

Proposition de corrigé

Concours : Banque PT

Année : 2024

Filière : PT

Épreuve : Informatique et Modélisation

Ceci est une proposition de corrigé des concours de CPGE, réalisée bénévolement par des enseignants de Sciences Industrielles de l'Ingénieur et d'Informatique, membres de l'[UPSTI](https://www.upsti.fr) (Union des Professeurs de Sciences et Techniques Industrielles), et publiée sur le site de l'association :

<https://www.upsti.fr/espace-etudiants/annales-de-concours>

A l'attention des étudiants

Ce document vous apportera des éléments de corrections pour le sujet traité, mais n'est ni un corrigé officiel du concours, ni un corrigé détaillé ou exhaustif de l'épreuve en question.

L'UPSTI ne répondra pas directement aux questions que peuvent soulever ces corrigés : nous vous invitons à vous rapprocher de vos enseignants si vous souhaitez des compléments d'information, et à vous adresser à eux pour nous faire remonter vos éventuelles remarques.

Licence et Copyright

Toute représentation ou reproduction (même partielle) de ce document faite sans l'accord de l'UPSTI est **interdite**. Seuls le téléchargement et la copie privée à usage personnel sont autorisés (protection au titre des [droits d'auteur](#)).

En cas de doute, n'hésitez pas à nous contacter à : corrigesconcours@upsti.fr.

Informez-vous !

Retrouvez plus d'information sur les [Sciences de l'Ingénieur](#), l'[orientation](#), les [Grandes Ecoles](#) ainsi que sur les [Olympiades de Sciences de l'Ingénieur](#) et sur les [Sciences de l'Ingénieur au Féminin](#) sur notre site : www.upsti.fr

L'équipe UPSTI

Salle blanche d'usine

Corrigé UPSTI

C Déterminer le nombre de FFU, leur répartition et ajustement de la vitesse de rotation

Question 23 Écrire une fonction `volume` qui prend en argument les trois dimensions de la pièce et qui fournit le volume de la pièce en sortie.

```
1 def volume(L, l, h):  
2     return L*l*h
```

Question 24 Écrire une fonction `nbre_ffu` qui prend en argument V , Q et ACH et qui retourne le nombre N de FFU à prévoir pour répondre à l'exigence du renouvellement d'air du cahier des charges.

```
3 import math  
4  
5 def nbre_ffu(V, Q, tx_renew):  
6     n = int(math.ceil(V*tx_renew/Q))  
7     return n
```

Question 25 Justifier que le système d'équations qui relie nx, ny, a et b est le suivant :

- $L = (nx + 1) \cdot u + nx$
- $\ell = (ny + 1) \cdot v + ny$
- $(nx \times ny) < n$

La longueur L doit contenir nx FFU de 1 m et $nx + 1$ espacements u .

La largeur l doit contenir ny FFU de 1 m et $ny + 1$ espacements v (le décalage de $0,5 * v$ en haut de colonne est compensé par le décalage de $1,5 * v$ en bas de colonne).

Le nombre total de FFU est de $nx \times ny$, qui doit être supérieure ou égale au nombre minimum n (et non strictement inférieur comme l'indique le sujet, sinon le cahier des charges ne sera pas respecté).

Question 26 Écrire la situation d'initialisation de la boucle de la fonction `repartition` qui est la situation où tous les FFU sont alignés sur une seule ligne. Déterminer les expressions de nx, ny, a et b dans cette situation initiale.

Question 27 Après l'initialisation des paramètres, la fonction `repartition` utilise une boucle `while` pour calculer les nouvelles grandeurs nx, ny, a et b dans la situation d'une répartition des FFU sur une ligne supplémentaire. Définir la condition d'arrêt de la boucle `while` COND1 et les expressions EXP2, EXP3 définies dans le script précédent.

```
8 def repartition(l, L, n):  
9     nx = n  
10    ny = 1  
11    u = (L-nx)/(nx + 1)  
12    v = (l-ny)/(ny+1)
```

```

13
14     while (u < 1.0 or v < 0.0) and ny < n:
15         ny = ny+1
16         nx = n//ny
17
18         u = (L-nx)/(nx + 1)
19         v = (1-ny)/(ny+1)
20     return nx, ny, u, v
21
22 def coeff(Q, V, nx, ny, tx_renew):
23     Qreel=V*tx_renew/(nx*ny)
24     return Qreel/Q

```

La valeur renvoyée est nx, ny, u, v et non nx, ny, a, b comme l'indique le sujet, puisque a et b ne sont pas définis dans le code à trou du sujet.

