L’utilisation de machine learning dans mécanique quantique avec réseaux de neurones

Étudiant : Kanlai Peng

Encadrants : Jérôme Margueron et Hubert Hansen

31/05/2021

Résumé

Dans ce stage, nous étudions dans un premier temps la capacité des réseaux de neurones artificiels à approximer des fonctions continues à support compact. Nous décrivons ensuite un programme original de minimisation de l'énergie d'une particule dans un puits de potentiel reposant sur cette capacité d'approximation qu'ont les réseaux et essayons de trouver l'état fondamental de ce système.

Introduction

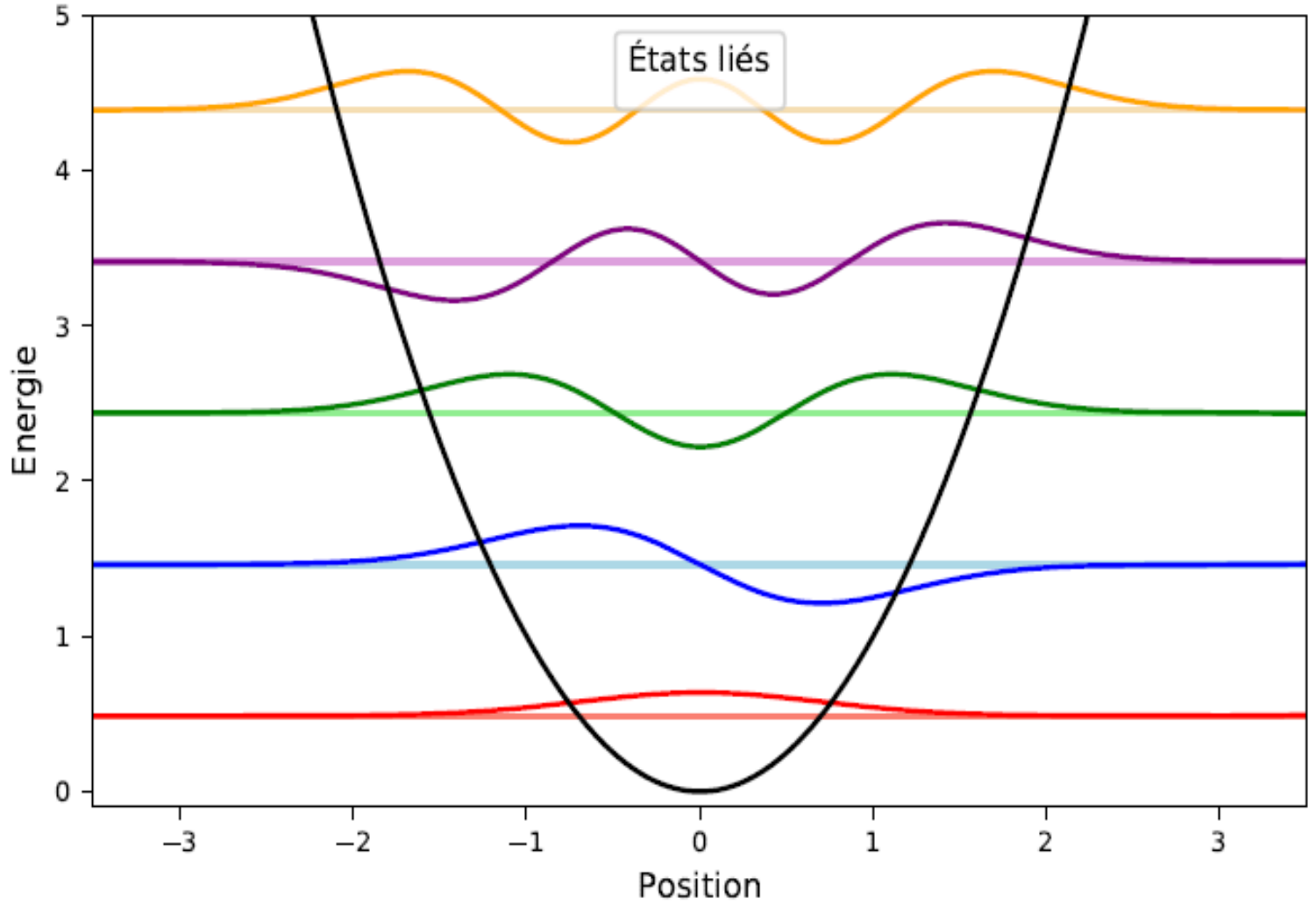
Le calcul des états et des énergies propres pour un potentiel quelconque est un problème général en mécanique quantique qui s'étend de la physique atomique à l'étude des quarks. On trouve ces états et leurs énergies par la résolution de l'équation de Schrödinger indépendante du temps mais l'absence de solution analytique pour la plupart des potentiels nous contraint à recourir à des solveurs numériques.

