Mines Informatique MP-PC-PSI 2022 — Corrigé

Ce corrigé est proposé par Cyril Ravat (professeur en CPGE) ; il a été relu par Virgile Andreani (ENS Ulm) et Benjamin Monmege (enseignant-chercheur à l'université).

Ce sujet aborde le thème du magnétisme des matériaux et la modélisation de son aspect aléatoire. Il contient une brève introduction décrivant la différence entre les matériaux paramagnétiques et ferromagnétiques, ainsi que l'importance de la température de Curie dans le comportement du matériau. Il est constitué de 4 parties indépendantes.

- Dans la partie I, on étudie l'évolution théorique de l'aimantation en fonction de la température. Cette évolution est définie à l'aide d'une relation non linéaire qu'il faut résoudre numériquement, ce qui donne l'occasion d'appliquer l'algorithme de recherche par dichotomie.
- La partie suivante fait intervenir des bases de données économiques (fournisseurs, matériaux, prix au kilogramme). Quatre questions de difficulté graduelle permettent de montrer ses compétences.
- La partie III, plus longue, étudie le modèle microscopique de l'aimantation à partir des sens *up* et *down* des spins présents à l'intérieur du matériau. L'échantillon est modélisé par un tableau à deux dimensions, où l'on cherche comment déterminer les éléments voisins notamment au bord de l'échantillon. Une recherche d'états typiques est réalisée par la méthode aléatoire de Monte-Carlo.
- Dans la dernière partie, il est question de reconnaissance de formes au sein des images produites précédemment. On réalise en particulier un marquage des domaines magnétiques dits de Weiss par la recherche des pixels voisins identiques. Une question aborde les fonctions récursives, une autre les piles.

Cette épreuve amène correctement un sujet complexe et le traite sous plusieurs aspects intéressants. Elle balaye l'ensemble du programme, avec des questions de difficulté progressive.

Indications

Partie I

3 Il faut reprendre l'algorithme de la recherche par dichotomie vu en cours et l'appliquer à cette fonction particulière qui possède un deuxième argument, constant.

4 La longueur de l'intervalle de recherche est divisée par 2 à chaque itération. Combien d'itérations sont réalisées avant d'arriver à la bonne précision ?

5 La construction de la liste est classique, réalisée à l'aide de la méthode `append`. Attention, deux cas sont possibles en fonction de la valeur de t.

Partie II

7 Des colonnes de deux tables différentes sont nécessaires, il faut faire une jointure.

8 On doit récupérer le prix minimum et s'en servir pour sélectionner les bonnes lignes de la table obtenue par jointure. L'utilisation de l'opérateur `ORDER BY` ne donne pas le bon résultat ici.

9 On peut générer la table contenant le prix moyen de tous les matériaux, puis y sélectionner uniquement les prix moyens voulus en imbriquant la requête précédente à l'intérieur d'une nouvelle requête.

Partie III

11 Il est possible d'adapter le code de la question précédente pour modifier les valeurs une ligne sur deux.

12 L'extraction des valeurs de `s` entre les indices `a` et `b` exclu se fait par `s[a :b]`.

13 Le plus simple est de traiter le cas général en premier, puis d'ajouter une modification de chaque voisin dans le cas où le spin se trouve sur un des bords de l'échantillon.

14 L'équation 3 de l'énoncé montre la nécessité d'une double boucle, sur l'ensemble des spins puis sur les voisins de chaque spin. On doit récupérer les voisins avec la fonction définie à la question 13.

15 Il faut retourner une valeur booléenne, vraie dans deux cas disjoints.

16 Quelle est la complexité de la fonction `energie` ?

17 Le tirage au hasard du spin à modifier est à réaliser avec la fonction `randrange`.

19 Comme à la question 18, la complexité du calcul de ΔE est essentiel.

20 Quelle est la différence d'énergie ΔE lorsqu'un spin change ? Peut-on simplifier le calcul, supprimer les sommes ?

Partie IV

22 La fonction `explorer_voisinage` regarde simplement chaque pixel voisin de `i` pour éventuellement en modifier la couleur et exécuter à nouveau la fonction sur ce pixel. On ne demande pas la fonction qui crée une nouvelle couleur.

23 L'utilisation d'une pile se fait à l'aide d'une boucle itérant tant que la pile est non vide. La pile doit être initialisée avec la valeur `i`.

24 Il faut exécuter la fonction précédente pour chaque pixel non encore traité, sans oublier d'incrémenter `num` à chaque fois.

