Projet de Traitement du Signal :
Récepteur MF-TDMA (*)
(*) Multiple Frequency – Time Division Multiple Access

Introduction :

Le but du projet est de comprendre l'apport des techniques de traitement du signal dans des applications multimédia par satellite. L'application retenue concerne des systèmes d'accès bidirectionnel à haut débit par satellite. Ces systèmes utilisent le standard DVB-RCS(*) qui permet d'offrir aux utilisateurs une interactivité à l'aide d'une voie retour par satellite.

La voie retour par satellite utilise l'accès MF-TDMA. Cet accès permet de partager en temps et en fréquence la ressource satellite entre les différents utilisateurs. Ce partage temporel et fréquentiel de la ressource repose sur une structure de trame normalisée appelée trame MF-TDMA.

(*) DVB-RCS : Digital Video Broadcasting – Return Channel via Satellite

Description d'un système DVB-RCS :

Un système DVB-RCS offre un accès Internet bidirectionnel à haut débit par satellite. Les utilisateurs disposent d'un terminal qui permet de les relier à une station d'interconnexion à travers une liaison satellite. La station d'interconnexion joue le rôle d'interface avec le monde de l'Internet et du réseau pour offrir des applications diverses telles que la navigation à travers le WEB, la messagerie ou le transfert de données :

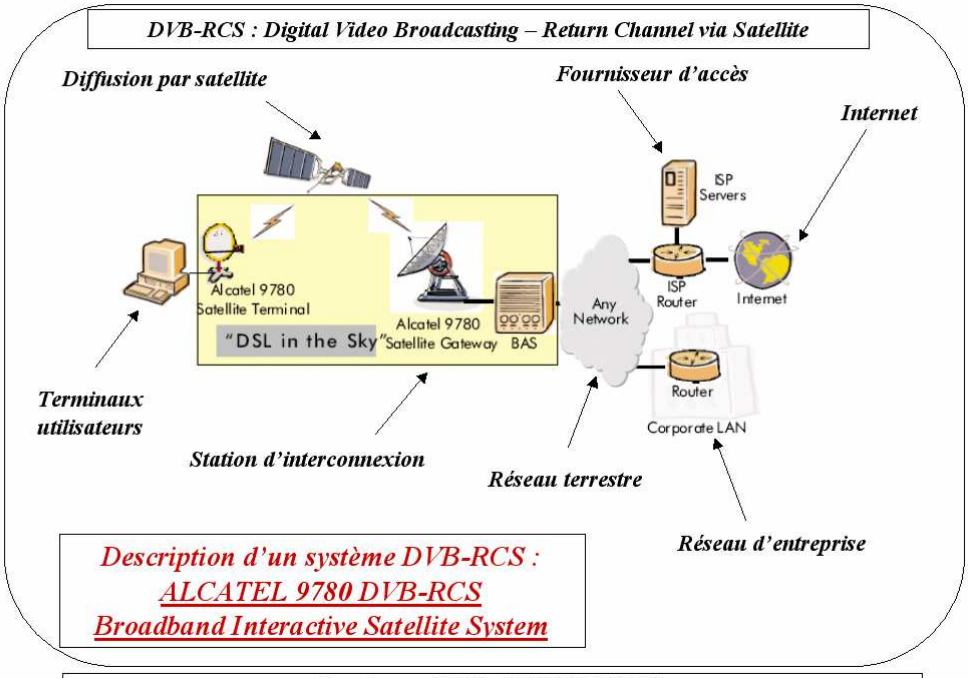
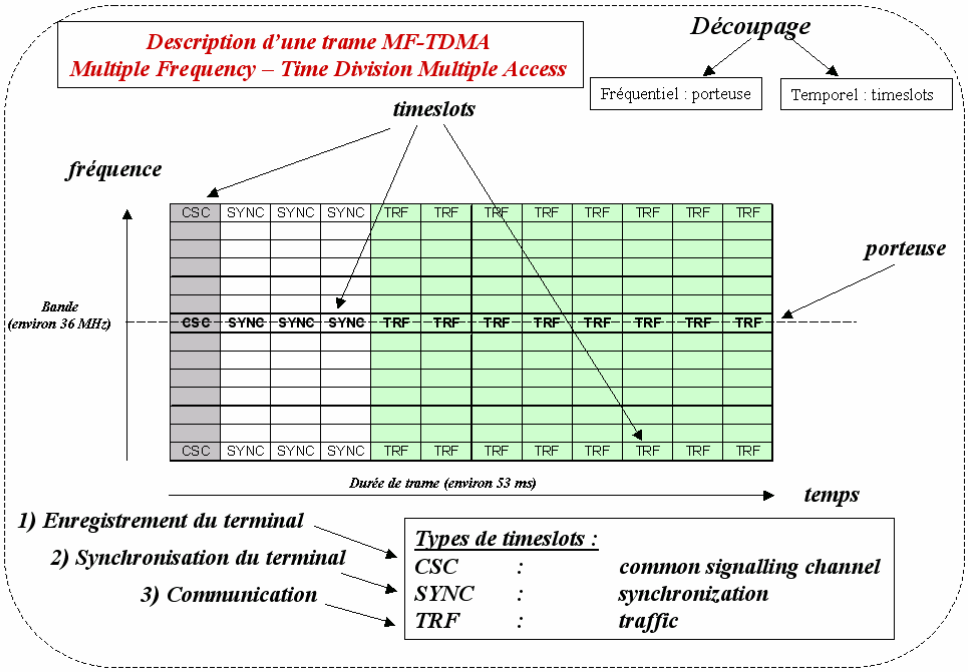


Figure de source ALCATEL : ALCATEL 9780 DVB-RCS

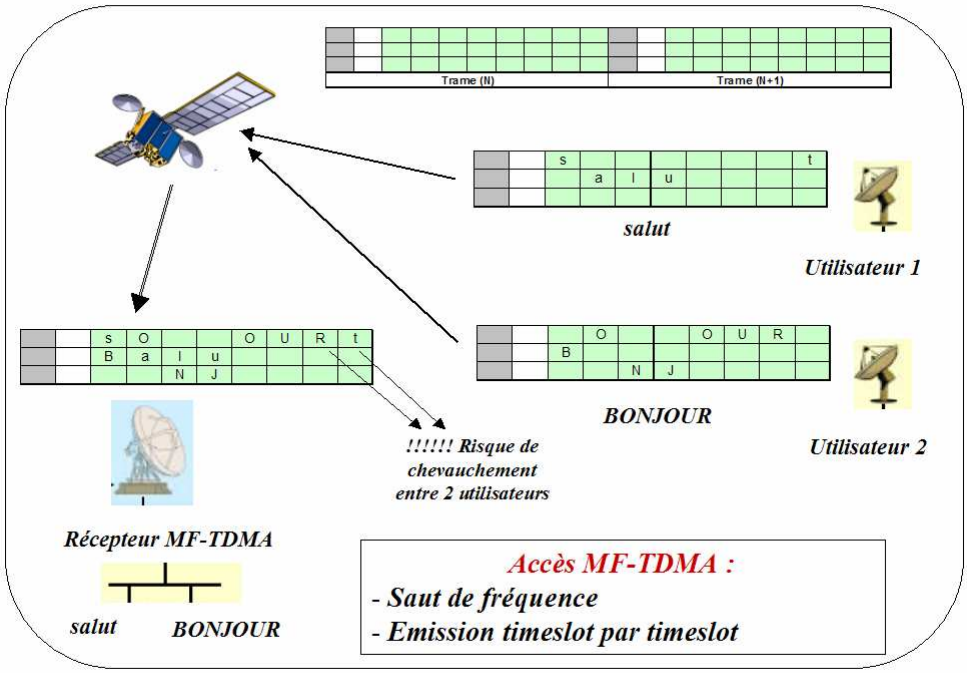
Description de la trame MF-TDMA :

Les utilisateurs du système accèdent à la station d'interconnexion en utilisant une trame MF-TDMA. Cette dernière correspond à un motif temps fréquence avec une durée temporelle et une bande fréquentielle fixées en fonction du système.

La bande fréquentielle de la trame est découpée en plusieurs porteuses dont la largeur dépend du rythme symbole. Chaque porteuse est découpée en plusieurs portions temporelles appelées timeslots et qui sont partagées entre les différents utilisateurs :

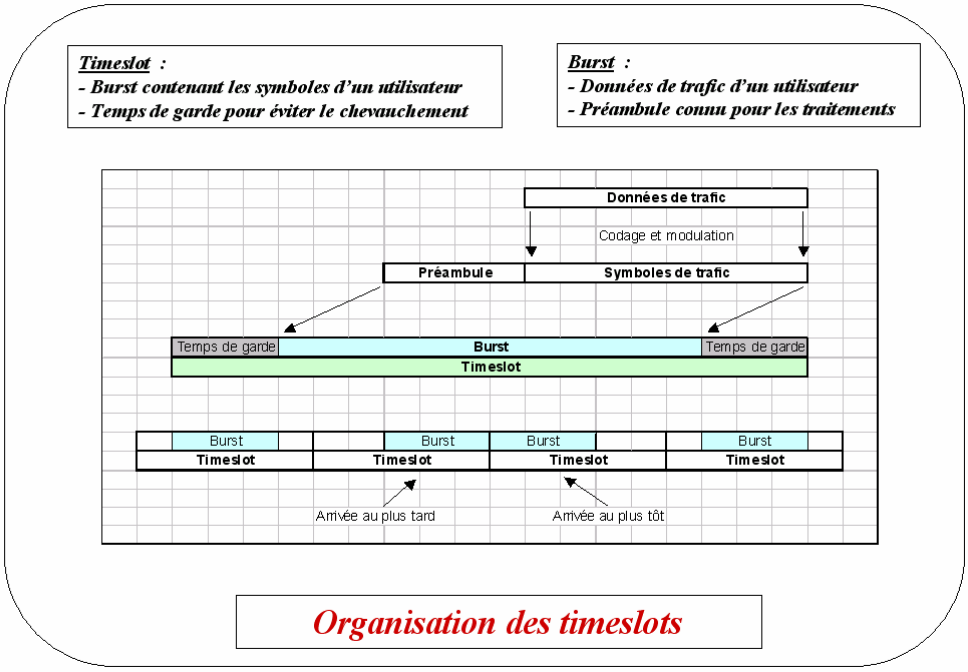


Chaque utilisateur transmet ses données en les répartissant dans les timeslots qui lui ont été alloués. L'allocation des timeslots entre les différents utilisateurs est prise en charge par la station d'interconnexion alors que le récepteur MF-TDMA restitue les données envoyées par chaque utilisateur :



Afin d'éviter le chevauchement entre 2 utilisateurs, chaque timeslot se compose d'un burst entouré d'un temps de garde et contenant les symboles de l'utilisateur.

