Projet Metropolis - IDL

Ali Janati Idrissi

April 2021

Contents

1 Présentation du sujet

Dans ce sujet, est considéré un modèle de spin bidimensionel : le **modèle XY**. L'état stationnaire de ce système est simulé par une méthode de Monte-Carlo, via l'**algorithme de Metropolis** sur Matlab. Cet algorithme permet de calculer la charge locale : *Cm*, et la magnétisation locale *Mm*.

Les spins sont en interactions avec leurs plus proches voisins et tendent à minimiser leur énergie. Dans l'état stationnaire, on voit apparaître des **défauts topologique** de valeur -1 ou +1.

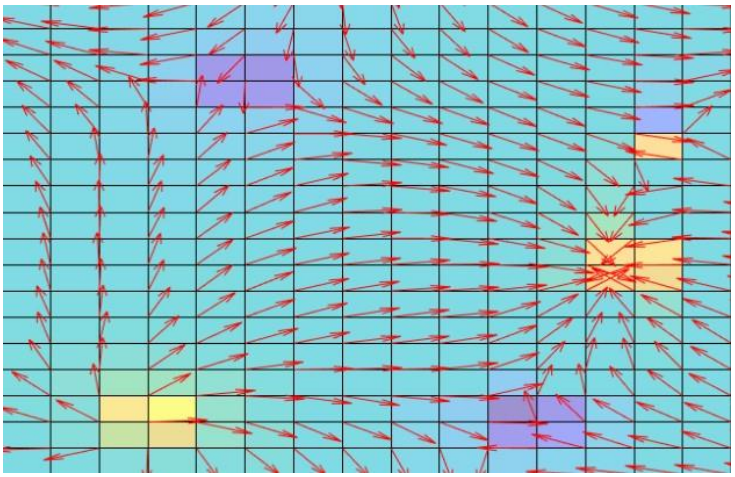

Figure 1: Exemple de **défauts topologique** : **-1** en jaune, **+1** en bleu. On a superposé les spins (flèche) et l'image *Cm*(1-Mm)*, qui met en évidence les défauts topologique.

Une propriété intéressante de l'état stationnaire de ce système, est que la somme des défauts topologiques (ou la somme des *Cm* comme elle est calculée dans l'algorithme de Metropolis) est **toujours nulle**, que les conditions limites soient périodiques ou non.

Nous avons en effet été amené à vérifier cette propriété dans le cas de conditions limites non périodiques (figure 2), qui s'est averé toujours valide, bien que nous n'ayons pas de démonstration à proposer.

