EBOULE Erwann
FREMAUX Julien

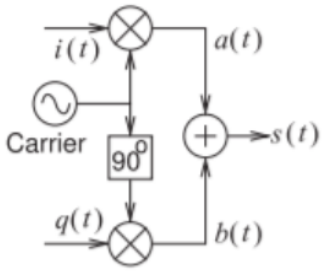
PROJET - Modulations Numériques : Modulation 8-PSK d'une chaîne de caractère

CNUM

2.1 Partie Théorique

Question n°1 :

La modulation à M phase peut être modélisée par la schéma-bloc suivant :



On a les signaux i(t) et q(t) de même amplitudes qui vont être déphasés de pi/, possèdent une phase phi.

D'où : $i(t) = cos(\varphi(t))$

$q(t) = sin(\varphi(t))$

On a alors : $a(t) = cos(\varphi(t)) \times cos(2\pi ft)$

$= \frac{1}{2}[cos(2\pi ft - \varphi(t)) + cos(2\pi ft + \varphi(t))]$

$b(t) = sin(\varphi(t)) \times (-sin(2\pi ft))$

$= -\frac{1}{2}[cos(2\pi ft - \varphi(t)) - cos(2\pi ft + \varphi(t))]$

Donc : $xmod(t) = cos(2\pi ft + \varphi(t))$

Question n°2 :

$\varphi_k \in \left\{\pm\frac{\pi}{M} \pm\frac{3\pi}{M} \dots \pm\frac{(M-1)\pi}{M}\right\}$

Question n°3 :

On multiplie le signal modulé par la porteuse et on cherche l'équivalent en bande de base afin de démoduler notre signal :

$xmod(t) \times cos(2\pi ft) = \frac{1}{2}[cos(4\pi ft + \varphi(t)) + cos(\varphi(t))]$

Notre calcul n'a pas abouti…

Question n°4 :

BPSK : Ts = Tb

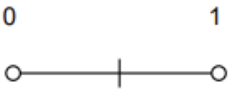
QPSK : Ts = 2Tb

8-PSK : Ts = 3Tb

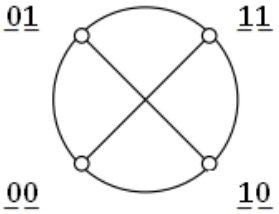
QAM16 : Ts = 4Tb

Question n°5 :

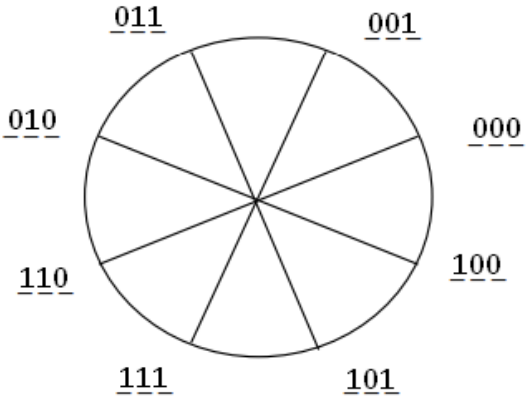
BPSK :



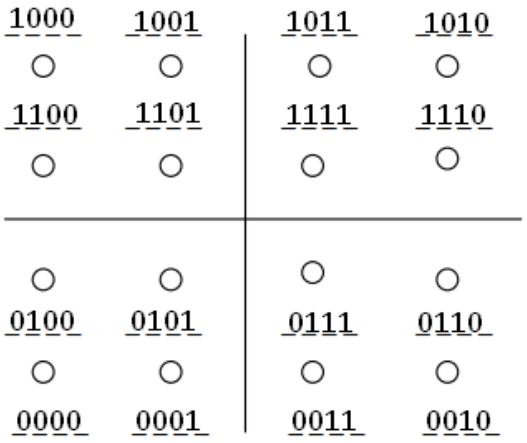
QPSK :



8PSK :



QAM16 :