Chaque burst se décompose d'un entête connu appelé préambule et d'une partie privée contenant les données de trafic de l'utilisateur :



Etapes du travail à effectuer :

Une des difficultés d'un système DVB-RCS est la mise en œuvre du récepteur MF-TDMA. Un tel récepteur, situé dans la station d'interconnexion, doit extraire et restaurer les données de tous les utilisateurs ayant accès au système. L'objectif du projet est la mise en œuvre de mécanismes simplifiés permettant d'effectuer les différents traitements classiques du récepteur :

- Démultiplexage des porteuses du canal
- Démodulation des porteuses
- Décodage des données des utilisateurs

Le scénario retenu pour le projet consiste en une trame MF-TDMA composée de 2 porteuses contenant chacune 5 timeslots :

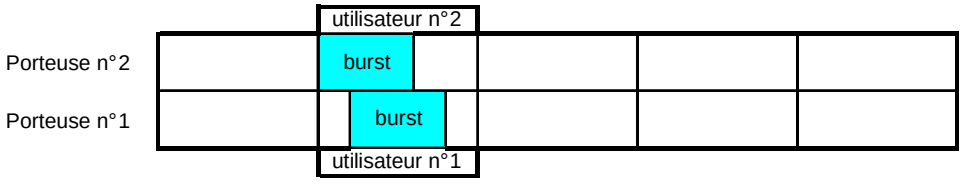
Porteuse n°2	1	2	3	4	5
Porteuse n°1	1	2	3	4	5

Les paramètres de la trame sont résumés dans le tableau suivant :

Durée de la trame	200	ms
Nombre de porteuses	2	
Nombre de timeslots par trame	5	
Durée du timeslot	40	ms
Durée du burst	38	ms
Nombre de symboles du préambule	40	
Rythme symbole	10	kHz
Fréquence de la porteuse n° 1	0	kHz
Fréquence de la porteuse n° 2	46	kHz
Fréquence d'échantillonnage	120	kHz

Deux utilisateurs vont accéder au système en utilisant la trame de la manière suivante :

- L'utilisateur n° 1 exploite le 2ème timeslot de la porteuse n° 1 avec un retard de 1 ms par rapport au début du timeslot
- L'utilisateur n° 2 exploite le 2ème timeslot de la porteuse n° 2 avec un retard nul.



Le travail à effectuer consiste à restaurer les données de l'utilisateur n° 1 en suivant 6 étapes :

1. Modélisation du signal MF-TDMA :

Cette étape consiste à générer un signal synthétique et représentatif d'un signal réel reçu par un récepteur MF-TDMA.

2. Analyse spectrale :

Cette étape permet d'étudier les représentations spectrales du signal MF-TDMA à partir des estimateurs classiques du périodogramme et du corrélogramme (biaisé et non biaisé). La présence de plusieurs porteuses sera particulièrement étudiée.

3. *Extraction de la porteuse par filtrage :*

Pour pouvoir isoler la porteuse de l'utilisateur n° 1, un filtrage doit être effectué sur le signal composite. Cette étape permettra de choisir le filtre approprié pour restaurer la porteuse cible.

4. *Détection de la présence du burst :*

Après avoir récupéré la porteuse cible, il faut identifier le numéro du timeslot occupé par le burst de l'utilisateur n° 1. Cette étape servira à étudier et implémenter le meilleur détecteur de présence du burst (détecteur de Neyman-Pearson).

5. *Localisation temporelle du burst :*

L'objectif de cette étape est de mesurer l'instant d'arrivée du burst dans le timeslot. Cette mesure est basée sur l'utilisation de l'intercorrélation glissante entre les échantillons du timeslot et les échantillons du préambule qui est une séquence connue par le récepteur.

6. *Restauration des données de l'utilisateur :*

Cette étape sert à restaurer les symboles de l'utilisateur n° 1 en utilisant la technique de filtrage adapté.

Etape 1 : Modélisation du signal MF-TDMA :

Cette étape consiste à générer un signal synthétique et représentatif d'un signal réel reçu par un récepteur MF-TDMA. Un tel signal peut être modélisé de la manière suivante :

$$s(t) = m(t) + b(t) \qquad 0 \leq t \leq T$$

- T correspond à la durée de la trame
- $m(t)$ est le signal utile MF-TDMA
- $b(t)$ est un bruit blanc gaussien

Le signal utile MF-TDMA peut être exprimé de la manière suivante :

$$m(t) = p_1(t) + p_2(t) \qquad 0 \leq t \leq T$$

- $p_1(t)$ est la porteuse n° 1
- $p_2(t)$ est la porteuse n° 2

La porteuse n° 1 est un signal nul dans lequel est inséré le burst de l'utilisateur n° 1. L'instant de début du burst (t_1) est fonction du numéro du timeslot et du retard du burst par rapport au début du timeslot (i.e. t_1 =*(numéro du timeslot)*(durée du timeslot)+(retard du burst dans le timeslot)*) :

$$p_1(t) = b_1(t-t_1) \quad t_1 \leq t \leq t_1+T_b$$
$$p_1(t) = 0 \text{ sinon}$$

- T_b est la durée du burst
- $b_1(t)$ est le burst de l'utilisateur n° 1

De même, la porteuse n° 2 est un signal nul dans lequel est inséré le burst de l'utilisateur n° 2. L'instant de début du burst (t_2) est fonction du numéro du timeslot et du retard du burst par rapport au début du timeslot :

$$p_2(t) = b_2(t-t_2) \quad t_2 \leq t \leq t_2+T_b$$
$$p_2(t) = 0 \text{ sinon}$$

- $b_2(t)$ est le burst de l'utilisateur n° 2

Chaque burst est un signal sinusoïdal modulé par les symboles associés :

$$b_1(t) = d_1(t)\ \cos(2\pi f_1 t) \qquad 0 \le t \le T_b$$
$$b_2(t) = d_2(t)\ \cos(2\pi f_2 t) \qquad 0 \le t \le T_b$$

- *$d_1(t)$ sont les symboles du burst de l'utilisateur n° 1*
- *$d_2(t)$ sont les symboles du burst de l'utilisateur n° 2*
- *f_1 est la fréquence de la porteuse n° 1*
- *f_2 est la fréquence de la porteuse n° 2*

Comme signalé auparavant, les symboles d'un burst sont constitués d'une séquence connue appelée préambule et d'une séquence aléatoire de type NRZ.

1- Avant de générer les échantillons du signal reçu par le récepteur, évaluer les paramètres suivants (cette partie pourra être effectuée avant la première séance de TP et les résultats pourront être reportés dans l'annexe "***liste des paramètres utiles au développement***") :

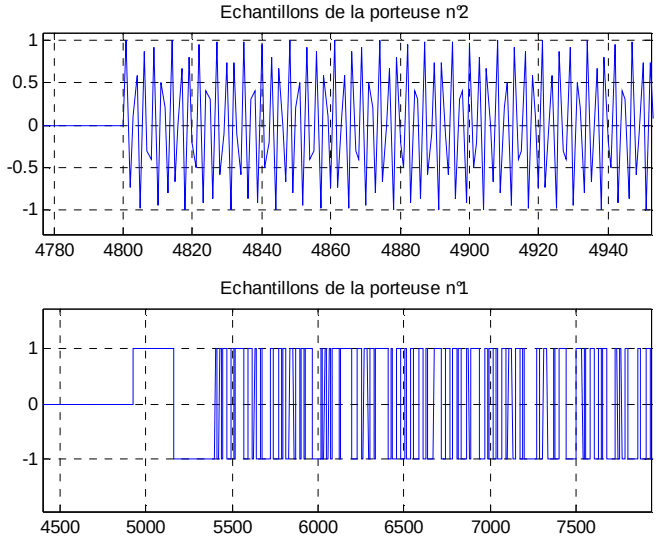
- Nombre d'échantillons par trame
- Nombre d'échantillons par timeslot
- Nombre d'échantillons par burst
- Nombre d'échantillons par symbole
- Nombre de symboles par burst
- Nombre de symboles de trafic

Note : utiliser les règles fournies en annexe pour nommer les différents paramètres et signaux lors du développement sous MATLAB.

2- Générer les symboles de chaque burst $d_1(t)$ et $d_2(t)$. Pour cela, utiliser un préambule composé de 20 symboles (+1) suivis de 20 symboles (-1) et compléter le préambule par une séquence aléatoire NRZ {-1, +1} (utiliser la fonction *rand*).

3- Générer les échantillons de chaque burst $b_1(t)$ et $b_2(t)$ en fonction de la fréquence porteuse associée (utiliser la fonction *kron*).

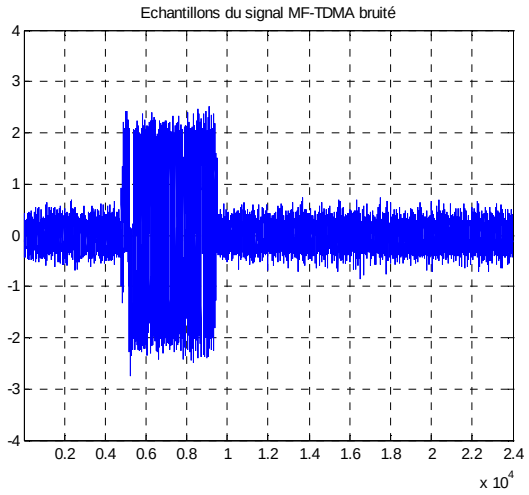
4- Evaluer l'instant de début de chaque burst et générer les échantillons de la porteuse associée. Un exemple de résultat est présenté ci-dessous :



5- Générer les échantillons du signal reçu par le récepteur en combinant les 2 porteuses et en ajoutant un bruit blanc gaussien de moyenne nulle et d'écart type 0.2 (utiliser la fonction *randn*).

6- Analyser et commenter les différents signaux obtenus.

La figure suivante donne un aperçu du résultat attendu pour la génération des échantillons du signal MF-TDMA synthétique :



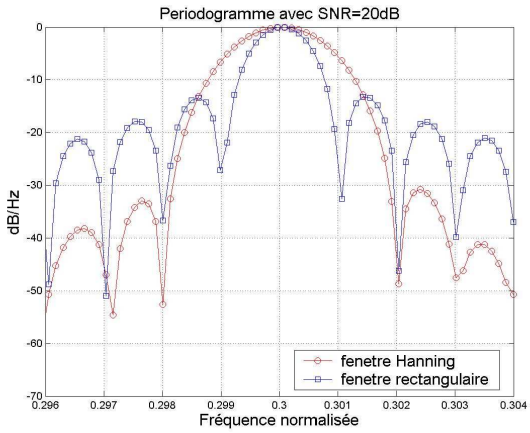
Note : Penser à sauvegarder les signaux de cette étape (utiliser la fonction *save*).