L’équation de Schrödinger indépendante du temps:

$$\left[\frac{-\hbar^2}{2m}\frac{d^2}{dx^2} + V(x)\right]\psi(x) = E\psi(x)$$

Après la résolution, on trouve:

$$E = \frac{\frac{-\hbar^2}{2m}\left([\psi\psi']_{-\infty}^{+\infty} - \int_{-\infty}^{+\infty}|\psi'|^2 dx\right) + \int_{-\infty}^{+\infty} \mathrm{V(x)}|\psi|^2 dx}{\int_{-\infty}^{+\infty}|\psi|^2 dx}$$



États liés d'un oscillateur harmonique. Les fonctions d'onde (non normalisées) sont alignées sur leurs énergies.

Introduction au concept des réseaux de neurones

Un réseau de neurones artificiels, ou réseau neuronal artificiel, est un système dont la conception est à l'origine schématiquement inspirée du fonctionnement des neurones biologiques, et qui par la suite s'est rapproché des méthodes statistiques. Les neurones artificiels composant le réseau sont tous dotés d’une fonction d’activation qui détermine l’activation ou la non activation des neurones pour une entrée donnée. Les réseaux de neurones sont généralement optimisés par des méthodes d’apprentissage de type probabiliste.

Selon le théorème d’approximation universelle, un réseau de neurone peut reproduire n’importe quelle fonction à valeurs réelles et à support compact. Nous essayons ici d’illustrer ce théorème en approximant l’état fondamental d’un oscillateur harmonique.

Reproduction de l'état fondamental d'un oscillateur harmonique

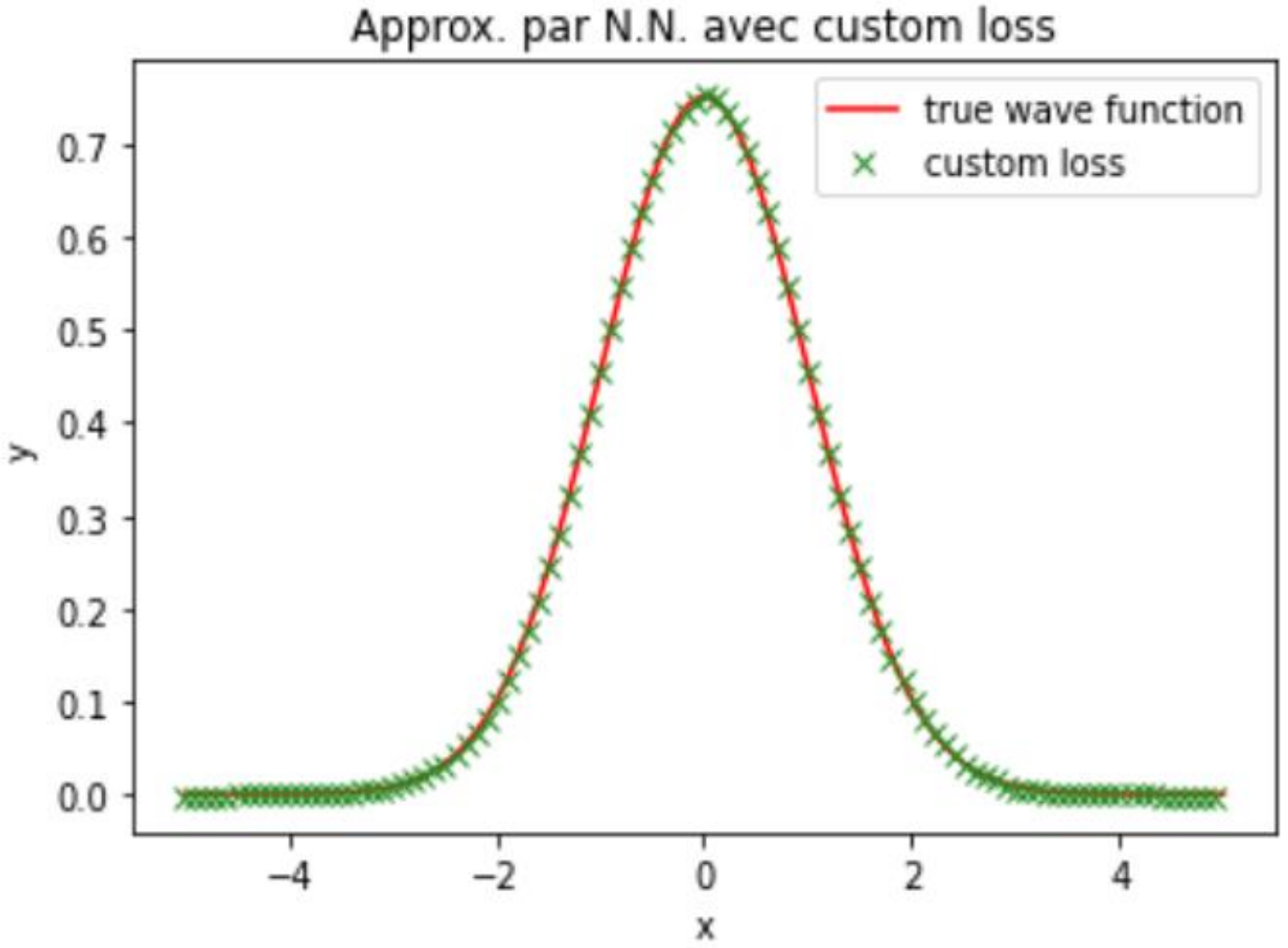
Une liste x discrétisée de −5 à 5 sur 1001 points