2.2 Partie pratique

2.2.1 Question n°1 : Manipulation de chaîne de caractères et d'entiers

On transforme une chaîne de caractère en un train binaire en utilisant la fonction char2bit, puis on complète ce dernier avec 2 zéros afin de pouvoir avoir un train binaire contenant une multiple de 3 de valeurs :

```
%Transformation d'une chaîne de caractères en une séquence binaire
c = 'Projet CNUM : modulations numériques : PSK 8';
c_send = char2bit(c);
c_send = [c_send, 0];
c_send = [c_send, 0];
```

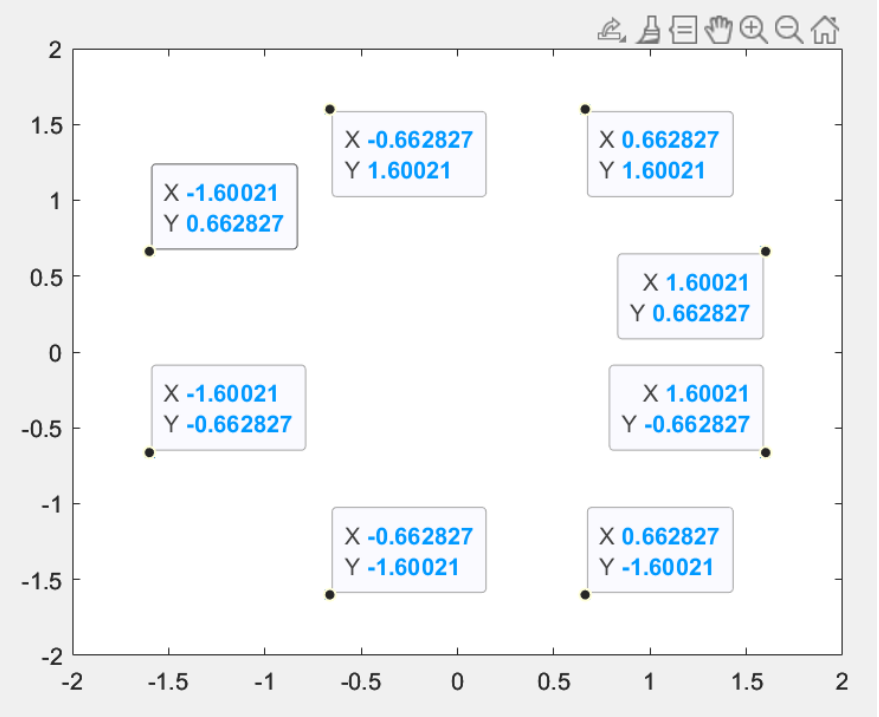
2.2.2 Question n°2 : Mapping 8-PSK

On commence par créer le cardinal de symbole, puis on vient ensuite créer un vecteur S contenant les différents symboles complexes après mapping des bits. Pour finir, on trace la constellation :

```
%Création d'un cardinal de symbole (avec racine de 3 afin d'obtenir des
%bits d'énergie égale à 1
symboles=sqrt(3)*[exp(j*pi/8);exp(j*3*pi/8);exp(j*7*pi/8);
    exp(j*5*pi/8);exp(j*15*pi/8);exp(j*13*pi/8);exp(j*9*pi/8);exp(j*11*pi/8)];

%Mapping des bits
A1=c_send(1:3:length(c_send));
A2=c_send(2:3:length(c_send));
A3=c_send(3:3:length(c_send));
S=symboles(4*A1+2*A2+A3+1);

%Traçage de la constellation
%plot(S, 'x');
```



Après toutes ces manipulations, on vient sur-échantillonner chaque symbole par un facteur 4, c'est-à dire que l'on vient ajouter 3 zéros entre chaque symbole complexe :

```
%Insertion des 3 zéros
S2=[];
for i = 1:118
    S2 = [S2; S(i); 0; 0; 0];
end
```