Etape 2 : Analyse spectrale :

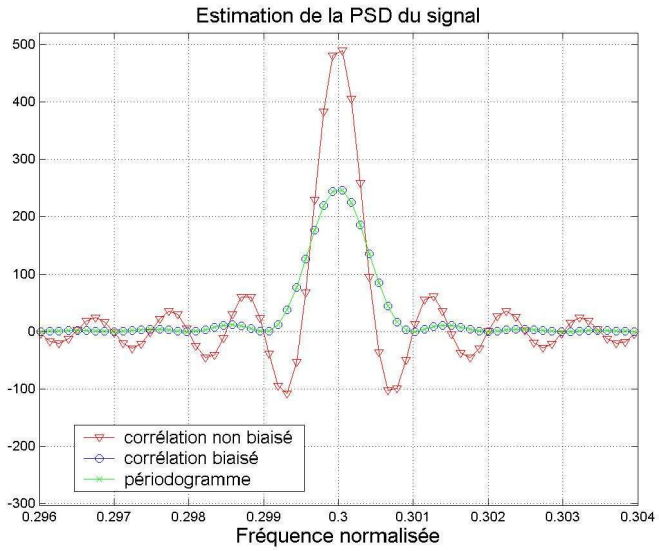
1- Cette première partie a pour but d'analyser les estimateurs de la densité spectrale de puissance obtenus par les méthodes du périodogramme et du corrélogramme exposées en annexe. Dans ce but, générer une sinusoïde à la fréquence de 60Hz perturbée par un bruit additif blanc gaussien (on pourra fixer une fréquence d'échantillonnage de 200Hz). Tracer les signaux temporels correspondants à divers rapports signal sur bruit :

$$SNR = 10\log_{10}\left(\frac{P_s}{P_b}\right),$$

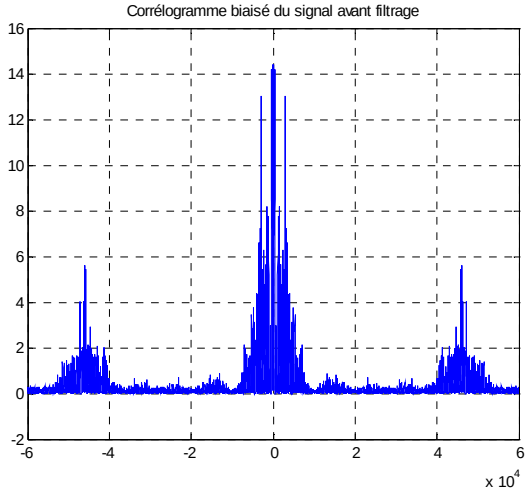
où P_s et P_b désignent les puissances du signal et du bruit. Calculer le périodogramme de la sinusoïde bruitée en utilisant les fenêtres rectangulaire et de Hanning :



Déterminer les fonctions d'autocorrélation biaisée et non biaisée de la sinusoïde bruitée (utiliser la fonction *xcorr*) puis les densités spectrales de puissance de ce signal (Lisez attentivement la remarque 3 de l'annexe). Une comparaison avec le périodogramme est illustrée ci-dessous :



2- Déterminer le spectre du signal MF-TDMA bruité à l'aide d'une des méthodes étudiées précédemment. Le cas du corrélogramme biaisé est illustré ci-dessous :



Afin d'analyser la figure précédente, déterminer la densité spectrale de puissance du signal aléatoire :

$$m(t) = d_1(t)\ \cos(2\pi f_1 t+\varphi_1) + d_2(t)\ \cos(2\pi f_2 t+\varphi_2)$$

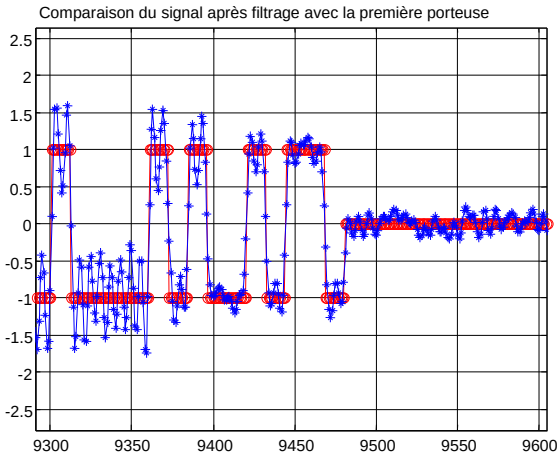
où φ_1 et φ_2 sont des phases aléatoires uniformément réparties sur $[0,2\pi]$.

En déduire le rapport des amplitudes associées aux fréquences $f_1 = 0$ et $f_2 \neq 0$ et confirmer ce rapport à partir du spectre obtenu.

Etape 3 : Extraction de la porteuse par filtrage :

Identifier quel type de gabarit de filtre est le mieux approprié pour récupérer les données de l'utilisateur n° 1. Proposer une fréquence de coupure pour le gabarit approprié et filtrer le signal MF-

TDMA (méthode fréquentielle). La figure suivante est une illustration du résultat attendu lorsqu'on observe la zone correspondante à la fin du burst :



Note : Penser à sauvegarder les signaux de cette étape (utiliser la fonction *save*).

Etape 4 : Détection de la présence du burst :

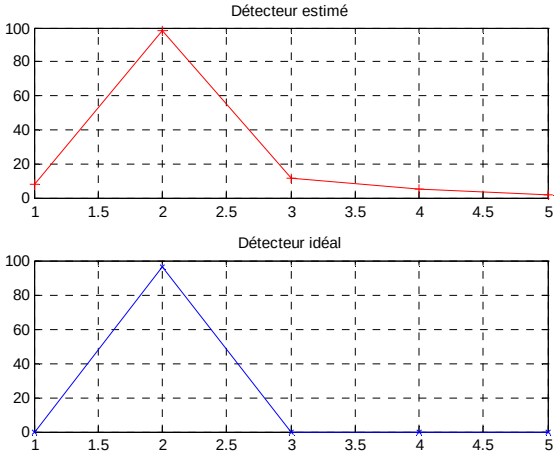
Après avoir récupéré la porteuse cible, cette étape va servir à identifier le numéro du timeslot occupé par le burst de l'utilisateur n° 1. On fait les hypothèses suivantes sur la statistique du signal filtré en absence et en présence du burst pour chaque timeslot :

$y_n = b_n$ *sous hypothèse* H_0 *(absence de burst)*
$y_n = d_n + b_n$ *sous hypothèse* H_1 *(présence de burst)*

où b_n est un bruit blanc gaussien de moyenne nulle et de variance σ^2 ($b_n \sim N(0, \sigma^2)$) et d_n est un signal NRZ (on supposera pour simplifier que le signal NRZ occupe toute la durée du timeslot).

1- Déterminer la densité de probabilité du signal y_n sous les deux hypothèses H_0 et H_1. En déduire la forme du détecteur de Neyman Pearson.

2- Pour chaque timeslot, appliquer l'algorithme de détection obtenu précédemment sur le signal filtré ainsi que sur le signal idéal (porteuse n° 1). Analyser les résultats obtenus illustrés ci-dessous :



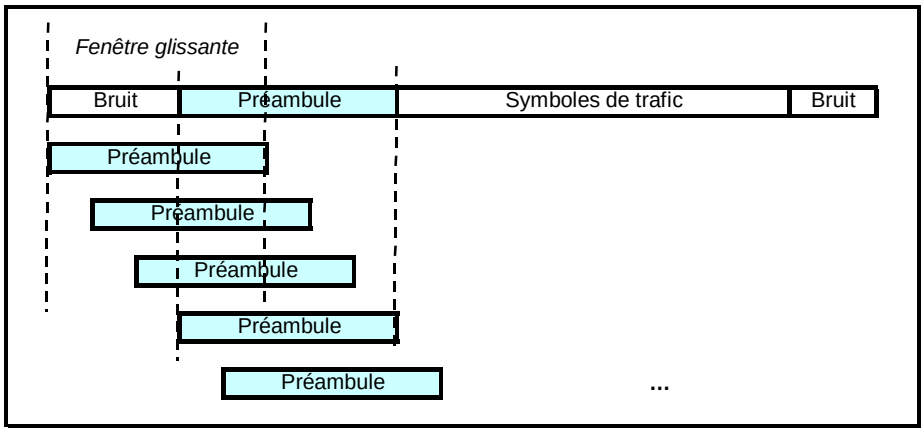
En particulier, retrouver analytiquement la valeur maximale observée lorsqu'on applique la procédure de détection (pour cela, déterminer le nombre de symboles positifs et le nombre de symboles négatifs du burst de l'utilisateur n° 1 et prendre en compte le nombre d'échantillons par symbole). En déduire une valeur appropriée du seuil de détection.

3- Extraire les échantillons du timeslot contenant le burst de l'utilisateur n° 1

Note : Penser à sauvegarder les signaux de cette étape (utiliser la fonction *save*).

Etape 5 : Localisation temporelle du burst :

L'objectif de cette étape est de mesurer l'instant d'arrivée du burst dans le timeslot. Cette mesure est basée sur l'utilisation de l'intercorrélation glissante entre les échantillons du timeslot et les échantillons du préambule qui est une séquence connue par le récepteur :



Pour cette étape, on utilise un estimateur de l'intercorrélation entre le préambule et une partie du timeslot correspondant à l'intervalle $[k+1, k+N_p]$:

$$C_{px}(k) = \frac{1}{N_p}\sum_{n=1}^{N_p} p(n)x(k+n), \qquad k = 0,...,N-N_p$$

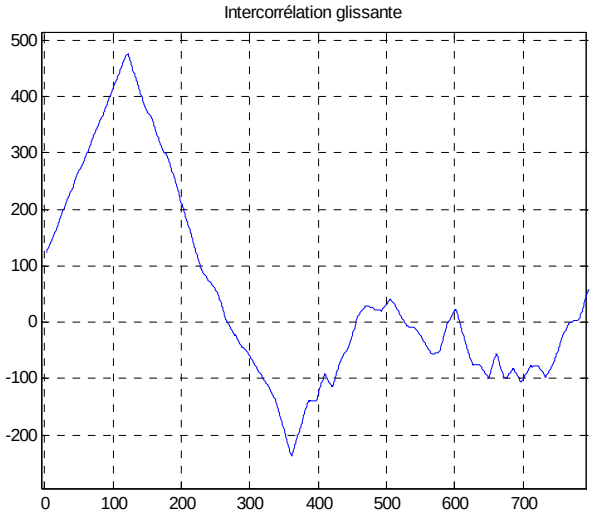
- $p(n)$ $(1 \le n \le N_p = 40x12)$ *représente les échantillons du préambule*
- N_p *est le nombre d'échantillons du préambule*
- $x(n)$ *représente les échantillons du timeslot de l'utilisateur n° 1*
- N *est le nombre d'échantillons du timeslot*

1- Sous certaines hypothèses d'ergodicité, $C_{px}(k)$ permet d'estimer l'intercorrélation :

$$K_{px}(\tau) = E\left[p(t)x(t+\tau)\right]$$

où $p(t)$ est le préambule et $x(t)$ est le signal du timeslot (i.e. succession d'un temps de garde, d'un préambule et d'un signal NRZ noyé dans un bruit additif de moyenne nulle). Déterminer $K_{px}(\tau)$ et montrer que cette fonction est maximale en $\tau = \tau_1$ où τ_1 est le retard du burst par rapport au début du timeslot.

2- Générer les échantillons associés au préambule $p(n)$ (utiliser la fonction *kron*) et tracer l'estimateur $C_{px}(k)$. La figure suivante est une illustration du résultat attendu :



3- Analyser le profil obtenu et proposer un estimateur de l'instant de début du préambule. Calculer la valeur théorique du maximum de $C_{px}(k)$ (effectuer un calcul numérique en prenant en compte le nombre d'échantillons par symbole).

4- Extraire les échantillons du burst de l'utilisateur n° 1

Note : Penser à sauvegarder les signaux de cette étape (utiliser la fonction *save*).

