Memo intro telecom

Traitement signaux

Intérêt de la numérisation d'un signal :
- tolérant au bruit
- facile à régénérer
- facile à implanter électroniquement
- formalisme identique pour la parole, l'image et les données

PCM `Pulse Code Modulation`

1. échantillonnage (Sampling)
2. quantification (Quantizing)
3. encodage (Encoding)

Représentation d'un signal analogique

Représentation temporelle

- le signal analogique devient un signal numérique discretisé en temps
- caractérisé par une **amplitude** et une **fréquence**
- modélisé par une sinusoïde

$$s(t) = Asin(\omega t + \phi) \tag{1}$$

- A l'amplitude (ou valeur crête)
- ω la pulsation (aussi : $\omega = 2\pi f = \frac{2\pi}{T}$, avec f la fréquence d'échantillonnage et T la période)
- t la période d'échantillonnage
- ϕ la phase à l'origine

Représentation fréquentielle (spectre)

- somme infinie de fonctions sinusoïdales
- ces fonctions sinusoïdales sont les harmoniques du signal
- permet de représenter tout signal périodique

Autrement :

$$s(t) = a_0 + \sum_{n=1}^{\infty}(a_n cos(2\pi n F_0 t) + b_n sin(2\pi n F_0 t)) \tag{2}$$

où a_n et b_n sont les **coefficients de la série de Fourier**.
On peut aussi écrire cette relation sous la forme d'un développement en **harmoniques** (composante dont la fréquence est un multiple de la fréquence fondamentale F_0 :

$$s(t) = a_0 + \sum_{n=1}^{\infty} c_n cos(2\pi F_0 t + \phi_n) \tag{3}$$

avec

$$c_n = \sqrt{a_n^2 + b_n^2} \ et \ \phi_n = arctg(-\frac{b_n}{a_n}) \tag{4}$$

ou encore

$$s(t) = \sum_{n=-\infty}^{\infty} S(nF_0)e^{j2\pi n F_0 t} \tag{5}$$

avec

$$S(nF_0) = \frac{1}{2}(a_n - jb_n) \tag{6}$$

Les valeurs négatives de n sont introduites dans un but de simplification, mais, étant donnée que $s(t)$ est réel, nous avons :

$$a_{-n} = a_n \ et \ b_{-n} = -b_n \tag{7}$$

Spectre

Le **spectre** en fréquence du signal représente l'amplitude du fondamentale F_0 et des différentes harmoniques en fonction de la fréquence $f = nF_0$.
ATTENTION : le spectre d'une fonction périodique est discontinu et composé de raies dont l'écart minimum est, sur l'axe des fréquences, F_0.
$S(nF_0)$ représente les composantes du spectre en fréquence de $s(t)$, grandeur en général complexe, qui a donc pour module :

$$|S(nF_0)| = \frac{1}{2}\sqrt{a_n^2 + b_n^2} = \frac{c_n}{2} \tag{8}$$

et pour phase :

$$\phi(nF_0) = arctg(-\frac{b_n}{a_n}) \tag{9}$$

En conclusion, le spectre $S(f)$ du signal périodique $s(t)$ est alors représenté par l'expression suivante :

$$S(f) = \sum_{n=-\infty}^{\infty} S(nF_0).\delta(f - nF_0) \tag{10}$$

avec :

$$S(nF_0) = |S(nF_0)|e^{-j\phi(nF_0)} \tag{11}$$

où δ représente un pic de Dirac.

exemple

on a le signal temporel suivant :

$$s(t) = cos(2\pi F_0 t)$$

On va représenter le spectre bilatéral de $s(t)$ sachant que la décomposition en série de Fourier est définie par :

$$a_0 = 0 \ , \ a_1 = 1 \ , \ et \ a_n = 0 \ \forall n > 1$$
$$b_n = 0 \ \forall n \geq 0$$

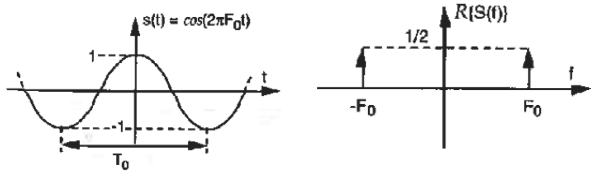
Il s'agit d'un spectre réel puisque la partie imaginaire est nulle $\forall$ n.
D'après (6),

$$S(F_0) = \tfrac{1}{2}(a_1 - jb_1) = \tfrac{1}{2} \ et \ S(-F_0) = \tfrac{1}{2}(a_{-1} - jb_{-1}) = \tfrac{1}{2}$$

