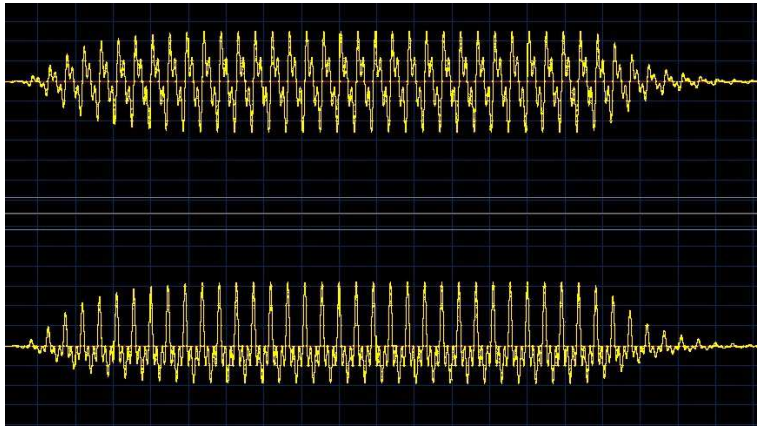
Mathématiques et applications :

Le son, un exemple physique à dimension humaine



Intervenant : Luc Lasne, Université de Bordeaux

luc.lasne@u-bordeaux.fr

- Introduction

- Les sons : des signaux aisément perceptibles pour lesquels l'analyse de notre cerveau est à la fois très évoluée et très accessible.

- Ecouter des sons permet de découvrir les concepts de fréquence, de forme d'onde, de superposition et de décomposition harmonique.

- En utilisant simplement les propriétés de la fonction sinus, il est possible d'écrire un algorithme qui permet de détecter les différentes fréquences qui composent un son complexe : cela s'appelle « l'étude harmonique ».

- Cette introduction à l'analyse dite « de Fourier » peut être menée en 1ère ou Terminale sous forme d'atelier et fait découvrir une théorie habituellement présentée au niveau Licence 2ème année.

- Introduction

Plan

I - Mathématiser les sons : fonction sinus et superposition

II - Générer des sons et des signaux : Audacity, Matlab, Python

III - La moyenne d'un produit de sinus : la clé de « l'analyse harmonique »

IV - Atelier : détection d'un son noyé dans du bruit

I - Mathématiser les sons : fonction sinus et superposition

Le son : une vibration qui se propage dans l'air, ou dans tout matériau le permettant.

Vibration, oscillation, onde : de nombreux termes désignent des fonctions « périodiques », c'est-à-dire présentant la répétition d'un « motif » de base dont la durée *T* est appelée la « période ».

La grandeur qui caractérise de façon prépondérante une onde est sa « fréquence » f, c'est-à-dire le nombre de périodes qui se produisent en une seconde.

$$f = \frac{1}{T} \quad unité : Herz\ (Hz)$$

Les sons divers sont complexes et habituellement composés de nombreuses sources et de nombreuses fréquences. Pour s'habituer à les distinguer, il est possible de s'intéresser au mélange de simples « tonalités » caractérisées par des fréquences différentes.

Tune 440Hz Tune 587 Hz Tune 660Hz Mix

- Introduction
- I) Mathématiser les sons

Le « son de base », la « tonalité » (tune) correspond à une fonction sinus (ou cosinus) dont l'amplitude correspond à l'intensité du son entendu, et dont la valeur de la fréquence est facile à déterminer dans la l'écriture de la fonction. Cela s'écrit :