Etape 6 : Restauration des données de l'utilisateur :

Cette étape sert à restaurer les symboles de l'utilisateur n° 1 en utilisant la technique de filtrage adapté. Il est donc impératif de consulter le paragraphe « ***Matched filtering*** » de l'article fourni en support au sujet :

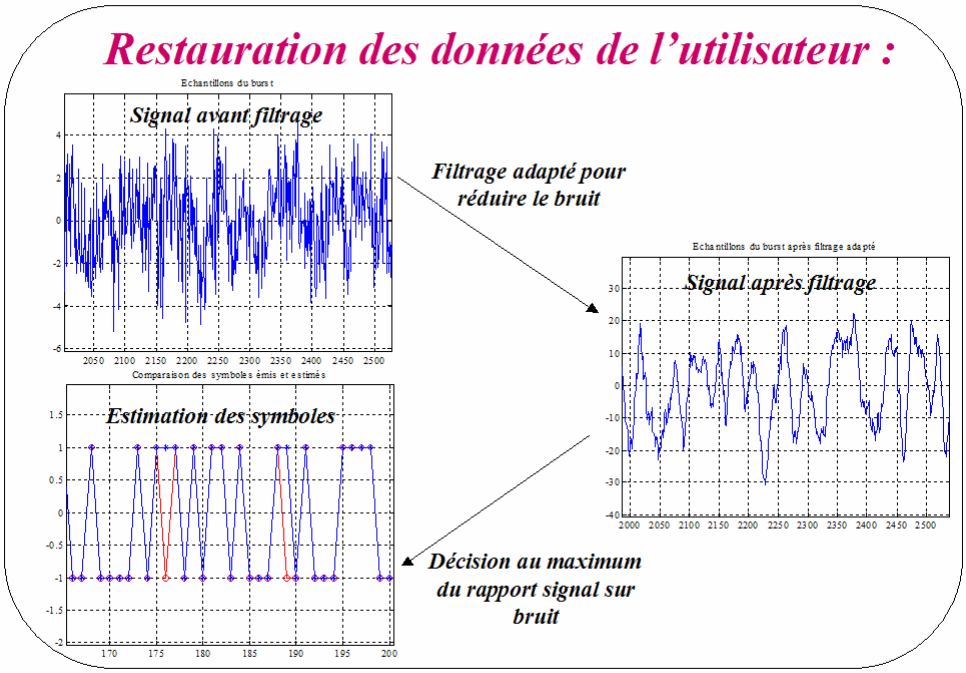
« ***Matched filtering and timing recovery in digital receivers*** »

1- Compte tenu de la nature du signal présent dans le burst, donner une formulation analytique de la réponse du filtre adapté.

2- Exprimer la réponse impulsionnelle du filtre adapté dans le cas d'un signal numérique en prenant en compte la fréquence d'échantillonnage.

3- Réaliser le filtrage adapté des échantillons du burst en utilisant la réponse impulsionnelle exprimée ci-dessus (utiliser la fonction *filter*). Comparer les signaux avant et après filtrage.

4- Estimer les symboles après filtrage adapté en comparant le signe des points obtenus après échantillonnage du signal filtré à l'instant optimal. Comparer les symboles estimés avec les symboles émis par l'utilisateur n°1 comme illustré par la figure suivante :



5- Evaluer le *TES* (Taux d'Erreurs par Symbole) défini comme étant le rapport entre le nombre de symboles erronés et le nombre de symboles émis.

6- Etudier l'influence de l'instant de décision sur le *TES*. Pour cela, tracer l'évolution du *TES* en avançant l'instant de décision d'un nombre d'échantillons allant de 1 à 6 par rapport à l'instant optimal.

7- Justifier théoriquement le résultat obtenu du *TES* quand l'instant de décision est avancé de 6 échantillons par rapport à l'instant optimal. Pour cela, étudier le comportement du filtre adapté avec la séquence suivante *{-1,+1,+1,-1,-1}*. En déduire les différents cas de décision quand le filtre adapté est appliqué à un signal NRZ et que l'instant de décision est avancé d'une demi période.

8- Etudier l'influence sur le *TES* du rapport signal sur bruit en fonction de l'instant de décision. Pour cela, ajouter un bruit blanc gaussien de moyenne nulle et d'écart type variable et évaluer le *TES* en fonction de l'instant de décision.

9- Commenter les différents profils obtenus et faire une synthèse sur l'influence des différents paramètres sur les performances du récepteur.

Annexe : Liste des paramètres utiles au développement

Trame MF-TDMA			
Durée de la trame	200	ms	D_Trame
Nombre de timeslots par porteuse	5		Nb_Timeslots_Trame
Durée du timeslot	40	ms	D_Timeslot
Durée du burst	38	ms	D_Burst
Nombre de symboles du préambule	40		Nsym_Preambule
Rythme symbole	10	kHz	Rs
Fréquence de la porteuse n°1	0	kHz	f1
Fréquence de la porteuse n°2	46	kHz	f2
Fréquence d'échantillonage	120	kHz	fe
Nombre d'échantillons par trame			Nech_Trame
Nombre d'échantillons par timeslot			Nech_Timeslot
Nombre d'échantillons par burst			Nech_Burst
Nombre d'échantillons par symbole			Nech_Symbole
Nombre de symboles par burst			Nsym_Burst
Nombre de symboles de trafic			Nsym_Trafic
Utilisateur n°1			
Numéro de la porteuse	1		
Numéro du timeslot	2		Num_Timeslot1
Retard du burst par rapport au timeslot	1	ms	Retard_Burst1
Instant de début du burst		ms	Position_Burst1
Indice du premier échantillon du burst			Debut_Burst1
Utilisateur n°2			
Numéro de la porteuse	2		
Numéro du timeslot	2		Num_Timeslot2
Retard du burst par rapport au timeslot	0	ms	Retard_Burst2
Instant de début du burst		ms	Position_Burst2
Indice du premier échantillon du burst			Debut_Burst2
Bruit			
Ecart type du bruit	0.2		Ecart_Type
Filtrage			
Fréquence de coupure du filtre passe bas		kHz	fc

Annexe : Liste des signaux utiles au développement

ETAPE	NOM	DESCRIPTION
Modélisation	*preambule*	Symboles du préambule
	trafic1	Symboles de trafic de l'utilisateur n°1
	trafic2	Symboles de trafic de l'utilisateur n°2
	data1	Symboles du burst de l'utilisateur n°1
	data2	Symboles du burst de l'utilisateur n°2
	burst1	Echantillons du burst de l'utilisateur n°1
	burst2	Echantillons du burst de l'utilisateur n°2
	porteuse1	Echantillons de la porteuse n°1
	porteuse2	Echantillons de la porteuse n°2
	mftdma	Echantillons du signal MF-TDMA
	signal	Echantillons du signal MF-TDMA bruité
Filtrage	*signal_filtre*	Echantillons du signal MF-TDMA filtré
Détection	*timeslot*	Extraction des échantillons du timeslot détecté
localisation	*sequence*	Echantillons du préambule
	burst	Extraction des échantillons du burst détecté
Restauration	*h*	Réponse impulsionnelle du filtre adapté
	burst_filtre	Echantillons du burst détecté après filtrage adapté
	data	Symboles du burst après filtrage adapté

Annexe : Périodogramme / Corrélogramme

La densité spectrale de puissance (DSP) d'un signal $x(t)$ à énergie finie est définie par :

$$s(f) = TF[K_x(\tau)] = |X(f)|^2$$

où $X(f)$ est la transformée de Fourier de $x(t)$ et $K_x(\tau)$ sa fonction d'autocorrélation. Il en découle deux méthodes d'estimation de la DSP appelées **périodogramme** et **corrélogramme.**

- ***Périodogramme :***

Lorsqu'on estime la transformée de Fourier avec l'algorithme de FFT rapide de MATLAB, on montre qu'un estimateur satisfaisant de la DSP du signal $x(t)$ appelé périodogramme est défini par :

$$\frac{1}{N}|TFD[x(n)]|^2$$

où $x(n)$ est obtenu par échantillonnage de $x(t)$.

- ***Corrélogramme :***

L'estimation de la DSP par corrélogramme comporte deux étapes :

1. Estimation de la fonction d'autocorrélation (*xcorr*) qui produit $\hat{K}_x(n)$
2. Transformée de Fourier discrète de $\hat{K}_x(n)f(n)$, où $f(n)$ est une fenêtre de pondération et $\hat{K}_x(n)$ est l'estimation biaisée ou non biaisée de la fonction d'autocorrélation.

<u>*Remarque* 1</u> :
il est important de noter que lorsque $\hat{K}_x(n)$ est l'estimateur biaisé de la fonction d'autocorrélation de $x(t)$, **le corrélogramme coïncide exactement avec le périodogramme**.

<u>*Remarque 2 : Estimateurs spectraux moyennés*</u>

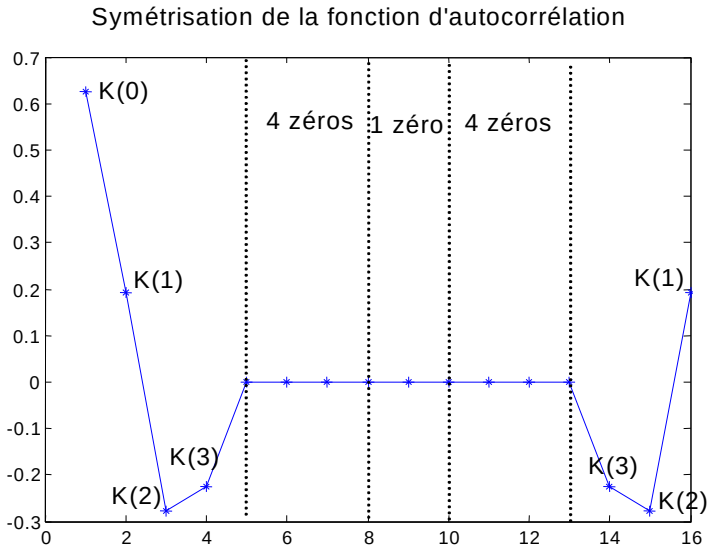
On montre que la variance des estimateurs de la DSP (corrélogramme et périodogramme) ne dépend pas de la durée du signal. Ces estimateurs ne sont donc pas convergents. Afin de réduire la variance des estimateurs, il est habituel de diviser le signal observé en plusieurs tranches et de moyenner les estimateurs obtenus sur chaque tranche. La variance des estimateurs moyennés est alors inversement proportionnelle au nombre de tranches, ce qui réduit la variance.

