Communications numériques

Travaux Pratiques (21h)

Modélisation et étude d'une chaîne de communication numérique

Université Paris 13, Institut Galilée, Ecole d'ingénieurs Sup Galilée
Parcours Informatique et Réseaux Alternance - 2ème année

2015-2016

Consignes

- Récupérer le fichier `TP.zip` sur le site

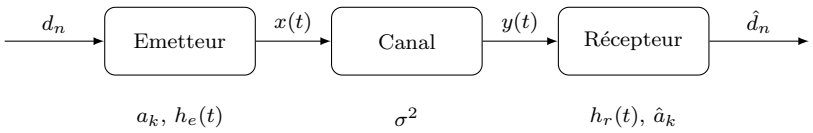
  `http://www.laurentoudre.fr/tns.html`

- Ouvrir MATLAB et créer un répertoire de travail. Dézipper le fichier `TP.zip` dans ce répertoire.
- A la fin de la séance, récupérer les scripts que vous avez écrits et vous les envoyer par e-mail afin de les conserver pour la prochaine séance.

Plan de l'étude

1 Transmission en bande de base

Le but de cette partie est d'étudier une chaîne de transmission en bande de base en présence de bruit blanc gaussien et illustrée ci-dessous.



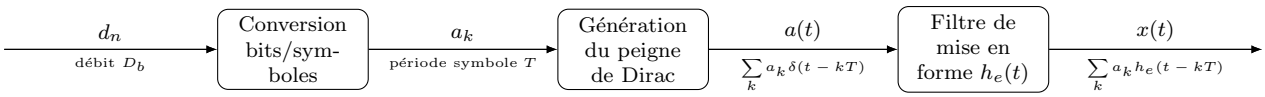
- A l'émetteur, on part d'une série de bits d_n, que l'on convertit en symboles a_k selon un dictionnaire choisi. On crée ensuite à partir de ces symboles un signal physique $x(t)$ grâce à un filtre de mise en forme $h_e(t)$ correctement choisi
- Le canal est modélisé uniquement par un bruit additif gaussien $b(t)$ de variance σ^2, créant un signal bruité $y(t) = x(t) + b(t)$.
- Au niveau du récepteur, on aura un filtre de réception $h_r(t)$ (choisi pour que le récepteur soit optimal), une étape d'échantillonnage puis de prise de décision pour retrouver les symboles $\hat{a}_k$. Enfin, on décodera les symboles pour tenter de retrouver le message binaire original $\hat{d}_n$.

Deux points fondamentaux seront étudiés dans cette partie :

- **L'étude des propriétés du signal en bande de base.** On étudiera en particulier l'influence des dictionnaires et des filtres de mise en forme sur les propriétés spectrales du signal ainsi construit (bande passante, annulations, raies fréquentielles, etc...).
- **L'étude détaillée des performances de la chaîne de communication** en terme notamment de taux d'erreur binaire. On étudiera l'influence des filtres d'émission et de réception, le choix du dictionnaire, ainsi que de la variance du bruit sur les performances de la chaîne de traitement.

1.1 Emetteur

Le plan de l'émetteur est détaillé sur le schéma bloc ci-dessous :



Les différentes étapes à réaliser, illustrées ci-dessus, sont les suivantes :

1. Conversion grâce à un dictionnaire du message binaire d_n (émis avec un débit binaire D_b) en une série de symboles a_k (période symbole T)
2. Construction du signal physique $a(t) = \sum_k a_k \delta(t - kT)$ (un symbole émis toutes les T secondes)
3. Convolution du signal $a(t)$ par la réponse impulsionnelle $h_e(t)$ du filtre d'émission, pour former $x(t)$

1.1.1 Conversion bits/symboles

Le but de l'étape de conversion bits/symboles est d'associer à chaque groupement de m bits un symbole pris dans un dictionnaire à $M = 2^m$ éléments. On a vu dans le cours deux types de dictionnaires à $M = 2^m$ élements :

- Codage M-aire unipolaire : $a_k \in \{0, 1, \cdots, M-1\}$
- Codage M-aire antipolaire : $a_k \in \{-(M-1), \cdots, -3, -1, 1, 3, \cdots, M-1\}$ (uniquement les éléments impairs)

Q1 - Créer une fonction MATLAB

```
[dico_sym,dico_dec] = dictionnaire(M,method_dic)
```

prenant en entrée la valence `M` et le type de dictionnaire `method_dic`, et retournant la liste des symboles du dictionnaire `dico_sym` et la représentation décimale des messages binaires associés à chaque symbole `dico_dec`

Indications

- Pour gérer les variables `M` et `method_dic`, on pourra utiliser l'instruction MATLAB `switch`
- On prendra `method_dic = 1` pour le dictionnaire unipolaire et `method_dic = 2` pour le dictionnaire antipolaire.
- On codera uniquement les dictionnaires suivants :
  - Codage binaire unipolaire (`M = 2`, `method_dic = 1`) : $0 \rightarrow 0$, $1 \rightarrow 1$
  - Codage binaire antipolaire (`M = 2`, `method_dic = 2`) : $0 \rightarrow -1$, $1 \rightarrow 1$
  - Codage 2B1Q (`M = 4`, `method_dic = 2`) : $00 \rightarrow -3$, $01 \rightarrow -1$, $11 \rightarrow 1$, $10 \rightarrow 3$

  On renverra un message d'erreur en dehors de ces trois cas.
- Afin de coder facilement les messages binaires associés à chaque symbole, on les représentera en base 10. Exemple : si le symbole -7 est associé au message 0 1 1, alors on stockera -7 dans le vecteur `dico_sym` et la représentation décimale 3 dans le vecteur `dico_dec`.

