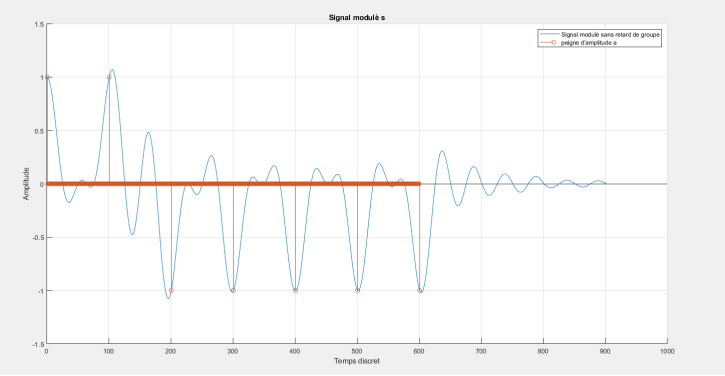
Filtrage Numérique
Transmission de signal, modulation MIA



Léa Yeromonahos

Table des matières

Modulation MIA en bande de base

Première étape : filtre d'émission

L'objectif est de simuler une modulation/démodulation numérique échantillonnée à deux états. Les amplitudes (ou symboles) possibles peuvent prendre les valeur +1 et -1. On choisit un facteur d'échantillonnage $K = 100$, autrement dit l'intervalle de temps correspondant à la période symbole est représenté par 100 échantillons.

Le filtre d'émission choisit est un RIF passe-bas obtenu par troncature de la réponse impulsionnelle (RI) sans fenêtre. On limite la longueur de la RI à ± 3 périodes symboles (soient $\pm$3K échantillons).
Afin de simplifier la démodulation, on réalise un filtre respectant la condition de Nyquist, c'est-à-dire :

$$h[pK] = \delta[p],\ p\{-3,3\} \tag{1}$$

Ainsi, $h[0]$ vaudra 1, et h vaudra 0 aux autres indices d'échantillons multiples de K.
Pour déterminer la fréquence de coupure f_c du filtre, on se réfère à la condition de Nyquist telle que :

$$f_c \geq 2f_{max} \tag{2}$$

Or ici notre fréquence maximum est égale à $\frac{1}{K}$, car nous avons des périodes de K échantillons. Notre fréquence de coupure f_c sera donc de $\frac{2}{K} = 0.02$ Hz.
On peut maintenant générer la RI, de la forme d'un sinus cardinal de fréquence f_c et de longueur $[-3K, 3K]$, que l'on troncature afin de la rendre causale :

```
K = 100;
N = 3*K;
n = (-N:N)';

fmax = 1/K;
fc = 2*fmax;

h = sinc(2*fc*(n - 2*n));
```

On définit d'abord notre période K, notre limite N ($\pm$3K), puis notre intervalle n allant de $-3K$ à $3K$. Une fois les fréquences f_{max} et f_c déterminées, on crée la RI h. La troncature s'effectue dans la définition de h : la RI commence à $-3K$, décalée afin de commencer à 0 et ainsi être causale.

La causalité a un coût : un retard de groupe. On peut le mesurer avec la fonction *grpdelay()* pour le compenser plus tard :

```
k = 500;
delay = grpdelay(h, 1, k);
delay = mean(abs(delay));
```

Le retard de groupe est calculé à partir de la RI, sur k échantillons (ici k est arbitrairement choisi à 500).
Le résultat obtenu est un vecteur de 500 échantillons, dont on prend la valeur moyenne.
Pour ce filtre d'émission, on obtient un retard de groupe de 300 échantillons.

Pour vérifier les caractéristiques de notre filtre, on affiche sa réponse impulsionnelle et sa réponse en fréquence :

