Programmation sous Matlab

Durée 1^{H30}.
Adresse pour récupérer les fichiers de travail : $/net/commun/HMEE104/$.

1 Restitution du travail

Pour restituer votre travail, vous devez placer tous les fichiers de vos programme dans un dossier **dont le titre est votre nom** sur le bureau de votre espace de travail – par exemple **DurandBruno**. Ce dossier doit contenir **au minimum** un fichier dont le titre est **examen104.m** contenant votre programme principal. En fin d'épreuve, vous sortez de la salle en laissant votre session ouverte. L'enseignant passera récupérer votre travail sur une clé USB et fermera votre session.

Dans votre intérêt, ce dossier ne doit contenir que les fichiers utiles.

Tous les documents sont autorisés, par contre les échanges quels qu'ils soient sont interdits. Vous pouvez utiliser des documents que vous auriez sur une clé USB, par contre vous ne pouvez charger ces documents que pendant la première demi-heure de l'épreuve. **Si une clé USB persiste sur votre ordinateur au bout d'une demi-heure, cela sera considéré comme une tentative de fraude**.

Attention !!! une heure limite de restitution de votre travail vous sera précisé en début de séance. Après cette heure, vous devez quitter la salle. Deux minutes de plus comptent pour un point de moins sur la note finale.

N'oubliez pas de commenter votre programme. La qualité de la présentation du programme comptera naturellement pour la note finale. N'oubliez pas que les majuscules et le minuscules sont différentes sous Matlab. Respectez les consignes. Pensez aussi à donner à vos variables des noms explicites et à ne fixer pratiquement aucune valeur (nombre d'échantillons des signaux par exemple). Pensez à tester chaque partie de votre programme avant de passer à la question suivante. Toutes les questions dépendent les unes des autres, et les difficulté sont croissantes. Bon travail.

2 But du travail

Le but de ce travail est de créer deux fonctions, l'une de chiffrement et l'autre de déchiffrement selon le principe de la permutation aléatoire. Le fichier à chiffrer est un fichier son que vous récupérerez à partir de l'onglet adéquat. C'est un signal codé sur 8 bits (donc chaque valeur est un entier entre 0 et 255).

3 Principe du chiffrement par permutation

Cette méthode de chiffrement est maintenant très peu utilisée car très facile à casser. Il a l'avantage ici de se programmer très facilement. Cela consiste à échanger le code de chaque échantillon par un autre code, rendant ainsi le signal complètement incohérent. Je donne ici un exemple simple basé sur un code à 2 bits. Dans ce cas les valeurs possibles du signal sont $[0, 1, 2, 3]$. Supposons que la table de permutation soit $[2, 0, 1, 3]$, dans ce cas le signal $\{0, 1, 2, 3, 2, 3, 2, 1, 0, 0\}$ devient $\{2, 0, 1, 3, 1, 3, 1, 0, 2, 2\}$.

4 Travail de programmation

Le travail qui vous est demandé est réalisé en quatre étapes.

1. Quantification du signal audio.
2. Création d'une table de permutation sans répétition.
3. Réalisation d'une fonction de chiffrement.
4. Réalisation d'une fonction de déchiffrement.

4.1 Quantification du signal audio

Le fichier que vous devez chiffrer porte le nom de *Whistle.wav*. Il est code sur 8 bits (donc a valeur dans $[0, 255]$). Cependant lorsque vous le chargez avec la fonction *audioread*, celle-ci le convertit automatiquement en un fichier son classique (donc a valeur dans $[-1, 1]$).

Votre premier travail va donc consister à convertir le signal audio à valeur réelle dans $[-1, 1]$ en un signal à valeur entière dans $[0, 255]$. Pour cela, il suffit d'appliquer une règle de proportion et transformer le codage en réel en un codage 8-bits.

Question 1
Dans votre programme principal, lisez le signal audio et affichez le dans une fenêtre en respectant bien l'axe des temps (servez vous de la fréquence d'échantillonnage). Vous intitulerez cette figure *Signal Audio Original*.

Question 2
Faites une fonction dont le prototype est *[Musique_8bits] = ConvertirReel8bits(Musique)* dont le rôle est de convertir le signal à valeur réelle $Musique$ en un signal à valeur entière codée sur 8 bits $Musique\_8bits$.

AIDE vous devez utiliser les fonction *floor* et *uint8*.

Question 3
Appelez votre fonction dans la suite de votre programme et, dans une deuxième fenêtre divisée en trois horizontalement, affichez votre signal codé sur 8 bits dans la partie supérieure de la fenêtre. Respectez l'axe des temps. Appelez cette figure *Experience de chiffrement*.