$$\psi_{EF}(x) = \left(\frac{m\omega}{\pi}\right)^{\frac{1}{4}} e^{\frac{-m\omega x^2}{2}}$$

La vraie fonction d'onde

Une approximation de la fonction d'onde par machine learning

Custom Loss



Approximation par réseau de neurones

Calcul de l'énergie de la reproduction

On crée une fonction 'energy compute' avec les paramètres : abscisse, fonction et le potentiel

Oscillateur harmonique à une dimension, donc le potentiel $V(x) = \frac{1}{2} m\omega x^2$ et la fonction d'onde de l'état fondamental

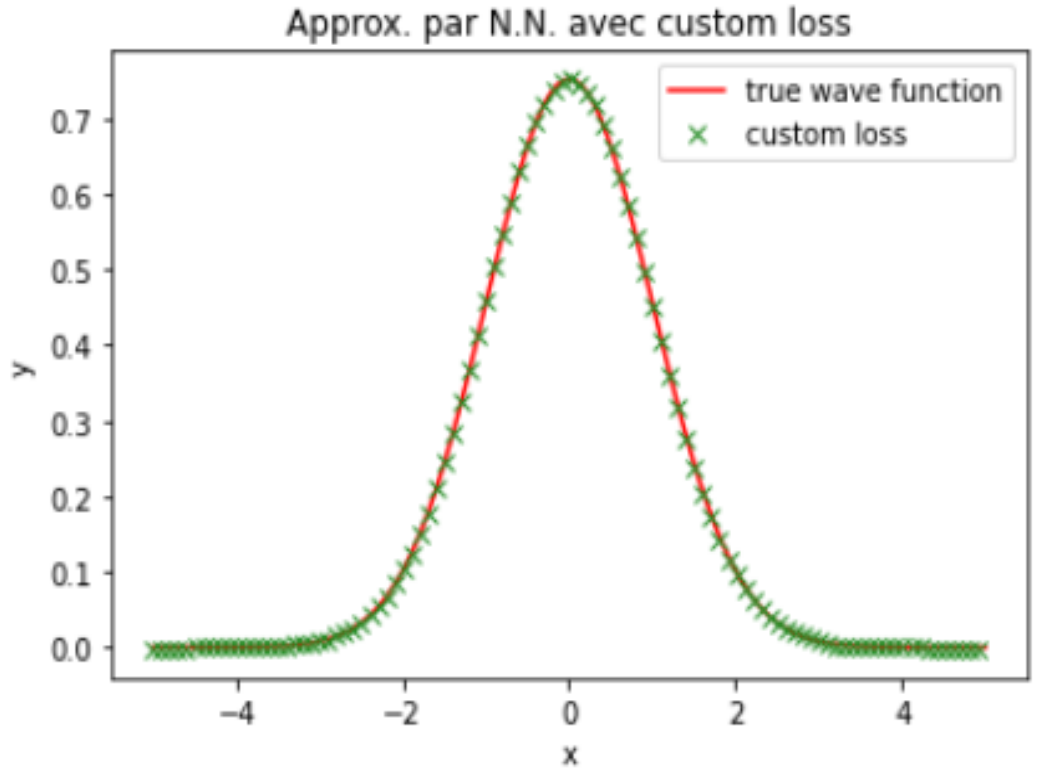
$$\psi_0(x) = \left(\frac{m\omega}{\pi}\right)^{\frac{1}{4}} e^{-\frac{m\omega x^2}{2}}$$