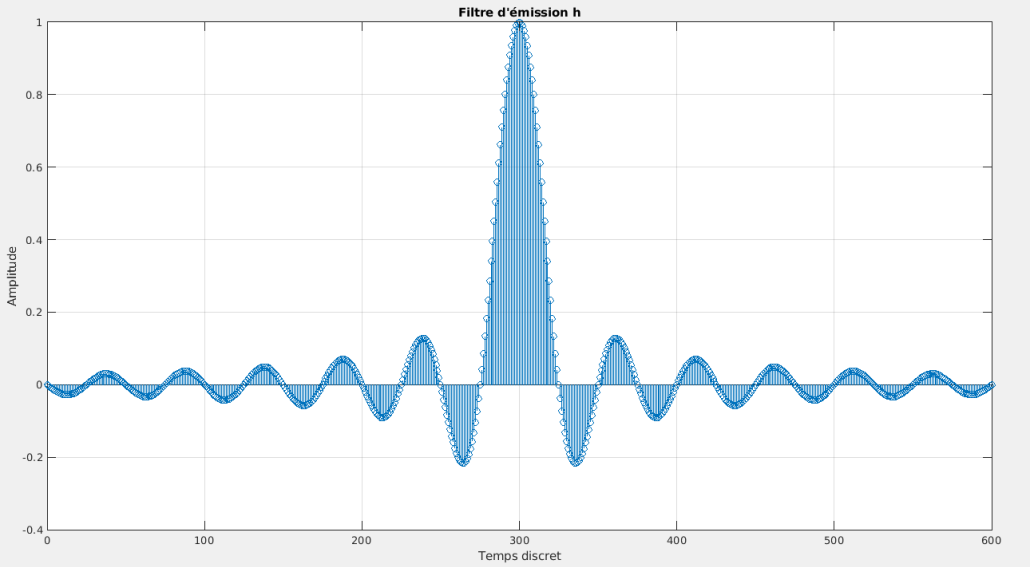


FIGURE 1 – Réponse impulsionnelle du filtre d'émission h

Comme attendu, la RI vaut 1 à $h[0]$ (rendu causal donc ici $h[0] = 300$), et 0 à tous les autres indices multiples de K (par exemple à $K = 100$ échantillons, à $2K = 200$ échantillons, etc...). Elle est comprise entre $-3K$ et $3K$ échantillons, décalés pour la causalité, soit entre 0 et $2 \times 3K$ échantillons.

Pour la réponse en fréquence, on s'attend à observer un filtre passe-bas de fréquence de coupure $f_c = 0.02$ Hz.
On la calcule ainsi :

```
[H, w] = freqz(h, 1, k);
tftd_h = 20*log10(abs(H));
```

La fonction *freqz(B, A, k)* permet de retourner la réponse en fréquence du filtre de coefficients B et A sous forme de vecteur H, avec les fréquences correspondantes w. Pour représenter le module, récupère ensuite la valeur absolue de cette réponse en fréquence, que l'on passe en dB.
On obtient donc :

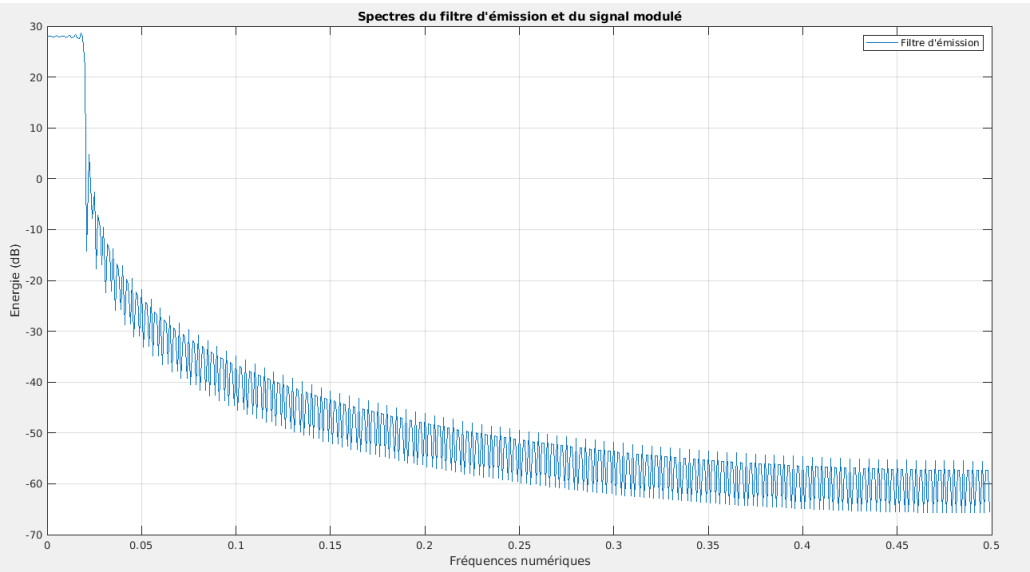


FIGURE 2 – Réponse en fréquence du filtre d'émission h

Le filtre obtenu est bien un passe-bas de fréquence de coupure $f_c = 0.02$ Hz, descendant assez rapidement à -30 dB. L'amplitude n'est pas à 0 dB, mais ce n'est pas problématique dans notre cas, car ce n'est pas ce que l'on recherche.
La largeur théorique de la bande occupée par la modulation est ici de 0.02 Hz.

Deuxième étape : séquence d'amplitude

Le but de la deuxième étape est de créer une séquence d'amplitudes aléatoires de longueur variable.
Pour cela, on crée une séquence de valeurs aléatoires de 100 échantillons avec la fonction *randn()*. On on trie ensuite les valeurs, qui vaudront -1 si elles sont négatives, 1 si elles sont positives, et 0 si elles sont nulles. On s'aide pour cela de la fonction *sign()*.
Enfin, pour tous les multiples de K, la séquence d'amplitude a sera non nulle. On va interpoler notre séquence par des 0, soit K-1 échantillons à 0 entre chaque échantillons de r. On utilise pour cela la fonction *upsample()* :

```
r = randn(100, 1);
a = upsample(r, K);
```

a, notre signal de séquence d'amplitude, est donc un vecteur de valeurs aléatoires valant -1, 1 et 0. On le représente ci-dessous :