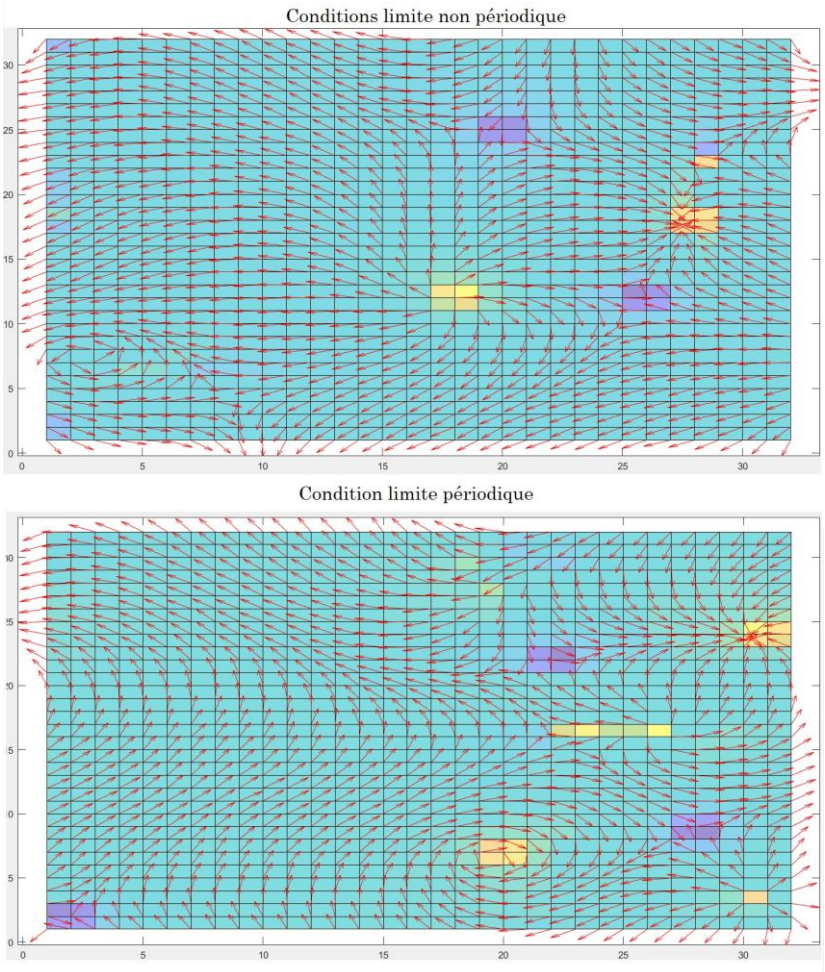

Figure 2: Etat stationnaire dans le cas de CL périodiques et de CL non périodiques. La somme des *Cm* est toujours nulle. Le code matlab XY_metropolis_CL permet d'avoir des CL non périodiques.

1.1 Problématique

L'objectif de ce projet est de voir s'il est possible d'utiliser un réseau de neurones pour classer les images selon que la charge topologique totale est nulle ou non.

Pour ce faire deux types d'images ont été générées:

1. Des images de dimension *32,32* aux conditions limites non périodiques. Sur ces images, **la somme des charges locales sur toute l'image est**

toujours nulle. Voir code *XY bdd generation.mtl*.

2. Des images de dimension *64,64* aux conditions limites non périodiques, coupées en 4. Parmi ces sous-images de dimension *32,32*, sont gardées celles dont **la somme des charges locales est non nulle**.
   Voir code *XY bdd generation_crop.mtl*.

Une image est un fichier *png* dont chaque pixel contient une valeur entre 0 et 1 (0 à 2π) correspondant à l'angle du spin à l'état stationnaire.

La raison pour laquelle nous avons préféré utiliser des images aux conditions limites non périodique, est qu'une image croppée (classe 2) ne peut pas avoir de condition limite périodique. Il est donc préférable que les deux classes ne soient pas séparable selon ce critère de périodicité.

Néanmoins, les images croppées ont toujours deux bords où les conditions limites ne sont pas exactement non périodique. Nous supposerons pour l'instant cet effet négligeable.

1.2 Dataset

Les deux classes sont définies dans la section précédente.

D'autres **paramètres** ont été arbitrairement choisis pour la génération de ce dataset via l'algorithme Metropolis sur Matlab :

1. La **température** $kT = 0.01$ **J**
2. Le nombre d'**itération de Monte-Carlo** permettant d'atteindre l'état stationnaire : $N_{MC} = 1000$
3. La **dimension** des images *32,32* (*numSpinsPerDim* = 2^5)
4. **Nombre d'images** : 8000 pour chaque classe, donc 16000 au total. 15000 images sont utilisées pour l'entrainement, et 1000 pour le calcul de la validité du modèle.

En pratique, le dataset ne fait pas 16000 images, mais aux alentours de 15000. C'est dû au fait que les images "croppées" ont parfois une charge topologique totale nulle, et dans ce cas nous les avons exclus de la seconde classe (sans pour autant les mettre dans la première).