<u>*Remarque 3 : Implantation numérique*</u>

Si le signal numérique $x(n)$ possède N_s points, la fonction *xcorr* calcule la fonction d'autocorrélation $\hat{K}_x(n)$ pour $n = -(N_s-1),...,-1,0,1,...,(N_s-1)$ (on a donc $2N_s-1$ points). On peut ``padder'' cette autocorrélation par des zéros afin d'avoir une représentation plus précise de la DSP. L'algorithme de transformée de Fourier discrète de MATLAB nécessite une symétrisation de la fonction d'autocorrélation de la façon suivante :

- Points d'autocorrélation $\hat{K}_x(0), \hat{K}_x(1),..., \hat{K}_x(N_s-1)$
- N_z zéros
- zéro central
- N_z zéros
- Points d'autocorrélation $\hat{K}_x(N_s-1),..., \hat{K}_x(1)$

Cette procédure de symétrisation est illustrée sur la figure suivante pour $N_s = 4$ et $N_z = 4$:



Annexe : Du filtrage analogique au filtrage numérique

- ***Filtrage Analogique :***

Les opérations de filtrage consistent à « éliminer » ou « mettre en évidence » certaine(s) partie(s) de la bande de fréquence occupée par un signal donné. Dans le domaine continu, la transformée de Fourier d'un signal *x(t)* est définie par $X(f) = \int_{-\infty}^{+\infty} x(t)e^{-j2\pi ft}dt$. Filtrer un signal consiste à sélectionner une bande de fréquence d'intérêt et la mettre en évidence par rapport au reste des composantes fréquentielles du signal. Dans le domaine fréquentiel (Fourier) l'opération « naturelle » de filtrage correspond donc à une multiplication du type : $X_{filtre}(f) = X(f)H(f)$, où $X(f) = TF\{x(t)\}$ est la transformée de Fourier du signal temporel et $H(f)$ est la **fonction de transfert** du filtre qui correspond au gabarit fréquentiel pour l'opération de filtrage désirée.

L'opération de multiplication dans le domaine fréquentiel trouve son équivalent dans la convolution dans le domaine temporel (et inversement). Ainsi une opération de filtrage temporel analogique est : $x_{filtre}(t) = x(t) * h(t)$, où h(t) désigne la réponse impulsionnelle du filtre et correspond à : $h(t) = TF^{-1}\{H(f)\}$ ou encore $H(f) = TF\{h(t)\}$.

- ***Filtrage Numérique :***

Si les opérations de filtrage analogique présentées précédemment sont physiquement réalisées par des circuits électroniques (du Hardware), mettant en jeu des circuits (R, L, C), le traitement numérique mis en œuvre par l'intermédiaire de µP et de programmation informatique connaît depuis plusieurs années un essor grandissant, offrant des possibilités beaucoup plus vastes (en termes de complexité, taille, reprogrammabilité…).

Le signal numérique $x(n), n \in \{1,...,N\}$ est obtenu par échantillonnage du signal continu qui représente le signal analogique associé $x(t)$. En toute rigueur, si T_e est la période d'échantillonnage, le signal numérique obtenu en sortie du convertisseur analogique numérique est $x(n) = x(nT_e)$. Il est

donc possible comme dans le cas de signaux continus de définir les opération de filtrage dans le domaine discret. Dans cet espace de représentation, la transformée en Z est le pendant de la transformée de Fourier pour le continu :

$$\boxed{X(z) = \sum_{k=-\infty}^{+\infty} x(k) z^{-k}}$$

Une des propriétés essentielle de la transformée en Z (notée TZ) est le théorème du retard :

$$X(z)z^{-m} = TZ[x(n)]z^{-m} = TZ[x(n-m)]$$

Ainsi un filtre dans le domaine discret se définit par une réponse impulsionnelle, une suite de points *h(n)* ($n \in N$), dont la transformée en Z désigne la fonction de transfert du filtre :

$$\boxed{H(z) = \sum_{k=-\infty}^{+\infty} h(k) z^{-k}}$$

Remarque *: en pratique on n'utilise que la transformée en Z dite unilatérale,* $X(z) = \sum_{k=0}^{+\infty} x(k) z^{-k}$ *qui suppose le signal x causal (x(n)=0 pour n<0). De la même manière, le principe de causalité (« l'effet ne peut précéder la cause ») fait que la réponse impulsionnelle de tout système physique est obligatoirement nulle pour t<0. Donc pour tout système causal on a : h(n)=0 pour n<0. C'est pourquoi nous ne noterons plus les indices de la TZ.*

L'opération de filtrage dans le domaine fréquentiel, en Z, tout comme pour le cas continu est un produit de l'entrée par la fonction de transfert du filtre :

$$X_{filtre}(z) = H(z)X(z)$$

Ainsi, en utilisant le théorème du retard on montre que l'opération de filtrage peut encore s'écrire dans le domaine temporel discret :

$$X_{filtre}(z) = H(z)X(z) = \left[\sum h(k) z^{-k}\right] X(z) = \sum h(k) X(z) z^{-k}$$

En utilisant le théorème du retard et la définition de la TZ, cette expression devient dans le domaine temporel :

$$x_{filtre}(n) = \sum h(k) TZ^{-1}\left[X(z)z^{-k}\right] \underset{\substack{\text{théorème} \\ \text{du retard}}}{=} \sum_k h(k) x(n-k)$$

qui n'est autre que l'expression de la formule de convolution discrète. C'est l'opération de filtrage qui est concrètement implantée dans un calculateur : la sortie à un instant donné n est une combinaisons linéaire des échantillons d'entrée aux instants inférieurs à n.

Cependant on vise généralement des opérations de filtrage dites temps réel (calcul de l'échantillon de sortie pendant l'intervalle de temps entre 2 échantillons d'entrée) ; or la charge de calcul de cette opération générale croit avec le temps (i.e., les indices sont tels que k varie entre 0 et l'infini dans l'équation précédente), plus on a d'échantillons passés plus le calcul d'un échantillon nécessite d'opérations élémentaires. Afin de limiter cette charge de calcul, on considère une classe de filtres linéaires ayant une charge calculatoire constante au cours du temps qui est la classe des filtres récursifs qui possèdent une expression générale de la forme :

$$x_{filtre}(n) = -\sum_{k=1}^{M} a_k x_{filtre}(n-k) + \sum_{k=0}^{N} b_k x(n-k)$$

Dans ce cas la fonction de transfert associée est une fraction rationnelle en Z :

$$H(z) = \frac{\sum_{k=0}^{N} b_k z^{-k}}{1 + \sum_{k=1}^{M} a_k z^{-k}}$$

Pour définir un tel filtre numérique, il suffit donc de déterminer les coefficients a_k et b_k intervenant dans $H(z)$ (sous MATLAB, voir la commande *filter*).

Matched filtering and timing recovery in digital receivers

A practical look at methods for signal detection and symbol synchronization.

By Louis Litwin

The acquisition of a signal in a digital communications system requires the convergence of several signal processing algorithms before the receiver can output meaningful data. These algorithms are adaptive in nature and need to process multiple received symbols before convergence is achieved. Because of the feedback nature inherent in these algorithms, the various adaptive receiver sections are often referred to as loops. Certain receiver loops depend on other loops. Depending on the algorithm implemented, it is possible that a given loop cannot converge until one or more previous loops have sufficiently converged. The major receiver loops are listed in the order that they typically need to converge, although the order may sometimes vary depending on the implementation.

Stage 1 – AGC

This stage scales the signal to a known power level. Automatic gain control (AGC) is typically handled in the analog domain to properly scale the signal for analog-to-digital (A/D) conversion because A/D converters have a limited dynamic range. If the received signal strength is too high, the A/D conversion process will introduce a type of distortion known as clipping. If the signal strength is too low, the signal variations will toggle only a few bits at the A/D, and distortion will occur because of severe quantization.

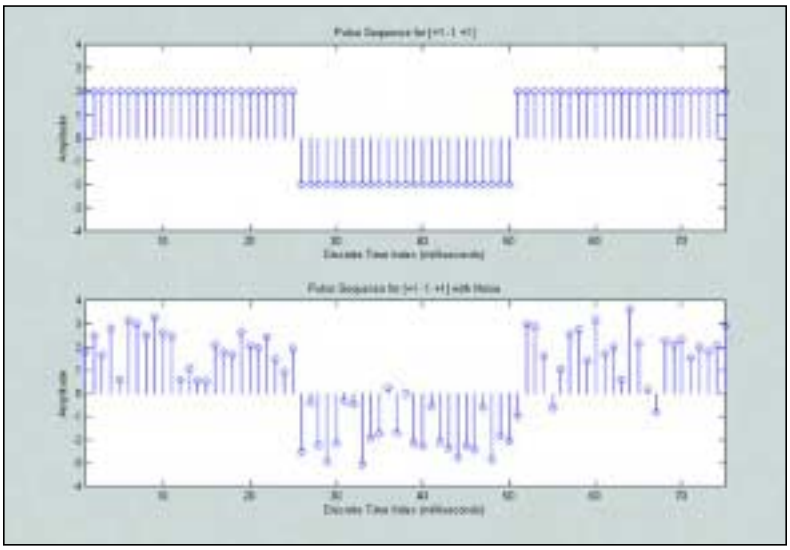


Figure 1. Rectangular pulse train shown with and without noise. The SNR for the lower plot is roughly 5 dB.

The convergence of the AGC loop is also required for several other receiver blocks. Certain parameters and gains for various adaptive algorithms, as well as boundaries for symbol decision regions at the slicer, are based on the signal being at a known power level. In addition to the analog AGC, many receivers implement an additional AGC in the digital domain for fine signal scaling.

Stage 2 – timing recovery

The purpose of the timing recovery loop is to obtain symbol synchronization. Two quantities must be determined by the receiver to achieve symbol synchronization. The first is the sampling frequency. Locking the sampling frequency requires estimating the symbol period so that samples can be taken at the correct rate. Although this quantity should be known (e.g., the system's symbol rate is specified to be 20 MHz), oscillator drift will introduce deviations from the stated symbol rate.

The other quantity to determine is sampling phase. Locking the sampling phase involves determining the correct time within a symbol period to take a sample. Real-world symbol pulse shapes have a peak in the center of the symbol period. Sampling the symbol at this peak results in the best signal-to-noise-ratio and will ideally eliminate interference from other symbols. This type of interference is known as intersymbol interference.

Stage 3 – carrier recovery

An oscillator at the transmitter generates a sinusoidal carrier signal that ideally exists at some known carrier frequency. Due to oscillator drift, the actual frequency of the carrier will deviate slightly from the ideal value. This carrier is multiplied by the data to modulate the signal up to a passband center frequency. At the receiver, the passband signal is multiplied by a sinusoid generated by the local oscillator.

Preferably, the frequency of the local oscillator will exactly match the frequency of the oscillator used at the transmitter. In practice, their frequencies differ and, instead of demodulation bringing the signal to baseband, the signal will be near baseband with some frequency offset. The presence of this frequency offset will cause the received signal constellation to rotate. This "spinning" effect must be removed before accurate symbol decisions can be made. The purpose of the carrier recovery loop is to remove this frequency offset so that the signal can be processed directly at baseband.