$$E = \frac{\frac{-\hbar^2}{2m}\left([\psi\psi']_a^b - \int_a^b |\psi'|^2 dx\right) + \int_a^b \mathrm{V(x)}|\psi|^2 dx}{\int_a^b |\psi|^2 dx}$$

comme valeur de retour de la fonction

Encore une fois, on fait une approximation de la fonction d'onde par machine learning et cette fois-ci on peut aussi calculer l'énergie associée à la fonction de prédiction approximée par le réseau de neurones et la comparer avec le résultat qu'on a trouvé précédemment.

```
Energy_wave =  0.49999999979537146
Energy_pred =  0.500424736694682
```



Calcul de l'énergie

L’évolution de l’énergie de prédiction au cours du entraînement et le taux de convergence

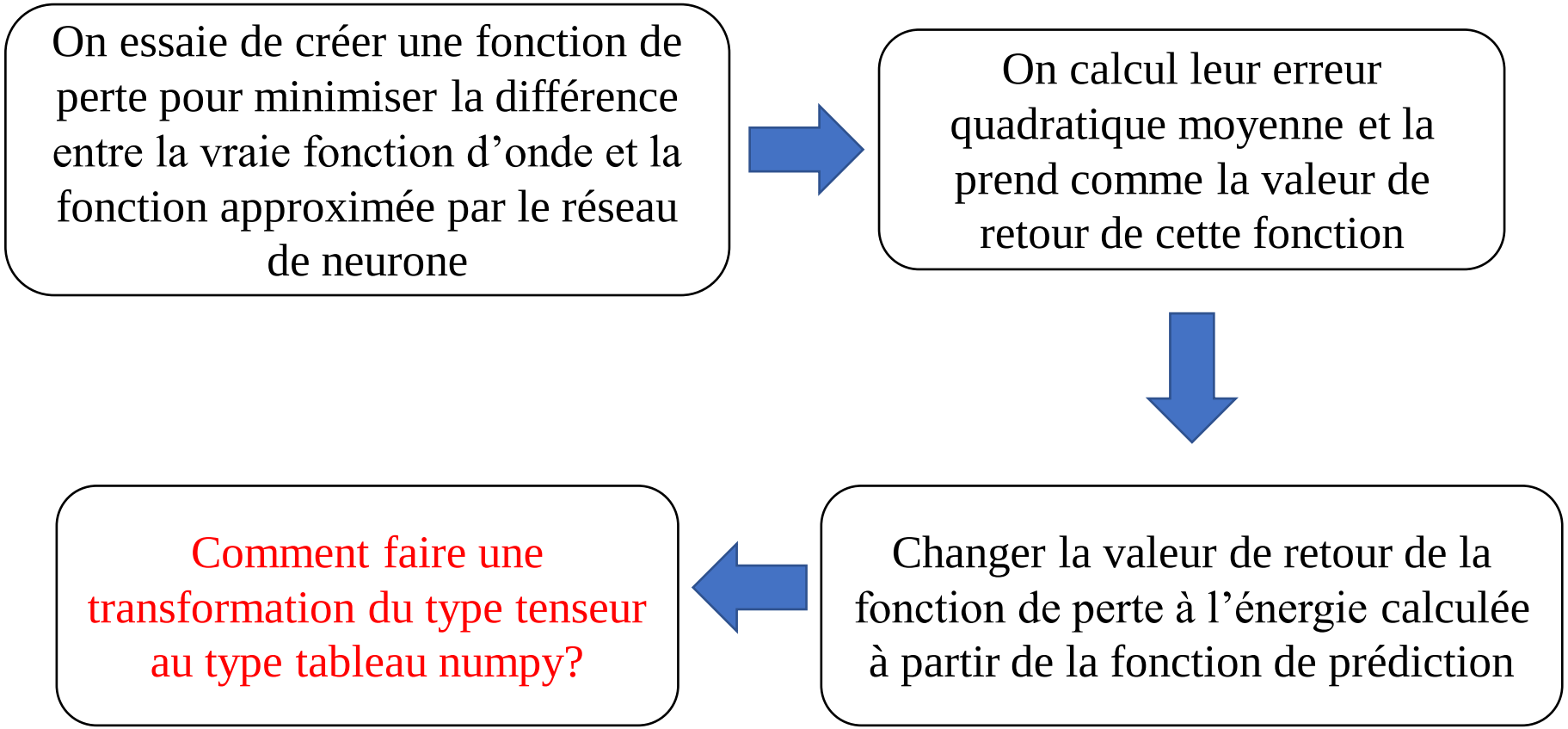
```
20/20 [==============================] - 0s 2ms/step - loss: 0.0978
0 Energy_pred =  1.944539493578198 , convergence rate =  1.944539493578198
20/20 [==============================] - 0s 2ms/step - loss: 0.0249
1 Energy_pred =  1.3760642303936363 , convergence rate =  -0.5684752631845618
20/20 [==============================] - 0s 2ms/step - loss: 0.0112
2 Energy_pred =  1.0130795823094794 , convergence rate =  -0.362984648084157
20/20 [==============================] - 0s 2ms/step - loss: 0.0085
3 Energy_pred =  0.7854222639597843 , convergence rate =  -0.22765731834969505
20/20 [==============================] - 0s 2ms/step - loss: 0.0054
4 Energy_pred =  0.7192702749954685 , convergence rate =  -0.06615198896431584
20/20 [==============================] - 0s 2ms/step - loss: 0.0034
5 Energy_pred =  0.6700134145768722 , convergence rate =  -0.04925686041859623
20/20 [==============================] - 0s 2ms/step - loss: 0.0025
6 Energy_pred =  0.7491183408044697 , convergence rate =  0.07910492622759746
20/20 [==============================] - 0s 2ms/step - loss: 0.0019
7 Energy_pred =  0.6355317893232151 , convergence rate =  -0.11358655148125463