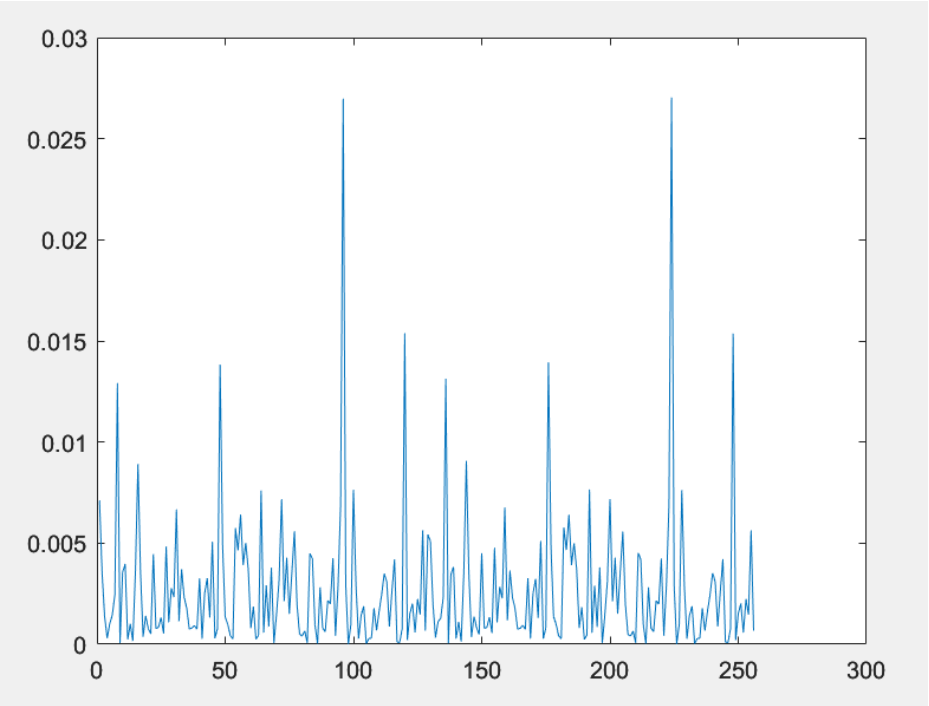
Pour finir, on vient appliquer un filtre $h(t) = \left\{\frac{1}{\sqrt{4}}\right\}$, $t \in \{1, 2, 3, 4\}$ sinon 0 :

```
%Application du filtre de mise en forme
b = [1/sqrt(4) 1/sqrt(4) 1/sqrt(4) 1/sqrt(4)];
filtered_signal = filter(b, 1, S2);
```

2.2.3 Question n°3 : Spectre

On vient tracer le spectre du signal sur-échantillonné que l'on interpole avec un zéro padding :

```
%Traçage de la densité spectrale de puissance
a= [S2; 0; 0; 0; 0; 0; 0; 0; 0]; %On ajoute 8 0 afin de pouvoir faire un zero padding
dsp = psd(a);
plot(dsp);
```



2.2.4 Question n°4 : Filtrage adapté

Ici, on remarque le filtre adapté au signal se révèle être le filtre $h(t)$ ainsi on utilise un code similaire :

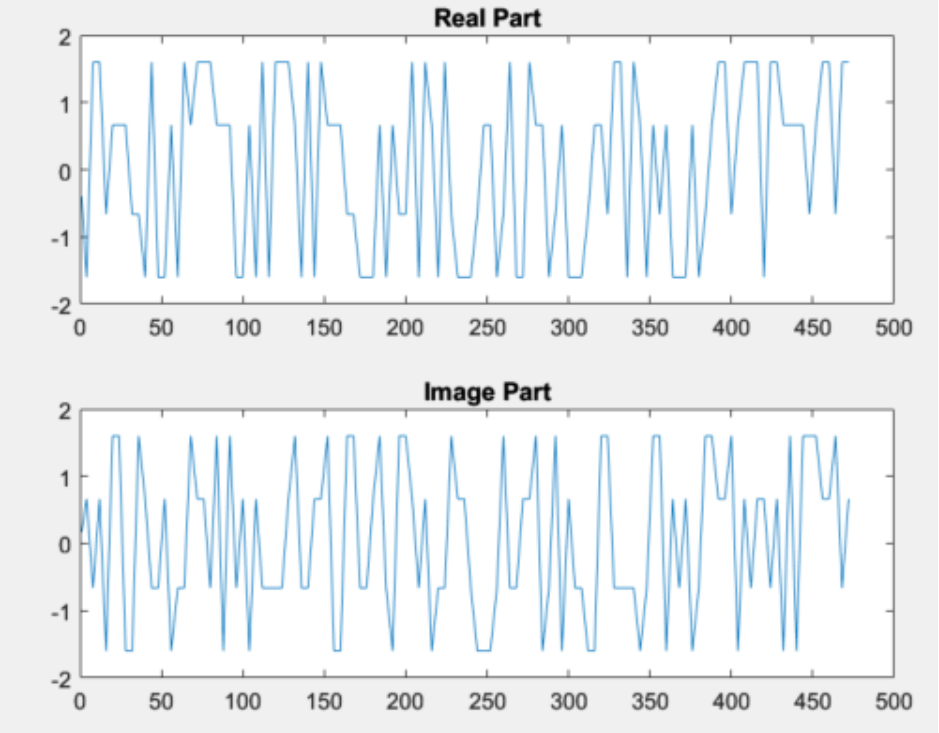
```
%Application du filtre adapté
adapted_filtered_signal = filter(b, 1, filtered_signal);
```