Donc d'après (10),

$$S(f) = \tfrac{1}{2}.\delta(f - F_0) + \tfrac{1}{2}.\delta(f + F_0)$$

Ce qui se dessine :



application

A partir de la série :
- il est possible à partir de sa série de Fourier, de retrouver un signal
- en identifiant les valeurs a_n et b_n il est possible de représenter son spectre (on choisit arbitrairement un nombre d'harmoniques).
- astuce : rammener l'expression de la série à une des formes vu plus haut

A partir du spectre :
- on peut en déduire la série de Fourier
- c'est un peu plus compliqué de déduire la série de Fourier

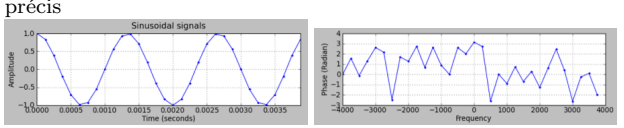
exemple

Si on a les paramètres suivant :
- le nombre de points $N = 32$
- la fréquence d'échantillonnage $F_s = 8000$
- la période d'échantillonnage $T_s = \frac{1}{F_s}$
- la résolution fréquentielle $freqStep = \frac{F_s}{N}$
- la fréquence du signal $f = 3 * freqStep$
- et $t \in N * T_s$

Aussi, on a le signal suivant :

$$s(t) = cos(2\pi f t) \tag{12}$$

Il faut remarque que plus N est grand, plus le signal serra précis



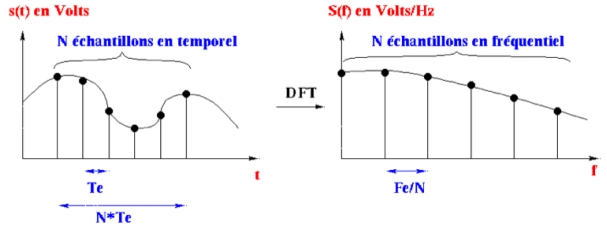
L'harmonique remarquable, par son amplitude, est celle située en 3, on peut donc en déduire que le signal $s(t)$ est composé *exclusivement* d'un cosinus de fréquence

$$3\Delta = 3\frac{F_s}{N} = 3 * \frac{8000}{32} = 750Hz \tag{13}$$

Transformée de Fourier

La transformation de Fourier est un analogue de la théorie des séries de Fourier pour les fonctions non périodiques, et permet de leur associer un spectre en fréquences.
- il faut travailler sur une discrétisation du signal $s(t)$, donc il faudra nécessairement tronquer le support temporel
- comme le signal est discret, la transformée de Fourier est dite *discrète*

La DFT réalise la correspondance entre 2 suites de N termes :



La *représentation fréquentielle* entre les points de la représentation fréquentielle est :

$$\Delta f = \frac{F_e}{N} \tag{14}$$

Elle dépend donc :
- de la période d'échantillonnage temporelle T_e
- **ET** du nombre de points fournis à la DFT

Echantillonnage

- pour quoi faire ? discrétiser un signal analogique (et ainsi le numériser)
- comment ? en utilisant Dirac sur une période d'échantillonnage $T_e = t_{i+1} - t_i \forall i$
- **ATTENTION** : le nombre de valeurs possibles est fini

Il faudra retenir que :
- Si on sur-échantillonne, le signal numérique deviendra trop lourd à transporter
- Si on sous-échantillonne, le signal numérique ne pourra pas être reconstruit en signal analogique

Fréquence d'échantillonnage idéale de *Shannon* :

$$F_e \geq 2.F_{max} \tag{15}$$

avec F_{max} la plus haute fréquence présente dans le spectre du signal.