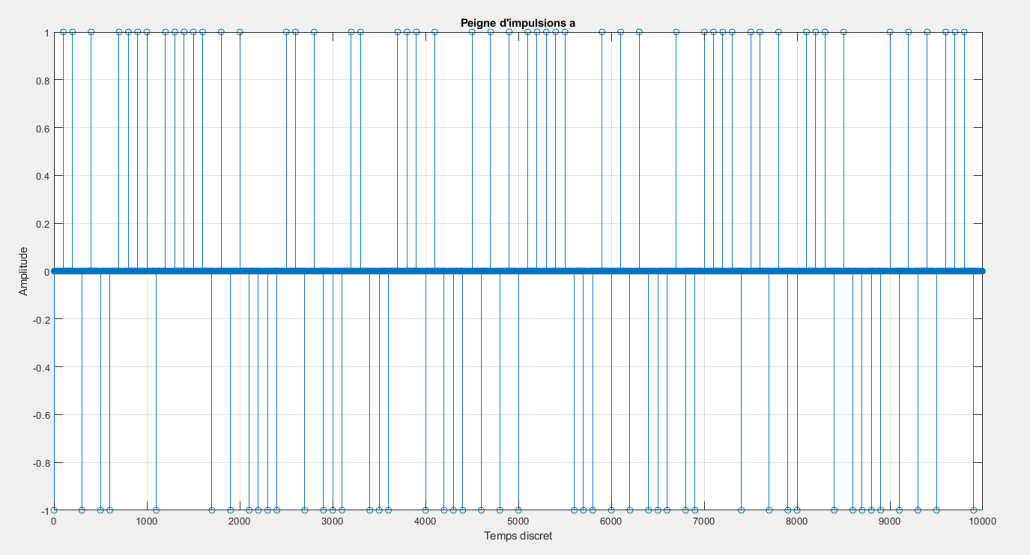


FIGURE 3 – Signal d'amplitudes a

On observe des amplitudes valant -1 ou 1 aux multiples de K, et 0 ailleurs, comme attendu. Les amplitudes sont non-nulles pour tout facteur de K.

Troisième étape : signal modulé

Pour moduler le signal d'amplitude, on réalise un filtrage de a par le filtre d'émission h.
Afin d'obtenir une meilleure résolution du signal émis, on réalise un zéro-padding sur h et a. Le zéro-padding consiste à rajouter des 0 en fin de signal, augmentant ainsi sa bande-passante sans modifier le signal :

```
h = [h; zeros(length(h), 1)];
a = [a; zeros(length(a), 1)];
s = filter(h, 1, a);
```

Pour le zéro-padding, on double la longueur des signaux avec un vecteur de 0.
Le signal modulé *s* est le résultat de la fonction *filter(B, A, x)* qui filtre le signal *x* par un filtre de coefficients *B* et *A*.
Afin de vérifier que le signal modulé suive bien les amplitudes, on enlève le retard de groupe :

```
sb = s;
sb(1:delay) = [];
```

où *sb* est le signal modulé sans le retard de groupe. On obtient donc :

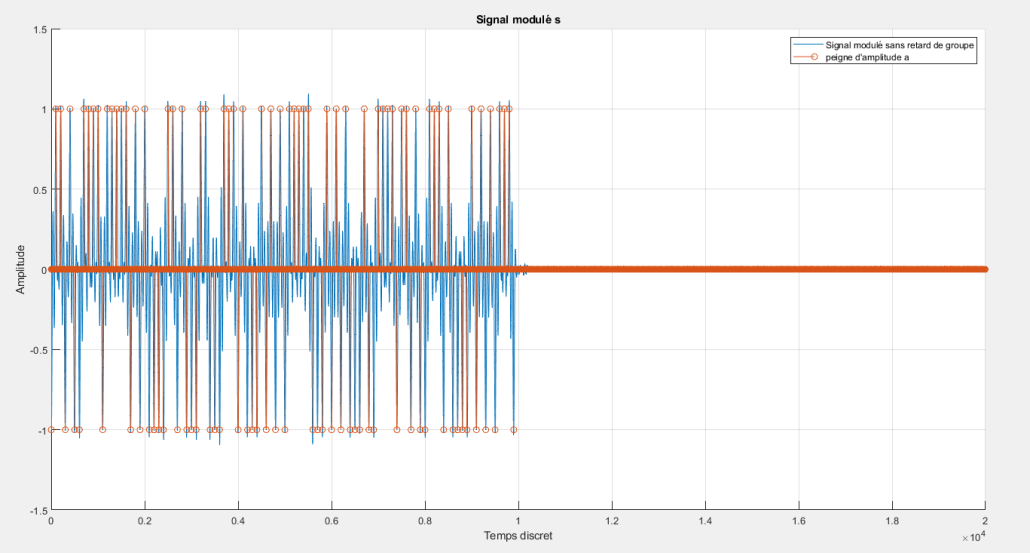


FIGURE 4 – Signal modulé *s* et séquence d'amplitudes *a*

Après compensation du retard de groupe, il apparaît que le signal modulé *s* (en bleu) suit fidèlement la séquence d'amplitudes *a*.
Pour le spectre de *s*, on obtient :