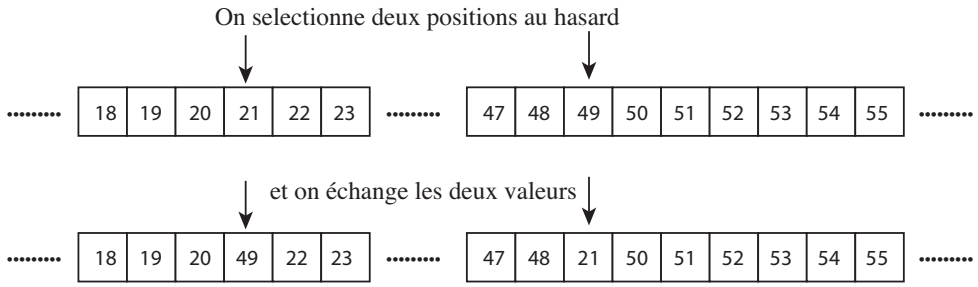

FIGURE 1 – Permutation de deux valeurs du vecteur pour obtenir la clé de chiffrement.

4.2 Création d'une table de permutation sans répétition

Pour créer une table de permutation sans répétition, une des méthodes les plus simples est de permuter aléatoirement un vecteur contenant les valeurs quantifiées. Dans notre cas, il s'agit de $2^8 = 256$ valeurs : $[0, 1, 2, \ldots, 255]$.

Question 4
Vous devez donc créer une fonction dont le prototype est :
CleDeChiffrement = CreationCleDeChiffrement(NombreDePermutation) ;
qui doit renvoyer une matrice 256×1 des éléments $[0, 1, 2, \ldots, 255]$ permutés.

Pour réaliser la permutation, vous devez choisir aléatoirement deux chiffres entre 0 et 255 et permuter leurs valeurs un certain nombre de fois. Ce nombre est fixé par la variable *NombreDePermutation* qui est l'argument d'entrée de votre fonction. Faites en sorte que, si la fonction est appelée sans argument, le nombre de permutation est fixé arbitrairement à 1000.

La Figure 1 est une illustration de la première permutation.

Appelez votre fonction dans votre programme principal.

4.3 Réalisation d'une fonction de chiffrement

La fonction de chiffrement consiste à remplacer chaque valeur du signal par son correspondant chiffré.

Question 5
Vous devez créer une fonction dont le prototype est :
[Musique_8bitsChiffre] = Chiffrement(Musique_8bits, CleDeChiffrement) ;
qui doit renvoyer un signal ayant subit la permutation présentée en début de ce document : chaque valeur du signal original est remplacé par la valeur correspondante dans la table de chiffrement. Par exemple, si le $578^{\text{ème}}$ échantillon du signal original vaut 21 et que la clé est celle présentée sur la Figure 1 alors le $578^{\text{ème}}$ échantillon du signal chiffré vaut 49.

Appelez votre fonction dans la suite de votre programme et, dans la deuxième fenêtre qui divisée en trois horizontalement, affichez votre signal chiffré dans la partie médiane de la fenêtre.

Question 6

Dans la suite du programme principal, transformez votre signal chiffre sur 8 bit $[0, 255]$ en un signal reel sur $[-1, 1]$ et enregistrez le sous le nom *WhistleChif.wav*. Vous joindrez ce fichier à votre compte rendu.

4.4 Réalisation d'une fonction de déchiffrement

La fonction de déchiffrement consiste à retrouver le signal original en utilisant la clé.

Question 7

Vous devez créer une fonction dont le prototype est :
[Musique_8bitsDechiffre] = Dechiffrement(Musique_8bitsChiffre, CleDeChiffrement) ;
qui doit renvoyer une matrice un signal ayant subit la permutation inverse. Pour réaliser cette permutation inverse, vous devez créer une clé de déchiffrement et faire exactement le même processus que pour le chiffrement.

La réalisation d'une clé de déchiffrement à partir de l'exemple d'un codage sur 2 bits vous est proposé ci-dessous.
Supposons que la clé de chiffrement soit $[2, 0, 1, 3]$, dans ce cas la clé de déchiffrement sera $[1, 2, 0, 3]$. On l'obtient en parcourant la clé de chiffrement : 2 est en position 0 donc je met 0 en position 2 dans la clé de déchiffrement ; 0 est en position 1 donc je met 1 en position 0 dans la clé de déchiffrement ...et ainsi de suite. **Attention** Matlab a des indices qui démarrent à 1 et non à 0

Si vous êtes malin, la réalisation de cette fonction est très simple.

Appelez votre fonction dans la suite de votre programme et, dans la deuxième fenêtre qui divisée en trois horizontalement, affichez votre signal chiffré dans la partie inférieure de la fenêtre.

Question 8

Calculez l'écart quadratique entre le signal original et le signal déchiffré et affichez cette valeur en fin de programme sous la forme : *Apres dechiffrement, l'ecart entre le signal dechiffre et le signal original est de ....*