CONCOURS BLANC – 1H30

IDENTIFICATION DU COMPORTEMENT D'UN SYSTÈME ASSERVI

Éléments de corrigé

1 Préambule

Question 1

Quel type de variable est présent dans le fichier texte ? Quel type de variable proposeriez vous pour stocker un élément de la première colonne ? un élément de la seconde colonne ? une colonne toute entière ?

Corrigé

- Fichier texte : chaîne de caractères.
- Élément de la première et de la seconde colonne : flottant (échantillonnage de 1 kHz $\Longrightarrow$ 1 mesure toutes les 0,001 seconde).
- Colonne entière liste (ou tableau) de flottants.

Question 2

Quelle sera la taille d'un fichier de mesure courant dans le cas le plus défavorable (mesure de 2 secondes) ?

Corrigé

Codage ASCII donc 1 octet par caractère. 2 secondes à 1 kHZ donc 2000 mesures.

- Codage des secondes (0,001) : 5 octets
- Espace : 1 octet
- Codage de la mesure (999,999) : 7 octets
- Retour à la ligne : 1 octet

On a donc approximativement 14 octets par ligne soit un fichier de 28 ko.

Question 3

On donne la séquence d'instruction suivante. Expliquer l'objectif de chacune des lignes.

Corrigé

```python
nom_fichier="fichier_mesure.dat"   # Affectation du nom du fichier
fid = open(nom_fichier,'r')        # Ouverture en lecture du fichier
donnees_fichier=fid.readlines()    # Stockage de chacune des lignes du fichier dans
                                   # un élément de la liste donnees_fichier
fid.close()                        # Fermeture du fichier
```

```python
def ordre(temps,mesures):
    j=0
    for i in range(len(mesures)-1):
        j=i
        if abs(mesures[i]-mesures[i+1])>0,1:
            break
    pente = (abs(mesures[j+1]-mesures[j]))/(temps[j+1]-temps[j])
    if pente <0,1:
        return 2
    else :
        return 1
```

Question 4

Expliquer l'objectif de la boucle for (lignes 3 à 6) ?

Corrigé

Le but des lignes 3 à 6 est de détecter le début de la mesure. Puis conserver l'indice de début de la mesure (stocké dans i)

Question 5

Expliquer l'objectif des lignes 7 à 11 ? À quoi correspond le test de la ligne 8 ?

Corrigé

Ces lignes permettent de retourner l'ordre (estimé) du système. Pour cela on définit un seuil de pente (arbitraire) à l'origine au delà duquel on à affaire à un premier ordre (et en deçà duquel on considère que l'ordre est 2).

2 Identification d'un système d'ordre 1

Question 6

Réaliser une fonction simple appelée calculGain prenant comme argument le tableau mesures et l'amplitude E0 de l'échelon d'entrée et retournant la valeur du gain statique K.

Corrigé

```
1  def calculGain(mesures,E0):
2      gain = (mesures[len(mesures)-1]-mesures[0])/E0
3      return gain
```

Question 7

Pour une plus grande fiabilité des résultats pour le calcul de K, on désire utiliser une moyenne sur les 20 derniers pour cents d'une mesure. Quel en est l'intérêt ? Implémenter une fonction nommée calculValeurFinale permettant de déterminer la valeur finale. Implémenter alors une fonction nommée calculGainMoyen permettant de déterminer K en tenant compte de ces nouvelles contraintes.

Corrigé

python

```
1  def calculValeurFinale(mesures):
2      i0 = int(len(mesures)*80/100)
3      moyenne=0
4      for i in range(i0,len(mesures)):
5          moyenne = moyenne+mesures[i]
6      moyenne = moyenne/(len(mesures)-i0)
7      return moyenne
8
9  def calculGainMoyen(mesures,E0):
10     gain = calculValeurFinale(mesures)/E0
11     return gain
```

Question 8

En prenant comme critère le fait que le signal a atteint 63% de sa valeur finale au bout d'un temps τ, compléter la fonction calculTau permettant de déterminer la constante de temps du système en utilisant un algorithme de recherche dichotomique. On rappelle l'existence de la fonction calculDebut.

Corrigé

```python
def calculTau(temps,mesures,E0):
    i0 = calculDebut(mesures)                  # Index du depart de la mesure
    ifin = len(mesures)                        # Index de la derniere mesures
    mes0 = mesures[i0]                         # Valeur initiale
    mesf = calculValeurFinale(mesures)         # Calcul de la valeur finale
    mes_63 = 0,63*(mesf-mes0)+mes0

    ind_g = i0
    ind_d = ifin
    mes_g=mesures[ind_g]
    mes_d=mesures[ind_d]

    while ind_g-ind_d>2:
        ind_m = int((ind_g+ind_d)/2)
        mes_m =mesures[ind_m]
        if mes_m <= mes_63 :
            ind_g = ind_m

        else :
            ind_d = ind_m

    return temps[ind_m]-temps[i0]
```

Question 9

Que pouvez-vous dire de la précision de la valeur de la constante de temps ? Serait-il possible de trouver une meilleure approximation ? Si oui, expliquer succinctement votre démarche.