Tests à effectuer

Tester les valeurs de sorties pour les 3 dictionnaires proposés, et vérifier que l'on obtient bien une erreur en dehors de ces 3 cas.

Q2 - Créer une fonction MATLAB

```
[a_k,T,K] = conversion_bitssymboles(d,Db,M,method_dic)
```

prenant en entrée une série de bits `d` émise avec un débit binaire `Db` et formant une série de `K` symboles `a_k` associés à une période symbole `T`. Les variables `M` et `method_dic` permettent de sélectionner le dictionnaire de symboles à utiliser.

Indications

- Il faudra d'abord vérifier que `M` est bien une puissance de 2, puis calculer la période symbole `T` en fonction du débit binaire `Db` et de la valence `M`.
- On traitera un par un les groupes de $m = \log_2(M)$ bits : on commencera par les représenter en base 10 (grâce à la fonction `bits2dec.m` fournie), puis on cherchera dans le dictionnaire le symbole associé à ce nombre décimal

Tests à effectuer

Pour tester la fonction, on pourra générer un signal binaire aléatoire de longueur `N` = 10 grâce à la ligne de code suivante :

```
d = round (rand(N,1));
```

On utilisera la valeur `Db` = 2 bits/sec, et on comparera, pour les 3 dictionnaires, les résultats obtenus (symboles + période symbole) avec les résultats théoriques trouvés à la main.

Q3 - Créer une fonction MATLAB

```
[d] = conversion_symbolesbits(a_k,M,method_dic)
```

prenant en entrée une série de symboles `a_k` et formant la série de bits `d` initiale. Les variables `M` et `method_dic` permettent de sélectionner le dictionnaire de symboles à utiliser.

Indications

On commencera par chercher la représentation en base 10 du symbole, que l'on convertira ensuite en message binaire grâce à la fonction `dec2bits.m` fournie.

Tests à effectuer

Si tout fonctionne, lorsqu'on convertit le message binaire en symboles puis les symboles en message binaire, on doit retrouver le message original. Tester que ceci est vrai pour les 3 dictionnaires avec des messages binaires aléatoires de taille `N` = 10 et un débit binaire `Db` = 2 bits/sec.

1.1.2 Génération du peigne de Dirac

Q4 - Créer une fonction

```
[a,t_a]=genere_dirac(a_k,T,Fs)
```

prenant en entrée une série de symboles `a_k` avec une période symbole `T`, et générant le signal physique `a` associé à un vecteur temps `t_a` et échantillonné à la fréquence d'échantillonnage `Fs`.

Indications

- Commencer par déterminer la durée `Nsym` (en échantillons) d'un symbole à partir de `T` et `Fs`
- Calculer ensuite le nombre de symboles `K` à transmettre et en déduire le vecteur temps `t_a` à utiliser. Attention, ce vecteur doit contenir $K \times N_{sym}$ échantillons.
- Déterminer les indices du vecteur `t_a` correspondant aux temps multiples de `T`
- Définir ensuite le signal `a`, de même taille que le vecteur temps `t_a`, et former le signal

$$a(t) = \sum_{k=0}^{K-1} a_k \delta(t - kT)$$

Tests à effectuer

Générer un signal binaire aléatoire de longueur `N` = 10 émis avec un débit binaire `Db` = 2 bits/sec, et le convertir en symboles grâce au dictionnaire de votre choix. Générer et afficher ensuite en fonction du temps le signal $a(t)$ obtenu. On prendra `Fs` = 100 Hz.

1.1.3 Filtre de mise en forme

Q5 - Créer une fonction MATLAB

```
[he,E_he]=create_filter(t_he,T,method_fil)
```

prenant en entrée un vecteur temps `t_he` et la période symbole `T` et formant la réponse impulsionnelle `he` d'un filtre de mise en forme. La variable `method_fil` permet de sélectionner le filtre que l'on veut générer et `E_he` représente l'énergie totale du filtre.

- `method_fil` = 1 : filtre NRZ

$$h_e(t) = \begin{cases} 1 & \text{si } 0 \leq t < T \\ 0 & \text{sinon} \end{cases}$$

- `method_fil` = 2 : filtre RZ

$$h_e(t) = \begin{cases} 1 & \text{si } 0 \leq t < \frac{T}{2} \\ 0 & \text{sinon} \end{cases}$$

- `method_fil` = 3 : filtre biphase Manchester

$$h_e(t) = \begin{cases} 1 & \text{si } 0 \leq t < \frac{T}{2} \\ -1 & \text{si } \frac{T}{2} \leq t < T \\ 0 & \text{sinon} \end{cases}$$

- `method_fil` = 4 : filtre en racine de cosinus surélevé

Indications

- Pour des raisons de précision numérique, il se peut que les inégalités strictes soient mal comprises par MATLAB. Il est donc préférable d'écrire $a < b$ comme $a < b - \epsilon$ avec par exemple $\epsilon = 10^{-10}$.
- L'énergie totale du filtre sera estimée comme la somme des éléments de la réponse impulsionnelle au carré.
- Pour réaliser le filtre en racine de cosinus surélevé, on utilisera la fonction `rootraisedcosine.m` fournie. Dans la suite on supposera que le paramètre du filtre est $\beta = 0.25$ (on pourra le changer par la suite si on le désire).