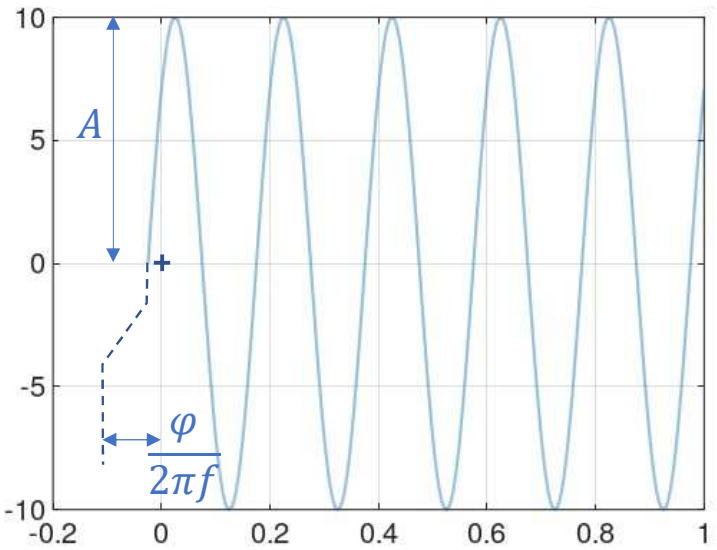
$$f(t) = A.\sin(2\pi.f.t + \varphi)$$

t : variable « temps » (en secondes)

A : « Amplitude » càd nombre qui multiplie la fonction puisque $sin(t) \in [-1\ ,1]$

f : Fréquence de l'onde. En une seconde, le terme $2\pi.f.t$ représente bien « f fois » un angle de 2π. La fonction se répète donc bien « f fois par secondes » (Le terme $2\pi.f$ est souvent appelé ω la « pulsation » de l'onde, en radians par seconde)

φ : Angle constant appelé « phase » qui permet de transcrire un décalage constant (avance ou retard) par rapport au zéro du repère choisi.



Exemple : A=10 , φ=π/4 , *f*=5 Hz

- Introduction
- I) Mathématiser les sons
- II) Générer des sons et signaux

II - Générer des sons et des signaux : Audacity, Matlab, Python

Pour analyser, « travailler », ou générer des sons, le logiciel gratuit Audacity s'avère complet et assez simple d'utilisation.

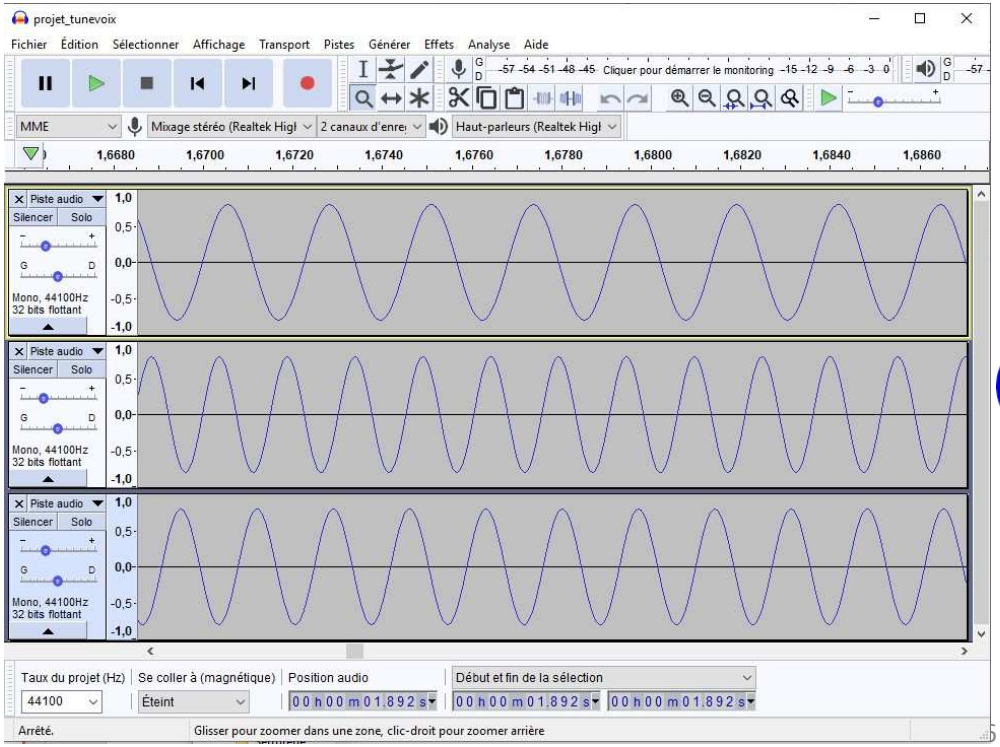
Activité « découverte » :