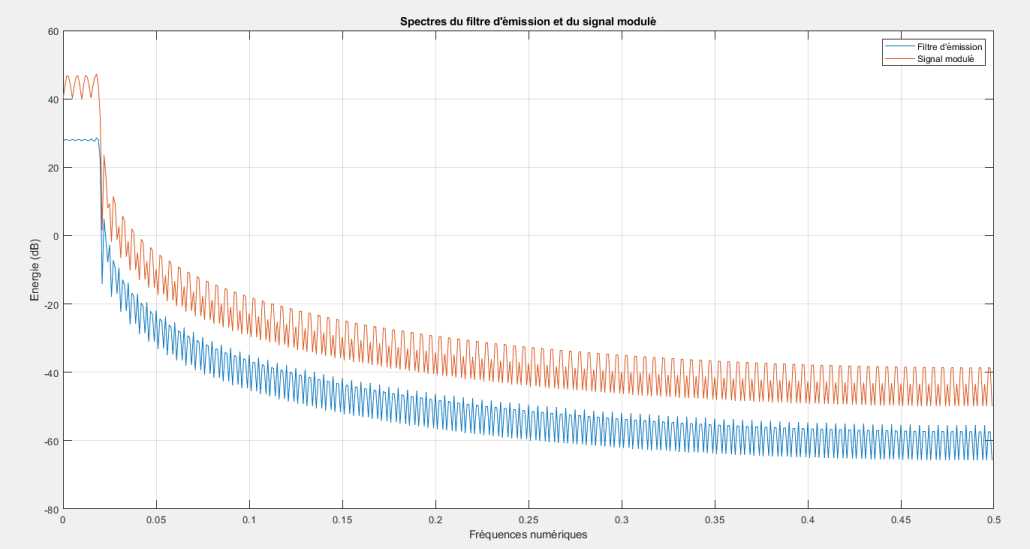


FIGURE 5 – Spectres du filtre d'émission h et du signal modulé s

Le spectre du signal modulé suit bien celui du filtre d'émission. L'énergie spectrale de s dépend entièrement du filtre d'émission, ainsi il est normal que le gain ne soit ni à 0 dB, ni égal à celui de h, car l'énergie de a est constante et non égale à 1. C'est le résultat du filtrage entre h et s.

Quatrième étape : démodulation

On veut maintenant récupérer les amplitudes de départ à partir de notre signal modulé. Pour cela, on le décime par K, puis on arrondit les résultats à -1 pour les valeurs négatives et 1 pour les positives, afin de rendre le filtre plus robuste :

```
A = downsample(sb, K);
```

A est notre séquence d'amplitudes que l'on a récupéré du signal sb (le signal modulé sans retard de groupe), en le sous-échantillonnant d'un facteur K grâce à la fonction *downsample()*. Autrement dit, on garde toutes les amplitudes multiples de K. On obtient la séquence suivante :

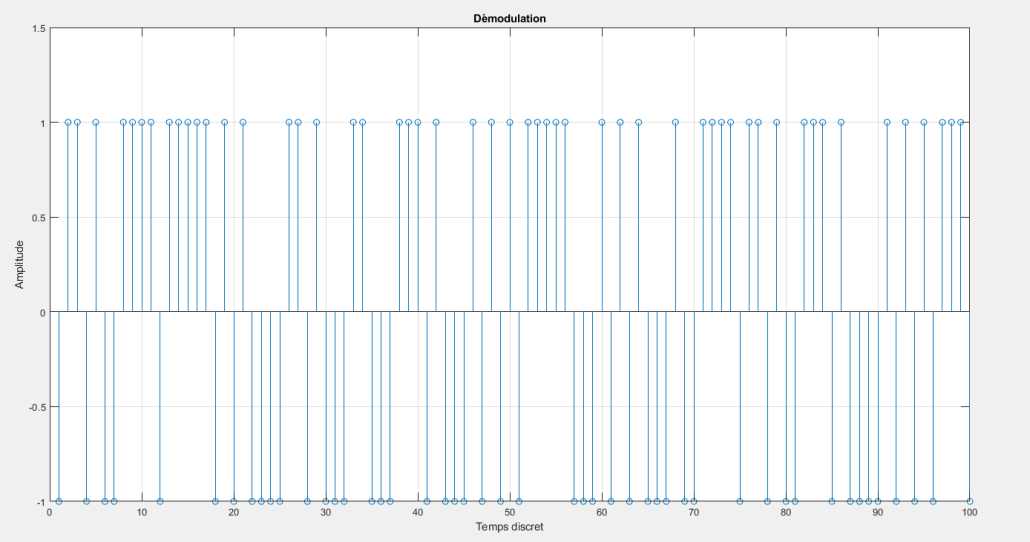
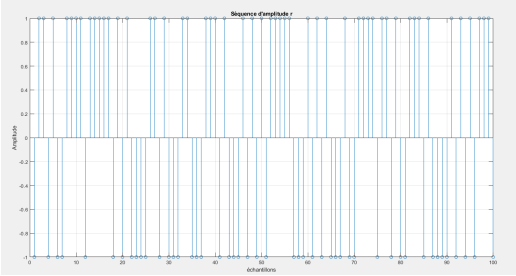
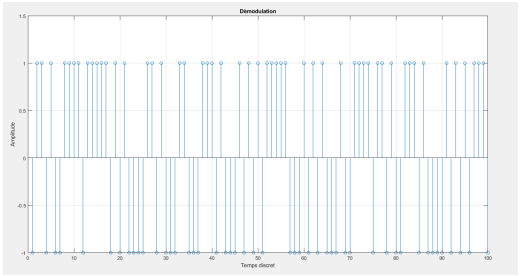