Tests à effectuer

Pour les tests, on pourra utiliser `T` = 0.5 secondes. Définir un vecteur temps `t_he` compris entre `-2T` à `2T` et échantillonné à `Fs` = 100 Hz. Tracer la réponse impulsionnelle des 4 filtres définis ci dessus en fonction du temps.

Q6 - Créer une fonction MATLAB

```
[x,t_x,E_he]=filtre_emission(a,t_a,T,Fs,method_fil)
```

qui prend en entrée un signal physique `a` (somme de dirac), le convolue avec un filtre de mise en forme spécifié par la variable `method_fil` et sort un signal `x`. `t_a` et `t_x` sont les vecteurs temps respectivement associés à `a` et `x` et `Fs` est la fréquence d'échantillonnage de ces deux signaux physiques. `T` est la période symbole. `E_he` est ici l'énergie du filtre d'émission utilisé.

Indications

- Pour réaliser la convolution entre le signal $a(t)$ et la réponse impulsionnelle du filtre de mise en forme $h_e(t)$ afin de former $x(t)$, on utilisera la fonction `prodconv.m` fournie.
- Pour cela, on considèrera des réponses impulsionnelles $h_e(t)$ du filtre pour des temps compris entre $-5T$ et $5T$.
- Attention, le signal obtenu en sortie n'a plus le même support temporel, il faut donc travailler ensuite avec le nouveau vecteur temps fourni en sortie de la fonction `prodconv.m`.

Tests à effectuer

Voir question suivante.

1.1.4 Finalisation de l'émetteur

Q7 - En utilisant les fonctions MATLAB précédemment codées, créer une fonction

```
[x,t_x,T,a_k,K,E_he]=emetteur(d,Db,Fs,M,methode_dic,methode_fil)
```

générant un signal `x` mis en forme à partir d'une série de bits `d`. `t_x` représente le vecteur temps associé au signal `x`, `T` la période symbole, `a_k` la série de `K` symboles et `E_he` l'énergie totale du filtre de mise en forme. `Db` représente ici le débit binaire, `Fs` la fréquence d'échantillonnage. `M` et `methode_dic` spécifient le dictionnaire à utiliser et `methode_fil` le filtre de mise en forme à utiliser.

Indications

L'émetteur réalisera successivement la conversion bits/symboles, la génération du peigne de Dirac, le calcul de la réponse impulsionnelle du filtre, puis la convolution entre le signal $a(t)$ et la réponse impulsionnelle du filtre $h_e(t)$.

Tests à effectuer

Générer aléatoirement un signal binaire de taille `N` $= 10$, émis à un débit binaire `Db` $= 2$ bits/seconde. Tracer en fonction du temps le signal en bande de base `x` obtenu pour tous les choix de dictionnaire et de filtres d'émission. Les signaux physiques seront échantillonnés à `Fs` $= 100$ Hz. Attention, à cause du produit de convolution, le vecteur temps `t_x` est défini pour des temps inférieurs à 0 et supérieurs à KT. On utilisera donc la commande `xlim` de MATLAB pour le tracer uniquement sur la période souhaitée.

JALON 1 : Propriétés des signaux en bande de base

ETUDE A MENER

Pour chacune des configurations possibles (dictionnaire + filtre de mise en forme), tracer et étudier la densité spectrale de puissance des signaux en bande de base. Pour cela, on génèrera des signaux aléatoires avec `N` = 20000, `Db` = 2 bits/seconde et `Fs` = 100 Hz. On estimera la densité spectrale de puissance du signal grâce à la fonction `compute_dsp.m` fournie. On pourra étudier ces densités spectrales de puissance soit en décibels (par défaut dans la fonction fournie) soit en échelle linéaire. Attention, il ne faut évaluer la DSP que sur les temps correspondant effectivement au signal physique généré.

```
compute_dsp(x(t_x>=0 & t_x<K*T),Fs);
```

L'étude abordera, pour chaque configuration, les points suivants :

- La largeur de bande du signal (exprimée en fonction de $\frac{1}{T}$)
- La présence d'annulations dans la DSP (et leur position exprimée en fonction de $\frac{1}{T}$)
- La présence de raies fréquentielles dans la DSP (et leur position exprimée en fonction de $\frac{1}{T}$)
- L'atténuation (en dB) du deuxième lobe par rapport au premier lobe

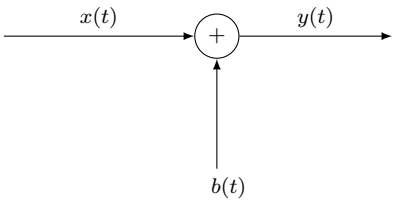
On résumera toutes ces informations sous la forme de plusieurs tableaux que l'on commentera. On pourra en particulier comparer les résultats expérimentaux aux résultats théoriques donnés dans le cours (pour une ou deux configurations). Pour le filtre en racine de cosinus surélevé, on fera varier β et on observera les conséquences notamment sur la largeur de bande du signal.