- Ouvrir Audacity et générer les tonalités précédentes (440Hz, 585Hz et 660 Hz)

- Ecouter les sons les uns après les autres puis tous ensemble (fonction « Silencer »)

- Enregistrer le mix en mp3

- Analyser en affichant le spectre



- Introduction
- I) Mathématiser les sons
- II) Générer des sons et signaux

Activité « atelier » en langage python :

Générer un signal sinusoïdal de fréquence *f*=440 Hz, d'amplitude *A*=0.1 , sur une durée d'observation *Tmax*=0.1 s. Le signal sera échantillonné à la fréquence *fe*=44 kHz

1) Définir et initialiser les variables utiles à l'établissement du signal
2) Définir et initialiser également le « nombre total d'échantillons » *Ne* ainsi que la « période d'échantillonnage » *Te*
3) Créer une liste de Ne éléments contenant les valeurs du temps t compris entre *0* et *Tmax* par « pas » de *Te*
4) Créer une liste de *Ne* éléments contenant les échantillons de $A.\sin(2\pi.f.t)$ pour (utiliser au besoin, mais ce n'est pas impératif, la librairie « numpy »)
5) Utiliser la librairie « matplotlib » pour tracer les valeurs du signal en fonction des valeurs du temps et donc l'allure de la fonction échantillonnée
6) Rajouter à ce signal une sinusoïde à 585 Hz et une sinusoïde à 660 Hz et observer le résultat. Mesurer sur le graphe la période et la fréquence du signal composite obtenu par l'addition des trois sinusoïdes

- Introduction
- I) Mathématiser les sons
- II) Générer des sons et signaux

Activité « atelier » : Codes pour la génération du signal et son enregistrement sous forme de fichier .wav sous Python et sous Octave (clone de Matlab qui est un logiciel fait pour le traitement du signal)

fonction_sinus.m

```
%%Generation d'une fonction sinus
%% de durée et fréquence connues
duree=1;  %%secondes
f=440;    %%Hz
A=1;      %%Amplitude  max = 1
Phi=pi/4; %%Phase
FileName='L:\Luc\Cours, TD, TP\Maison pour la science\Formation ma
Fech=44000;  %%Frequence d'échantillonnage 44kHz
Tech=1/Fech; %%Période d'échantillonnage

t=0:Tech:duree;  %%tableau "temps"
signal=A.*sin(2*pi*f*t+Phi);
signal_8bits=int8(127*signal);
plot(t,signal_8bits,'LineWidth',2);
set(gca,'FontSize',20);
sound(signal_8bits,Fech);
audiowrite(FileName,signal_8bits,Fech,'BitsPerSample',8);
```

GNU Octave

generer_sinus_filewav.py - L:/Luc/Cours, TD, TP/Maison pour la science/Formation maths - lycees 2019_2020/generer_si...

