C++ : Contrôle Continu 1

Exercice 1 (12 points)

Nous nous intéressons dans cet exercice à la mise en œuvre d'une méthode d'approximation au sens des moindres carrés. Le but est d'approcher un nuage de points par un polynôme (ici d'ordre 1 ou 2). Ce nuage de point pourrait représenter par exemple des mesures qui seraient bruitées et la méthodologie des moindres carrés nous permettra de retrouver les mesures réelles.

En pièce jointe, vous trouverez un fichier texte nommé ex1_data.txt contenant un ensemble de mesures. Ce fichier est basé sur le format du CSV (Comma-separated values) qui est un format informatique très utilisé permettant de représenter des données tabulaires sous forme de valeurs séparées par des virgules. Les données dans le fichier sont représentées sous forme de colonne. La première ligne correspond au nom de chaque donnée.

Fichier au format .csv

```
x, y, z
0, 2.3, 3.4
0.5, 3.2, 6.8
1, 2.7, 5.3
1.5, 4.1, 2.4
2, 4.4, 2.2
```

Représentation tabulaire

x	y	z
0	2.3	3.4
0.5	3.2	6.8
1	2.7	5.3
1.5	4.1	2.4
2	4.4	2.2

Ce fichier permet de définir 3 séries de valeurs notées $x = \{x_i\}_{i=1}^N$, $y = \{y_i\}_{i=1}^N$ et $z = \{z_i\}_{i=1}^N$. La figure 1 (resp 2) illustre ces données avec en abscisse les valeurs associées à x et en ordonnée les valeurs associées à y (resp z). Les points noirs sont les mesures, la ligne bleu est l'approximation que l'on souhaite calculer.

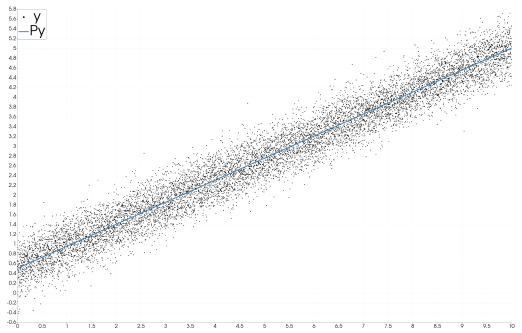


FIGURE 1 – Mesures y en fonction de x

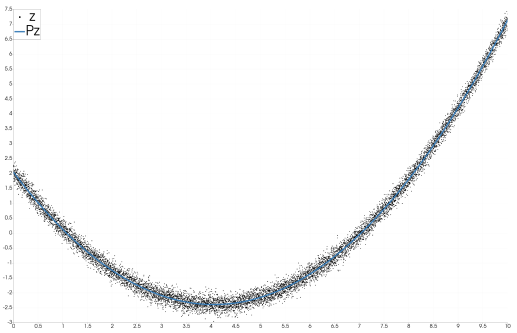


FIGURE 2 – Mesures z en fonction de x

a) Vous devez écrire une classe **Mesures** qui contiendra les valeurs lues depuis le fichier fourni. Cette classe disposera de 3 tableaux C++ contenant les données x, y et z. Ces tableaux seront de même taille égale à $N = 10000$ (correspondant au nombre de ligne du fichier sans la première ligne avec x, y, z. Le constructeur de cette classe prendra en paramètre une chaine de caractère représentant le nom du fichier à lire. Ainsi, lors de la construction de l'objet, les tableaux C++ seront initialisés par la lecture du fichier.

Une fois les données en mémoire, vous devrez calculer le coefficient de corrélation linéaire entre x et y (noté r(x,y)) puis entre x et z (noté r(x,z)).

$$r(x,y) = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{N}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{N}(y_i - \bar{y})^2}} \quad \text{avec } \bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i \text{ et } \bar{y} = \frac{1}{N}\sum_{i=1}^{N} y_i$$

Les valeurs attendues sont $r(x,y) \approx 0.973811$ et $r(x,z) \approx 0.58882$.

