**ÉCOLE DES PONTS PARISTECH,
ISAE-SUPAERO, ENSTA PARISTECH,
TELECOM PARISTECH, MINES PARISTECH,
MINES SAINT-ÉTIENNE, MINES NANCY,
IMT Atlantique, ENSAE PARISTECH.**

**Concours Centrale-Supélec (Cycle International),
Concours Mines-Télécom, Concours Commun TPE/EIVP.**

CONCOURS 2018

ÉPREUVE D'INFORMATIQUE COMMUNE

Durée de l'épreuve : 1 heure 30 minutes

L'usage de la calculatrice et de tout dispositif électronique est interdit.

Cette épreuve est commune aux candidats des filières MP, PC et PSI

*Les candidats sont priés de mentionner de façon apparente
sur la première page de la copie :*

INFORMATIQUE COMMUNE

L'énoncé de cette épreuve comporte 10 pages de texte.

Si, au cours de l'épreuve, un candidat repère ce qui lui semble être une erreur d'énoncé, il le signale sur sa copie et poursuit sa composition en expliquant les raisons des initiatives qu'il est amené à prendre.

Téléchargé gratuitement sur Doc-Solus.fr .

Mesures de houle

Le sujet comporte des questions de programmation. Le langage à utiliser est `Python`.

On s'intéresse à des mesures de niveau de la surface libre de la mer effectuées par une bouée (représentée sur la Figure 1) [1]. Cette bouée contient un ensemble de capteurs incluant un accéléromètre vertical qui fournit, après un traitement approprié, des mesures à étudier [2].

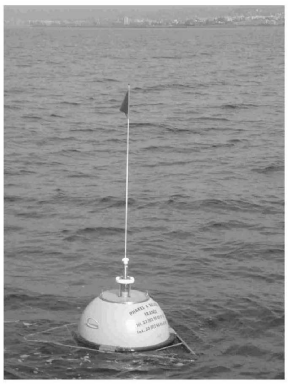

Figure 1 : bouée de mesure de houle

Les mesures réalisées à bord de la bouée sont envoyées par liaison radio à une station à terre où elles sont enregistrées, contrôlées et diffusées pour servir à des études scientifiques. Pour plus de sûreté, les mesures sont aussi enregistrées sur une carte mémoire interne à la bouée.

Partie I. Stockage interne des données

Une campagne de mesures a été effectuée. Les caractéristiques de cette campagne sont les suivantes :

- durée de la campagne : 15 jours ;
- durée d'enregistrement : 20 min toutes les demi-heures ;
- fréquence d'échantillonnage : 2 Hz.

Les relevés de la campagne de mesure sont écrits dans un fichier texte dont le contenu est défini comme suit.

Les informations relatives à la campagne sont rassemblées sur la première ligne du fichier, séparées par des points-virgules (";"). On y indique différentes informations importantes comme le numéro de la campagne, le nom du site, le type du capteur, la latitude et la longitude de la bouée, la date et l'heure de la séquence.

1. Cette étude utilise des résultats extraits de la base de données du Centre d'Archivage National des Données de Houle In Situ. Les acquisitions ont été effectuées par le Centre d'Etudes Techniques Maritimes Et Fluviales.
2. L'ensemble des paramètres des états de mer présent dans la base CANDHIS est calculé par les logiciels :
   - Houle5 (CETMEF) : analyse vague par vague (temporelle) ;
   - PADINES (EDF/LNHE) : analyse spectrale et directionnelle (fréquentielle).

Téléchargé gratuitement sur Doc-Solus.fr .

Les lignes suivantes contiennent les mesures du déplacement vertical (m). Chaque ligne comporte 8 caractères dont le caractère de fin de ligne. Par exemple, on trouvera dans le fichier texte les 3 lignes suivantes :

```
+0.4256
+0.3174
-0.0825
...
```

❏ **Q1** – On suppose que chaque caractère est codé sur 8 bits. En ne tenant pas compte de la première ligne, déterminer le nombre d'octets correspondant à 20 minutes d'enregistrement à la fréquence d'échantillonnage de 2 Hz.