File Edit Format Run Options Window Help

```
# Code source Python
# Generation d'un fichier .wav stereo tonalité de fréquence et durée paramétrées
# codage sur un octet de chaque échantillon à 44000 ech/s

import wave
import math
import binascii

print("Génération d'un fichier .wav PCM 8 bits stéréo 44000 Hz")
print("Tonalité sinusoïdale\n")

FileName = 'signal_python.wav'
Signal = wave.open(FileName,'w') # instanciation de l'objet Monson

nbCanal = 2    # stéreo
nbOctet = 1    # taille d'un échantillon : 1 octet = 8 bits
fech = 44010   # fréquence d'échantillonnage

frequency = 440 #Hz
level = 1
MaxTime = 5       #secondes

nbEchantillon = int(MaxTime*fech)
print("Nombre d'échantillons :",nbEchantillon)

params = (nbCanal,nbOctet,fech,nbEchantillon,'NONE','not compressed')
Signal.setparams(params)    # création de l'en-tête

# niveau max dans l'onde positive : +1 -> 255 (0xFF)
# niveau max dans l'onde négative : -1 ->   0 (0x00)
# niveau sonore nul :                 0 -> 127.5 (0x80 en valeur arrondi)

amplitude = 127.5*level

print('Processing...')
for i in range(0,nbEchantillon):
    value = wave.struct.pack('B',int(128.0 + amplitude*math.sin(2.0*math.pi*frequency*i/fech)))
    Signal.writeframes(value + value) # écriture frame 2 canaux

Signal.close()

File = open(FileName,'rb')
data = File.read()
FileSize = len(data)
print('\nTaille du fichier',FileName, ':', FileSize,'octets')
print("Nombre d'octets de données :",FileSize - 44)  #Entête = 44bits
File.close()
```

python™

- Introduction
- I) Mathématiser les sons
- II) Générer des sons et signaux

Intérêt de l'activité :

- Aborder la numérisation des données : Il est nécessaire de définir une fréquence d'échantillonnage et un format. Contrairement à l'étude de fonctions en maths, les données numérisées ne comportent qu'un nombre fini de valeurs (discrétisation).

- Utiliser des boucles *for* ou bien manipuler des tableaux de données de façon à former la fonction $f(t) = A.\sin(2\pi.f.t + \varphi)$.

- Tracer la fonction et vérifier la conformité avec le résultat attendu.

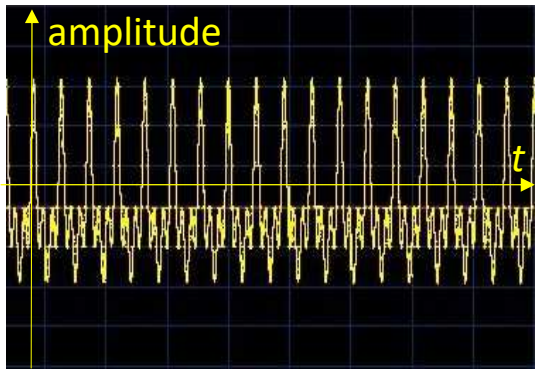
- Convertir le tableau de valeurs en signal .wav, écouter le son obtenu.

- L'activité doit donner l'envie de « complexifier » le signal : changer de fréquence, ajouter plusieurs sinus de fréquences différentes, etc.

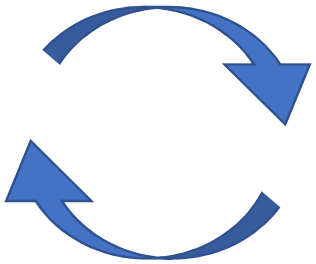
- Introduction
- I) Mathématiser les sons
- II) Générer des sons et signaux
- III) La clé de l'analyse harmonique

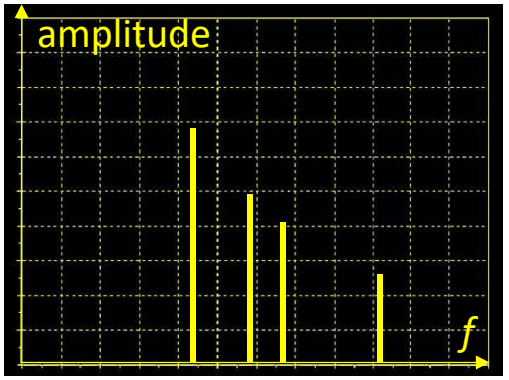
III - La moyenne d'un produit de sinus : la clé de « l'analyse harmonique »

L'analyse des signaux périodiques, appelée « analyse harmonique » ou « théorie de Fourier » permet de faire correspondre à un « signal » l'ensemble des sinusoïdes (de fréquences différentes) qui le composent.
Le signal devient alors compréhensible à travers les valeurs des fréquences qui y apparaissent (et des amplitudes correspondantes). On parle alors d'analyse « spectrale » ou encore « d'analyse harmonique ».