D Étude de résultats expérimentaux afin de modéliser le couple résistant ramené sur l'arbre moteur

1 Extraction des résultats expérimentaux d'un essai

Question 28 Écrire une fonction python nommée `remplace_virgule_par_point(chaine)` qui prend en argument une chaîne de caractères appelée chaine et qui renvoie une nouvelle chaîne de caractères obtenue en remplaçant les virgules "," par des points ".". On rappelle que le type string est un conteneur numérique comme une liste mais non mutable. Ainsi on fabriquera la nouvelle chaîne, caractère par caractère.

```

25 def replace_virgule_par_point(chaine):
26     res = ""
27     for c in chaine:
28         if c == ",":
29             res += "."
30         else:
31             res += c
32     return res

```

Question 29 La capture précédente fournit le script incomplet de la fonction `extraction_resultats_essai(nom_fi` qui prend en argument le nom du fichier de l'essai analysé et qui renvoie les quatre listes des grandeurs physiques numériques type flottants de l'essai étudié `Ltemps`, `Lconsigne`, `Lvitesse`, `Lcourant`, soit les données présentes dans la première, la deuxième, la troisième et la quatrième colonne du fichier à partir de la 14ème ligne. Définir les arguments `EZR1`, `EZR2`, `EZR3` et `EZR4` définis dans le script précédent.

```

33 def extraction_resultats_essai(chemin_fichier):
34     # initialisation des listes vides
35     Ltemps=[]
36     Lconsigne=[]
37     Lvitesse=[]
38     Lcourant=[]
39     fichier=open(chemin_fichier,"r")
40     # pour passer les 13 premières lignes
41     for i in range(13):
42         ligne=fichier.readline()
43     # pour traiter les lignes suivantes
44     for ligne in fichier:
45         ligne2=replace_virgule_par_point(ligne)
46         liste=ligne2.split("\t")
47         # remplissage des données en transformant les str en float

```

```

48     Ltemps.append(float(liste[0]))
49     Lconsigne.append(float(liste[1]))
50     Lvitesse.append(float(liste[2]))
51     Lcourant.append(float(liste[3]))
52     fichier.close()
53     return Ltemps, Lconsigne, Lvitesse, Lcourant

```

Question 30 Que signifient ces deux instructions et décrire la structure de données choisie ? Préciser son intérêt.

La première instruction construit un dictionnaire, la seconde le renvoie.

Un dictionnaire permet de stocker différentes valeurs de façon non ordonnée (contrairement à une liste), mais en les associant à une clé (ici, "temps", "consigne..."), ce qui permet ici de facilement sélectionner les données qui nous intéressent pour les traiter dans la suite.

Question 31 Écrire un script python qui permet d'extraire les résultats de l'essai 1 qui seront stockés dans un dictionnaire nommé `donnees`, puis qui affiche la liste des temps.

```

54 donnees = extraction_resultats_essai("essai1.txt")
55 print(donnees["temps"])

```

Question 32 Écrire une fonction `affichage(donnees)` qui prend en argument le dictionnaire stockant les résultats d'un essai et qui renvoie la courbe des vitesses mesurées d'un essai en fonction du temps.

```

56 def affichage(donnees):
57     temps = donnees["temps"]
58     vitesses = donnees["vitesse"]
59
60     plt.plot(temps, vitesses)
61
62     # Mise en forme du graphique
63     plt.xlabel("Temps (s)")
64     plt.ylabel("Vitesse de rotation (tr/min)")
65     plt.legend("Vitesse réelle")
66     plt.grid()
67
68     plt.show()

```

2 Détermination du couple résistant

Question 33 Écrire la fonction `moyenne_sur_intervalle(Liste, deb, fin)` qui permet de déterminer la valeur moyenne des nombres compris entre les indices `deb` (inclus) et `fin` (exclu) de la liste `Liste`.

```

69 def moyenne_sur_intervalle(Liste, deb, fin):
70     return sum(Liste[deb:fin])/(fin-deb)
71
72 # Autre version
73 def moyenne_sur_intervalle(Liste, deb, fin):
74     s = 0 # somme partielle
75     for i in range(deb, fin):
76         s += Liste[i]
77     return s/(fin-deb)