❏ **Q2** – En déduire le nombre approximatif (un ordre de grandeur suffira) d'octets contenus dans le fichier correspondant à la campagne de mesures définie précédemment. Une carte mémoire de 1 Go est-elle suffisante ?

❏ **Q3** – Si, dans un souci de réduction de la taille du fichier, on souhaitait ôter un chiffre significatif dans les mesures, quel gain relatif d'espace mémoire obtiendrait-on ?

❏ **Q4** – Les données se trouvent dans le répertoire de travail sous forme d'un fichier `donnees.txt`. Proposer une suite d'instructions permettant de créer à partir de ce fichier une liste de flottants `liste_niveaux` contenant les valeurs du niveau de la mer. On prendra garde à ne pas insérer dans la liste la première ligne du fichier.

Deux analyses sont effectuées sur les mesures : l'une est appelée "vague par vague", l'autre est appelée "spectrale".

Partie II. Analyse "vague par vague"

On considère ici que la mesure de houle est représentée par un signal $\eta(t) \in \mathbb{R}$, $t \in [0, T]$, avec η une fonction C^1.

On appelle niveau moyen m la moyenne de $\eta(t)$ sur $[0, T]$.

On définit Z_1, Z_2, ..., Z_n l'ensemble (supposé fini) des Passages par le Niveau moyen en Descente (PND) (voir Figure 2). A chaque PND, le signal traverse la valeur m en descente.

On suppose $\eta(0) > m$, et $\dfrac{\mathrm{d}\eta}{\mathrm{d}t}(0) > 0$.

On en déduit que $\eta(t) - m \geq 0$ sur $[0,\ Z_1]$.

Les hauteurs de vagues H_i sont définies par les différences :

$$\begin{cases} H_1 = \max\limits_{t\in[0,\ Z_1]} \eta(t) - \min\limits_{t\in[Z_1,\ Z_2]} \eta(t) \\ H_i = \max\limits_{t\in[Z_{i-1},\ Z_i]} \eta(t) - \min\limits_{t\in[Z_i,\ Z_{i+1}]} \eta(t) \quad \text{pour } 2 \leq i < n \end{cases}$$

.

On définit les *périodes de vagues* par $T_i = Z_{i+1} - Z_i$.

❏ **Q5** – Pour le signal représenté sur la Figure 2, que valent approximativement H_1, H_2 et H_3 ? Que valent approximativement T_1 et T_2 ?

On adopte désormais une représentation en temps discret du signal. On appelle *horodate*, un ensemble (fini) des mesures réalisées sur une période de 20 minutes à une fréquence d'échantillonnage de 2 Hz. Les informations de niveau de surface libre d'un horodate sont stockées dans une

Téléchargé gratuitement sur Doc-Solus.fr .

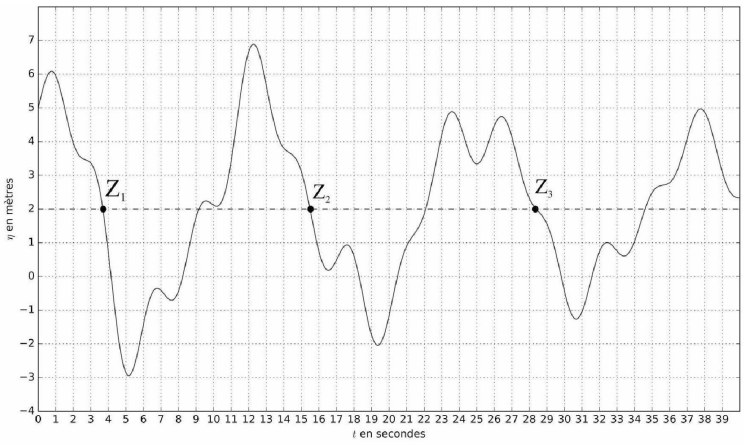

FIGURE 2 : Passages par le Niveau moyen en Descente (PND). Ici la moyenne m vaut 2

liste de flottants `liste_niveaux`. On suppose qu'aucun des éléments de cette liste n'est égal à la moyenne.