Corrigé

Dans ce cas, on prend la mesure la plus proche des 63% de la valeur finale. On pourrait interpoler la mesure pour avoir une meilleure approximation du temps de réponse.

Question 10

Quelle est la complexité de cet algorithme ? Serait-il possible d'estimer le nombre d'itération de la boucle while ? Quelle serait la complexité d'un algorithme de recherche naïf ? Quel est l'algorithme le plus efficace temporellement ?

Corrigé

L'algormithme de dichotomie a une complexité $\mathcal{O}(log(n))$.

Dans le pire des cas, pour un tableau de n éléments, on cherche le nombre d'itération k tel que $\frac{n}{2^k} < 1$ soit $k > \frac{\log n}{\log 2}$.

L'algorithme naïf aurait ici une complexité linéaire ce qui est moins efficace qu'un algorithme logarithmique.

Question 11

Réaliser la fonction identificationOrdre1 permettant de calculer le gain et la constante de temps. Cette fonction prendra comme argument les tableaux temps et mesures.

Corrigé

```python
def identificationOrdre1(temps,mesures,E0):
    K = calculGainMoyen(mesures,E0)
    tau = calculTau(temps,mesures,E0)
    return K,tau
```

3 Identification d'un système d'ordre 2

Question 12

Donner une méthode permettant de déterminer automatiquement le premier dépassement à partir d'une mesure.

Corrigé

Il s'agit de rechercher le maximum s_{max} de la liste. On réalise ensuite l'opération $d = \frac{s_{max} - s_{fin}}{s_{fin} - s_0}$.

Question 13

Réaliser la fonction premierDepassement permettant de savoir quand a lieu le temps de premier dépassement et quelle en est sa valeur.

Corrigé

Selon l'idée précédente, il faut réaliser l'algorithme permettant de rechercher le maximum de la liste des mesures.

```python
# Fonction à tester
def premierDepassement(temps,mesure):
    maxi = 0
    for i in range(len(mesure)):
        if mesure[i]>mesure[maxi]:
            i = maxi
    d = (mesure[maxi]-mesure[len(mesure)-1])/(mesure[len(mesure)-1]-mesure[0])
    res=[temps[i], dep]
    return res
```

Question 14

On souhaite savoir à quel moment a lieu l'intersection entre la mesure et l'asymptote horizontale lors de la première montée. Donner une méthode permettant de déterminer ce temps de montée.

Corrigé

On fait une recherche dichotomique comme dans la partie précédente.

La fonction pseudoPeriode permet de calculer la pseudo période à partir du fichier de mesures.

Question 15

Réaliser la fonction identificationOrdre2 permettant de calculer le gain la pulsation propre et le coefficient d'amortissement. Cette fonction prendra comme argument les tableaux temps et mesures ainsi que E_0 (amplitude de l'échelon d'entrée).

Corrigé

```python
# Fonction à tester
def identificationOrdre2 (temps,mesure,E0):
    om0=pseudoPeriode(temps,mesure)
    K = calculGainMoyen(mesure,E0)
    dep = premierDepassement(temps,mesure)
    m = calculAmortissement(dep)
    res=[K,om0,dep]
    return res
```

4 Synthèse

Question 16

Réaliser la fonction identification permettant de déterminer dans un premier temps si la mesure réalisée est d'ordre 1 ou d'ordre 2. Dans un second temps, cette fonction devra calculer les paramètres caractéristiques des fonctions de transfert. Cette fonction prendra comme argument une chaîne de caractère correspondant au nom du fichier de mesures. Elle retournera les caractéristiques du système.

Corrigé

```python
# Fonction à tester
def identification(fichierMesure):
    temps,mesure=traitementDonnees(fichierMesure)
    ord = ordre(temps,mesures)
    if ord == 1:
        return identificationOrdre1(temps,mesure)
    elif ord == 2:
        return identificationOrdre2(temps,mesure)
    else:
        return False
```

Question 17

On désire avoir recours à la dérivée du signal mesuré. Réaliser la fonction deriveMesure prenant comme argument les tableaux temps et mesures. Cette fonction renvoie une liste de couples [temps,derivee].

Corrigé

```python
# Fonction à tester
def deriveMesure(temps,mesure):
    res = []
    for i in range(len(temps)-1):
        t = temps[i]
        d = (mesure[i+1]-mesure[i])/(temps[i+1]-temps[i])
        res.append([t,d])
    return res
```

Question 18

On désire avoir recours à l'intégrale du signal mesuré. Réaliser la fonction integreMesure prenant comme argument les tableaux temps et mesures. Cette fonction renvoie une liste de couples [temps,integrale].

Algorithme de calcul d'une intégrale numérique. (Méthode des rectangles à gauche)

Corrigé

```python
# Fonction à tester
def integreMesure(temps,mesure):
    res = []
    for i in range(len(temps)-1):
        t = temps[i]
        i = mesure[i]*(temps[i+1]-temps[i])
        res.append([t,i])
    return res
```