RENDU

- Rapport présentant les résultats et l'analyse de l'étude (format PDF)
- Fonctions des questions Q1 à Q7 + script réalisant les tests unitaires des différentes fonctions (fichier ZIP)
- Tous les fichiers (rapport + fonctions + script) doivent contenir vos noms et prénoms

1.2 Canal

Le plan du canal de transmission est détaillé sur le schéma bloc ci-dessous :



Le canal est modélisé uniquement par un bruit additif gaussien $b(t)$ de variance σ^2, créant un signal bruité $y(t) = x(t)+b(t)$. Nous savons que la puissance moyenne totale des signaux en bande de base dépend notamment du filtre de mise en forme, mais aussi du dictionnaire utilisé. Si l'on souhaite comparer les performances des différentes configurations, il est donc plus pertinent de le faire en terme de rapport signal sur bruit. Il faut donc paramétrer la valeur de σ^2 en fonction du rapport $\left.\frac{E_{bit}}{N_0}\right|_{dB}$ en décibels.

Q8 - Créer une fonction MATLAB

```
[E_bit]=energie_moyenne_bit(M,method_dic,E_he)
```

calculant l'énergie moyenne par bit `E_bit` pour un dictionnaire spécifié par `M` et `method_dic`, et un filtre de mise en forme d'énergie totale `E_he`.

Indications

On utilisera pour cela les formules théoriques du cours

Tests à effectuer

On vérifiera en prenant `E_he` = 1 que l'on retrouve bien les résultats théoriques pour les 3 dictionnaires considérés.

Q9 - Créer une fonction MATLAB

```
[y,sigma2] = canal(x,EbitN0_dB,M,method_dic,E_he)
```

qui prend en entrée le signal `x` et lui rajoute un bruit blanc additif gaussien de variance `sigma2` calculée à partir du rapport `EbitN0\_dB` en décibels, du dictionnaire spécifié par `M` et `method_dic`, et de l'énergie totale du filtre de mise en forme `E_he`.

Indications

- La première étape consiste à calculer `\sigma2`. On commencera pour cela à calculer `E_bit` grâce à la fonction `energie_moyenne_bit`, puis on utilisera le fait que $\sigma^2 = \frac{N_0}{2}$ et que $\left.\frac{E_{bit}}{N_0}\right|_{dB} = 10\log_{10}\left(\frac{E_{bit}}{N_0}\right)$.
- Pour créer un bruit blanc additif gaussien de variance `sigma2`, on pourra utiliser la commande

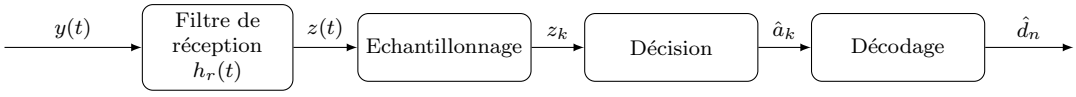
```
b=sqrt(sigma2)*randn(length(x),1)
```

Tests à effectuer

On pourra visualiser les signaux bruités obtenus avec des valeurs de `EbitN0_dB` allant de 0 à 20 dB. Pour les simulations, on prendra `N` =10, `Db` = 2 bits/seconde, et `Fs` = 100 Hz, et on génèrera un signal $x(t)$ avec un dictionnaire et un filtre de mise en forme au choix.

1.3 Récepteur

Le plan du récepteur est détaillé sur le schéma bloc ci-dessous :



Les différentes étapes à réaliser, illustrées ci-dessus, sont les suivantes :

1. Convolution du signal $y(t)$ par la réponse impulsionnelle $h_t(t)$ du filtre de réception, pour former $z(t)$
2. Echantillonnage du signal $z(t)$ pour tous les temps $t = kT$ avec $k \in [\![0, K-1]\!]$, afin de récupérer z_k
3. Décision pour estimer les symboles $\hat{a}_k$ à partir des z_k et de E_{he}
4. Décodage pour retrouver le message binaire $\hat{d}_n$ à partir des symboles estimés $\hat{a}_k$

On supposera ici que le récepteur est optimal (rappel : on a donc $h_r(t) = h_e(-t)$ et $h = h_e * h_r$ qui est un filtre de Nyquist).

1.3.1 Filtre de réception

Q10 - Créer une fonction MATLAB

```
[z,t_z,E_hr]=filtre_reception(y,t_y,T,Fs,method_fil)
```

qui prend en entrée un signal bruité `y`, le convolue avec un filtre de réception spécifié par la variable `method_fil` et sort un signal `z`. `t_y` et `t_z` sont les vecteurs temps respectivement associés à `y` et `z` et `Fs` la fréquence d'échantillonnage. `T` est la période symbole et `E_hr` est ici l'énergie du filtre d'émission h_r utilisé (qui est aussi égale à celle du filtre d'émission).

Indications

- Pour créer la réponse impulsionnelle du filtre de réception, il suffit de créer celle du filtre d'émission mais avec $t = -t$
- On prendra toujours la réponse impulsionnelle du filtre entre $-5T$ et $5T$
- Attention, la convolution change le vecteur de temps donc `t_z` n'est pas le même que `t_y`.

Tests à effectuer

On pourra visualiser le signal $z(t)$ obtenu avec différentes valeurs de `EbitN0_dB`. Pour les simulations, on prendra `N` =10, `Db` = 2 bits/seconde, et `Fs` = 100 Hz, et on génèrera un signal $x(t)$ avec un dictionnaire et un filtre de mise en forme au choix. Faire varier le filtre de mise en forme et observer les conséquences sur le signal $z(t)$.

