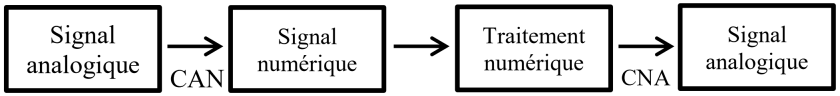
TP n°2 Electronique: Filtrage numérique avec python

Capacités expérimentales exigibles: **Utiliser un convertisseur analogique-numérique et un convertisseur numérique-analogique.**
Capacités numériques exigibles: **à l'aide d'un langage de programmation, simuler un filtrage numérique et visualiser son action sur un signal périodique.**

Dans un studio d'enregistrement, le signal audio est enregistré numériquement (ou échantillonné). Ce signal, devenu numérique, peut alors subir de multiples opérations : filtrage, compression, effets divers... Lors de la restitution du son (quand vous l'écoutez) le signal numérique est alors transformé en signal analogique avant d'être transmis, le plus souvent après amplification, vers des haut-parleurs. On peut résumer le procédé par le schéma suivant:



avec les abréviations **CAN** pour **C**onversion **A**nalogique **N**umérique et **CNA** pour **C**onversion **N**umérique **A**nalogique.

On se propose de réaliser une telle chaîne d'opérations, mais dans un cadre restreint:

- Le signal analogique $s(t)$ à traiter sera constitué de la somme de deux fonctions sinusoïdales, de fréquences $f_1 = 200Hz$ et $f_2 = 2000\ Hz$.
- Le traitement numérique sera ici un filtrage passe haut ou passe bas sur le signal.

En plus des appareils usuels, on dispose du matériel suivant:

- Un sommateur, qui peut recevoir en entrée une tension s_1, une tension s_2, et fournir la tension $s_1 + s_2$. Ce dispositif est alimenté $(+15\ V, 0\ V, -15\ V)$ et ne peut fournir de tension supérieure à $15\ V$ en valeur absolue.
- Un haut-parleur.
- Un amplificateur (ampli GBF) de puissance qui permet, à partir d4un signal de faible puissance, d'alimenter des dispositifs qui en nécessitent davantage.
- Un boitier d'acquisition Sysam® qui permet, par l'intermédiaire du logiciel LatisPro, de réaliser les conversions CAN et CNA (cf. annexe 1).
- Un outil de traitement de données qui permet de réaliser les traitements numériques sur les signaux. Le choix retenu est d'utiliser le langage Python pour traiter numériquement les données (cf annexe 2).

L'objectif du TP est de permettre, à partir du signal $s(t)$, l'écoute d'un son de fréquence soit f_1, soit f_2.

1 Acquisition d'un signal analogique

Proposer un montage pour réaliser le signal voulu en entrée, et faire son acquisition par l'intermédiaire du logiciel Latis Pro. Pour le signal étudié, choisir le nombre de points d'acquisition N et le pas d'échantillonnage T_e ($T_e = 1/f_e$) de manière:

- A respecter le critère de Nyquist-Shannon
- A avoir une bonne résolution en fréquence

On justifiera les choix retenus pour ces deux paramètres.

2 Filtrage numérique

En vous aidant des annexes, réaliser, par une ou plusieurs méthodes, le filtrage numérique du signal de manière à isoler les composantes hautes et basse fréquence. On rappelle l'expression des fonctions de transfert des filtres passe-bas et passe-haut d'ordre 1:

$$H_{PB} = \frac{H_0}{1 + j\omega\tau} \quad \text{et} \quad H_{PH} = \frac{j\omega\tau H_0}{1 + j\omega\tau}$$

où $\omega_0 = \dfrac{1}{\tau}$ est la pulsation de coupure des filtres.

3 Restitution d'un signal analogique

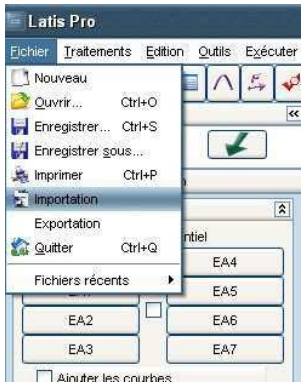
Il ne reste plus qu'à utiliser le signal filtré pour alimenter le haut-parleur, et apprécier l'effet du filtrage.