Signal : fonction du temps





Spectre : fonction de la fréquence

- Introduction
- I) Mathématiser les sons
- II) Générer des sons et signaux
- III) La clé de l'analyse harmonique

Pour aborder cette théorie sans prérequis, une question doit se poser : « comment calculer et « détecter » les fréquences et les amplitudes des sinusoïdes qui composent un signal quelconque ? »
La réponse est basée sur une observation très simple : le calcul de la valeur moyenne sur une période du produit de deux sinusoïdes.

Valeur moyenne

Soit un signal $S(t)$ périodique et échantillonné (discrétisé), c'est-à-dire connu sur N échantillons nommés : $S_i \;\; i \in \mathbb{N}\,, i \in [1, N]$. La « valeur moyenne » de ce signal sur le nombre N d'échantillons s'écrit :

$$E(S) = \frac{1}{N}.\sum_{i=1}^{N} S_i$$

Activité « atelier » : Reprendre les programmes de la partie II et rajouter dans le script le calcul de la valeur moyenne des signaux sinusoïdaux générés.
Intérêt : On réalise que la moyenne de la fonction sinus sur une période est nulle et facile à programmer. Pour la suite, on pourra utiliser les fonctions « mean( ) » …

Produit de deux sinusoïdes

Soient $S_i(t) = A_i.\sin(2\pi.f_i.t)$ et $\sin(2\pi.f.t)$ deux fonctions sinusoïdales.
Que vaut la moyenne : $E(S_i(t).\sin(2\pi.f.t))$?
Le résultat est facile à prévoir, en effet :

$$S_i(t).\sin(2\pi.f.t) = A_i.\sin(2\pi.f_i.t).\sin(2\pi.f.t)$$

Formule de trigonométrie : $\sin(a).\sin(b) = \frac{1}{2}.(\cos(a-b) - \cos(a+b))$

Ainsi :

$$S_i(t).\sin(2\pi.f.t) = \frac{A_i}{2}.(\cos(2\pi.(f_i - f).t) - \cos(2\pi.(f_i + f).t))$$

Moyenne du produit de deux sinusoïdes

Ainsi, la moyenne sur une période du produit de ces deux sinusoïdes s'écrit :

$$E(S_i(t).\sin(2\pi.f.t)) = \frac{A_i}{2}.E(\cos(2\pi.(f_i - f).t)) - \frac{A_i}{2}.\underbrace{E(\cos(2\pi.(f_i + f).t))}_{=0}$$

Il ne subsiste ainsi que deux cas, la moyenne sur une période d'une fonction cosinus étant toujours nulle :

Si $f_i \neq f$: $E(\cos(2\pi.(f_i - f).t)) = 0$ et donc : $E(S_i(t).\sin(2\pi.f.t)) = 0$

Si $f_i = f$: $E(\cos(2\pi.(f_i - f).t)) = E(\cos(0)) = 1$ et donc : $E(S_i(t).\sin(2\pi.f.t)) = \frac{A_i}{2}$

La moyenne du produit de deux sinusoïdes est donc très simple à appréhender : si les deux sinusoïdes sont de même fréquences, le résultat obtenu est la moitié du produit des amplitudes, sinon le résultat est nul