❏ **Q6** – Proposer une fonction `moyenne` prenant en argument une liste non vide `liste_niveaux`, et retournant sa valeur moyenne.

❏ **Q7** – Proposer une fonction `integrale_precise` prenant en argument une liste non vide `liste_niveaux`, et retournant la valeur approchée de l'intégrale de η sur une période de 20 minutes. On demande d'utiliser la méthode des trapèzes. En déduire une fonction `moyenne_precise` prenant en argument une liste non vide `liste_niveaux` et retournant une estimation de la moyenne de η sur une période de 20 minutes.

❏ **Q8** – Proposer une fonction `ind_premier_pzd(liste_niveaux)` retournant, s'il existe, l'indice du premier élément de la liste tel que cet élément soit supérieur à la moyenne et l'élément suivant soit inférieur à la moyenne. Cette fonction devra retourner -1 si aucun élément vérifiant cette condition n'existe.

❏ **Q9** – Proposer une fonction retournant l'indice i du *dernier* élément de la liste tel que cet élément soit supérieur à la moyenne et l'élément suivant soit inférieur à la moyenne. Cette fonction devra retourner -2 si aucun élément vérifiant cette condition n'existe. On cherchera à proposer une fonction de complexité $O(1)$ dans le meilleur des cas.

On souhaite stocker dans une liste `successeurs`, les indices des points succédant (strictement) aux PND (voir Figure 3).

❏ **Q10** – On propose la fonction `construction_successeurs` en annexe (algorithme 1). Elle retourne la liste `successeurs`. Compléter (sur la copie) les lignes 6 et 7.

❏ **Q11** – Proposer une fonction `decompose_vagues(liste_niveaux)` qui permet de décomposer une liste de niveaux en liste de vagues. On omettra les données précédant le premier PND et celles

Téléchargé gratuitement sur Doc-Solus.fr .

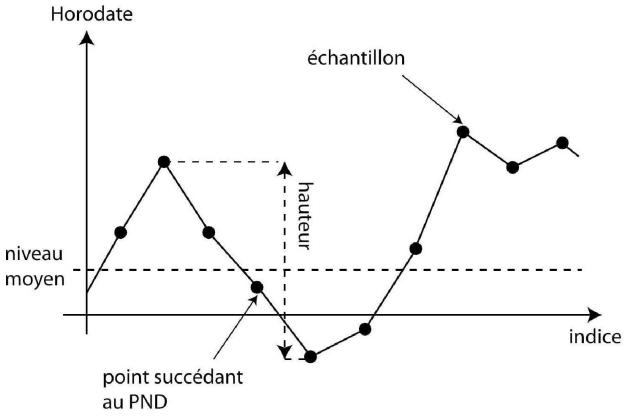

FIGURE 3 : propriétés d'une vague

succédant au dernier PND. Ainsi `decompose_vagues([1, -1, -2, 2, -2, -1, 6, 4, -2, -5])` (noter que cette liste est de moyenne nulle) retournera `[[-1, -2, 2], [-2, -1, 6, 4]]`.

On désire maintenant caractériser les vagues.

Ainsi, on cherche à concevoir une fonction `proprietes(liste_niveaux)` retournant une liste de listes à deux éléments `[Hi,Ti]` permettant de caractériser *chacune des vagues i* par ses attributs :

- `Hi`, sa hauteur en mètres (m) (voir Figure 3),
- `Ti`, sa période en secondes (s).

❏ **Q12** – Proposer une fonction `proprietes(liste_niveaux)` réalisant cet objectif. On pourra utiliser les fonctions de Python `max(L)` et `min(L)` qui retournent le maximum et le minimum d'une liste `L`, respectivement.

Partie III. Contrôle des données

Plusieurs indicateurs sont couramment considérés pour définir l'état de la mer. Parmi eux, on note :

- H_{max} : la hauteur de la plus grande vague observée sur l'intervalle d'enregistrement $[0, T]$;
- $H_{1/3}$: la valeur moyenne des hauteurs du tiers supérieur des plus grandes vagues observées sur $[0, T]$;
- $T_{H1/3}$: la valeur moyenne des périodes du tiers supérieur des plus grandes vagues observées sur $[0, T]$.