1.3.2 Echantillonnage

Q11 - Créer une fonction MATLAB

```
[z_k]=echantillonnage(z,t_z,T,K,Fs)
```

qui prend un signal `z` échantilonné à `Fs` et associé à un vecteur temps `t_z`, et renvoie un vecteur `z_k` de longueur `K`, stockant les valeurs de `z` correspondant aux temps $t = kT$ avec $k \in [\![0, K-1]\!]$.

Indications

- Dans un premier temps, il s'agit de calculer la durée `Nsym` (en échantillons) d'un symbole
- On cherchera ensuite l'indice du vecteur `t_z` correspondant au temps $t = 0$ grâce à la commande `find` de MATLAB.
- A partir de là, et en s'inspirant de ce qui a été fait pour la fonction `genere_dirac`, il s'agit de trouver les indices du vecteur `t_z` correspondant aux temps multiples de `T`. Attention, il faut s'arrêter à $t = (K-1)T$!)

Tests à effectuer

Tester la fonction sur un des signaux $z(t)$ généré à la question précédente et vérifier en comparant à la figure que l'on a bien échantillonné aux bons instants.

1.3.3 Décision

Q12 - Créer une fonction

```
[a_k_hat]=decision(z_k,E_he,M,method_dic)
```

qui prend une série de valeurs `z_k` et associe à chacune un symbole `a_k_hat` du dictionnaire spécifié par `M` et `method_dic`. `E_he` est ici l'énergie du filtre d'émission h_e utilisé.

Indications

Comme nous l'avons vu dans le cours, étant donnée une valeur de z_k, le symbole détecté est le symbole du dictionnaire le plus proche de la valeur $\frac{z_k}{E_{he}}$ (au sens de la distance euclidienne).

Tests à effectuer

On pourra vérifier qu'en absence de bruit, on retrouve exactement les symboles a_k.

1.3.4 Décodage

Cette partie a déjà été réalisée, il s'agit de la fonction `conversion_symbolesbits` écrite dans la question Q3.

1.3.5 Finalisation du récepteur

Q13 - En utilisant les fonctions MATLAB précédemment codées, créer une fonction

```
[d_hat,a_k_hat,E_he]=recepteur(y,t_y,T,K,Fs,M,method_dic,method_fil)
```

qui prend en entrée un signal bruité `y` (échantillonné à `Fs` et associé à un vecteur temps `t_y`) et renvoyant une estimation du message binaire originellement envoyé `d_hat`. `a_k_hat` représente l'estimation des symboles envoyés et `E_he` l'énergie du filtre de mise en forme. `T` est la période symbole, `K` le nombre de symboles à retrouver, et `M`, `method_dic` et `method_fil` précisent respectivement le dictionnaire et les filtres à utiliser.

Indications

Le récepteur réalisera successivement la convolution par le filtre de réception, l'échantillonnage, la décision et le décodage.

Tests à effectuer

Pour les simulations, on prendra comme toujours `N` =10, `Db` = 2 bits/seconde, et `Fs` = 100 Hz. Si tout fonctionne bien, en prenant $y = x$ (absence de bruit), on doit retrouver exactement les bits envoyées.

JALON 2 : Etude des performances d'une chaîne de transmission numérique en bande de base

ETUDE A MENER

Pour les différentes configurations possibles (dictionnaire + filtre de mise en forme), on étudiera le taux d'erreur binaire de la chaîne de transmission, défini par :

$$TEB = \frac{\text{nombre de bits mal transmis}}{\text{nombre total de bits emis}}$$

Pour cela, on génèrera des signaux aléatoires avec `N` $= 24000$, `Db` $= 2$ bits/seconde et `Fs` $= 100$ Hz et on calculera à chaque fois le taux d'erreur binaire. Afin de réduire le nombre des configurations à étudier, on s'interessera uniquement aux dictionnaires antipolaires. On pourra calculer le TEB grâce à la commande MATLAB suivante :

```
TEB = sum(d~=d_hat)/length(d);
```

L'étude abordera les points suivants :

- Le taux d'erreur binaire dépend-il du filtre ? Expliquer pourquoi.
- Le taux d'erreur binaire dépend-il de la valence M ? Expliquer pourquoi.
- Tracer pour chacune des configurations le taux d'erreur binaire en fonction du rapport $\left.\frac{E_{bit}}{N_0}\right|_{dB}$ (On prendra des valeurs entre -5 et 10 dB).
- Comparer les résultats expérimentaux aux résultats théoriques donnés dans le cours pour un dictionnaire antipolaire :

$$TEB \approx \frac{M-1}{M \log_2 M} erfc\left(\sqrt{\frac{E_{bit}}{N_0} \frac{3 \log_2 M}{M^2-1}}\right)$$

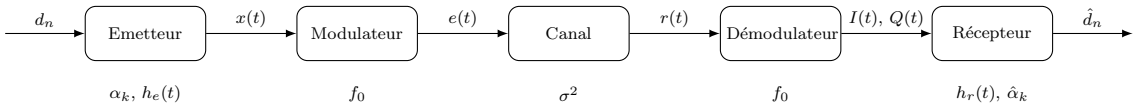
On pourra également étudier les performances des différentes configurations en terme d'efficacité spectrale, en réutilisant les résultats du Jalon 1. Expliquer alors les choix et compromis qui doivent être effectués lors du dimensionnement d'une chaîne de transmission numérique en bande de base.