2 Résultats

2.1 Classifieur linéaire

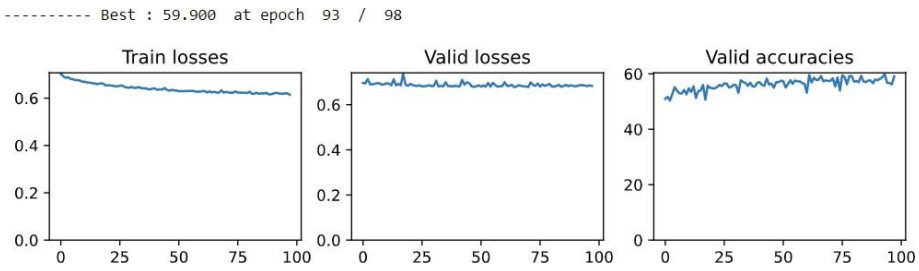

Figure 3: Classifieur linéaire. Le *batch size* est de 150 images. Calcul de la *loss* sur le set d'entrainement (à gauche), puis de validité (pour vérifier l'absence d'*overfitting*). Le troisième graphe (*valid accuracies*) donne le pourcentage de réussite sur le set de validité à chaque *epoch*.

Ce modèle simple n'apprend pas à classer les images en deux catégories : pas d'amélioration sur le set de validité. Cela signifie que l'écart entre la prédiction et le *label* de l'image n'évolue pas.

2.2 Une couche cachée

En optimisant manuellement la vitesse d'apprentissage (notée l_r dans le code), il semblerait qu'avec une **couche cachée de 50 neurones**, et suffisament de données, le modèle converge vers une **bonne classification** (75% sur la figure 4).

Néanmoins, sur les courbes de la figure 4, il semblerait qu'avec plus de données, le modèle arriverait à améliorer sa précision (absence de palier).

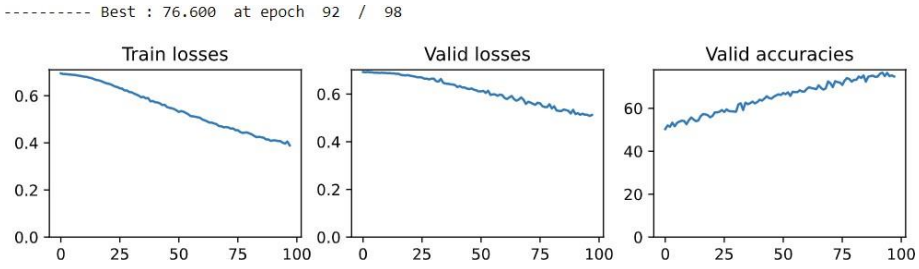

Figure 4: Réseau de neurones avec une couche cachée. Le batch size est de 150 images. La *valid loss* décroit : le modèle apprend donc bien à classer.

2.3 CNN

```
---------- Best : 90.800  at epoch  64  /  70
(0.0, 92.625)
```

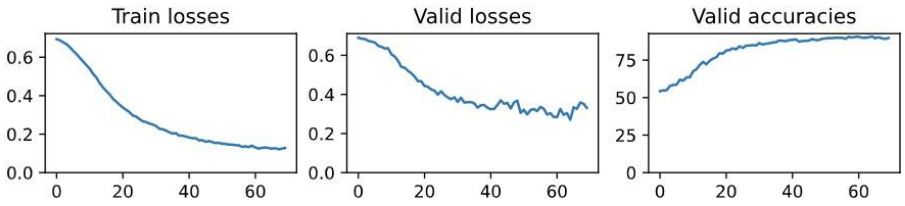

Figure 5: Réseau de convolution. Le batch size est de 200 images.

Avec ce modèle, l'apprentissage est beaucoup plus rapide. Un palier est atteint aux alentour de 90%.

La classification est donc possible et efficace.

Une optimisation poussée de l'architecture du réseau, ou du dataset en entrée permetterait peut être de gagner encore en précision. Nous n'avons pas exploré le premier point, mais nous avons essayé d'améliorer le dataset d'entrée pour voir l'impact sur l'apprentissage (dans la section suivante).