FIGURE 6 – Démodulation du signal sb

A première vue, en comparant visuellement la séquence d'amplitude avec A :



(a) Séquence d'amplitude r



(b) Séquence d'amplitude retrouvée par démodulation A

Les séquences semblent identiques. Toutefois, afin d'en être rigoureusement sûr, on calcule le taux d'erreur entre la séquence initiale r et la démodulée A :

```
seuilErreur = 10^-15;
e = abs(A-r);
nbErr = 0;
for i = 1:length(e)
    if e(i) < seuilErreur
        e(i) = 0;
        nbErr = nbErr;
    else
        e(i) = e(i);
        nbErr = nbErr + 1;
    end
end

tauxErreur = nbErr/(length(e))*100;
```

Pour une précision arbitrairement choisie à 10^{-15}, on calcule e l'erreur entre A et r, puis on calcule le taux en %.
Dans notre cas, le taux est à 0%, soit une transmission parfaite. C'était le résultat attendu, dans le sens où aucun bruit ne perturbait la transmission. Dans le cas contraire, le taux d'erreur aurait indiqué un problème de programmation.

Modulation MIA sur fréquence porteuse

Maintenant que l'on sait transmettre un signal, on voudrait pouvoir réaliser une transmission à plus longue portée. On va pour cela ajouter un signal porteur à notre signal en bande de base (soit le filtre d'émission h et le signal d'amplitude a), sous la forme d'un cosinus de fréquence $\nu_p = 0.2$ Hz.
En pratique, on multiplie la porteuse au signal en bande de base :

```
longueurSequence = 100;
np = (-longueurSequence^2:longueurSequence^2-1)';

port = cos(2*pi*n0*np);

sa1 = filter(h, 1, a);

sA1 = sa1 .* port;
```

Afin d'avoir un signal porteur couvrant l'intégralité des informations de notre signal en bande de base, on crée un vecteur suffisamment grand (où la longueur de la séquence est celle des amplitudes), on crée la porteuse *port*, puis on la multiplie au signal $sA1$ le signal modulé.
Pour vérifier nos calculs, on affiche le spectre du signal modulé :

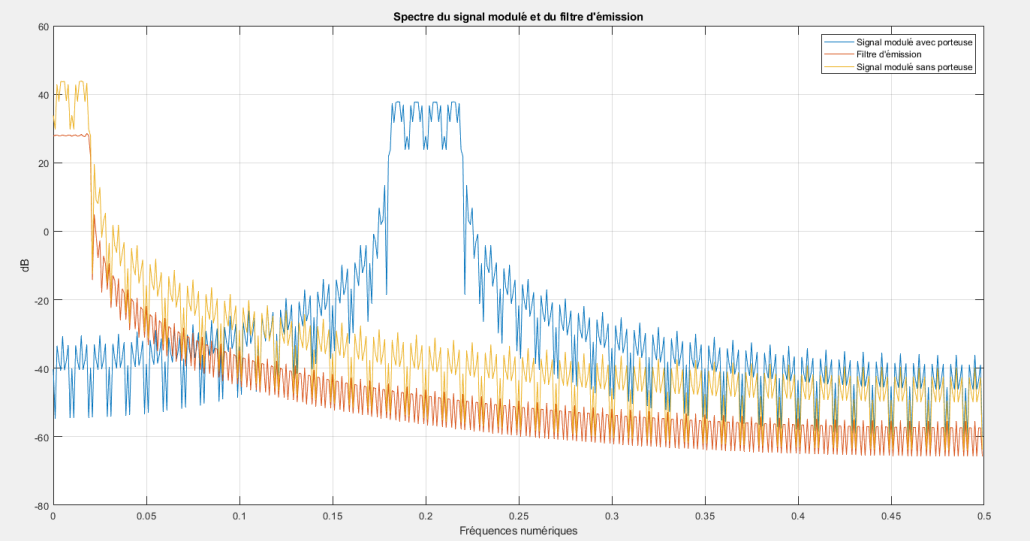


FIGURE 8 – Spectres du signal modulé avec porteuse, sans porteuse, et du filtre d'émission

Le spectre du signal modulé avec porteuse (en bleu) est à la fréquence 0.2 Hz, soit celle de la porteuse, contrairement au spectre du signal modulé sans porteuse (en jaune) qui lui est bien à la fréquence du filtre d'émission 0.02 Hz (dont le spectre est aussi affiché, en rouge).