b) Nous allons commencer par analyser les mesures données par y à l'aide d'une approximation par moindre carré linéaire. L'objectif va être de trouver les coefficients du polynôme d'ordre 1 (une droite) décrit par $P_y(x) = a_o + a_1 x$ qui réalise la meilleure approximation au sens des moindres carrés. En d'autres termes, cette méthode consiste à trouver a_0 et a_1 qui vont minimiser la quantité suivante :

$$R(a_0, a_1) = \sum_{i=1}^{N} \left(y_i - (a_0 + a_1 x)\right)^2$$

Après application de la méthode mathématique, la détermination des coefficients a_0 et a_1 se ramène à la résolution d'un système linéaire A U = F, avec A une matrice 2×2, U et F des vecteurs de dimension 2. Ces matrices/vecteurs sont alors exprimés de la manière suivante :

$$A = \begin{pmatrix} N & \sum_{i=1}^{N} x_i \\ \sum_{i=1}^{N} x_i & \sum_{i=1}^{N} x_i^2 \end{pmatrix} \quad , \quad U = \begin{pmatrix} a_0 \\ a_1 \end{pmatrix} \quad , \quad F = \begin{pmatrix} \sum_{i=1}^{N} y_i \\ \sum_{i=1}^{N} x_i y_i \end{pmatrix}$$

Pour trouver les coefficients a_0 et a_1, il suffit d'inverser la matrice, c'est-à-dire de calculer $U = A^{-1}F$. Le calcul de l'inverse d'une matrice 2×2 est donnée en Annexe.

Une fois la méthodologie comprise, vous devez implémenter une classe nommé **Matrice2x2** qui définira les éléments suivants :

- des accesseurs en lecture
- des accesseurs en lecture/écriture
- une méthode calculant le déterminant
- une méthode qui retourne l'inverse de la matrice

A l'aide de cette classe, vous devez rajouter une méthode **analyse_y** dans la classe **Mesures** permettant de calculer les coefficients a_0 et a_1 en utilisant les mesures y_i. La méthode devra renvoyer les valeurs des 2 coefficients. Les valeurs attendues sont $a_0 \approx 0.51$ et $a_1 \approx 0.45$.

c) On s'intéresse maintenant aux mesures données par z. Visuellement, on peut voir que l'approximation par une droite affine ne sera pas bonne. On va alors chercher le polynôme d'ordre 2 approchant au mieux ces mesures au sens des moindres carrés. Cela revient à trouver les coefficients a_0, a_1 et a_2 du polynôme $P_z(x) = a_o + a_1 x + a_2 x^2$. Comme pour le cas affine, ces coefficients peuvent être déterminés en définissant un système $AU = F$, avec A une matrice 3×3, U et F des vecteurs de dimension 3. Ils s'expriment de la manière suivante :

$$A = \begin{pmatrix} N & S_1 & S_2 \\ S_1 & S_2 & S_3 \\ S_2 & S_3 & S_4 \end{pmatrix} \quad , \quad U = \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} \quad , \quad F = \begin{pmatrix} W_0 \\ W_1 \\ W_2 \end{pmatrix}$$

avec

$$S_k = \sum_{i=1}^{N} x_i^k \quad , \quad W_k = \sum_{i=1}^{N} z_i x_i^k$$

Comme pour la question précédente, il faudra implémenter une classe **Matrice3x3** et ajouter une méthode **analyse_z** dans la classe **Mesures** qui déterminera les coefficients. Les valeurs attendues sont $a_0 \approx 2.0475$ et $a_1 \approx -2.19$ et $a_2 \approx 0.27$.

d) Les mesures y et z ont été générées par un programme informatique en évaluant la quantité $y_i = P_y(x_i) + \epsilon_i$ (idem pour z). Le paramètre ϵ_i est un bruit généré aléatoirement à l'aide d'une loi normale. Ce générateur de nombre aléatoire est ainsi paramétrisé par l'écart type que l'on note σ_y (resp σ_z) pour les mesures y_i (resp z_i). Il est alors possible de retrouver les 2 valeurs σ_y et σ_z en utilisant les formules suivantes :

$$\sigma_y = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - P_y(x_i))^2} \quad \text{et} \quad \sigma_z = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(z_i - P_z(x_i))^2}$$

Implémenter le calcul de σ_y et σ_z, puis afficher les valeurs ainsi obtenues. Les valeurs attendues sont $\sigma_y \approx 0.3$ et $\sigma_z \approx 0.17$.

Exercice 2 (8 points)

Dans cet exercice, on s'intéresse à l'évaluation de l'indice de masse grasse (IMG), exprimé en pourcentage, permettant de juger de la proportion de tissus adipeux d'une personne adulte, en fonction de la taille, du poids, de l'âge et du sexe.