Stage 4 – channel equalization

Transmitting a signal through a multipath channel results in a received signal that consists

Continued on page 36

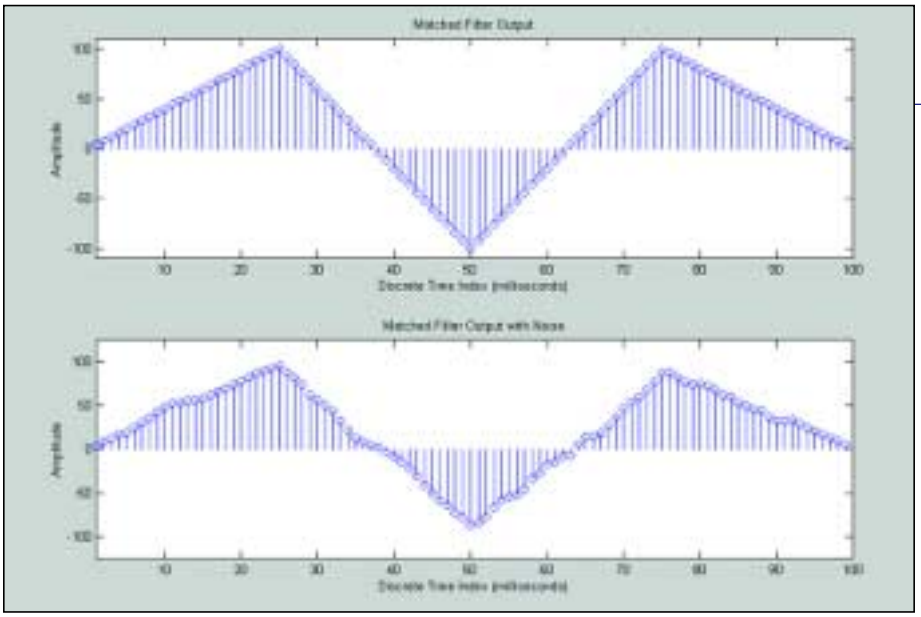


Figure 2. Matched filter outputs for signals in Figure 1.

of several delayed and scaled versions of the transmitted signal. Multiple versions of the signal occur because the receiver may pick up the signal that traveled the direct path from transmitter to receiver, as well as multiple reflected paths. The multipath channel can be viewed as a linear filter. The equalizer is an adaptive filter that attempts to remove intersymbol interference by undoing the filtering effects of the multipath channel.

Timing recovery algorithms adaptively determine the correct time to sample the symbol pulse shape. Thus, before entering into a discussion on timing recovery, some background material will be provided on the topics of matched filtering and pulse shaping.

Signal detection

A basic problem in digital communications is the detection and estimation of a transmitted pulse in the presence of additive white Gaussian noise (AWGN). Imagine the simple case of a rectangular pulse, such as that shown in the top half of Figure 1. A data symbol of +1 is indicated by transmitting a pulse with an amplitude of +2, and similarly, a data symbol of –1 is indicated by transmitting a pulse with an amplitude of –2. The period of these pulses, T, is 25 ms. Note that the one pulse is simply a negated version of the other.

When two pulse shapes are used that have the same energy and a cross-correlation of –1, the signaling set is said to be antipodal. The estimation of the transmitted pulse shape is trivial for the case of no noise. The receiver simply takes one sample every T seconds and determines whether the sample equals +2 or –2.

Such a scheme no longer works in the presence of AWGN. White noise has infinite average power and can therefore easily drown out the received signal that is of limited power. The lower half of Figure 1 shows the same pulse sequence for the case of noise with a signal-to-noise ratio (SNR) of 5 dB. Note that the noise has severely distorted the signal, even flipping the sign of some samples. Because all practical communications systems have some non-trivial noise level, a more robust signal estimation scheme is needed.

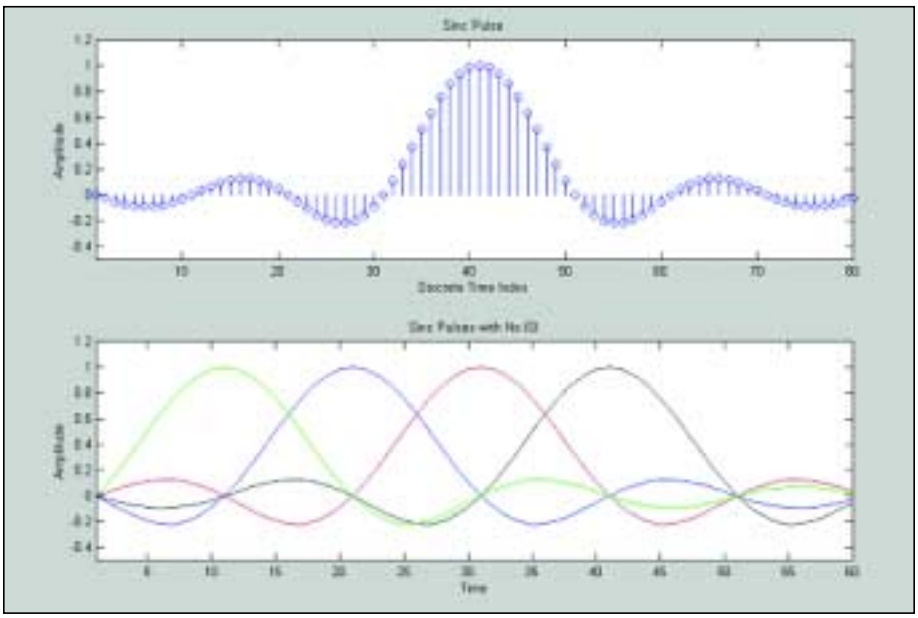


Figure 3. Sinc pulse examples.

Matched filtering

Practical receivers estimate the transmitted signal by using a technique known as matched filtering. A receiver employing such a technique filters the received signal with a filter whose shape is "matched" to the transmitted signal's pulse shape. The output of the filter is then sampled at time T. The matched filter's pulse shape is a time-reversed version of the transmit pulse shape. Thus, if the transmit pulse shape $h(t)$ is defined as:

$h(t)$ for $0 \le t \le T$

then the ideal matched filter's response $h_m(t)$ is:

$h_m(t) = h(T–t)$ for $0 \le t \le T$

Such processing has two advantages. One advantage is that typical pulse shapes have a low-pass response. By filtering the received signal with such a filter at the receiver, the frequencies containing the data signal are passed while the remaining frequencies are attenuated. This matched filtering limits the amount of the noise spectrum that is passed on to subsequent stages in the receiver. A second advantage is that a matched filter correlates the received signal with the transmit pulse shape over the symbol period T.

Recall that passing a signal $r(t)$ through a filter $h_m(t)$ is a convolution operation. The convolution of these two signals can be written as:

$$y(t) = \int_{T}^{0} r(t) h_m(T - t)\, dt$$

where $y(T)$ represents the output of the matched filter sampled at time T. However, the matched filter's response

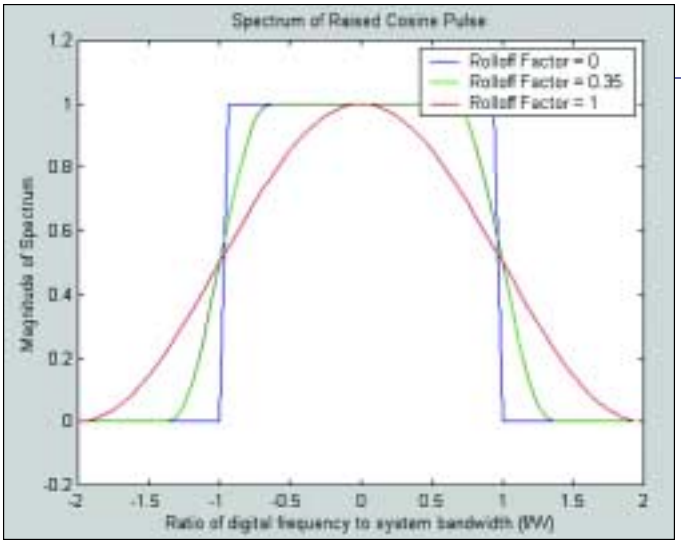


Figure 4. Spectrum of raised cosine pulse for different values of the roll-off factor.

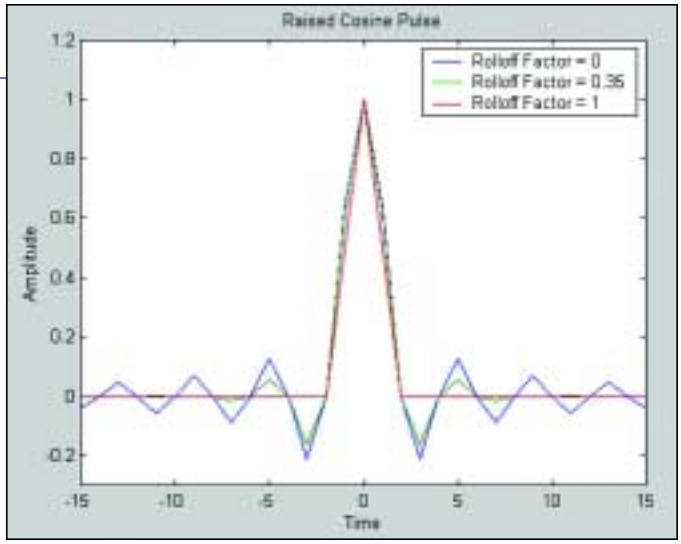


Figure 5. Raised cosine pulse for different values of the rolloff factor.

was defined as $h_m(t) = h(T - t)$. By substituting this definition into the above equation, the following integral is obtained:

$$y(t) = \int_0^T r(t)h(T-(T-t))dt = \int_0^T r(t)h(t)dt$$

The above equation is the cross-correlation (sampled at time T) of $r(t)$ with $h(t)$ for a lag of 0. Thus, this simple derivation has illustrated how matched filtering effects the correlation of the received signal with the matched filter. Such processing results in a correlation gain by integrating the received signal energy while averaging out the zero-mean AWGN.

An example of matched filtering is shown in Figure 2. The received signals for the top and bottom halves of the figure are the signals shown in Figure 1. The matched filter used was:

$$h_m(t) = 2 \text{ for } 0 \le t \le T$$

Note that sampling the matched filter output at time T = 25 ms provides the sample with the highest SNR. The samples from Figure 1 had an amplitude of 2, whereas the matched filter output (when sampled properly) has a value of 100. The value of 100 represents the integral, over the time period T, of the received signal pulse shape exactly lined up with the matched filter response. The value of this peak can be calculated as follows:

$$y(t) = \int_0^{25} (2 \bullet 2)dt = 25(4) - 0(4) = 100$$

This simple example illustrates how matched filtering provides the receiver with a stronger signal to work with compared to directly sampling the received signal. The processing gain of matched filtering is especially apparent for the example with an SNR of 5 dB.

Note that the received signal is severely distorted by noise, but the matched filter's output is still close to its ideal value for the case of no noise. This result is possible because the matched filter filters out the higher frequency noise and then integrates the remaining lower frequency noise over a time period of T ms. Because AWGN is zero-mean, this integration effectively averages out the noise.

As can be seen from Figure 2, it is important to sample the matched filter's output exactly at time T to obtain the sample with the highest SNR. Sampling the matched filter's output at some time $T + \Delta$, (where Δ represents a receiver timing offset) will significantly reduce the effective SNR seen by subsequent receiver blocks. This example shows the importance of keeping Δ as close to zero as possible and thus provides motivation for the inclusion of a timing recovery loop in the receiver.

Before discussing specific timing recovery algorithms, the next sections will first illustrate the problems inherent in using this rectangular pulse shape. A more practical pulse shape known as a root-raised cosine pulse will then be introduced.