RENDU

- Rapport présentant les résultats et l'analyse de l'étude (format PDF)
- Fonctions des questions Q8 à Q13 + script réalisant les tests unitaires des différentes fonctions (fichier ZIP)
- Tous les fichiers (rapport + fonctions + script) doivent contenir vos noms et prénoms

2 Transmission en bande modulée

Le but de cette partie est d'étudier une chaîne de transmission en bande modulée en présence de bruit blanc gaussien et illustrée ci-dessous.



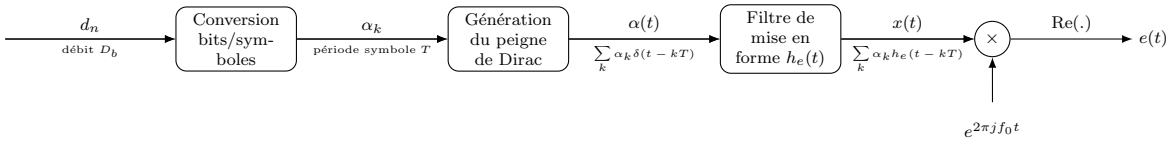
- A l'émetteur, on part d'une série de bits d_n, que l'on convertit en symboles complexes α_k selon le type de modulation choisi. On crée ensuite à partir de ces symboles un signal en bande de base $x(t)$ grâce à un filtre de mise en forme h_e correctement choisi. Ce signal est ensuite modulé par une sinusoïde complexe de fréquence fondamentale f_0 pour former un signal $e(t)$.
- Le canal est modélisé uniquement par un bruit additif gaussien $b(t)$ de variance σ^2, créant un signal bruité $r(t) = e(t) + b(t)$.
- Au niveau du récepteur, on aura tout d'abord une étape de démodulation et de filtrage pour récupérer les composantes en phase et en quadrature de phase $I(t)$ et $Q(t)$. Ces composantes passeront par un filtre de réception $h_r(t)$ (adapté à $h_e(t)$), par une étape d'échantillonnage puis de prise de décision pour retrouver les symboles complexes $\hat{\alpha}_k$. Enfin, on décodera les symboles $\hat{\alpha}_k$ pour tenter de retrouver le message binaire original $\hat{d}_n$.

On supposera ici que le récepteur est optimal, ce qui guidera le choix des filtres d'émission et de réception (rappel : on doit avoir $h_r(t) = h_e(-t)$ et $h = h_e * h_r$ qui doit être un filtre de Nyquist).

On étudiera en particulier l'influence des filtres d'émission et de réception et du type de modulation sur les performances de la chaîne de transmission.

2.1 Emetteur

Le plan de l'émetteur est détaillé sur le schéma bloc ci-dessous :



Les différentes étapes à réaliser, illustrées ci-dessus, sont les suivantes :

1. Conversion grâce à un dictionnaire du message binaire d_n (émis avec un débit binaire D_b) en une série de symboles complexes α_k (période symbole T)
2. Construction du signal physique $\alpha(t) = \sum_k \alpha_k \delta(t - kT)$ (un symbole émis toutes les T secondes)
3. Convolution du signal $a(t)$ par la réponse impulsionnelle $h_e(t)$ du filtre d'émission, pour former $x(t)$
4. Multiplication de $x(t)$ par une exponentielle complexe $e^{2\pi j f_0 t}$ puis prise de la partie entière, pour former $e(t)$

En dehors de la première et de la dernière étape, toutes les fonctions utilisées en bande de base pourront être réutilisées en bande modulée.

Q14 - Modifier la fonction `dictionnaire` en une fonction

```
[dico_sym,dico_dec] = dictionnaire_mod(M,method_mod)
```

prenant en entrée la valence `M` et le type de modulation `method_mod`, et retournant la liste des symboles du dictionnaire `dico_sym` et la représentation décimale des messages binaires associés à chaque symbole `dico_dec`. On codera ici plusieurs types de modulations :

- `method_mod = ASK` : modulation ASK symétrique à `M` = 2, 4 ou 8 éléments
- `method_mod = PSK` : modulation PSK à `M` = 2, 4 ou 8 éléments
- `method_mod = QAM$` : modulation QAM à `M` = 4 ou 16 éléments

Indications

- On utilisera les constellations données dans le cours pour établir la correspondance entre messages binaires et symboles.
- On retournera un message d'erreur en dehors de ces 8 cas.

Tests à effectuer

Tester les valeurs de sorties pour les 8 modulations proposées, et vérifier que l'on obtient bien une erreur en dehors de ces 8 cas.

Q15 - Modifier les fonctions `conversion_bitssymboles` et `conversion\_symbolesbits` en fonctions

```
[alpha_k,T,K] = conversion_bitssymboles_mod(d,Db,M,method_mod)
```

```
[d] = conversion_symbolesbits_mod(alpha_k,M,method_mod)
```

pour qu'elles utilisent dorénavant la fonction `dictionnaire_mod`

Tests à effectuer

Pour tester la fonction, on pourra générer un signal binaire aléatoire de longueur `N` = 12, émis avec un débit binaire`Db` = 2 bits/sec. On comparera, pour les 8 modulations, les résultats obtenus (symboles + période symbole) avec les résultats théoriques trouvés à la main. On s'assurera aussi que lorsqu'on convertit le message binaire en symboles puis les symboles en message binaire, on retrouve bien le message original.

