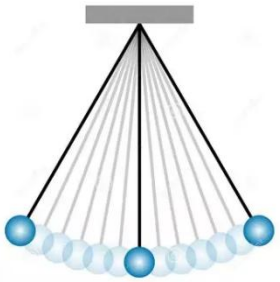
Intelligence Artificielle et Analyse de données

TP1 : Prédiction de trajectoires (réseaux récurrents)



On s'intéresse à la dynamique d'un pendule :

connaissant la valeur de l'angle θ aux instants $t_0, t_1, \dots, t_H$, on cherche à prédire la valeur aux instants $t_{H+1}, t_{H+2}, \dots, t_f$.

Pour cela, 3 *datasets* (*train, valid, test*) sont fournis.

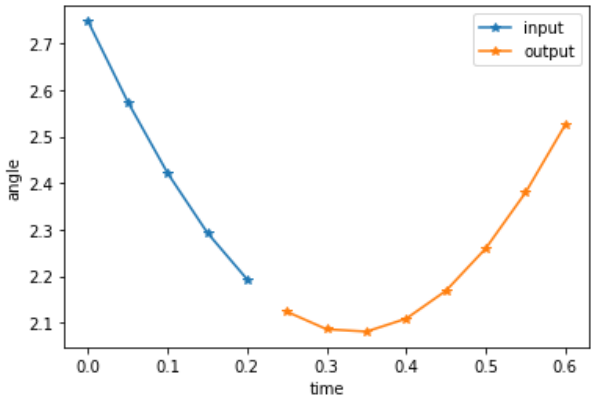
Une donnée d'un *dataset* est constituée de $(t_{in}, \theta_{in}, t_{out}, \theta_{out})$ où

- $t_{in} = t_0, t_1, \dots, t_H$ et $t_{out} = t_{H+1}, t_{H+2}, \dots, t_f$ sont des vecteurs de temps (pas nécessaires à l'entraînement, car le temps d'échantillonage est constant),
- θ_{in} un vecteur d'angle aux instants t_{in} (données d'entrée) et
- θ_{out} un vecteur d'angle aux instants t_{out} (données de sortie, à prédire).

1- <u>Comprendre les données :</u>

Tracer quelques trajectoires du dataset *train*, sur le même graphique.

Voici un exemple d'une trajectoire.



2- <u>Entraîner un réseau RNN (Elman) :</u>

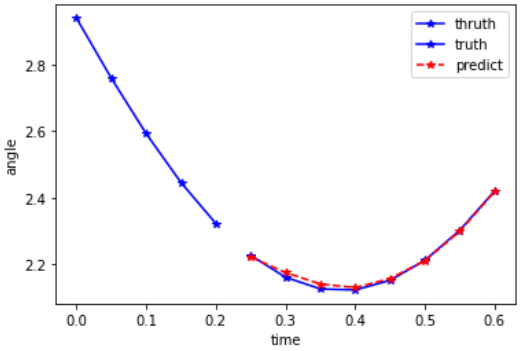
Implémenter la boucle d'entraînement (*epochs*).

Mémoriser les valeurs de coût dans les tableaux *train_losses* et *valid_losses*. En fin d'entraînement, tracer leur évolution, et analyser (stabilité, sur-apprentissage, …).

Modifier les hyperparamètres pour améliorer les performances.

3- <u>Exploitation du modèle :</u>

Tracer une courbe prédite (utiliser le dataset *test*) par le réseau RNN et comparer avec la donnée.



4- <u>Prédictions avec un MLP :</u>

Le MLP servira de *baseline* pour mieux évaluer les performances du RNN.

Conseil : Utiliser la fonction flatten (*x_flat = x.flatten(start_dim=1)*) pour adapter les données séquentielles en vecteurs (dans la fonction *forward* du MLP).