2.2.5 Question n°5 : Synchronisation échantillon

On commence par afficher la partie réelle et imaginaire de notre signal sur-échantillonné et filtré :

```
%Observation des parties réelles et imaginaires du signal
subplot(2,1,1);
real_part = real(adapted_filtered_signal);
plot(real_part);
title('Real Part')

subplot(2,1,2);
image_part = imag(adapted_filtered_signal);
plot(image_part);
title('Image Part')
```

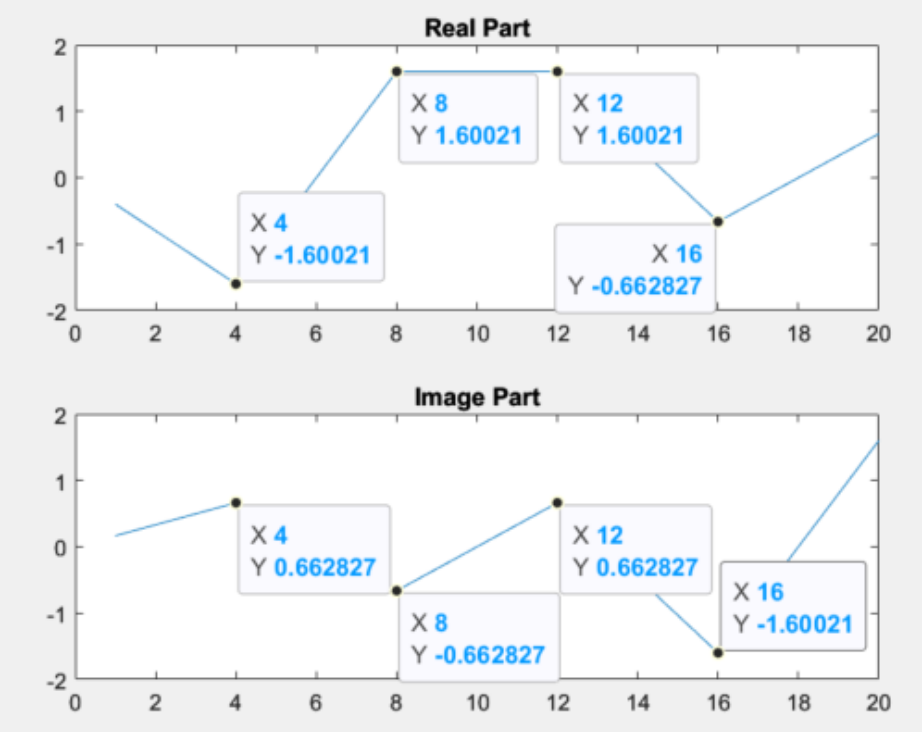


On remarque que le premier symbole arrive pour $x = 4$ donc on en conclut qu'il y a un décalage de 4.