Stratégie de l'analyse harmonique

- Imaginons un signal complexe composé d'une somme de sinusoïdes :

$$S(t) = \sum_{i=1}^{N} A_i . \sin(2\pi . f_i . t)$$

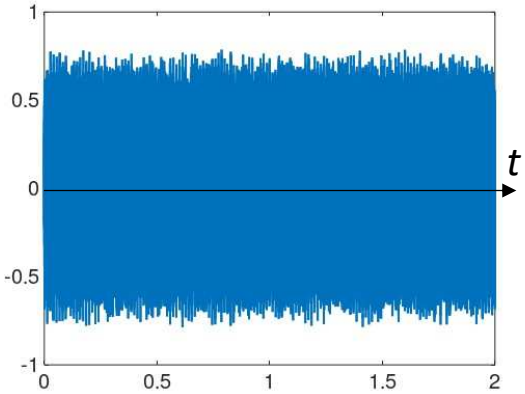
- Choisissons une valeur de fréquence f et calculons $E(S(t). \sin(2\pi . f . t))$
- Si le résultat est nul c'est que la fréquence f n'apparaît pas dans les sinusoïdes qui composent le signal complet.
- Si le résultat est non nul, cela veut dire qu'une sinusoïde de même fréquence existe dans la composition du signal complet, et il ne reste qu'à multiplier le résultat de la moyenne calculée par 2 pour connaître l'amplitude correspondante.
- Il suffit ensuite de réaliser ce calcul sur toute un plage de valeurs de la fréquence f pour détecter et calculer l'ensemble des sinusoïdes qui composent le signal.
- Le résultat peut être représenté par un graphe donnant les amplitudes en fonction des fréquences « balayées » : le spectre.

- Introduction
- I) Mathématiser les sons
- II) Générer des sons et signaux
- III) La clé de l'analyse harmonique
- IV) Atelier : détection d'un signal noyé dans le bruit

IV – Atelier : détection d'un signal noyé dans du bruit

L'atelier final consiste en trois actions à réaliser dans l'ordre, et peut ensuite ouvrir l'étude sur l'extension de l'algorithme à des cas plus généraux ou plus complexes.

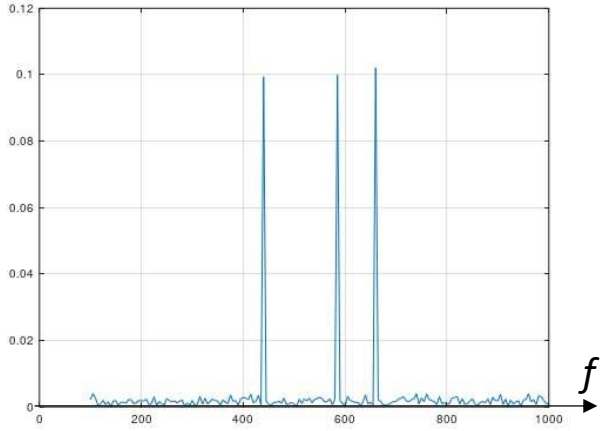
1) Activité « signal » : génération d'un signal composé de plusieurs sinusoïdes, de différentes fréquences et amplitudes. Génération d'un signal aléatoire d'amplitude supérieure à celle des sinusoïdes. Obtention du « signal bruité » sous forme de tableau de données et aussi de fichier son .wav). Ecoute et analyse du résultat.



Signal « bruité »

- Introduction
- I) Mathématiser les sons
- II) Générer des sons et signaux
- III) La clé de l'analyse harmonique
- IV) Atelier : détection d'un signal noyé dans le bruit

2) Activité « Fourier » : écriture de l'algorithme permettant de calculer la moyenne du produit du signal avec une série de sinusoïdes dont les fréquences sont réparties uniformément entre f_{min} et f_{max} (définition d'un « pas »). Après calcul on représente les valeurs de $2 \times E(S(t).\sin(2\pi.f.t))$ en fonction des fréquences f pour obtenir le « spectre ».