```

Question 34 Écrire une fonction `point_de_fonctionnement(donnees)` qui prend en argument le dictionnaire stockant les résultats de l'essai étudié et qui renvoie la valeur de la vitesse de rotation `N` en régime

permanent et du couple résistant Cr de l'essai étudié en tenant compte uniquement du courant entre 50 et 70 s.

```

78 def point_de_fonctionnement(donnees):
79     K = 0.13 # En Nm/A
80     vitesse = moyenne_sur_intervalle(donnees["vitesse"], 500, 701)
81     courant = moyenne_sur_intervalle(donnees["courant"], 500, 701)
82     return vitesse, courant*K

```

3 Analyse de l'ensemble des essais

Question 35 Écrire un script python qui pour chaque essai présent dans le dossier, détermine le couple résistant Cr et la vitesse de rotation Vr sur l'intervalle de temps compris entre 50 et 70 s et mémorise Cr et N dans deux listes L_vit et L_Cr .

```

83 L_vit, L_Cr = [], []
84 for i in range(1, 18+1):
85     nomfichier = "essai" + str(i) + ".txt"
86     donnees = extraction_resultats_essai(nomfichier)
87     res = point_de_fonctionnement(donnees)
88     L_vit.append(res[0])
89     L_Cr.append(res[1])

```

4 Modélisation du comportement couple résistant Cr en fonction de la vitesse du moteur N par une régression linéaire

Question 36 Justifier que la méthode de descente de gradient est une méthode d'apprentissage supervisé. Expliquer la différence entre un apprentissage supervisé avec classification et avec régression.

C'est une méthode :

- d'apprentissage : on utilise des données d'entraînement pour déterminer a_1 et a_0 , pour ensuite les utiliser dans la phase d'inférence.
- supervisé : les données d'apprentissage font partie de données d'entraînement, qui sont étiquetées.

Une méthode d'apprentissage supervisé avec :

- classification trie les données dans différents groupes. Le résultat est une fonction permettant d'indiquer à quel groupe appartient la donnée d'entrée.
- régression associe à une entrée un résultat numérique (non discret).

Question 37 Qu'est-ce qu'une régression linéaire multiple ? Quel serait dans ce cas l'expression du modèle de comportement de Cr à rechercher ?

Une régression linéaire multiple permet d'inférer une valeur en fonction de plusieurs variables.

Par exemple, on pourrait obtenir un modèle de comportement de Cr en fonction de la vitesse de rotation et (comme proposé par le sujet) de la température, de l'humidité et du nombre d'heures d'utilisation. En supposant une relation linéaire entre Cr et ces paramètres, on pourrait avoir un modèle de comportement de la forme :

$$Cr = a_4 \times T_{\text{celsius}} + a_3 \times T_{\text{humidite}} + a_2 \times H_{\text{util}} + a_1 \times w^2 + a_0$$

Question 38 Écrire un script python qui permet de construire la matrice X afin de prendre en compte le biais du modèle affine.

```
90 X = np.hstack((V_N2, np.ones([m, 1])))
```

Question 39 Écrire la fonction `modele` qui prend en argument X et `theta` et retourne la matrice Y .

```
91 def modele(X, theta):
92     return np.dot(X, theta)
```

Question 40 Écrire la fonction de coût $Q(a_0, a_1)$ en fonction de a_0 , a_1 , les $y^{(k)}$ et $x^{(k)}$ et m .

$$Q(a_0, a_1) = \sum_{i=1}^m (a_1 \times x^{(k)} + a_0 - y^{(k)})^2$$

Question 41 Écrire la fonction de coût $Q(\theta)$ sous forme matricielle en fonction de X , θ et Y .

$$Q(\theta) = (X \cdot \theta - Y)^T \cdot (X \cdot \theta - Y)$$

On obtient une matrice 1×1 , que l'on assimilera ici à un nombre.