❏ **Q13** – Proposer une fonction prenant en argument la liste `liste_niveaux` de la question 12 et retournant H_{max}.

Afin de déterminer $H_{1/3}$ et $T_{H1/3}$, il est nécessaire de trier la liste des propriétés des vagues. La méthode utilisée ici est un *tri rapide* (*quick sort*). On donne en annexe un algorithme possible pour la fonction `triRapide` (algorithme 2). Trois arguments sont nécessaires : une liste `liste`, et deux indices `g` et `d`.

❏ **Q14** – Préciser les valeurs que doivent prendre les arguments `g` et `d` au premier appel de la fonction `triRapide`. Compléter (sur la copie) la ligne 2.

Téléchargé gratuitement sur Doc-Solus.fr .

Lorsque le tri rapide est utilisé et que le nombre de données à traiter devient petit dans les sous-listes (de l'ordre de 15), il peut être avantageux d'utiliser un "tri par insertion". On appelle `triInsertion` la fonction qui permet d'effectuer un tri par insertion. Elle admet en argument une liste `liste`, et deux indices `g` et `d`. Ces deux indices permettent de caractériser la sous-partie de la liste à trier (indices de début et de fin inclus).

❏ **Q15** – Donner les modifications à apporter à la fonction `triRapide` pour que, lorsque le nombre de données dans une sous-liste devient inférieur ou égal à 15, la fonction `triInsertion` soit appelée pour terminer le tri.

Le code incomplet de la fonction `triInsertion` est donné en annexe : algorithme 3.

❏ **Q16** – La fonction `triInsertion` admet trois arguments : une liste de données `liste`, et deux indices `g` et `d`. Elle trie dans l'ordre croissant la partie de la liste comprise entre les indices `g` et `d` inclus. Compléter cette fonction (avec sur la copie le nombre de lignes de votre choix).

La distribution des hauteurs de vague (voir Figure 4) lors de l'analyse *vague par vague* est réputée être gaussienne. On peut contrôler ceci par des tests de *skewness* (variable désignée par S) et de *kurtosis* (variable désignée par K) définis ci-après. Ces deux tests permettent de quantifier respectivement l'asymétrie et l'aplatissement de la distribution.

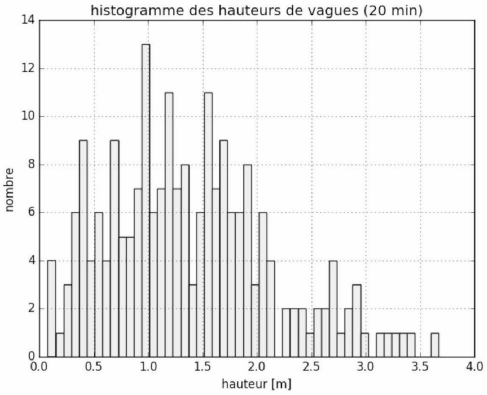

FIGURE 4 : histogramme des hauteurs de vague

On appelle $\overline{H}$ et σ^2 les estimateurs non biaisés de l'espérance et de la variance, n le nombre d'éléments H_1, H_2, ..., H_n.

On définit alors

$$S = \frac{n}{(n-1)(n-2)} \times \left(\frac{1}{\sigma^3}\right) \times \sum_{i=1}^{n} \left(H_i - \overline{H}\right)^3$$

$$K = \frac{n}{(n-1)(n-2)(n-3)} \times \left(\frac{1}{\sigma^4}\right) \times \sum_{i=1}^{n} \left(H_i - \overline{H}\right)^4 - \frac{3(n-1)^2}{(n-2)(n-3)}$$

Le test suivant est appliqué :

Téléchargé gratuitement sur Doc-Solus.fr .

– si la valeur absolue de S est supérieure à 0,3 alors l'horodate est déclaré non valide ;
– si la valeur de K est supérieure à 5 alors l'horodate est déclaré non valide.