Cet indice s'exprime par la formule suivante :

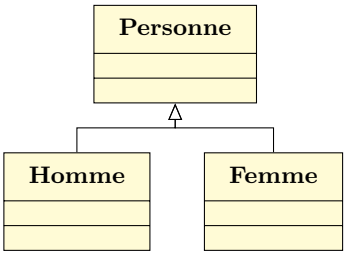
$$IMG = (1.2\frac{P}{T^2}) + (0.23\ A) - (10.8\ S) - 5.4$$

avec P le poids en kg, T la taille en m, A l'âge et S le sexe ($S = 0$ pour les femmes, et $S = 1$ pour les hommes).

a) Définir 3 classes nommées **Personne**, **Homme** et **Femme** en utilisant le diagramme d'héritage ci-contre. Les classes **Homme** et **Femme** fourniront un constructeur à 3 arguments :

- la taille (type **double**)
- le poids (type **double**)
- l'âge (type **int**)

La classe mère contiendra une méthode virtuelle pure que l'on nommera par **estHomme()** et renverra un booléen égale à vrai si l'objet est un homme et faux sinon.



Le code suivant devra pouvoir être exécuté :

```
1  Personne * homme = new Homme(1.70,85,33);
2  std::cout << "exemple homme : "<< *homme<< "\n";
3  Personne * femme = new Femme(1.70,85,33);
4  std::cout << "exemple femme : "<< *femme << "\n";
```

```
user $ ./myprog
exemple homme : [1] 26.6841 (1.7,85,33)
exemple femme : [0] 37.4841 (1.7,85,33)
```

avec le format d'affichage suivant : premier nombre entre crochet le type (homme ou femme), puis l'IMG, puis la taille, le poids et l'âge.

Consignes : Il faut utiliser les atouts de l'héritage en évitant de dupliquer du code C++.

b) Soit N un entier positif qui est donnée par l'utilisateur comme un argument de l'exécutable. On rappelle qu'il faut utiliser l'instruction **int N = std::stoi(c)** pour convertir une chaine de caractère c en un entier. Écrire ensuite une fonction qui permet d'initialiser une collection d'objet de type **Homme** ou **Femme** de taille N. Cette collection sera construite aléatoirement à l'aide d'une variable aléatoire valant 0 ou 1 pour le choix **Homme** ou **Femme**. Pour les autres paramètres définies aléatoirement, on utilisera les hypothèses suivantes :

	Taille [min - max]	Poids [min - max]	Age [min - max]
Homme	1.65 - 2.0	55 - 110	18 - 50
Femme	1.50 - 1.90	45 - 90	18 - 50

Une fois le tableau initialisé, vous afficherez le contenu de cette collection. Vous effectuerez également les calculs suivants en respectant les consignes décrites :

— Compter le nombre d'homme dans le tableau en utilisant la méthode **estHommme()**.

— Compter le nombre de femme dans le tableau en utilisant le transtypage dynamique.

c) La définition d'une fonction est fournie dans le fichier ex2_tri.hpp. Elle représente un algorithme de tri qui va permettre de trier le tableau initialisé précédemment en fonction de l'IMG. Vous devez inclure ce fichier dans votre programme (en recopiant le code ou en faisant un **#include "ex2_tri.hpp"** mais attention il faut le faire après la définition de la classe **Personne**). Une fois fait, vous devez appeler la fonction avec votre tableau en argument ainsi que sa taille. Le programme ne va probablement pas compiler, il faut alors comprendre le problème et corriger les classes qui ont été développées.

Consignes : Vous pouvez modifier le fichier fourni dans le but de comprendre le problème mais le programme doit ensuite fonctionner avec ce fichier non modifié.

Annexes

Matrice 2×2

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \quad \Longrightarrow \quad \det(A) = ad - bc \quad \text{et} \quad A^{-1} = \frac{1}{\det(A)} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$$

Matrice 3×3

$$A = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

$$\det(A) = aei + bfg + cdh - ceg - bdi - afh$$

$$A^{-1} = \frac{1}{\det(A)} \begin{pmatrix} ei - fh & ch - bi & bf - ce \\ fg - di & ai - cg & cd - af \\ dh - eg & bg - ah & ae - bd \end{pmatrix}$$