Outils Numériques pour l'Ingénieur·e en Physique

5N-028-PHY / ONIP-1

Bloc AM - Modulation AM (50%)

Concepts étudiés

[PHYS] Modulation d'amplitude
[MATH] Transformée de Fourier
[NUM] Signaux numériques
[NUM] Figures scientifiques

Mots clefs

Fichier CSV ; Graphique scientifique ; Transformée de Fourier ; Modulation d'Amplitude ; Démodulation

Sessions

0 Cours(s) - 1h30
0 TD(s) - 1h30
4 TD(s) Machine - 2h00
0 TP(s) - 4h30

Travail

En binôme

Institut d'Optique
Graduate School, *France*
https://www.institutoptique.fr

GitHub - Digital Methods

https://github.com/IOGS-Digital-Methods

Ce document est accessible en ***version numérique*** *sur le* ***site du LEnsE****, rubrique Année / Première Année / Outils Numériques / Bloc AM.*

Afficher et traiter des données provenant d'instruments de mesure

Les **expériences scientifiques**, les **essais industriels** sur des systèmes ou bien encore des **résultats de simulation** produisent énormément de **données**. Ces données sont souvent sauvegardées sous forme de **fichiers formatés** (format normalisé ou interne aux entreprises/laboratoires).

Il est alors indispensable de pouvoir **afficher les données** contenues dans ce type de fichier de manière claire et sans ambiguïté, avant d'en **extraire des informations pertinentes** par un traitement adapté.

Données à traiter

Dans cette séquence, vous serez amenés à utiliser des données provenant de 3 fichiers différents :

- DATA_01.CSV contenant l'enregistrement d'une **transmission d'informations modulées en amplitude** par un signal porteur sinusoïdal - acquis par un oscilloscope numérique
- DATA_02.TXT contenant un signal sonore modulé en amplitude à déchiffrer...
  - Format de données binaire 64 / Modulante sinusoïdale / Fichier sonore : 24 kHz / 16 bits
- B3_DATA_03.TXT contenant un ensemble de signaux modulés en amplitude à l'aide de différentes porteuses.
  - Format de données binaire 64 / Modulantes sinusoïdales / Fichier sonore : 160 kHz / 16 bits

Ressources

Cette séquence est basée sur le langage Python. Vous pouvez utiliser l'environnement **PyCharm** (édition Community 2023) et **Python 3.10** (inclus dans la distribution Anaconda 3).

Des aides sur Python sont disponibles sur la page suivante :

https://iogs-lense-training.github.io/python-for-science/

Acquis d'Apprentissage Visés

En résolvant ce problème, les étudiant·e·s seront capables de :

1. **valider un modèle physique** simple et fourni à l'aide d'un outil de calcul scientifique
   - Transcrire/Traduire un modèle physique donné (sous forme d'équations) en algorithme numérique
   - Choisir les paramètres de tests adaptés et réfléchis (discrétisation du signal, échantillonnage correct...)
   - Analyser la pertinence des résultats obtenus (erreurs de calcul, divergence...)
2. **générer des graphiques scientifiques** légendés
   - Réaliser le graphique
   - Décrire les axes avec les grandeurs et les unités associées
   - Légender le graphique (titre, légende des courbes...)
3. **écrire un script réutilisable dans un langage de haut niveau** (à but scientifique)
   - Utiliser des fonctions du langage avec des paramètres adaptés
   - Ecrire des fonctions dans un langage de haut niveau afin de rendre des parties du code réutilisable
   - Fournir un code lisible et réutilisable (convention d'écriture dans le langage, commentaires, documentation...)
4. **calculer, afficher et utiliser la transformée de Fourier discrète** d'un signal
   - Représenter l'axe des fréquences
   - Savoir repérer graphiquement les composantes fréquentielles d'un signal dans un spectre
   - Lister les contraintes de la FFT / Hypothèses et propriétés (signaux périodiques, symétrie hermitienne...)