On utilise la fonction `moyenne` pour estimer la valeur de $\overline{H}$, et on suppose disposer de la fonction `ecartType` qui permet de retourner la valeur de l'écart type non biaisé σ.

❑ **Q17** – Un codage de la fonction `skewness` pour une liste ayant au moins 3 éléments est donné en annexe (algorithme 4). Le temps d'exécution est anormalement long. Proposer une modification simple de la fonction pour diminuer le temps d'exécution (sans remettre en cause le codage des fonctions `ecartType` et `moyenne`).

❑ **Q18** – Doit-on s'attendre à une différence de type de la complexité entre une fonction évaluant S et une fonction évaluant K ?

Partie IV. Base de données relationnelle

On dispose d'une base de données relationnelle `Vagues`.

La première table est `Bouee`. On se limite aux attributs suivants : le numéro d'identification `idBouee`, le nom du site `nomSite`, le nom de la mer ou de l'océan `localisation`, le type du capteur `typeCapteur` et la fréquence d'échantillonnage `frequence`.

`Bouee`

idBouee	nomSite	localisation	typeCapteur	frequence
831	Porquerolles	Mediterranee	Datawell non directionnelle	2.00
291	Les pierres noires	Mer d'iroise	Datawell directionnelle	1.28
...	...	...	...	...

La seconde table est `Campagne`.

On se limite aux attributs suivants : le numéro d'identification `idCampagne`, le numéro d'identification de la bouée `idBouee`, la date de début `debutCampagne` et la date de fin `finCampagne`.

`Campagne`

idCampagne	idBouee	debutCampagne	finCampagne
08301	831	01/01/2010 00h00	15/01/2010 00h00
02911	291	15/10/2005 18h30	18/10/2005 08h00
...	...	...	...

La troisième table est `Tempete`. Les informations fournies relatives à un événement "tempête" sont les suivantes :

– date de début et fin de tempête,
– évolution des paramètres $H_{1/3}$ et H_{max} en fonction du temps,
– le détail de certains paramètres non définis ici, obtenus au pic de tempête.

On se limite aux attributs suivants : le numéro d'identification de la tempête `idTempete`, le numéro d'identification de la bouée `idBouee`, la date de début `debutTempete`, la date de fin `finTempete`, la valeur maximale de hauteur de vague `Hmax`.

`Tempete`

idTempete	idBouee	debutTempete	finTempete	Hmax
083010	831	07/01/2010 20h00	09/01/2010 15h30	5.3
029012	291	16/10/2005 08h30	18/10/2005 09h00	8.5
...	...	...	...	...

Le schéma de la base de données est donc : `Vagues`={`Bouee`, `Campagne`, `Tempete`}.

❑ **Q19** – Formuler les requêtes SQL permettant de répondre aux questions suivantes :
– "Quels sont le numéro d'identification et le nom de site des bouées localisées en Méditerranée ?"
– "Quel est le numéro d'identification des bouées où il n'y a pas eu de tempêtes ?"
– "Pour chaque site, quelle est la hauteur maximale enregistrée lors d'une tempête ?"

Téléchargé gratuitement sur Doc-Solus.fr .

Partie V. Analyse "spectrale"

L'analyse spectrale (fréquentielle) du niveau, permet elle aussi de caractériser l'état de la mer, qui peut, en première approximation, être modélisé par une superposition linéaire d'ondes sinusoïdales indépendantes.

