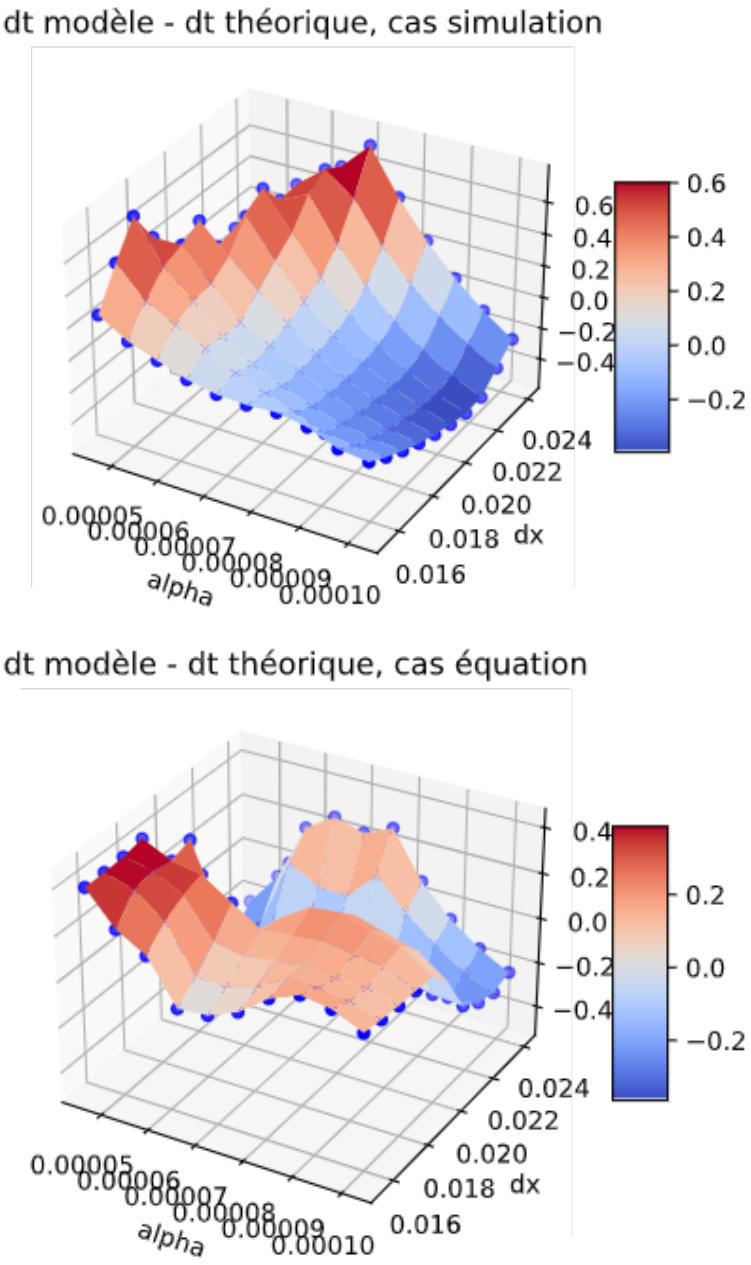

FIGURE 4 – Surfaces de comparaison des prédictions du réseau contre le résultat théorique.