Etapes

Etape 1 Lire un fichier de points
- Lire un fichier texte ou décoder un fichier binaire (Base64).
- Récupérer les données dans un vecteur
- Afficher le signal contenu dans le fichier

Etape 2 Extraire des informations du spectre d'un signal
- Calculer et afficher la FFT des signaux
- Identifier les informations utiles

Etape 3 Démodulation du signal
- Générer un signal sinusoïdal à la fréquence d'une porteuse et multiplier le signal initial avec cette porteuse
- Afficher la FFT de ce nouveau signal
- Démoduler le signal (et le jouer si c'est un signal audio)

Etape 4 Générer des signaux modulés et afficher leur spectre
- Générer des signaux modulés par des porteuses sinusoïdales
- Calculer et afficher la FFT de ces signaux modulés

Livrables attendus

Vous aurez 5 minutes lors de la séance 4 pour présenter l'ensemble de vos résultats et de vos analyses.

Vos scripts devront être déposés sur la plateforme eCampus la veille de votre passage à l'oral.

Le nom du fichier doit être de la forme suivante : *Gx_NOM1_NOM2.py* où x est le numéro de votre groupe de TD.

Lors de l'oral, vous serez amené à **présenter la démarche** ayant mené à vos résultats. Vous devrez **exécuter au moins une fois votre code** afin de valider son bon fonctionnement. *Vous pouvez également générer des graphiques légendés permettant de faciliter votre présentation.*

Une discussion de 5 minutes avec un·e encadrant·e suivra votre présentation.

Outils Numériques pour l'Ingénieur.e en Physique

Bloc AM / Fichiers de données

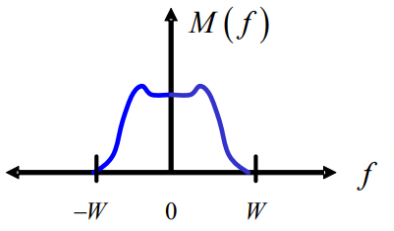
Rappel sur la modulation d'amplitude

Afin de faciliter le transport de signaux électriques (i.e. permettre le transport spécifique de plusieurs informations sur un canal de transmission), on utilise de la **modulation**. La plus facile à mettre en œuvre est la **modulation d'amplitude** (AM).

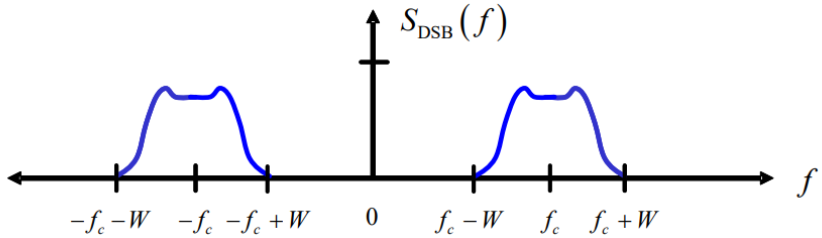
Elle consiste à moduler l'amplitude d'un signal porteur *p(t)* par un signal modulant *m(t)*.
Dans le cas de signaux sinusoïdaux, on a : **m(t)** un signal quelconque de pulsation maximale ω_m et $\mathbf{p(t) = A_p \, . \, sin\,(\omega_p \, . \, t)}$ avec $\omega_p >> \omega_m$

On obtient alors le signal modulé ***s(t) = m(t) . p(t)***.
Dans le cas des GBF Agilent, le signal modulé en sortie est du type : ***s(t) = (K . m(t) + 1) . p(t)*** où K est le taux de modulation.