Quantification

"Sur combien de niveaux va t-on représenter l'information ?"
- on essaye de prendre un ensemble de valeurs discrètes, qu'on va séparer en niveaux
- ça compresse l'information (mais avec pertes :/)
- peut être [non] uniforme (l'erreur est constante)
- s'il y a du bruit, on préfère utiliser des quantificateurs non uniformes (logarithmique)

Encodage

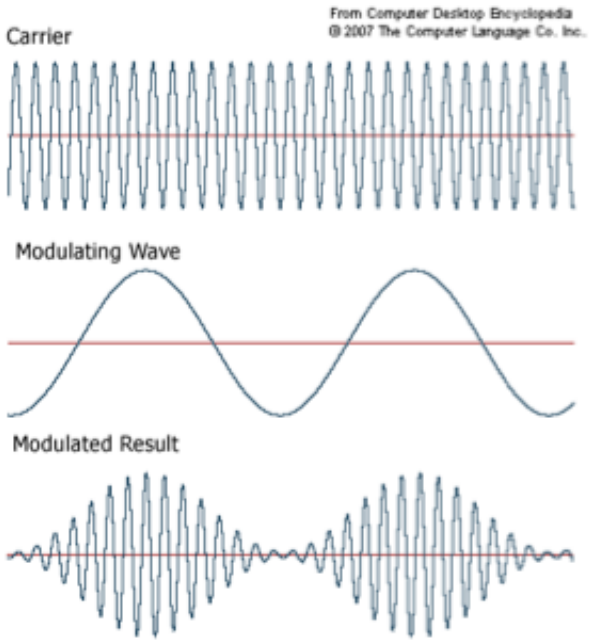
"J'ai X niveaux, de combien de bits ai-je besoin pour coder un échantillon ?"

La bande passante

- c'est une *taille*
- le *canal* par lequel on fait transiter l'information
- l'intervalle des fréquences (des harmoniques) admissibles forme la bande passante
- téléphone : $300Hz - 3400Hz \Rightarrow f_e = 6800Hz$, on prend $8kHz$ [$64kbps$(8000 échantillons par seconde x 8 bits par échantillon)]
- haute fidélité : $20kHz \Rightarrow f_e = 40kHz$, on prend $44,1kHz$ [$0,7Mbps$ (16 bits par échantillon, quantification linéaire)]
- télé numérique : $5MHz \Rightarrow f_e = 10kHz$, on prend $10MHz$ [$80Mbps$ (8 bits par échantillon)]

Modulation

Transformation d'un signal source en une forme adaptée au canal de transmission (en faisant varier l'amplitude ou la *phase/fréquence* d'une onde sinusoïdale appelée porteuse)



ATTENTION : ⇒ sinon on peut émettre en **bande de base** si le signal et le canal sont sur les mêmes fréquences.

FM

Il existe 2 types de modulation de fréquence :
- VHF (Very High Frequency) avec une bande de fréquence de 30 MHz à 300 MHz
- UHF (Ultra Hight Frequency) avec une bande de fréquence de 300 MHz à 3 GHz

Le soucis : la portée est limitée par l'horizon, il faut donc mettre des antennes relais sur les points hauts
On utilise les **ondes courtes** (< 30 MHz) qui utilise le phénomène de *propagation ionosphérique*.

Commutation

Il n'est pas possible de relier tout le monde, fil par fil. A la place on a mis en place le système de commutation ! La connexion d'une personne à une autre est faite **dynamiquement**. (comme le téléphone, ou RTC Réseau Téléphonique Commuté)

Commutation spaciale

C'est le système avec la standardiste qui met en connexion manuellement 2 personnes qui pourraient être éloignée géographiquement.

Commutation de circuit

C'est le système automatisé, ou le numéro de téléphone va indiquer le *chemin* à prendre pour établir la connexion, ce qui prend quelques secondes.
désavantage : sous utilisation de la ligne.
avantages :
- simple à implanter aussi bien sur un réseau analogique que numérique
- capacité du lien et débit de transmission garanti

Multiplexage/Démultiplexage

- permettre à plusieurs utilisateurs de se partager un même support physique de transmission
- avoir l'impression d'être le seul à l'utiliser
- parce qu'une communication (audio par exemple) est constituée à 90% de blanc, on chercher à optimiser l'utilisation du médium de transmission
- le coût élevé du montage des liaisons de transmissions sur de longues distances et/ou de grandes capacité

Multiplexage temporel

par exemple l'ordonnancement des tâches dans un ordinateur.
C'est en fait la possibilité de faire transiter sur un même canal plusieurs signaux en accordant une tranche de temps fixe à chacun.