Spectre du signal « bruité »
Fmin=100Hz, fmax=1000Hz, df=5Hz

- Introduction
- I) Mathématiser les sons
- II) Générer des sons et signaux
- III) La clé de l'analyse harmonique
- IV) Atelier : détection d'un signal noyé dans le bruit

3) Activité « reconstruction du signal » : une fois les amplitudes et les fréquences des différentes sinusoïdes retrouvées, il suffit de les utiliser pour reconstruire le signal exempt de bruit.
Ceci étant réalisé, il reste quelques axes de réflexion :
- Comment faire de même si le signal caché est déphasé par rapport à l'origine ?
- Comment choisir la plage de fréquences scannées et le pas de calcul ? Penser à une échelle logarithmique …
- Comment traiter un signal qui évolue dans le temps (voix, musique, etc) ?

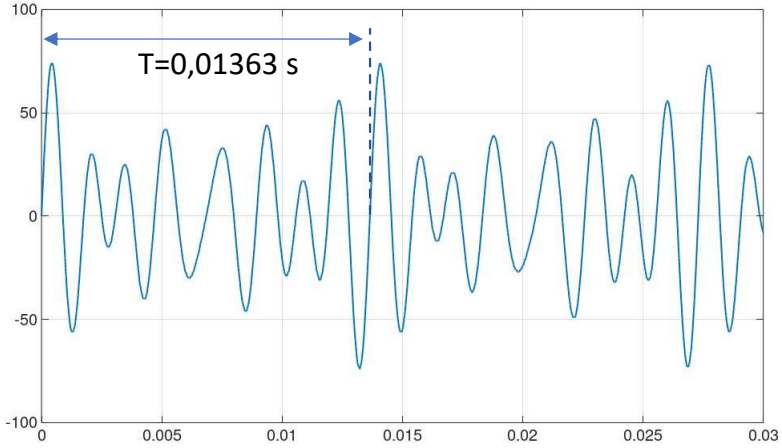
- Introduction
- I) Mathématiser les sons
- II) Générer des sons et signaux
- III) La clé de l'analyse harmonique
- IV) Atelier : détection d'un signal noyé dans le bruit

Extension de l'atelier : la notion d'analyse « harmonique »

En réalité, une « harmonique » désigne une sinusoïde dont la fréquence est un multiple entier d'une fréquence de base appelée « fondamentale ».

Atelier : avec Audacity ou Octave, générer une tonalité de fréquence 440Hz, puis une de 880Hz (harmonique 2). Écouter les sons correspondants. Revenir ensuite au mix de tonalités de 440Hz, 585Hz et 660Hz.

Une question se pose : quand on a mixé les fréquences de 440Hz, 585Hz et 660Hz, qui ne sont pas des multiples, quelle était la fréquence fondamentale du signal résultant ?



$$f_{fond} = \frac{1}{0{,}01363} = 73{,}333\ Hz$$

$$\frac{440}{73{,}333} = 6$$

$$\frac{585}{73{,}333} = 8$$

$$\frac{440}{73{,}333} = 9$$

- Introduction
- I) Mathématiser les sons
- II) Générer des sons et signaux
- III) La clé de l'analyse harmonique
- IV) Atelier : détection d'un signal noyé dans le bruit

Les trois sinusoïdes initiales présentent en réalité des fréquences qui sont des multiples entiers de la fréquence fondamentale, celle-ci n'étant trouvée qu'en inversant la période du signal ce qui n'est pas compliqué.

Une fois déterminée cette fréquence fondamentale, on SAIT à l'avance que toutes les composantes sinusoïdales du signal correspondront à des fréquences multiples (entiers). Cela réduit donc énormément les fréquences à scanner pour déceler les harmoniques par l'algorithme précédent et rend impossible le fait de « louper » une composante à cause du pas de calcul.

Voilà pourquoi on parle « d'analyse harmonique ».

Fin de la présentation