Annexe 1: Manipulation de données avec LatisPro

1 Import de données vers LatisPro

Pour importer des données contenues dans un fichier `.txt` :

Menu Fichier ⇒ Importation



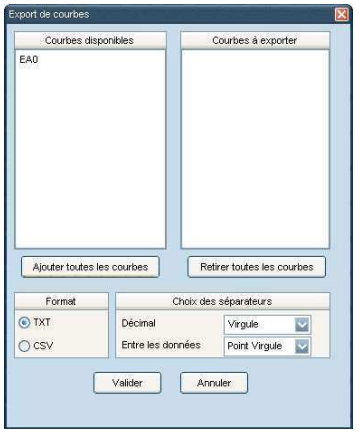
Il suffit ensuite de faire glisser les colonnes que l'on veut importer.

2 Export des données contenues dans LatisPro

Pour exporter des données de LatisPro vers un autre logiciel:

Menu Fichier ⇒ Exportation

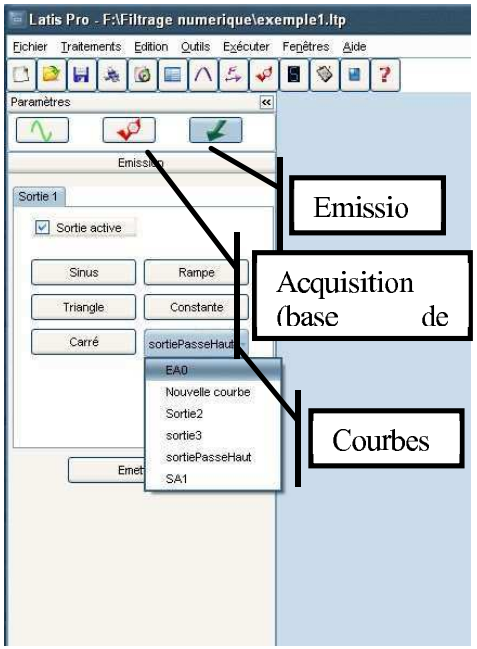
Il suffit ensuite de faire glisser la courbe voulue dans la fenêtre de droite, et de régler les options comme indiqué sur la figure:



3 Fonctionnement en CAN-CNA du boitier Sysam®

On peut à l'aide de LatisPro piloter le boitier pour réaliser:

- Une acquisition ou Conversion Analogique Numérique (CAN)
- Une émission de signal ou Conversion Numérique Analogique (CNA)



3.1 Acquisition

Pour une acquisition à l'aide de LatisPro, on se reportera au TP détaillant ce procédé, et on veillera notamment à réaliser une acquisition compatible avec le critère de Nyquist-Shannon.
Recommandation importante pour ce TP: le nombre total de points d'acquisition doit être une puissance de 2 (512, 1024,...).

3.2 Emission

Sur le boitier Sysam®, on dispose d'un connecteur Sortie SA1, qui permet de générer une tension analogique (entre Sortie et la masse) à partir d'un fichier numérique. Pour piloter cette émission, sélectionner l'onglet Emission de la boite Paramètres, et choisir une des courbes proposées (ces courbes doivent avoir été importées ou crées précédemment). Cocher Sortie active.

Le réglage de la base de temps pour l?émission s?effectue avec la même boite de dialogue que pour l'acquisition.

ATTENTION: pour un signal échantillonné à la période T_e, puis traité numériquement, il faut absolument que la valeur de T_e soit identique lors de l'émission. En effet c'est cette valeur (accessible par la boite acquisition) qui impose le pas temporel lors de l'émission, et pas la colonne de données temps qui pourrait figurer dans vos données.

Annexe 2: Filtrage sous Python

Le filtrage numérique peut être réalisé avec Python, et on fournit un fichier FiltrageNumerique_eleve.py pré rempli dans lequel figurent les principales instructions pour réaliser ce filtrage:

- lecture du fichier de point exporté en format `.txt` par Latispro, puis
- action sur ces signaux (filtrage temporel ou fréquentiel), et enfin
- écriture des signaux de sortie dans un fichier `.txt` utilisable par Latispro.

Vous téléchargerez ce fichier depuis sur le site de la classe; il est disponible dans la page `Travaux pratiques → TP de physique` à l'adresse:
`https://mp3montaignebdx.legtux.org/tp-de-physique/`
L'enregistrer ensuite dans votre répertoire personnel sur le réseau réseau du lycée avant toute utilisation. Les fichiers `.txt` utilisés par ce script Python devront être situés dans le même répertoire que le script lui-même. Les fichiers .txt générés y seront placés également.