Dans le cas de signaux périodiques quelconques, dont on connaît le spectre, on obtient alors le spectre suivant après modulation (tiré de http://wcours.gel.ulaval.ca/2017/a/GEL3006/default/5notes/index.chtml) :

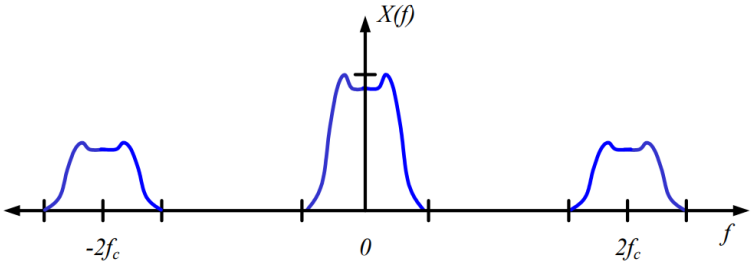


spectre initial

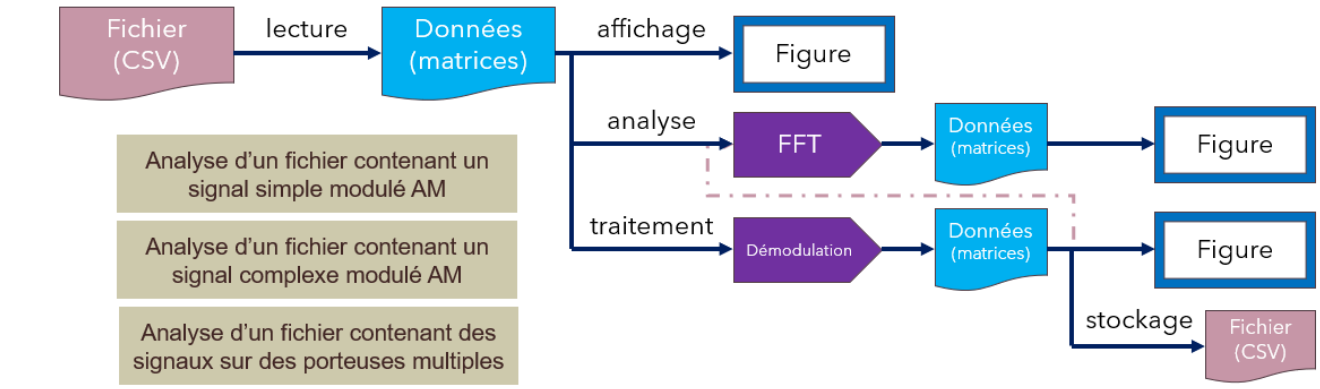


spectre après une modulation d'amplitude avec une porteuse de fréquence f_c

La **démodulation** d'un tel signal se fait en multipliant le signal modulé par la porteuse.
Ainsi : ***d(t) = s(t) . p(t)*** et on obtient le spectre résultant suivant (avec fc la fréquence de la porteuse). Il suffit alors de filtrer la partie centrale du spectre pour retrouver le signal modulé *m(t)*.



Travail demandé



INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech | LEnsE Laboratoire d'Enseignement Expérimental

Fonctions à maîtriser

- **lire des fichiers CSV** — **numpy .genfromtxt**, **pandas .read_csv**
- **créer de vecteurs / matrices** — **numpy .linspace .logspace**, **numpy .ones .zeros**
- **afficher des figures** — **pyplot .figure .plot .title .xlabel .ylabel .legend**
- **calculer la FFT** — **numpy.fft .fft .fftshift .ifft .fftfreq**
- **autres** — **size, numpy.abs, .shape …**
- **transcodage / Numpy types** — **numpy .frombuffer .astype**
- **encodage B64** — **base64 .b64encode .b64decode**
- **encodage WAV** — **scipy.io .wavfile.read .wavfile.write**

Exemple d'encodage et décodage en base 64

```
import base64
encoded = base64.b64encode(b'data to be encoded')
        >> b'ZGF0YSB0byBiZSBlbmNvZGVk'
data = base64.b64decode(encoded)
        >> b'data to be encoded'
```

Exemple de lecture d'un fichier binaire encodé en base 64

```
with open('B3_data_03.txt', 'rb') as file_to_decode :
    binary_file_data = file_to_decode.read()
    data = np.frombuffer(base64.b64decode(binary_file_data), np.int16)
```