I. Transition paramagnétique/ferromagnétique sans champ magnétique extérieur

1 Pour importer uniquement quelques fonctions à partir de modules, il faut utiliser la syntaxe

```
from math import exp, tanh
from random import randrange, random
```

2 L'équation donnée doit être résolue avec la variable m et se met sous la forme

$$f(x,t) = x - \tanh\left(\frac{x}{t}\right) = 0$$

d'inconnue x.

```
def f(x, t):
    return x - tanh(x/t)
```

3 On utilise la recherche par dichotomie d'une solution à l'équation $f(x,t) = 0$ où la variable est x alors que t est un paramètre fixé. Dans cet algorithme, on conserve deux valeurs correspondant aux deux bornes de l'intervalle dans lequel se trouve la solution, initialement $[a,b]$. À chaque itération, on divise la largeur de l'intervalle par 2, car en appelant m le milieu de l'intervalle,

- soit $f(a,t)$ et $f(m,t)$ sont de signes opposés et cela signifie que la solution se trouve entre a et m, on peut donc affecter m à la variable b ;
- soit $f(a,t)$ et $f(m,t)$ sont de même signe et cela signifie que la solution ne se trouve pas entre a et m, on peut donc affecter m à la variable a.

On continue les itérations jusqu'à ce que la largeur de l'intervalle soit inférieure à $2\,\varepsilon$ pour obtenir un résultat final égal au milieu de l'intervalle. La solution réelle est nécessairement à une distance inférieure à ε de ce résultat.

```
def dicho(f, t, a, b, eps):
    while b - a > 2*eps:
        m = (a + b) / 2
        if f(a, t) * f(m, t) < 0:
            b = m
        else:
            a = m
    return (a + b) / 2
```

4 À chaque itération, la largeur de l'intervalle de recherche est divisée par 2 à l'aide d'un nombre d'opérations toujours identique et indépendant de a et b. En nommant n le nombre d'itérations, on peut dire que l'algorithme est de complexité linéaire en n. Au bout de n itérations, la largeur de l'intervalle de recherche est $(b-a)/2^n$. Il faut ainsi, pour sortir de la boucle, que

$$\frac{b-a}{2^n} \leqslant 2\,\varepsilon$$

c'est-à-dire

$$\frac{b-a}{\varepsilon} \leqslant 2^{n+1}$$

soit

$$n \geqslant \log_2\left(\frac{b-a}{\varepsilon}\right) - 1$$

donc

La complexité de la fonction `dicho` est en $\mathrm{O}\left(\log_2\left(\frac{b-a}{\varepsilon}\right)\right)$.

5 Il faut, pour chacune des 500 valeurs t uniformément réparties entre t_1 et t_2, appliquer la fonction `dicho` si $t < 1$ (le matériau est alors ferromagnétique), ou simplement définir une aimantation nulle si $t \geqslant 1$ (le matériau est alors paramagnétique).

```
def construction_liste_m(t1, t2):
    m = []
    n = 500
    pas = (t2 - t1) / (n - 1)
    for i in range(n):
        t = t1 + pas * i
        if t < 1:
            m.append( dicho(f, t, 0.001, 1, 1e-6) )
        else:
            m.append(0)
    return m
```

> La figure 1 de l'énoncé peut être obtenue avec le code
>
> ```
> t = [ 1.5*t/500 for t in range(1,501) ]
> m = construction_liste_m(t[0],t[-1])
> plt.plot(t,m)
> plt.grid()
> plt.show()
> ```

II. Recherche dans une base de données de matériaux magnétiques

6 Pour obtenir le nom des matériaux ayant une température de Curie inférieure à 500 kelvins, il faut exécuter la sélection

```
SELECT nom FROM materiaux WHERE t_curie < 500;
```

7 Les noms de fournisseurs sont dans la table `fournisseurs` alors que les prix sont dans la table `prix`. Il faut donc réaliser une jointure avant la sélection, soit la requête

```
SELECT nom_fournisseur, prix_kg * 4.5
FROM fournisseurs JOIN prix ON id_fournisseur = id_four
WHERE id_mat = 8713;
```

8 Le prix minimal peut être obtenu par l'utilisation de la fonction d'agrégation `MIN` mais pas le fournisseur correspondant, car une telle fonction n'agit que sur une seule colonne du résultat. Ce n'est donc pas la projection (après l'opérateur `SELECT`) mais la sélection (après l'opération `WHERE`) qu'il faut modifier, en y incluant une condition sur le prix à l'aide d'une sous-requête, entre parenthèses.

```
SELECT nom_fournisseur, prix_kg * 4.5
FROM fournisseurs JOIN prix ON id_fournisseur = id_four
WHERE id_mat = 8713
AND prix_kg = (SELECT MIN(prix_kg) FROM prix WHERE id_mat = 8713);
```

> Si l'on conserve la requête de la question 7 et que l'on classe les lignes à l'aide de `ORDER BY prix`, on ne peut alors pas savoir si plusieurs fournisseurs sont aussi compétitifs. Ce n'est donc pas une réponse valable.