On vient ensuite tracer les 20 premiers échantillons du signal :

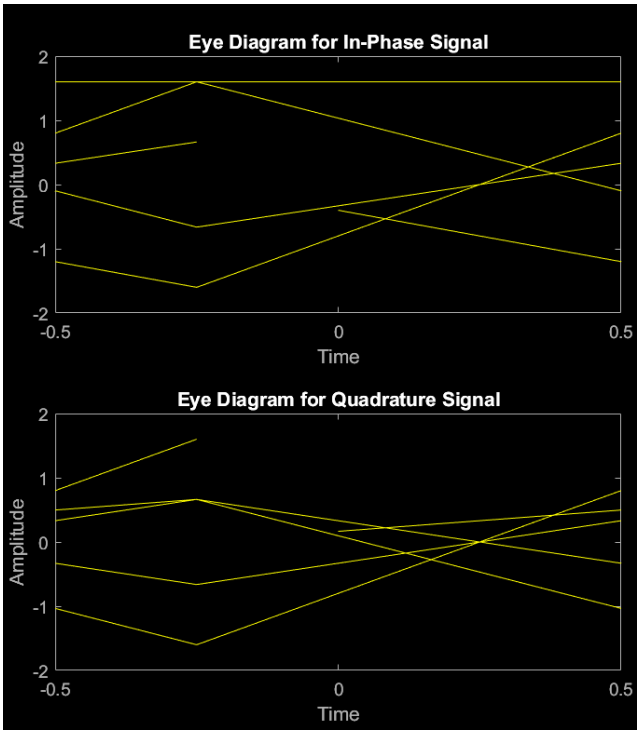
```
%Observation des parties réelles et imaginaires des 20 premiers échantillon du signal
echantillon = adapted_filtered_signal(1:1:20);
subplot(2,1,1);
real_part2 = real(echantillon);
plot(real_part2);
title('Real Part')

subplot(2,1,2);
image_part2 = imag(echantillon);
plot(image_part2);
title('Image Part')
eyediagram(echantillon, 4);
```



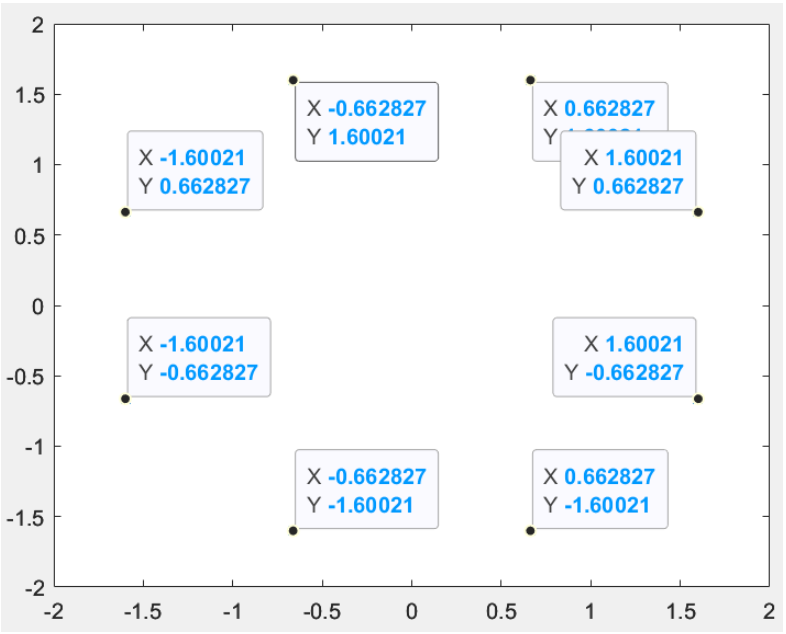
On constate bien que les différents symboles apparaissent avec un pas de 4

Si on trace le diagramme de l'oeil à partir de ce signal on obtient :



De plus, après le sous-échantillonnage du signal filtré on trace la constellation 8-PSK correspondante :

```
t = adapted_filtered_signal(4:4:length(adapted_filtered_signal));
plot(t, "x")
```