Afin de simuler une transmission réaliste, on ajoute deux canaux supplémentaires à celui de base, de fréquences $\nu_p - \delta_\nu$ et $\nu_p + \delta_\nu$. On aura donc 3 canaux, dont celui de base sera au centre, de fréquence ν_p. Ils sont séparés par δ_ν, appelé "bande de garde".
Pour implémenter les nouveaux canaux, on va créer deux nouvelles séquences d'amplitudes indépendantes de la première, les moduler avec leur propre porteuse, et enfin additionner ces nouveaux signaux à celui de base :

```
dn = 0.05; %bande de garde

h = sinc(2*fc*(n - 2*n)); %filtre d'emission tronque
port = cos(2*pi*n0*np);
port2 = cos(2*pi*(n0+dn)*np);
port3 = cos(2*pi*(n0-dn)*np);
```

```
delay = grpdelay(h, 1, k); %calcul du retard de groupe
delay = mean(abs(delay));

r1 = sign(randn(longueurSequence, 1)); %generation sequences aleatoires
r2 = sign(randn(longueurSequence, 1));
r3 = sign(randn(longueurSequence, 1));

a = upsample(r1, K);
b = upsample(r2, K);
c = upsample(r3, K);

h = [h; zeros(length(h)-1, 1)]; %zero-padding
a = [a; zeros(length(a), 1)];
b = [b; zeros(length(b), 1)];
c = [c; zeros(length(c), 1)];

sa1 = filter(h, 1, a); %signal module h et a
sA1 = sa1 .* port; %on multiplie a la porteuse

sb1 = filter(h, 1, b); %signal module h et b
sB1 = sb1 .* port2; %on multiplie a la porteuse

sc1 = filter(h, 1, c); %signal module h et c
sC1 = sc1 .* port3; %on multiplie a la porteuse

sFinal = sA1 + sB1 + sC1;
```

Comme précédemment, on crée les séquences d'amplitudes $r1$, $r2$ et $r3$, que l'on interpole de $K-1$ échantillons (sur-échantillonnage). On réalise ensuite le zéro-padding pour chaque signal, que l'on module individuellement avec leur porteuse respective et le filtre d'émission h.
Enfin, on crée le signal final *sFinal* comme la somme des 3 signaux modulés. Pour vérifier nos calculs, on affiche le spectre de ce nouveau signal :

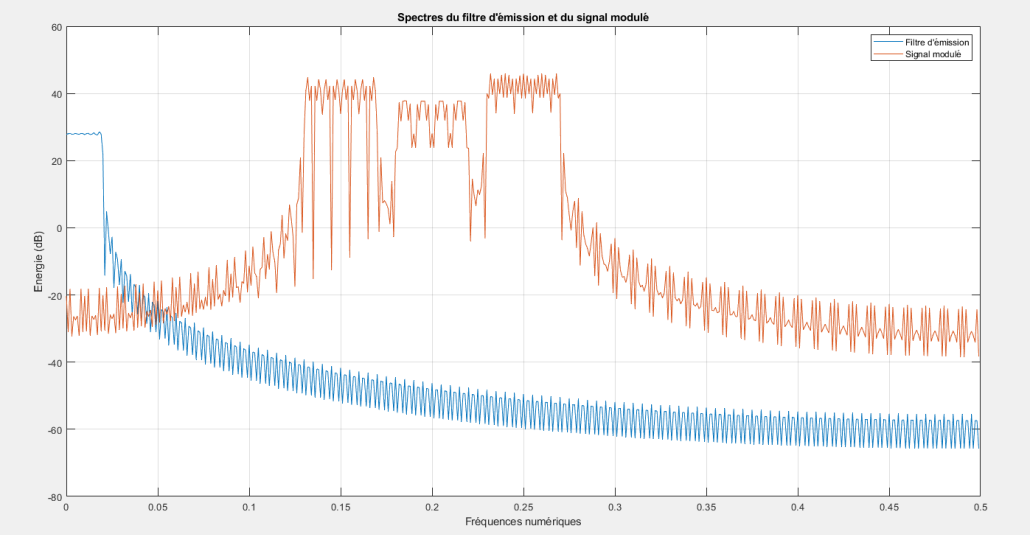


FIGURE 9 – Spectre du signal final modulé

On observe bien 3 canaux, centrés à la fréquence 0.2 Hz et séparés de δ_ν, soit ± 0.05 Hz.
Pour la démodulation, on ne s'intéressera qu'au canal central (notre premier signal).