Question 42 Écrire en python la fonction `cout` qui prend en argument trois vecteurs X , Y et `theta` et qui renvoie la valeur du coût $Q(\theta)$.

```
93 def cout(X, Y, theta):
94     diff = np.dot(X, theta) - Y
95     res = np.dot(diff, np.transpose(diff))
96     return res
```

Question 43 Écrire les dérivées partielles de la fonction $Q(a_0, a_1)$: $\frac{\partial Q}{\partial a_1}$ et $\frac{\partial Q}{\partial a_0}$.

$$Q(a_0, a_1) = \sum_{i=1}^m (a_1 \times x^{(k)} + a_0 - y^{(k)})^2 = \sum_{i=1}^m a_1^2 \times (x^{(k)})^2 + a_0^2 + (y^{(k)})^2 + 2a_1 a_0 x^{(k)} - 2a_1 x^{(k)} y^{(k)} - 2a_0 y^{(k)}$$

$$\frac{\partial Q}{\partial a_1} = \sum_{i=1}^m 2a_1 \times (x^{(k)})^2 + 2a_0 x^{(k)} - 2x^{(k)} y^{(k)}$$

$$\frac{\partial Q}{\partial a_0} = \sum_{i=1}^m 2a_0 + 2a_1 x^{(k)} - 2y^{(k)}$$

Question 44 Écrire le vecteur gradient de la fonction de coût $\overrightarrow{\text{grad}}(Q)$ sous forme matricielle en fonction de X , θ et Y et montrer que : $\overrightarrow{\text{grad}}(Q) = 2 \cdot X^T (X\theta - Y)$ avec X^T la transposée de la matrice X .

$$\overrightarrow{\text{grad}}(Q) = \begin{pmatrix} \frac{\partial Q}{\partial a_1} \\ \frac{\partial Q}{\partial a_0} \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^m 2a_1 \times (x^{(k)})^2 + 2a_0 x^{(k)} - 2x^{(k)} y^{(k)} \\ \sum_{i=1}^m 2a_0 + 2a_1 x^{(k)} - 2y^{(k)} \end{pmatrix}$$

$$= 2 \times \left[\underbrace{\left(\frac{\sum_{i=1}^m a_1 \times (x^{(k)})^2 + a_0 x^{(k)}}{\sum_{i=1}^m a_1 x^{(k)} + a_0} \right)}_{X^T(X\theta)} - \underbrace{\left(\frac{\sum_{i=1}^m x^{(k)} y^{(k)}}{\sum_{i=1}^m y^{(k)}} \right)}_{X^T Y} \right] = 2 \cdot X^T(X\theta - Y)$$

Question 45 — Écrire en python une fonction `gradient` qui prend en argument trois vecteurs `X`, `Y` et `theta` et qui renvoie le vecteur gradient du coût $\nabla_{\theta} Q(\theta)$. On pourra utiliser la fonction `np.transpose(M)` qui transpose la matrice `M`.

```
97 def gradient(X, Y, theta):
98     return 2 * np.dot(np.transpose(X), (np.dot(X, theta) - Y))
```

Question 46 — Écrire la fonction `descente_gradient(X,Y,theta_init,alpha,n)` qui prend en argument les données `X` et `Y`, le vecteur initial noté `theta_init`, la vitesse d'apprentissage `alpha` choisie, et `n` le nombre d'itérations choisi et qui renvoie le vecteur `theta` obtenu par la méthode de descente de gradient après les `n` itérations.

```
99 def descente_gradient(X, Y, theta_init, alpha, n):
100     theta = theta_init
101     for i in range(n):
102         theta = theta - alpha * gradient(X, Y, theta)
103     return theta
```

E Etude du comportement du moteur électrique entraînant le ventilateur

1 Etude du régime établi : recherche de la tension de commande moteur

Question 47 — Écrire une fonction `affichage` qui prend en argument les valeurs de `theta_0` et `theta_1` et qui permet de tracer l'allure $Cr(Vr)$ pour Vr compris entre 0 et 250 rad/s par pas de 10.

```
104 def affichage(a0, a1):
105     Omega = []
106     Cr = []
107     for w in np.linspace(0, 250, int(250/10 + 1)): # Calcul du nombre de points (pour éviter d'utiliser
108         Omega.append(w)                                range)
109         Cr.append(a1*w**2 + a0)
110
111     plt.plot(Omega, Cr)
112     plt.show()
```