Q16 - En utilisant les fonctions MATLAB précédemment codées, créer une fonction

```
[e,t_e,alpha_k,T,K,E_he] = emetteur_mod(d,Db,Fs,M,method_mod,method_fil,f0)
```

générant un signal `e` modulé avec une fréquence fondamentale `f0` à partir d'une série de bits `d`. `t_e` représente le vecteur temps associé au signal `e`, `alpha_k` la liste des symboles complexes qui ont été transmis, `T` la période symbole, `E_he` l'énergie du filtre de mise en forme. `Db` représente ici le débit binaire, `Fs` la fréquence d'échantillonnage, `M` la taille du dictionnaire, `method_mod` le type de modulation à utiliser et `method_fil` le filtre de mise en forme à utiliser.

Indications

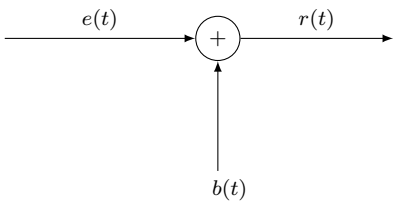
L'émetteur réalisera successivement la conversion bits/symboles, la génération du peigne de Dirac, le calcul de la réponse impulsionnelle du filtre, la convolution entre le signal $\alpha(t)$ et la réponse impulsionnelle du filtre $h_e(t)$, la multiplication par l'exponentielle complexe et la prise de partie entière.

Tests à effectuer

On pourra tester la fonction avec un message binaire de taille `N` = 12, un débit binaire `Db` = 2 bits/seconde, une fréquence d'échantillonnage `Fs` = 100 Hz, et une fréquence fondamentale `f0` = 13 Hz. On observera les signaux obtenus pour différentes modulations.

2.2 Canal

Le plan du canal de transmission est détaillé sur le schéma bloc ci-dessous :



Q17 - Modifier les fonctions `energie_moyenne_bit` et `canal` en fonctions

```
[E_bit]=energie_moyenne_bit_mod(M,method_mod,E_he)
```

```
[r,sigma2] = canal(e,EbitN0_dB,M,method_mod,E_he)
```

pour qu'elles utilisent dorénavant la fonction `dictionnaire_mod`

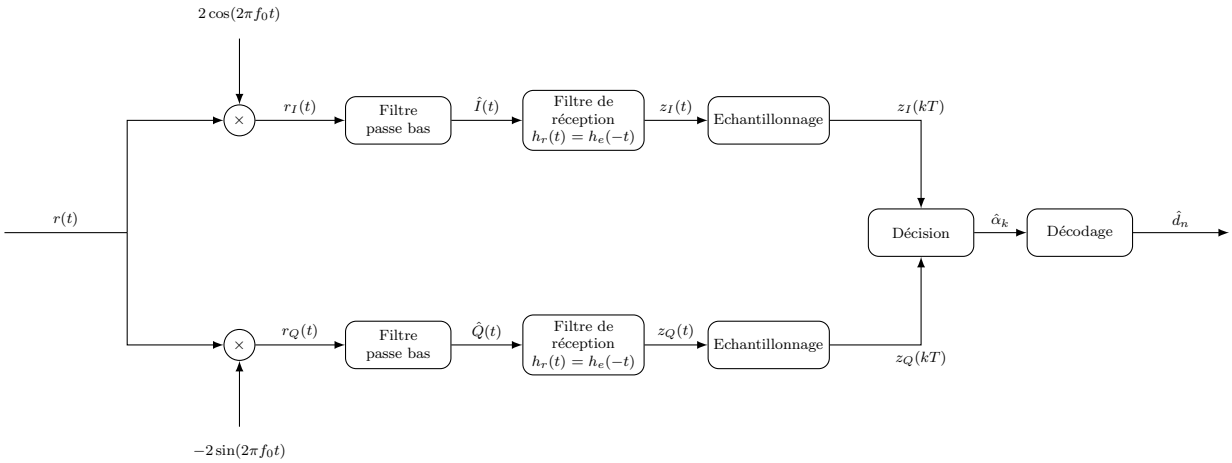
Attention, jusqu'à présent la notion de rapport signal sur bruit était définie en bande de base. Comme on ajoute ici le bruit sur le signal modulé $e(t)$ qui a une puissance moyenne deux fois moins grande sur que $x(t)$, il faut dans le canal modulé appliquer un bruit de variance $\frac{\sigma^2}{2}$ et non $sigma^2$.

Tests à effectuer

On pourra visualiser les signaux bruités obtenus avec des valeurs de `EbitN0_dB` allant de 0 à 20 dB. Pour les simulations, on prendra `N` = 12, un débit binaire `Db` = 2 bits/seconde, une fréquence d'échantillonnage `Fs` = 100 Hz, et une fréquence fondamentale `f0` = 13 Hz, et on génèrera un signal $e(t)$ avec une modulation et un filtre de mise en forme au choix. On pourra aussi observer la densité spectrale de puissance du signal $e(t)$ et la comparer à celle du signal $r(t)$.