La démodulation avec porteuse s'effectue en multipliant le signal modulé par la porteuse une nouvelle fois, afin de décaler le spectre en fréquence, puis ensuite d'appliquer un filtre passe-bas afin de ne récupérer que la première partie du nouveau spectre. C'est le processus de la démodulation cohérente.

L'utilisation de cette méthode se justifie par la quasi-impossibilité de récupérer les amplitudes de départ par simple décimation, à cause des erreurs de calages. Ces dernières nécessitent l'application d'un filtre passe-bas **avant** la décimation, donc un filtre avec une pente très raide et d'une bande passante très étroite, ce qui est en pratique quasiment irréalisable.

On applique donc la méthode de démodulation cohérente :

```
demod = sFinal .* port;
```

Le signal démodulé *demod* est le résultat du produit entre la somme des 3 signaux modulé et la porteuse du canal central.
Afin de déterminer la fréquence de coupure du filtre passe-bas, on regarde le spectre de *demod* :

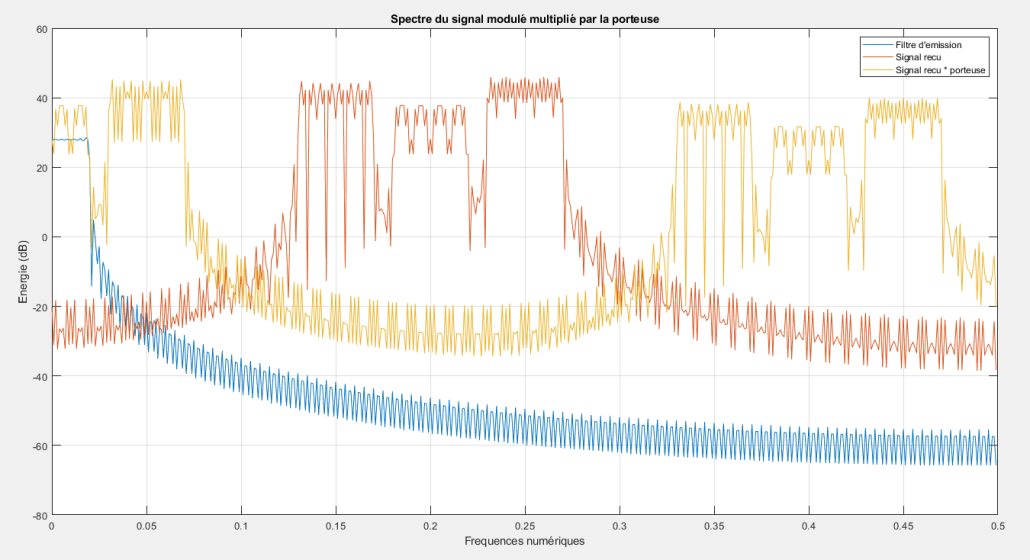


FIGURE 10 – Spectre du signal modulé multiplié par la porteuse

Si l'on veut récupérer seulement le canal central du spectre jaune, alors une fréquence de coupure de 0.025 Hz sera nécessaire. On crée donc ce filtre :

```
ordre = 25;
fPB = fir1(ordre, 2*0.025, 'low', hann(ordre+1));

signalFinalDemod = filter(fPB', 1, demod);

A = sign(downsample(signalFinalDemod, K));
```

La fonction *fir1()* permet de créer un filtre RIF (Réponse Impulsionnelle Finie) d'un ordre choisi, avec la méthode de la fenêtre (ici une fenêtre de Hanning). Pour essayer ce filtre, on choisit un ordre de départ de 25, que l'on modifiera par la suite.
On applique ce filtre à notre signal *demod*, puis on récupère par décimation (sous-échantillonnage) d'un facteur *K* notre séquence d'amplitude.
On obtient :

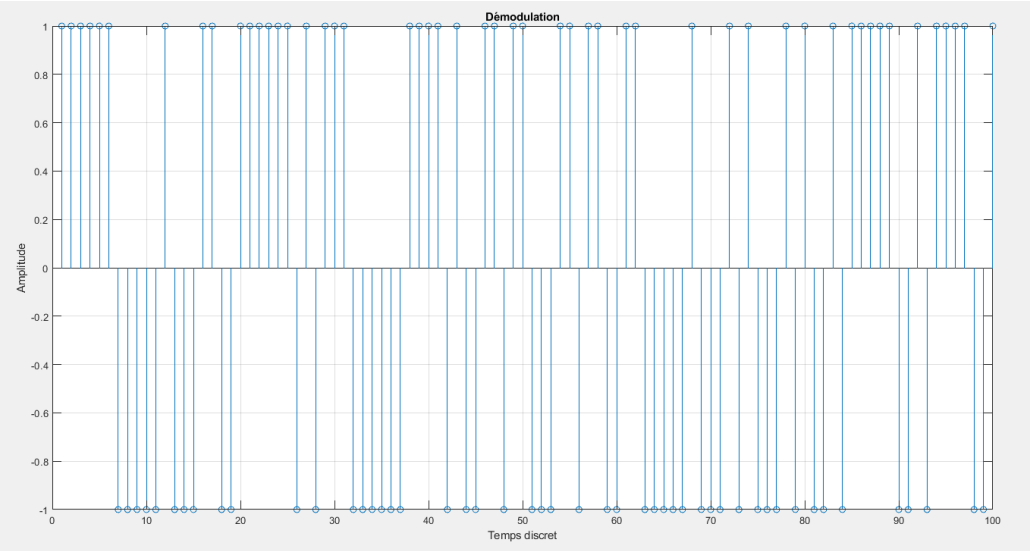


FIGURE 11 – Séquence d'amplitude démodulée, FPB d'ordre 25 et bande de garde à 0.05 Hz