Multiplexage fréquentiel

chaque entrée possède sa plage de fréquences en sortie.
Les spectres de fréquences en entrée sont mis dans la sortie du multiplexeur fréquentiel grâce à la **modulation**.

Le medium

Attenuation et affaiblissement

- L'attenuation d'un signal est le coefficient d'extinction en décibel par km
- le retard de conduction dans une paire de conducteurs en métal :

$$T(secondes) = C * R \tag{16}$$

avec C la capacité et R la résistance.
- affaiblissement

$$\alpha = \sqrt{\pi f C R} \tag{17}$$

La rapidité de modulation est le nombre de symboles par seconde :

$$R = \frac{1}{\Delta} Bauds \tag{18}$$

avec Δ la durée d'un symbole.
Dans notre cas, Bauds = Nombre de bits par seconde.
On a $RMAX = 2W$ avec W la bande passante du support.
Par exemple sur le RTC la bande passante est comprise entre 300Hz et 3400Hz, donc W=3400, d'où R_{max}=6800 Bauds.
ATTENTION :
- la bande passante limite le nombre de bauds
- un symbole peut représenter plus qu'un bit ! (par exemple la Modulation QAM

La valence

Le nombre d'états (ou symboles) significatifs que peut prendre le signal.

$$D = R * log_2(V) \tag{19}$$

avec D le débit en ligne, et V la valence.
Par exemple :

modulation de phase à 8 états (2^3), à 1600 Bauds, on obtient :

$$D = 1600 * 3 = 4800 bits/s$$

Modulation QAM (Quadrature Amplitude Modulation

C'est l'association d'une modulation d'amplitude et d'une modulation de phase.
Avec une constellation à 64 points, on peut transférer 6 bits par Baud.

xDSL

Modulation de la phase pour multiplexer plusieurs canaux à la foisn donc si on dispose de :

- bande de fréquence de 1 MHz
- on veut des canaux de 4 Hz
- $\frac{1000}{4} = 250$ canaux possibles

Si de base le canal permettait de faire passer 33,6 Kbps, on dispose d'un débit total de :

$$\frac{250*33,6}{1000} = 8,4 \text{ Mbps}$$

La fibre optique

- énorme bande passante
- indifférent au bruit
- de petite taille
- faible rayon de courbure

Les réseaux

Topologies de réseaux

adéquation	étoile	anneau	bus	arbre	maillage
symétrie	non	oui	oui	non	oui
ordre	non	oui	non	non	non
directionnel	non	a priori	non	non	non
nombre de brins	$N-1$	N	1	$N-1$	$N(N-1)/2$
confidentialité	oui	non	non	possible	oui

Protocoles

définition : Les protocoles sont les régles et les formats définissants les caractéristiques d'une communication entre entités de *même niveau*.
Le protocole peut devoir gérer l'échange en présence d'incidents :

- Pannes matérielles
- Congestion du réseau
- Retard ou perte d'unité de données
- Altération de données
- Duplication ou déséquencement d'unité de données.

La couche Application

- protocoles user applications
- protocoles support applications

La couche Transport

- fournit des services de communication *end to end* aux applications
- TCP
- UDP

La couche Internet

- controle et gestion de la connexion
- par exemple IP

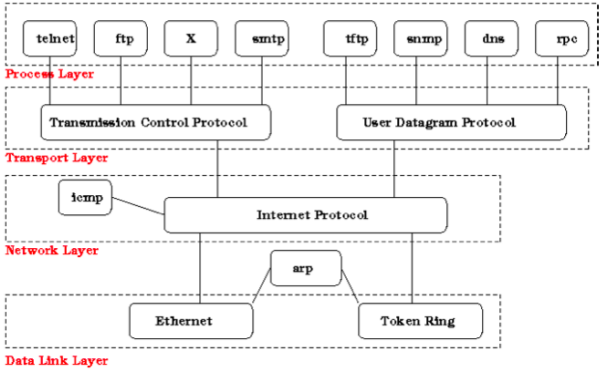
La couche Liaison

- s'occupe de l'adressage MAC
- découpe les données a transporter en trames de [72-1526] octets
- la détection d'erreur (pas la réparation !)
- en fait si, avec la FCE (Forward Correction Error)
- la contention (évitage de collision)

Switch et Hub

Un **hub** va re-émettre sur tout le réseau (en étoile) qu'il interconnecte.
Le **switch** va *router* en regardant les informations relatives à l'IP, dans le but de ne pas monopoliser le réseau en entier.