1 Lecture de fichier

L'instruction:
`fichier = open('FiltrageNumeriquePointsEntree.txt','r')` permet l'ouverture du fichier `FiltrageNumeriquePointsEntree.txt` en lecture seule.

- Adapter cette instruction pour ouvrir votre fichier de points.

2 Filtrage temporel

2.1 Présentation

NB: cette partie reprend, dans les grandes lignes, les éléments vus en cours sur le filtrage numérique temporel des signaux échantillonnés.

Il est possible de déterminer la réponse d'un système linéaire à un signal d'entrée en utilisant une approche temporelle. Le système est ici décrit par l'équation différentielle linéaire à coefficients constants liant l'entrée et la sortie, que l'on va réécrire sous une forme discrète approchée de manière à pouvoir écrire le lien entre les signaux d'entrée et de sortie sous forme échantillonnée.

Pour des signaux d'entrée et de sortie $e(t), s(t)$, on introduit la suite des valeurs $e_n = e(nT_e), s_n = s(nT_e)$, où T_e est le pas d'échantillonnage des signaux. Le but est donc de déterminer une relation de récurrence qui permet de calculer de manière approchée l'ensemble des s_n à partir des valeurs discrètes du signal d?entrée e_n, cette relation dépendant évidemment du filtre étudié.

- Cas du filtre passe-bas du 1er ordre: cette relation a été établie en cours; on la rappelle ici:

$$s_{n+1} = \frac{1-\alpha}{1+\alpha} s_n + \frac{\alpha}{1+\alpha}(e_{n+1} + e_n) \quad \text{avec } \alpha = \frac{T_e}{2\tau}$$

- Cas du filtre passe-haut du 1er ordre: **Etablir la relation entre e_n et s_n dans ce cas**.

2.2 Mise en oeuvre

A partir du tableau E des points du signal d'entrée, on désire créer deux tableaux $S1$ et $S2$ correspondant aux sorties après passage dans les filtres passe-bas et passe-haut du premier ordre. La relation de récurrence est codée à l'intérieur des boucles for.

- Compléter le contenu de la première boucle afin de réaliser le filtrage passe-bas.
- Compléter le contenu de la deuxième boucle afin de réaliser le filtrage passe-haut.

3 Filtrage fréquentiel

3.1 L'outil Fast Fourier Transform (FFT)

La transformée de Fourier rapide (TFR ou Fast Fourier Transform (FFT)) est un algorithme permettant de calculer très efficacement (complexité temporelle en $\ln_2(n)$) une transformée de Fourier discrète (TFD); elle permet de déterminer les spectres de signaux numérisés. Pour un signal échantillonné à la fréquence F_e ($T_e = 1/F_e$), avec N points de mesure (attention, N doit être une puissance de 2 pour la TFR), l'algorithme renvoie la valeur des coefficients de Fourier aux fréquences discrète $F_n = nF_e/N$, pour $N \in [0, N-1]$. On peut également réaliser le processus inverse, par l'algorithme IFFT (Inverse Fast Fourier Transform).

On peut utiliser le module `scipy.fftpack` de Python qui permet :

- De réaliser la `FFT` d'un signal discrétisé à l'aide de l'instruction `fft()`.
- De réaliser la transformation inverse à l'aide de l'instruction `ifft()`.
- De générer le tableau des fréquences pour lesquelles `fft()` calcule les coefficients de Fourier, à l'aide de `fftfreq(n,te)`.

3.2 Mise en oeuvre

Le script nécessite de définir les fonctions de transfert $H1$ (passe-bas) et $H2$ (passe-haut).

- Ecrire les lignes permettant de définir H1 et H2
- Expliquer l'enchainement des instructions permettant de réaliser le filtrage.

4 Ecriture d'un fichier de sortie

L'instruction: `fichier = open(nom+'.txt','w')`
permet l'ouverture du fichier `nom.txt` en écriture.

- Expliquer le fonctionnement de la fonction ecrit(T,S,nom)
- Sauvegarder sous des noms différents les signaux filtrés par différentes méthodes et différents filtres

Les signaux filtrés peuvent désormais être utilisés par tout autre dispositif acceptant les données numériques.