Ideal pulse shaping

Although the use of matched filtering gives the optimum performance in the presence of AWGN, there is still a problem with using a rectangular pulse shape. Recall from Fourier theory that a rectangular pulse in the time domain is equivalent to a sinc pulse in the frequency domain. Because the tails of the sinc pulse extend to infinity, such a pulse shape would require a system with infinite bandwidth.

The ideal pulse shape should have two properties. It should have a limited bandwidth to allow transmission on practical band-limited systems. The pulse shape should also have zero intersymbol interference if sampled at the correct time interval. That is, when a pulse train is sampled every T seconds, the value of the sample at time T should only be due to the current pulse. And there should be no interference from the other transmitted pulses.

In other words, ideally, $h(t) = 1$ for $t = 0$ and $h(t) = 0$ for $t = \pm kT$ where k is a non-zero integer. An ideal pulse shape that meets these requirements is a time-domain sinc pulse. An example of a sinc pulse for which $T = 10$ is shown in Figure 3. Note that the pulse takes on a value of 1 at its peak and its zero-crossings occur at intervals of integer multiples of ±10 samples away from the peak. The lower half of the figure shows a pulse train of four pulses. This example illustrates how the peak of any given pulse lines up with the zero-crossings of the remaining pulses. Therefore, there is no ISI.

Practical pulse shaping

Although the sinc pulse represents the ideal pulse shape, it cannot be implemented in practice because the pulse extends in time for infinite duration. The infinite signal duration is due to the discontinuities in the sinc pulse's rectangular-shaped spectrum. Signals with discontinuities in their spectrum are physically unrealizable. However, practical pulse shapes can be formed by smoothing the roll-off of the spectrum and allowing it to occupy excess bandwidth beyond that which is needed for

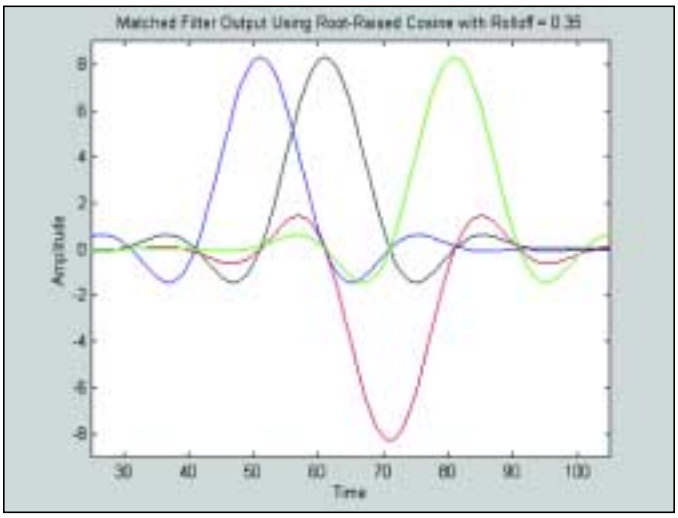


Figure 6. Zero ISI at output of matched filter using a root-raised cosine pulse.

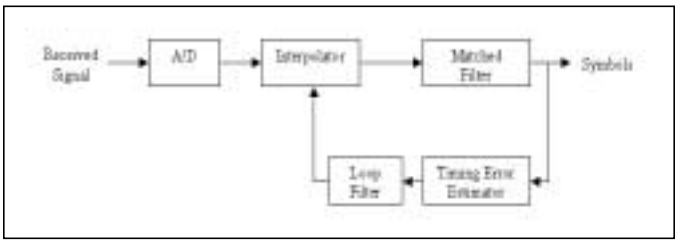


Figure 7. Example of an all-digital timing recovery loop.

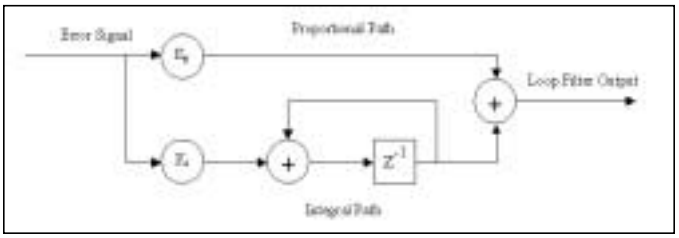


Figure 8. Structure of a typical second-order loop filter.

the spectrum of the ideal sinc pulse.

One pulse shape that has properties similar to the sinc pulse, but without the frequency-domain discontinuities, is the raised cosine pulse. The raised cosine pulse has a parameter known as the rolloff factor. The value of the rolloff factor determines how rapidly the frequency-domain spectrum of the pulse rolls off. The raised cosine pulse is identical to the sinc pulse when the rolloff factor is equal to zero. As the rolloff factor is increased, the spectrum begins to decay more gradually and this increased rolloff causes the pulse to occupy more bandwidth. When the rolloff reaches its maximum value of one, the spectrum requires twice as much bandwidth as the pulse with a rolloff of zero. Practical digital communications systems often use a rolloff factor of between 0.10 and 0.35. A pulse with a rolloff factor of 0.35 occupies 35% more bandwidth than the ideal sinc pulse. Figure 4 shows the effect of the rolloff factor on the pulse's spectrum. Figure 5 contains time-domain raised cosine pulses for the same rolloff factors used in Figure 4.

The pulses in Figure 5 exhibit zero-crossings at integer multiples of the symbol period. Thus, even with non-zero roll-off factors, the raised cosine pulse maintains this desirable (from the standpoint of no ISI) property of the sinc pulse. The choice of the roll-off factor is a trade-off between required bandwidth and the duration of the time-domain pulse. Note that the tails of the time-domain pulse are reduced for higher values of the roll-off factor. The smaller tails are desirable from a timing recovery standpoint because, in the presence of a timing offset, they will contribute less to ISI compared to the larger tails of the sinc pulse.

The most popular pulse shape used in practical communications systems is the root-raised cosine pulse. The root-raised cosine pulse is formed by taking the square root of a raised cosine pulse. This pulse shape is used to split the spectral characteristics of the raised cosine pulse equally between the transmitter and receiver.

By matched-filtering the root-raised cosine pulse and then sampling it at the symbol period, the root-raised cosine pulse is essentially squared. Thus, the output of the matched filter has a raised cosine pulse response.

An example of the matched filter output for a pulse train of root-raised cosine pulses with a rolloff factor of 0.35 is shown in Figure 6. Note that the matched filter output exhibits zero ISI because of the locations of the zero crossings for the case of perfect timing.

Timing recovery

The previous sections have shown how intersymbol interference can be avoided by sampling the matched filter output at its peak, which occurs every T seconds. The purpose of the timing recovery loop is to alter, as necessary, the sampling frequency and sampling phase to sample the matched filter at the peaks. If the timing recovery loop is operating properly, it will provide the downstream processing blocks with symbols that were sampled at the highest SNR points available.

An example of a typical all-digital timing recovery loop is shown in Figure 7. After A/D conversion, the signal is passed through an interpolator. The interpolator is able to generate samples in between those actually sampled by the A/D (i.e., it interpolates). By generating these intermediate samples as needed, the interpolator can adjust the effective sampling frequency and phase.

Interpolation is accomplished by first inserting $N–1$ zeros in between the data samples (upsampling by a factor of N). The upsampled signal passes through a lowpass interpolation filter to remove the aliases caused by upsampling. The resulting interpolated signal is a smoothed version of the original signal and it contains N times as many samples.

Following interpolation, the output of the matched filter is sent to a timing error estimator that can use a number of different algorithms to generate a timing error. The control signal for the interpolator is formed by filtering this error signal using a standard second-order loop filter containing a proportional and an integral section. An example of a typical second-order loop filter is shown in Figure 8.

The second-order loop filter consists of two paths. The proportional path multiplies the timing error signal by a proportional gain K_p. From control theory, it is known that a proportional path can be used to track out a phase error; however, it cannot track out a frequency error. For a timing recovery loop to track out a sampling frequency error, a loop filter containing an integral path is needed. The integral path multiplies the error signal by an integral gain K_i

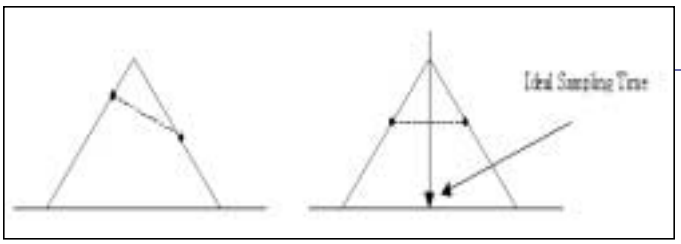


Figure 9. Method of generating error for early-late gate algorithm. The left plot shows where sampling is occurring too late. When sampling occurs at the right time, the early and late samples will be at the same amplitude.

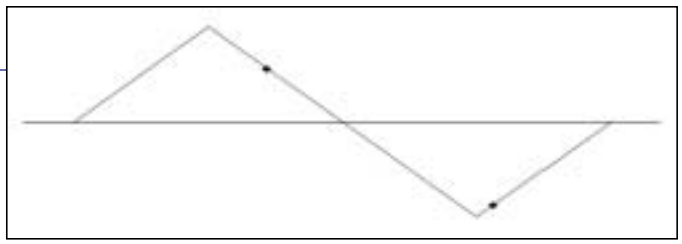


Figure 11. Timing is fast: $e_n = (-0.8 \bullet 1) - (-1 \bullet 0.5) = -0.3$.

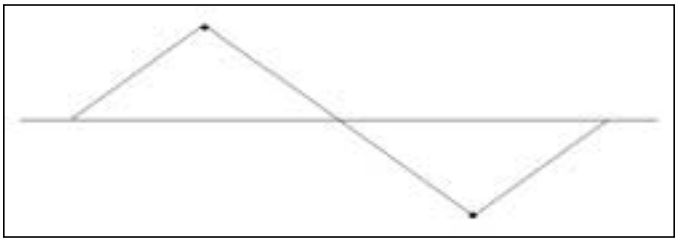


Figure 10. Correct timing: $e_n = (-1 \bullet 1) - (-1 \bullet 1) = 0$.

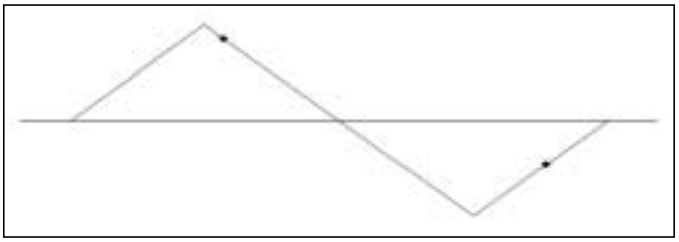


Figure 12. Timing is slow: $e_n = (-0.5 \bullet 1) - (-1 \bullet 0.8) = 0.3$.

and then integrates the scaled error using an adder and a delay block. A second-order filter, such as that shown in Figure 8, can track out both a sampling phase and a sampling frequency error.