2.4 CNN : Modification du Dataset

Etant donné que les images de spin prennent pour chaque pixel des valeurs entre 0 et 1 codant l'angle, des discontinuités apparaissent lorsqu'on passe de 2π à 0 pour des spins voisins.

Pour contrer ce problème, nous avons pensé à augmenter le nombre de *channel* d'entrée du modèle CNN (une image RGB a 3 *channels* par exemple), en mettant le **cosinus** et le **sinus** de l'image (voir figure 6).

Ce changement du dataset d'entrée ne permet pas une meilleure précision du réseau (le palier à 90% n'est pas franchi).

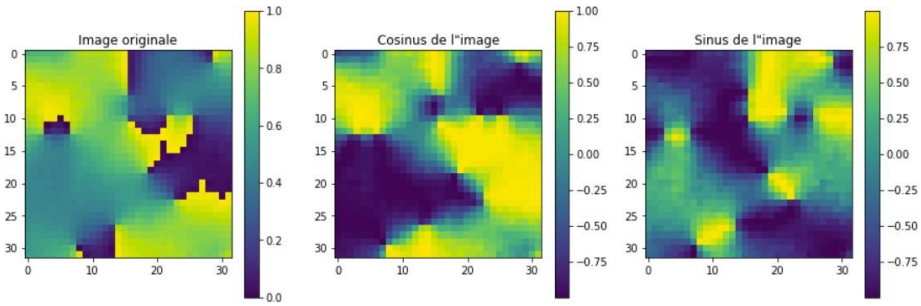

Figure 6

```
---------- Best : 88.200  at epoch  70  /  73
(0.0, 89.93)
```

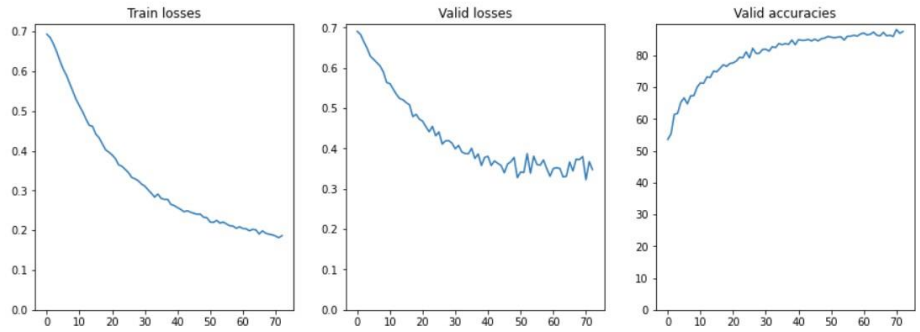

Figure 7: CNN avec deux channels. Le batch size est de 200 images.

3 Limite : un modèle dépendant de la température

(Code disponible dans l'Annexe 1 du Jupyter Notebook)

Jusque là, l'étude a été réalisé avec un dataset d'entrainement et de validité obtenus sur des images à $kT = 0.01J$.

Lorsque l'on change la température du set de validité (mais pas d'entrainement), **le modèle n'arrive plus à classer avec la même précision**. Le modèle est donc dépendant de la température.

Par ailleurs, dans la section *1.1 Problématique*, nous avons fait l'hypothèse que la classification ne se ferait pas selon des problèmes de bords. La faible précision du réseau à d'autres température mais avec les mêmes caractéristiques de bords valide cette hypothèse.