2.2.6 Question n°6 : Prise de décision

A la suite de ce sous-échantillonnage du signal on effectue la prise de décision, c'est-à dire que l'on vient déterminer les symboles les plus probablement émis (avec l'utilisation de différents seuils). Ensuite, on reforme le train binaire correspondant :

```
%Prise de décision et reformation du train binaire
Symboles_recup = [];
for i=1:length(t)
    if(real(t(i)) > 0 && real(t(i)) <= 1.2 && imag(t(i)) < 1.8 && imag(t(i)) >= 1.2)
        Symboles_recup = [Symboles_recup, 0, 0, 1];

    elseif(real(t(i)) > 1.2 && real(t(i)) <= 1.8 && imag(t(i)) < 1.2 && imag(t(i)) >= 0)
        Symboles_recup = [Symboles_recup, 0, 0, 0];

    elseif(real(t(i)) > 1.2 && real(t(i)) <= 1.8 && imag(t(i)) < 0 && imag(t(i)) >= -1.2)
        Symboles_recup = [Symboles_recup, 1, 0, 0];

    elseif(real(t(i)) > 0 && real(t(i)) <= 1.2 && imag(t(i)) < -1.2 && imag(t(i)) >= -1.8)
        Symboles_recup = [Symboles_recup, 1, 0, 1];

    elseif(real(t(i)) > -1.2 && real(t(i)) <= 0 && imag(t(i)) < -1.2 && imag(t(i)) >= -1.8)
        Symboles_recup = [Symboles_recup, 1, 1, 1];

    elseif(real(t(i)) > -1.8 && real(t(i)) <= -1.2 && imag(t(i)) < 0 && imag(t(i)) >= -1.2)
        Symboles_recup = [Symboles_recup, 1, 1, 0];

    elseif(real(t(i)) > -1.8 && real(t(i)) <= -1.2 && imag(t(i)) < 1.2 && imag(t(i)) >= 0)
        Symboles_recup = [Symboles_recup, 0, 1, 0];

    else
        Symboles_recup = [Symboles_recup, 0, 1, 1];
    end
end
```

2.2.7 Question n°7 : Assemblage de l'ensemble

A cette étape on assemble l'ensemble du code que nous avons créé dans les questions précédentes. A cela, on vient ajouter la suppression des 2 zéros que nous avions ajoutés au tout début du projet afin de pouvoir retrouver la trame binaire que nous avons transmis. Pour finir, on transpose la trame binaire en une chaîne de caractères à l'aide de la fonction bit2char :

```
%Suppresion des 2 zéros qui ont été ajouté à la trame binaire au début du
%projet
Symboles_recup([length(Symboles_recup), length(Symboles_recup)-1]) = [];

%Reconstitution de la chaîne de caractères
char_recu = bit2char(Symboles_recup);
```

Lorsque nous exécutons le code lorsqu'il est complété et que l'on cherche à obtenir la valeur de la chaîne de caractère reçue, on obtient ceci :

```
char_recu =

    'Projet CNUM : modulations numériques : PSK 8'
```

Soit exactement la même trame qui a été transmise au début du projet.

2.2.8 Question n°8 : Ajout du bruit

A cette étape on vient ajouter un bruit gaussien à notre signal en sortie du filtre adapté :

```
%Ajout d'un bruit blanc gaussien
n = gauss(0, 0.5, 2000);
adapted_filtered_signal = adapted_filtered_signal + n;
```

Lorsque l'on relance le programme on remarque que nous obtenons cette fois ci une chaîne de caractères illisibles :

```
char_recu =

    'Mòoiås-³NU<Û9¦modumçsiono&numémçques-°ÛMÓK-¶Ûu·Km¶Ûm¶Ûm¶Ûm¶Ém¶Ém¶Ën¶Ëm¶ËmÖËm¶Ëm¶[e¶
```

On peut donc en conclure que notre signal est assez sensible au bruit, une synchronisation plus précise pourrait certainement réduire cette sensibilité au bruit.

Etant donné que nous avons pris un bruit de variance $\sigma^2 = 0,5 = \frac{N_0}{2} \Leftrightarrow N_0 = 1$

Nous avons également la connaissance de l'énergie par bit, en effet nous avons moduler le signal de tel sorte à ce que l'énergie d'un bit soit égale à 1.

Ainsi on peut dire : $\frac{E_b}{N_0} = 1$