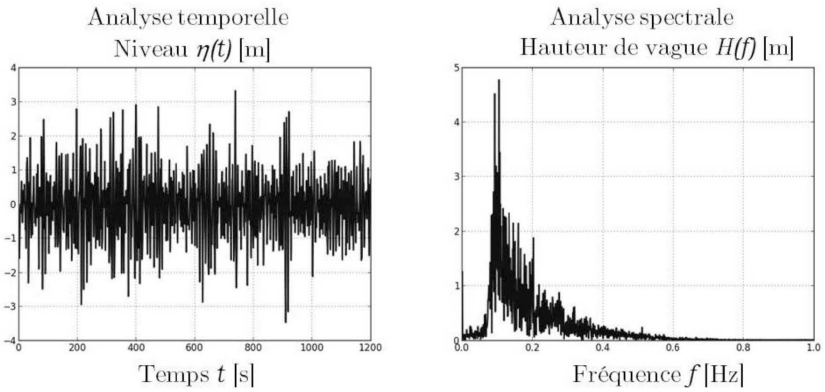

FIGURE 5 : analyses temporelle et spectrale

Des coefficients estimateurs de l'état de la mer issus de l'analyse spectrale ont donc été définis. Parmi eux, on note par exemple H_{m0} la hauteur significative spectrale des vagues ou T_P la période de pic barycentrique.

Pour leur calcul, il est nécessaire d'introduire la Transformation de Fourier Discrète (TFD).

Sa définition pour un signal numérique x de N échantillons, est la suivante :

$$X_k = \sum_{i=0}^{N-1} x_i \times e^{-2\pi j k \frac{i}{N}} \ , \ 0 \leq k < N \text{ et } j^2 = -1 \qquad (1)$$

Il existe plusieurs méthodes dites de "transformée de Fourier rapide". On étudie dans la suite l'algorithme de Cooley – Tukey adapté de celui de Gauss. On propose ici une réécriture de (1) appelée entrelacement temporel (DIT decimation-in-time).

Dans toute la suite, on suppose que N **est une puissance de 2**.

On note $w = e^{-\frac{2\pi j}{N}}$ (qui est une racine N-ième de l'unité).

On pose

$$P_k = \sum_{i=0}^{N/2-1} x_{2i} \times e^{-\frac{2\pi j}{N/2} ik}$$

$$I_k = \sum_{i=0}^{N/2-1} x_{2i+1} \times e^{-\frac{2\pi j}{N/2} ik}$$

P_k : TFD des indices pairs, I_k : TFD des indices impairs

On montre alors que pour $0 \leq k < \frac{N}{2}$, $\left| \begin{array}{l} X_k = P_k + w^k I_k \\ X_{k+N/2} = P_k - w^k I_k \end{array} \right.$

L'algorithme est de type "diviser pour régner" : le calcul d'une TFD pour N éléments se fait à l'aide de deux TFD de $N/2$ éléments.

❑ **Q20** – Quelle est la complexité en temps de cet algorithme en fonction de N ? Justifier en une ou deux lignes.

❑ **Q21** – Écrire une fonction *récursive* prenant en argument la liste de données `x` et retournant la liste `X` obtenue par transformée de Fourier discrète rapide. La longueur de `x` est une puissance de 2.

Téléchargé gratuitement sur Doc-Solus.fr .

Annexe

Algorithme 1

```
def construction_successeurs(liste_niveaux):
    n=len(liste_niveaux)
    successeurs=[]
    m=moyenne(liste_niveaux)
    for i in range(n-1):
        if                    # A completer
                              # A completer
    return successeurs
```

Algorithme 2

```
def triRapide(liste,g,d):
    pivot=                        # A completer
    i=g
    j=d
    while True:
        while i<=d and liste[i][0]<pivot:
            i=i+1
        while j>=g and liste[j][0]>pivot:
            j=j-1
        if i>j:
            break
        if i<j:
            liste[i],liste[j]=liste[j],liste[i]
        i=i+1
        j=j-1
    if g<j:
        triRapide(liste,g,j)
    if i<d:
        triRapide(liste,i,d)
```

Algorithme 3

```
def triInsertion(liste,g,d):
    for i in range(g+1,d+1):
        j=i-1
        tmp = liste[i]
        while                          # A completer






        liste[j+1]=tmp
```

Algorithme 4

```
def skewness(liste_hauteurs):
    n=len(liste_hauteurs)
    et3=(ecartType(liste_hauteurs))**3
    S=0
    for i in range(n):
        S+=(liste_hauteurs[i]-moyenne(liste_hauteurs))**3
    S=n/(n-1)/(n-2)*S/et3
    return S
```

Fin de l'épreuve.

Téléchargé gratuitement sur Doc-Solus.fr .