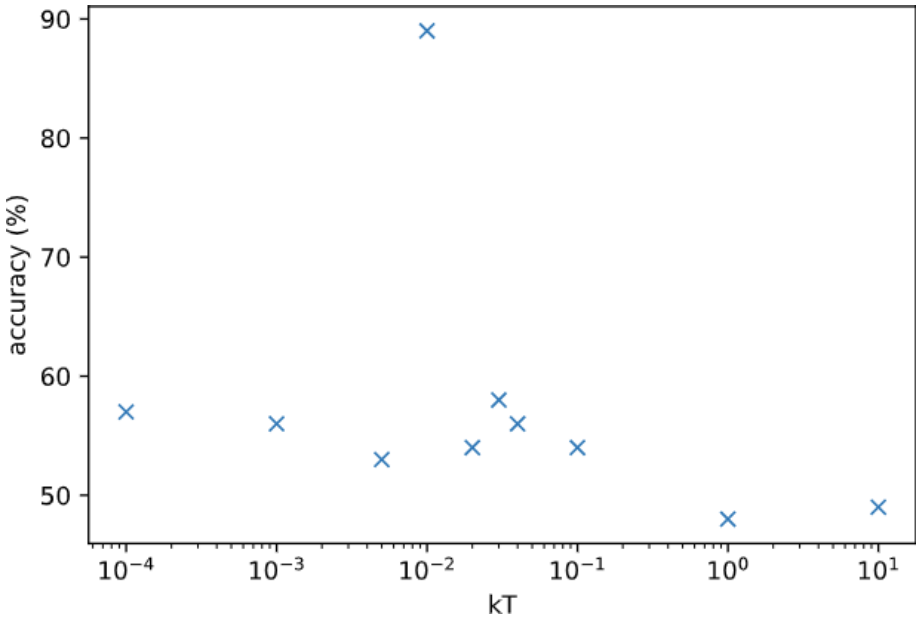

Figure 8: Précision du modèle entrainé à kT=0.01 sur des set de validité de 1000 images (500 de chaque classe) obtenus à d'autres températures. Notons le passage en dessous des 50% lorsqu'on dépasse la **température de Curie** (kTc=0.89).

4 Ouverture : vers un modèle indépendant de la température ?

La propriété générale qu'est la charge topologique totale nulle est valable quelque-soit la température. D'après la figure 8, il est compliqué d'avoir un modèle valable quelque soit la température.

Néanmoins, en dessous de la température de Curie, le système et les états stationnaires générés semblent admettre un certain niveau de ressemblance (qui se traduit par une précision supérieur à 50% sur des sets à d'autres température).

Le modèle proposé est le suivant :

1. L'architecture du CNN est la même que celle utilisée en section 2.3
2. Le dataset est différent : 2000 images (1000 de chaque classe) pour les kT suivants : 0.0001 ; 0.001 ; 0.005 ; 0.01 ; 0.02 ; 0.03 ; 0.04. Soit 14000 images au total.
3. Le dataset est ensuite mélangé aléatoirement, et 1000 images avec leur *label* sont récupérées pour former un set de validité. Le reste constitue le

dataset d'entrainement.

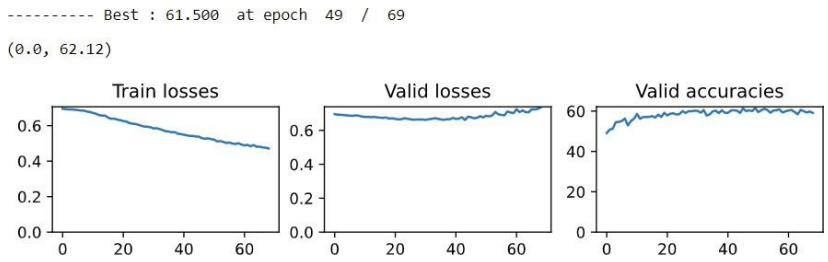

Figure 9: Résultat obtenu avec le modèle décrit ci dessus. La *valid loss* ne décroit pas, les prédictions du modèle ne s'améliorent donc pas vraiment. C'est une forme d'*overfitting*, présente malgré un dropout de 0,5.

Nous n'avons donc pas réussi à mettre en place un modèle capable de classifier indépendamment de la température de manière satisfaisante, mais nous espérons avoir pu ouvrir la voie à l'exploration d'autres pistes.