Lorsque l'on calcule le taux d'erreur de cette démodulation, on obtient 0%. La transmission est parfaite.
Si on cherche l'ordre du filtre minimal à une transmission parfaite, on obtient 22. En-dessous, la transmission présente des erreurs.
Maintenant, si l'on réduit la bande de garde, par exemple à 0.03 Hz, l'ordre minimal du filtre est de 52.
A partir de 0.02 Hz, l'ordre du filtre dépasse les 200 et il reste toujours des erreurs. $\delta_\nu = 0.02$ Hz est la limite de la bande de garde. Ensuite, les canaux sont trop proches pour la fréquence de coupure donnée, et la réduire rendrait le filtre trop compliqué à réaliser. La fréquence de coupure du filtre d'émission étant à 0.02 Hz, si la bande de garde n'est pas supérieure, alors les canaux vont s'entrecouper et ainsi on ne récupérera jamais **que** les amplitudes du canal central.

Etude de la résistance au bruit

Finalement, pour simuler une transmission réaliste, on ajoute du bruit qui serait créé par le canal de transmission. Ce bruit sera un bruit blanc gaussien centré de variance σ^2 :

```
sigma = 0.7;
bruit = sigma*randn(length(sA1), 1);
sFinal = sA1 + sB1 + sC1 + bruit;
```

On multiplie donc σ à un vecteur de valeurs aléatoires de la taille du signal modulé du canal central (pris arbitrairement entre les 3), puis l'on ajoute le bruit au signal final.
On obtient :

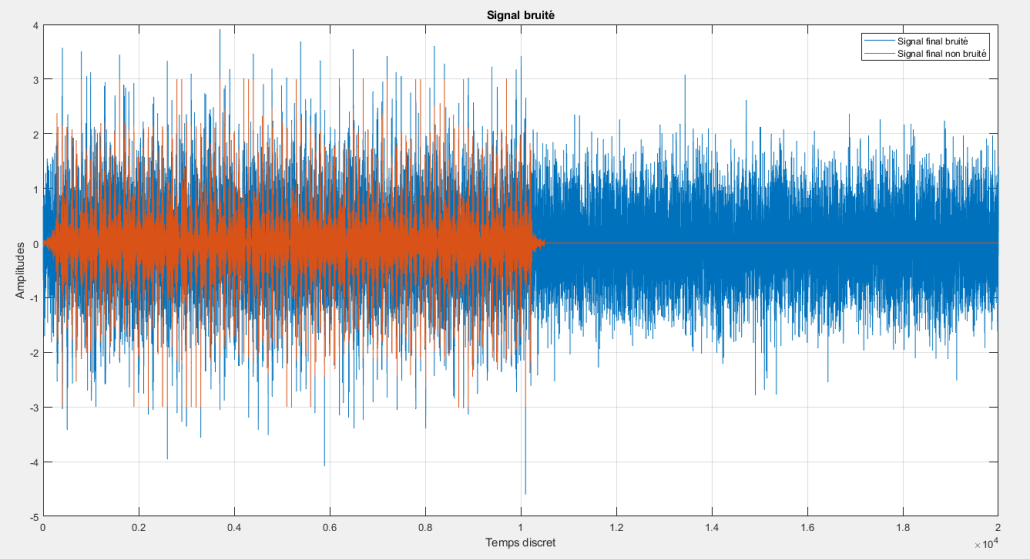


FIGURE 12 – Signaux finaux bruité et non bruité, $\sigma = 0.7$

Pour un σ de 0.7, on distingue très mal le signal original.
Pour une bande de garde de 0.03 Hz, et un ordre de filtre passe-bas de 52, une transmission parfaite n'est possible qu'avec 0.005 de variance de bruit. Par contre, si on augmente l'ordre du filtre, alors on peut aller jusqu'à $\sigma = 0.3$, pour un ordre de 85.
Pour une bande de garde de 0.05 Hz, et un ordre de filtre passe-bas de 22, la transmission parfaite n'est possible qu'à $\sigma = 0.005$. Par contre, pour un ordre de 40, σ peut monter jusqu'à 0.7.

Plus la bande de garde est élevée, plus le bruit peut être important sans impacter la démodulation. Néanmoins, il est quand même nécessaire d'augmenter l'ordre du filtre passe-bas, on ne peut pas garder les valeurs limites sans bruit.