Early-late gate algorithm

This timing recovery algorithm generates its error by using samples that are early and late compared to the ideal sampling point. The generation of the error requires at least three samples per symbol. The method of generating the error is illustrated in Figure 9. The left plot is for the case where sampling is occurring late. Note that the early and late samples are at different amplitudes. This difference in amplitude is used to derive an error for the timing recovery loop. Once the timing recovery loop converges, the early and late samples will be at equal amplitudes. The sample to be used for later processing is the sample that lies in the middle of the early and late samples. One drawback of the early-late gate algorithm is that it requires at least three samples per symbol. Thus, it is impractical for systems with high data rates.

Mueller and Muller Algorithm

The Mueller and Muller algorithm only requires one sample per symbol. The error term is computed using the following equation:

$$e_n = (y_n \bullet y_{n-1}) - (y_n \bullet y_{n-1})$$

where y_n is the sample from the current symbol and y_{n-1} is the sample from the previous symbol. The slicer (decision device) outputs for the current, and previous symbol are represented by $\hat{y}_n$ and $\hat{y}_{n-1}$, respectively. Examples of the value for the Mueller and Muller error for the cases of different timing offsets are shown in Figure 10, Figure 11 and Figure 12. One drawback of this algorithm is that it is sensitive to carrier offsets, and thus carrier recovery must be performed prior to the Mueller and Muller timing recovery.

Gardner algorithm

The Gardner algorithm has seen widespread use in many practical timing recovery loop implementations. The algorithm uses two samples per symbol and has the advantage of being insensitive to carrier offsets. The timing recovery loop can lock first, therefore simplifying the task of carrier recovery. The error for the Gardner algorithm is computed using the following equation:

$$e_n = (y_n - y_{n-2})\, y_{n-1}$$

where the spacing between y_n and y_{n-2} is T seconds, and the spacing between y_n and y_{n-1} is $T/2$ seconds.

The following figures illustrate how the sign of the Gardner error can be used to determine whether the sampling is correct (Figure 13), late (Figure 14) or early (Figure 15). Note that the Gardner error is most useful on symbol transitions (when the symbol goes from positive to negative or vice-versa). The Gardner error is relatively small when the current and previous symbol have the same polarity.

A simulation was run for a timing recovery loop that used the Gardner algorithm and the results are shown in Figure 16 (page 48). The top plot shows the matched filter output samples for the in-phase component of the signal.

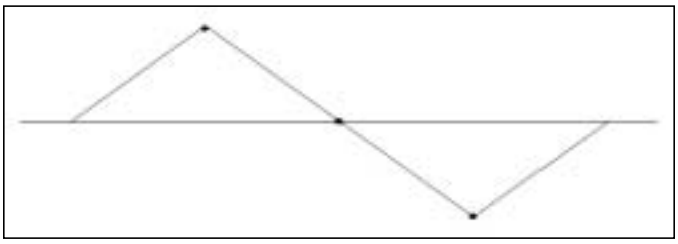


Figure 13. Correct timing: $e_n = (-1 - 1) \bullet 0 = 0$.

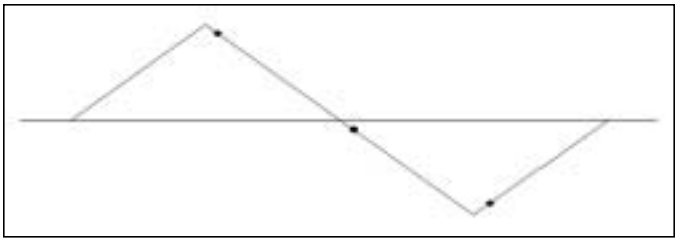


Figure 14. Timing is late: $e_n = (-0.8 - 0.8) \bullet (-0.2) = 0.32$.

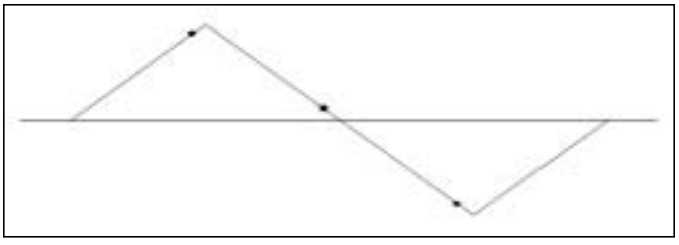


Figure 15. Timing is early: $e_n = (-0.8 - 0.8) \bullet (0.2) = -0.32$.

Notice that the timing recovery loop converges after about 600 symbols have been processed. At this point, the output of the matched filter takes on values of +1 and –1. The values are fairly constant because the matched filter output is being sampled near the ideal center point. During the first 600 symbols, when the loop is still converging, the matched filter samples take on a wide range of amplitudes. This variance in the matched filter output is because of ISI caused by sampling the output at points other than the ideal center point. The bottom plot shows the Gardner error e_n vs. time.

Other methods of timing recovery

The ideal case is to have the transmitter and receiver running off of the same clock. Although this situation is typically impossible in a wireless communications system, it can be implemented in some wired systems, such as computer networks. In such an ideal system, a timing recovery loop is not needed because synchronization is explicit.

Another alternative is to have the clock frequency transmitted along with the data. The receiver can recover this clock signal with a narrowband bandpass filter tuned to that frequency. Although this method is used in some practical systems, it is generally inefficient because the transmission of the clock signal consumes both bandwidth and transmitter power that could have otherwise been used for sending user data. In addition, other decision-directed and non-decision-directed algorithms exist for generating an error signal for a timing recovery.

The Gardner's algorithm presented here represents a good starting point for practical implementations because of its robustness to carrier offsets, simple implementation and modest oversampling requirement of two samples per symbol. The interested reader can learn more about timing recovery algorithms by consulting the references listed at the end of this article.

Conclusions

This article presents the problem of detecting pulses transmitted across an AWGN channel. Merely sampling the pulses at the receiver once every symbol period is found to be ineffective because of the signal distortion due to the presence of noise with an infinite bandwidth. The concept of the matched filter receiver is introduced as a way to limit the noise at the receiver, as well as to provide a high SNR sampling point due to the correlation gain. The implementation of symbol timing synchronization is shown to be a vital process in obtaining the best SNR sampling point while also avoiding intersymbol interference.

RF

Acknowledgments

The author would like to thank Kumar Ramaswamy and Paul Knutson (both of Thomson Multimedia) for their help in proofreading this article and also for the numerous (and sometimes late-night) discussions on the topic of timing recovery.

Continued on page 48

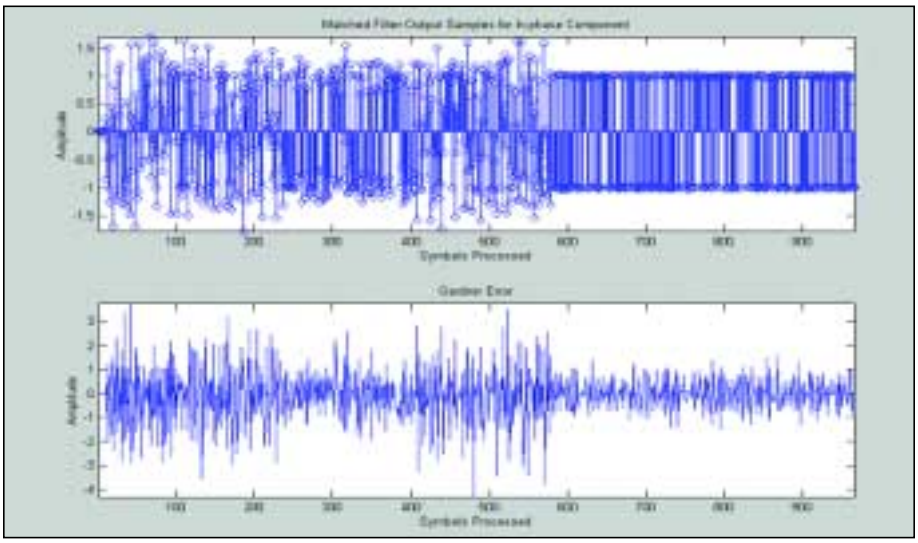


Figure 16. Simulation results using Gardner algorithm on QPSK data.

References

[1] F. M. Gardner, "A BPSK/QPSK Timing Error Detector for Sampled Receivers," *IEEE Transactions on Communications*, vol. COM-34, pp. 423-429, May 1986.

[2] D. N. Godard, "Passband Timing Recovery in an All-digital Modem Receiver," *IEEE Transactions on Communications*, vol. COM-26, pp. 517-523, May 1978.

[3] S. Haykin, Communication Systems, Wiley, NY, 1994.

[4] K. H. Mueller and M. S. Muller, "Timing Recovery in Digital Synchronous Data Receivers," *IEEE Transactions on Communications*, vol. COM-24, pp. 516-531, May 1976.

[5] J. G. Proakis, Digital Communications, McGraw-Hill, NY, 1995.

About the author

Louis Litwin is a member of the technical staff with Thomson Multimedia Corporate Research where he is working on 3G CDMA technology for mobile applications. He received his M.S. degree in electrical engineering from Purdue University and his B.S. degree in electrical engineering from Drexel University.

Liste des normes de la version 10 au mois d'août 2007

Normes	Description
Transmission	
DVB-S	Transmission satellite
DVB-S2	Transmission satellite version 2
DVB-C	Transmission câble
DVB-CS	Transmission satellite
DVB-T	Transmission terrestre
DVB-T2	Transmission terrestre version 2
DVB-H	Récepteur portable
DVB-SH	Récepteur portable transmission par satellite
DVB-MDS	Transmission satellite multipoint
DVB-DSNG	Transmission satellite temporaire
Multiplexage	
DVB-SI	Définition des tables
DVB-DATA	Diffusion (*broadcast*) des données
DVB-SSU	Mise à jour logicielle des récepteurs
DVB-TVA	Enregistreur digital personnel
DVB-GSE	Données génériques
Encodage des sources	
DVB-MPEG	Utilisation d'un système MPEG2
Sous-titrage	
DVB-SUB	Sous-titres
Accès conditionnel	
DVB-CSAS	Support de l'encryptage et de l'accès conditionnel
DVB-SIM	Simulcrypt
Analyse et mesure	
DVB-M	Analyse et mesure des flux DVB/MPEG

Interactivité	
DVB-NIP	Service interactif, protocole non spécifié
DVB-RCC	Service interactif par câble
DVB-RCP	Service interactif par réseau commuté
DVB-RCD	Service interactif par DECT
DVB-RCL	Service interactif par LMDS
DVB-RCG	Service interactif par GSM
DVB-RCCS	Service interactif par satellite
DVB-RCS	Service interactif par satellite
DVB-RCT	Service interactif par transmission terrestre
DVB-RCGPRS	Service interactif par GPRS
MHP	
DVB-MHP	Plate-forme multimédia
DVB-PCF	Contenu au format pour portable
Protection du contenu/Gestion des droits	
DVB-CPCM	Protection du contenu/Gestion des droits
Interfaçage	
DVB-PDH	Réseau PDH
DVB-SDH	Réseau SDH
DVB-ATM	Réseau ATM
DVB-HAN	Réseau HAN
DVB-HLN	Réseau HLN
DVB-CI	Interface commune pour l'accès conditionnel
DVB-PI	Interface pour CATV/SMATV et ASI
DVB-IRDI	Interface entrée/sortie décodeur
Protocole internet	
DVB-IPTV	Par réseau IP
DVB-IPDC	IP datacast sur réseau DVB-H