2.3 Récepteur

Le plan du récepteur est détaillé sur le schéma bloc ci-dessous :



Les différentes étapes à réaliser, illustrées ci-dessus, sont les suivantes :

1. Multiplier le signal $r(t)$ respectivement par $2\cos(2\pi f_0 t)$ et $-2\sin(2\pi f_0 t)$ pour obtenir les signaux $r_I(t)$ et $r_Q(t)$
2. Filtrer ces signaux grâce à un filtre passe-bas pour obtenir des estimées $\hat{I}(t)$ et $\hat{Q}(t)$
3. Convolution des signaux $\hat{I}(t)$ et $\hat{Q}(t)$ par la réponse impulsionnelle $h_t(t)$ du filtre de réception, pour former $z_I(t)$ et $z_Q(t)$
4. Echantillonnage des signaux $z_I(t)$ et $z_Q(t)$ pour tous les temps $t = kT$ avec $k \in [\![0, K-1]\!]$, afin de récupérer $z_I(kT)$ et $z_Q(kT)$
5. Décision pour estimer les symboles $\hat{\alpha}_k$ à partir des $z_I(kT)$, $z_Q(kT)$ et de E_{he}
6. Décodage pour retrouver le message binaire $\hat{d}_n$ à partir des symboles estimés $\hat{\alpha}_k$

On supposera ici que le récepteur est optimal (rappel : on a donc $h_r(t) = h_e(-t)$ et $h = h_e * h_r$ qui est un filtre de Nyquist).

Q18 - En utilisant les fonctions MATLAB précédemment codées, créer une fonction

```
[d_hat,alpha_k_hat,z_k]=recepteur_mod(r,t_r,T,K,Fs,M,method_mod,method_fil,f0)
```

qui prend en entrée un signal `r` échantillonné à `Fs`, modulé par une porteuse de fréquence `f0` et bruité (auquel correspond un vecteur temps `t_r`) et renvoyant une estimation du message binaire originellement envoyé `d_hat`. `alpha_k_hat` représente l'estimation des symboles complexes envoyés et `z_k` les symboles complexes obtenus à la sortie du récepteur avant décision. `T` est la période symbole, `K` le nombre de symboles à retrouver, `M` et `method_mod` les paramètres de la modulation et `method_fil` le type de filtre à utiliser.

Indications

- Pour la prise de décision, il faudra créer une nouvelle fonction `decision_mod` en s'inspirant de ce qui a été fait précédemment. Il faudra pour cela reformer des symboles complexes $z_k = z_I(kT) + jz_Q(kT)$
- Le filtrage passe-bas pourra être fait grâce à la commande suivante :

```
[b,a] = butter(5,f0*2/Fs);

y = filtfilt(b,a,x);
```

Tests à effectuer

On pourra tester la fonction avec un message binaire à `N` = 12 bits, un débit binaire `Db` = 2 bits/seconde, une fréquence d'échantillonnage `Fs` = 100 Hz, et une fréquence fondamentale `f0` = 13 Hz. Si tout fonctionne bien, en prenant $r = e$, on doit retrouver exactement les bits envoyées.

JALON 3 : Propriétés et performances d'une chaîne de transmission numérique en bande modulée

ETUDE A MENER

Pour les différentes configurations possibles (modulation + filtre de mise en forme), on étudiera les différentes propriétés de la chaîne de transmission en bande modulée : efficacité spectrale et taux d'erreur binaire. Pour cela, on génèrera des signaux aléatoires avec `N` = 24000, `Db` = 2 bits/seconde, `Fs` = 100 Hz et `f0` = 23 Hz. On testera toutes les modulations, mais uniquement les filtres d'émission NRZ.

L'étude abordera les points suivants :

- Etudier les densités spectrales de puissance (largeur de bande, annulations, etc...) des signaux modulés pour différentes modulations. Comparer à ce qui se passe en bande de base. Interpréter ces résultats en terme d'efficacité spectrale. Résumer dans un tableau les efficacités spectrales des différentes modulations (avec un filtre NRZ).
- Regarder dans le plan complexe les symboles $\frac{z_k}{E_{h_e}}$ obtenus pour différentes modulations et différentes valeurs du rapport $\frac{E_{bit}}{N_0}$ et commenter. On pourra utiliser pour cela la commande MATLAB

```
plot(real(z_k/E_he),imag(z_k/E_he),'.')
```

- Evaluer le taux d'erreur binaire pour différentes valeurs du rapport $\frac{E_{bit}}{N_0}$ et différentes modulations.
- Comparer les résultats expérimentaux aux résultats théoriques donnés dans le cours pour toutes les modulations :

$$TEB_{ASK} \approx \frac{M-1}{M\log_2 M} erfc\left(\sqrt{\frac{E_{bit}}{N_0}\frac{3\log_2 M}{M^2-1}}\right)$$

$$TEB_{PSK} \approx \frac{1}{\log_2 M} erfc\left(\sqrt{\frac{\log_2 M E_{bit}}{N_0}}\sin\left(\frac{\pi}{M}\right)\right) \text{ pour } M > 2$$

$$TEB_{QAM} \approx \frac{\sqrt{M}-1}{\sqrt{M}\log_2\sqrt{M}} erfc\left(\sqrt{\frac{E_{bit}}{N_0}\frac{3\log_2\sqrt{M}}{M-1}}\right)$$

RENDU

- Rapport présentant les résultats et l'analyse de l'étude (format PDF)
- Fonctions des questions Q14 à Q18 + script réalisant les tests unitaires des différentes fonctions (fichier ZIP)
- Tous les fichiers (rapport + fonctions + script) doivent contenir vos noms et prénoms