20/20 [==============================] - 0s 2ms/step - loss: 8.7510e-04
8 Energy_pred =  0.5675872755545127 , convergence rate =  -0.06794451376870236
20/20 [==============================] - 0s 2ms/step - loss: 7.9317e-04
9 Energy_pred =  0.6192276642313073 , convergence rate =  0.051640388676794546

Energy_wave =  0.49999999979537146
```

Minimisation de l’énergie pour trouver l’état fondamental

Jusqu’ici nous avons étudié la qualité d’approximateur universel des réseaux de neurones mais notre problème n’est pas résolu. Notre but maintenant est de développer une méthode pour que le programme trouve l’état fondamental sans le connaitre au préalable.

Dans cette partie, nous supposons que nous ne connaissons pas la fonction d'onde propre et faisons comme si le potentiel était n'importe quel potentiel, c’est-à-dire le système pourrait être représenté par n'importe quel potentiel. Et c'est un test de la méthode sur un résultat connu.

Création d'une fonction de perte (loss function)



Conclusion

Le programme utilisant les réseaux de neurones permet de reproduire l’état fondamental d’un oscillateur harmonique et calculer l’énergie de reproduction, mais cette méthode n'est pour l'instant pas du tout efficace. Dans le cas de problème beaucoup plus compliqué, on espère qu'elle deviendra efficace et c’est pourquoi nous avons choisi les réseaux de neuronaux comme le sujet.

Des problèmes & questions .....

- Comment faire une transformation correcte de tenseur au tableau numpy?
- Y a-t-il une autre solution pour réaliser la minimisation de l'énergie, par exemple changer les arguments de la fonction 'energy compute' aux tenseurs et faire les calculs directement en tenseurs?
- Quelles sont des performances des réseaux de neurones sur des temps plus longs?
- Quelles sont les performances du programme à deux et trois dimensions ?

Merci pour votre attention!