Architecture d'internet



Port

Un **port** est une destination abstraite sur une machine identifiée par un numéro (dit de port) qui sert d'interface a l'application (et éventuellement aux processus sous-jacents) pour recevoir et émettre des données, son rôle est donc de permettre a plusieurs programmes d'application de communiquer.
Il demultiplexe les infos reçues vers différents programmes d'application en utilisant les numéros de port(Ceci étant, il est possible que sur un même ordinateur plusieurs connexions partagent un même numero de port) .
Bien qu'UDP ne fragmente pas ses messages, IP le fera si le MTU du réseau physique ne permet pas d'acheminer le message sans modifier sa taille.

UDP

UDP fournit deux services supplémentaires par rapport à IP :

1. Il introduit la notion de port, ceci permet de distinguer plusieurs applications destinatrices sur la meme machine par l'intermediaire de ports différents. C'est la notion de multiplexage- demultiplexage appliquée à une pile de protocoles.
2. Il fournit (de façon optionnelle) un checksum qui permet de vérifier l'integrité des données. Motivation : Le service proposé est *grosso modo* une *interface transport* pour IP
3. permet le multicast

TCP

Le protocole TCP (Transmission Control Protocol) procure un service de flux d'octets oriente connexion et fiable. Il dispose de 5 particularités :

1. Orientation Connexion (une machine client et l'autre serveur. Connexion requise avant tout dialogue)
2. Circuits virtuels
3. Transferts tamponnes
4. Connexions non structurées
5. Connexions bidirectionnelles simultanées (full duplex).Motivation :Ce service était nécessaire, sous peine de voir chaque programme d'application développer ses propres mécanismes de contrôle.
6. ne permet pas le multicast

Fenêtre glissante

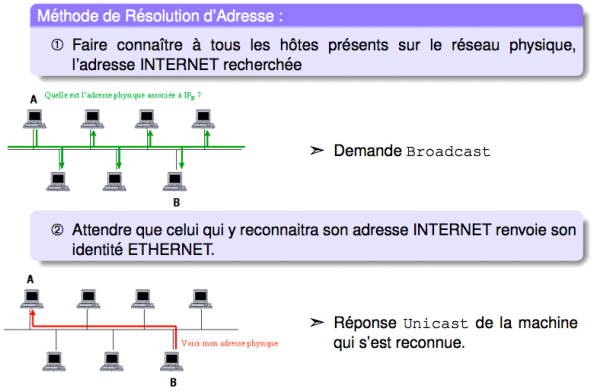
son but est d'améliorer l'efficacité de la transmission du flux, elle permet aussi de réguler le trafic en fonction de la charge des routeurs et du débit des réseaux traverses.

Algorithme de Nagle

: plus les ACK reviennent vite plus les données sont envoyés plus vite. Transmettre un message signifie donc un transfert de haut en bas a travers les couches logicielles, puis un transfert de bas en haut.

ARP

Déroulement du protocole ARP dans Ethernet :



MTU

Maximum Transmit Unit.
C'est la taille maximale d'une trame sur un réseau, la trame en question est réajustée dans le cas ou elle doit transiter sur un réseau intermédiaire ou le MTU est plus petit.
ATTENTION : la réciproque est fausse

Mémo Code Python

génération d'un signal sinusoïdal

```
def make_sin(a=1, ph=0, freq=440.0, N=100, n=30):
    T = 1.0/freq
    omega = (2*math.pi)/T
    te= T/n       # Sampling period
    print te
    print 1.0/te  # Sampling frequency
    sig_t = []
    sig_s = []
    for i in range(N):
        t = i*te
        sig_t.append(t)
        sig_s.append(a*math.sin((omega*t)+ph))
    # Faire un tableau a partir d'une liste
    x = numpy.array(sig_t)
    y = numpy.array(sig_s)
    return x, y
```

