Outils Numériques pour l'Ingénieur·e en Physique

2023-2024

5N-xxx-Phy / ONIP

Bloc 1 - Python et calcul scientifique (33%)

Concepts étudiés

- [Num] Bases de Python pour l'ingénieur·e en Physique
- [Phys] Mise en équation de systèmes régis par des équations différentielles
- [Math] Systèmes d'équations linéaires
- [Num] Résolutions numériques : calcul formel, équations différentielles, systèmes

Mots clefs

Python; Matrices (Numpy); Calcul formel (Sympy); Méthode d'Euler; Systèmes (control)

Sessions

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

Travail

Par équipe de 2

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

Démystifier Python et résoudre des problèmes à l'aide d'outils numériques

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.

Vous traiterez dans cette séquence une **information modulée en amplitude**, acquise par un **oscilloscope numérique** et stockée dans un **fichier de type tableur**.

Acquis d'Apprentissage Visés

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

Côté Numérique

1. **Générer des signaux numériques** à partir de fonctions mathématiques
2. **Définir et documenter des fonctions** pour générer des signaux numériques
3. **Produire des figures** claires et légendées à partir de signaux numériques - incluant un titre, des axes, des légendes
4. [Bonus] **Construire des bibliothèques de fonctions**

Côté Physique

1. **Analyser le contenu spectral** d'un signal électrique
2. **Déterminer les paramètres** d'une modulation d'amplitude
3. **Décoder** un signal modulé en amplitude

Livrables attendus

Pour valider cette session, vous devez fournir les **livrables suivants** :

1. **Fonctions commentées** (selon la norme PEP 257) pour générer des signaux numériques appropriés
2. **Graphiques légendés** incluant toutes les données nécessaires à la bonne compréhension des données présentées : signal initial, transformée de Fourier du signal initial, signaux générés pour démoduler le signal, transformées de Fourier intermédiaires, signal démodulé
3. **Analyse des figures** en insistant sur la démarche ayant amené à la démodulation du signal

Ces livrables pourront prendre la forme d'un **compte-rendu** incluant une introduction à la problématique, les figures demandées ainsi que leur analyse.

Ce compte-rendu sera accompagné des **fichiers** *main.py* et *signal_processing.py* contenant le programme principal permettant la génération des figures et de leurs légendes et les différentes fonctions commentées selon la norme PEP 257.

Données de départ

Dans cette séquence, vous serez amenés à utiliser des données provenant d'un fichier de points issu d'un **oscilloscope**. Le fichier se nomme B2_DATA_01.CSV.

Le signal qu'il contient est un enregistrement d'une **transmission d'informations modulées en amplitude** par un signal porteur sinusoïdal.

Il existe également un second fichier, nommé B2_DATA_02.CSV contenant un message caché...

Ressources

Cette séquence est basée sur le langage Python.

Vous pouvez utiliser l'environnement **JupyterHub@Paris-Saclay** - https://jupyterhub.ijclab.in2p3.fr/ ou l'environnement **Spyder 5** inclus dans *Anaconda 3*.

Des tutoriels Python (et sur les bibliothèques classiques : Numpy, Matplotlib or Scipy) sont disponibles à l'adresse : http://lense.institutoptique.fr/python/. Parmi ces tutoriels, nous vous suggérons de lire les suivants :

- Comment créer une fonction proprement (incluant des commentaires docstring)
- Comment créer des vecteurs (Numpy)
- Comment afficher des données depuis des vecteurs (Matplotlib et Numpy)

Outils Numériques

Fonctions et bibliothèques conseillées :

- **Numpy** gestion de matrices :
  - **arange**
  - **linspace**
  - **logspace**
- **Matplotlib** affichage de données :
  - **plotly**
  - **figure, plot**
  - **subplot**
  - **legend, title**
  - **xlabel, ylabel**
  - **show**
- **Scipy** fonctions scientifiques :
  - **fftpack** sublibrary
  - **fft, ifft**
  - **fftshift**
  - **fftfreq**

Outils avancés :

- **rcParams** de MatPlotLib.pyplot pour l'amélioration de l'affichage de courbes