Question 48 — Écrire une fonction `dichotomie(U,deb,fin,e)` qui prend en argument la valeur de la tension d'alimentation imposée `U` (en V), les bornes de l'intervalle de `w` $\in [deb, fin]$, la précision du résultat `e` et qui retourne les coordonnées (w, U) solution de l'équation $f(w, U) = 0$ à `e` près.

```
113 def f(w, U):
114     a1 = 5.3e-6
115     a0 = 0.05
116     return a1*w*w+a0-0.839*(U-1.17*w)
117
118 def dichotomie(U, deb, fin, e):
119     gauche = deb
```

```

120 droite = fin
121 fgauche = f(gauche, U)
122 fdroite = f(droite, U)
123 while abs(fgauche - fdroite) > e:
124     milieu = (gauche+droite) / 2
125     fmilieu = f(milieu, U)
126     if fgauche * fmilieu > 0:
127         gauche = milieu
128         fgauche = fmilieu
129     else:
130         droite = milieu
131         fdroite = fmilieu
132 return (gauche+droite) / 2, f((gauche+droite) / 2, U)

```

2 Etude en régime dynamique : prévision des caractéristiques de la réponse du FFU à un échelon de courant

Question 49 Écrire l'équation différentielle régissant la vitesse de rotation du rotor sous la forme $\frac{d\omega}{dt} = g(\omega(t), t)$. On précise que la fonction g ne dépend pas explicitement du temps t .

$$\frac{d\omega}{dt} = \frac{k}{J} \cdot I - \frac{b}{J} - \frac{a}{J} \omega(t)^2 = g(\omega(t), t)$$

Question 50 A l'aide d'un développement limité de la variable $\omega(t+T)$ à l'ordre 1, montrer que l'approximation de la dérivée peut s'écrire :

$$\frac{d\omega(t)}{dt} \approx \frac{\omega(t+T) - \omega(t)}{T} \text{ avec } T \ll t$$

$$\omega(t+T) = \omega(t) + T \times \frac{d\omega(t)}{dt} + o(T^2) \approx \omega(t) + T \times \frac{d\omega(t)}{dt}$$

$$\frac{d\omega(t)}{dt} \approx \frac{\omega(t+T) - \omega(t)}{T}$$

Question 51 Déterminer l'expression de $\omega(t+T)$ en fonction de T , $g(\omega(t), t)$ et $\omega(t)$.

Par un développement limité d'ordre 1, en substituant $\frac{d\omega(t)}{dt}$ par $g(\omega(t), t)$:

$$\omega(t+T) \approx \omega(t) + T \times g(\omega(t), t)$$

Question 52 Proposer une fonction `Euler_Explicite` dont les arguments sont g , $w0$, $t0$, $t1$, T permettant de résoudre l'équation différentielle $\frac{d\omega}{dt} = g(\omega(t), t)$ avec comme condition initiale $\omega(0) = w0$, par itérations successives de pas de temps T sur l'intervalle $[t0, t1]$ en utilisant la méthode d'Euler explicite, et renvoyant la liste des temps LT et la liste Lw des solutions approchées associées. On impose d'utiliser une boucle `while`.

```

133 def Euler_Explicite(g, w0, t0, t1, T):
134     Lw = [w0]
135     LT = [t0]
136

```

```
137     i = 0
138     t = t0
139     while t < t1:
140         t = t0 + i * T
141         LT.append(t)
142         w = Lw[-1] + T * g(Lw[-1], t)
143         i += 1
144
145     return LT, Lw
```

Question 53 Les listes LT et Lw étant calculées, écrire un script python permettant de déterminer puis d'afficher le temps de réponse et la vitesse de rotation en régime permanent du système.

On suppose que le temps de réponse correspond au temps de réponse à 95%.

```
146 LT, Lw = Euler_Explicite(...) # Déjà calculées
147
148 winit = Lw[0]
149 wfinale = Lw[-1]
150 w95pc = (wfinale-winit)*0.95
151
152 i = 0
153 while Lw[i] < w95pc and i < len(Lw):
154     i+=1
155 print("Le temps de réponse est : ", LT[i], " secondes")
156 print("La vitesse de rotation en régime permanent est : ", wfinale, " rad/s")
```