génération d'un signal carré

```
def make_square(a=2, freq=440.0, r=0.5, N=100, n=30):
    sig_t = []
    sig_s = []
    T = 1.0/freq
    te= T/n       # Sampling period
    for i in range(N):
        t = i*te
        sig_t.append(t)
        if ((i%n)<(n*r)):
            sig_s.append(+a)
        else:
            sig_s.append(-a)
    # Faire un tableau a partir d'une liste
    x = numpy.array(sig_t)
    y = numpy.array(sig_s)
    return x, y
```

génération d'un signal en scie

```
def make_saw(a=1, freq=440.0, N=100, n=30):
    x = numpy.zeros(N,dtype=numpy.float32)
    y = numpy.zeros(N,dtype=numpy.float32)
    T = 1.0/freq
    te= T/n       # Sampling period
    for i in range(N):
        t = i*te
        x[i] = t
        y[i] =  2*a*(t/T - math.floor((t/T)) -0.5)
    return x, y
```

additionner deux signaux

Je considère que $s1$,$s2$ et $s3$ sont des signaux existants.
$s4$ serra le signal *somme*.

```
s4 = []
  for i in range(len(s1[0])):
    s4.append(s1[1][i] + s2[1][i] + s3[1][i])
```

afficher un signal

On affiche le signal res en bleu pointillé :

```
pylab.plot(res[0], res[1], 'b.')
pylab.ylim([-3, 3])
pylab.xlim([0, 1])
pylab.show()
```

afficher une image et utilisation de la FFT

```
im = Image.open('train.png').convert('I')
im = im.rotate(90)
SZ = im.size[0]
print "debug de l'image:",im.format, im.size, im.mode
plt.figure(1)
plt.clf()
plt.imshow(im,
  cmap=plt.cm.gray_r,
  interpolation="Nearest",
  origin="lower")
F1 = np.fft.fft2(im)
F2 = np.fft.fftshift(F1)
# Inverse fft
imm = np.fft.ifft2(F1)
plt.figure(5)
plt.clf()
#print imm
plt.imshow(imm.real, cmap=cm.gray)
plt.show()
```

ecouter au micro

Retourne les données vers le fd donné en paramètre.

```
def ecouteur(nbSecondes=5,fd=1):
    inp = alsaaudio.PCM(alsaaudio.PCM_CAPTURE,
           alsaaudio.PCM_NORMAL)
    inp.setchannels(1)
    inp.setrate(8000)
    inp.setformat(alsaaudio.PCM_FORMAT_S16_LE)
    inp.setperiodsize(512)
    fini = False
    startFlag1 = time.time()
    while not fini:
            l,data = inp.read()
            if l:
                tailleDonneesEcrites = os.write(fd, data)
            endFlag1 = time.time()
            if (endFlag1 - startFlag1) >= nbSecondes:
                fini = True
    os.close(fd)
```

vers le haut parleur

Lit depuis le fd donné en paramètre vers le haut parleur.

```
def toSpeakers(nbSecondes=5,fd=1):
    out = alsaaudio.PCM(alsaaudio.PCM_PLAYBACK,
           alsaaudio.PCM_NORMAL)
    out.setchannels(1)
    out.setrate(8000)
    out.setformat(alsaaudio.PCM_FORMAT_S16_LE)
    out.setperiodsize(512)
    fini = False
    startFlag2 = time.time()
    data = os.read(fd, 512*2)
    while not fini:
        while data <> "":
            out.write(data)
            endFlag2 = time.time()
            if (endFlag2-startFlag2) >= (nbSecondes):
                fini = True
            data = os.read(fd, 512*2)
    os.close(fd)
```

serveur

```
def serveur(p):
    s = socket.socket(socket.AF_INET,
             socket.SOCK_DGRAM)
    s.bind(("",p))
    sortie = False
    while not sortie:
        datagrm, addr = s.recvfrom(1024)
        if datagrm:
            print datagrm
        if datagrm == "fini":
            sortie = True
    s.close()
```

client

```
def stresseur(addr,p,limite):
    s = socket.socket(socket.AF_INET,
             socket.SOCK_DGRAM)
    i = 0
    while i < limite:
        s.sendto("des données"), (addr, p))
        i += 1
    s.sendto("fini", (addr, p))
    s.close()
```

Serveurs UDP et TCP