Mode d'Emploi :

Lecture de fichiers CDF avec Python

DE GELIS Pierre-Marie
M2 Space Sciences and Applications

2 Octobre 2024

1 Introduction

Les fichiers CDF (*Common Data Format*) sont utilisés pour stocker des ensembles de données multidimensionnels, souvent dans des applications scientifiques, notamment pour les données spatiales, climatiques ou géophysiques. Ce document présente un mode d'emploi sur la façon de lire et manipuler ces fichiers en Python à l'aide de la bibliothèque `spacepy`.

La plupart des fichiers CDF de ce type peuvent être lus de manière similaire en adaptant les variables spécifiques, telles que les noms des champs magnétiques, les masques de qualité, et les autres métadonnées propres à chaque fichier.

Ce code est conçu pour traiter et visualiser des données CDF de type "multi-mag-rpw-scm-merged-rtn-256", contenant diverses variables telles que des champs magnétiques dans le référentiel RTN (Radial-Tangential-Normal), des masques de qualité, ainsi que des métadonnées associées comme la date, les fréquences d'échantillonnage

2 Installation des bibliothèques

Avant de pouvoir lire des fichiers CDF, il est nécessaire d'installer les bibliothèques Python requises. La bibliothèque principale utilisée est `spacepy`, ainsi que `numpy` pour la manipulation de tableaux et `matplotlib` pour la visualisation des données.

Pour cela, il faut ouvrir le terminal de commande et écrire :

```
pip install spacepy
```

3 Lecture d'un fichier CDF

3.1 Importation des bibliothèques

Voici les bibliothèques à importer pour lire et manipuler un fichier CDF.

```
import numpy as np
from spacepy import pycdf
import matplotlib.pyplot as plt
```

3.2 Fonction de lecture des données CDF

Cette fonction permet de charger un fichier CDF et d'extraire les principales variables de données.

```
def load_merged_data(file_path):
    """
    Charge les données à partir d'un fichier CDF et filtre selon des critères spécifiques.

    Parameters:
    - file_path: str, chemin vers le fichier CDF

    Returns:
    - B_merged: np.ndarray, données du champ magnétique dans le cadre RTN.
    - epoch: np.ndarray, horodatages pour chaque point de données.
    - t_event: np.ndarray, horodatages au format Unix.
    - qbm: np.ndarray, données de masques de qualité.
```

```
    - qbm_mag: np.ndarray, masques de qualité magnétique.
    - qbm_scm: np.ndarray, masques de qualité SCM.
    - freq: np.ndarray, taux d'échantillonnage.
    """

    # Ouvrir le fichier CDF
    cdffile1 = pycdf.CDF(file_path)

    # Extraire les données nécessaires
    epoch = cdffile1["Epoch"][...]
    B_merged = cdffile1["B_RTN"][...]
    qbm = cdffile1["QUALITY_BITMASK"][...]
    qbm_mag = cdffile1["QUALITY_BITMASK_MAG"][...]
    qbm_scm = cdffile1["QUALITY_BITMASK_SCM"][...]

    # Convertir les données de temps au format Unix (secondes depuis le 1er janvier 1970)
    t_event = np.array([e.timestamp() for e in epoch])

    # Extraire la fréquence d'échantillonnage
    freq = cdffile1["SAMPLING_RATE"][...]

    # Fermer le fichier CDF pour libérer des ressources
    cdffile1.close()

    return B_merged, epoch, t_event, qbm, qbm_mag, qbm_scm, freq
```

Cette fonction est personnalisable pour s'adapter à tout fichier CDF qui suit une structure similaire, comme des données de champ magnétique, des masques de qualité, ou des taux d'échantillonnage.

3.3 Chargement des données

Pour charger les données d'un fichier CDF spécifique avant de les visualiser, assurez-vous de spécifier correctement le chemin du fichier.

```

# Définir le chemin et le nom du fichier
path1 = "/Users/Adminlocal/Desktop/PROJET/"
file1 = "nom_du_fichier.cdf"  # exemple de fichier de données

# Charger les données du fichier CDF
B_merg, epoch, t_event, qbm, qbm_mag, qbm_scm, freq = load_merged_data(path1 + file1)
```

3.4 Lister les variables du fichier

Cette fonction permet de lister toutes les variables disponibles dans un fichier CDF. Cela peut être utile pour explorer les données avant d'effectuer des analyses spécifiques.

```
def lister_variables_cdf(file_path):
    """
    Liste toutes les variables disponibles dans un fichier CDF.

    Parameters :
    - file_path: str, chemin vers le fichier CDF

    Returns :
    - Les noms de toutes les variables dans le fichier CDF.
    """
    # Ouvrir le fichier CDF
```

```
    with pycdf.CDF(file_path) as fichier_cdf:
        # Lister et afficher toutes les variables
        print("Variables disponibles dans le fichier CDF :")
        for variable in fichier_cdf:
            print(variable)

# Exemple d'utilisation
lister_variables_cdf(path1 + file1)
```

4 Visualisation des données

Une fois les données extraites, il est souvent utile de les visualiser. Voici des exemples pour tracer les composantes du champ magnétique ainsi que sa norme.

4.1 Tracer les composantes du champ magnétique

```
# Tracer les composantes du champ magnétique B (Bx, By, Bz)
plt.figure(figsize=(10, 6))
plt.plot(t_event, B_merg[:, 0], label="Bx (RTN)")
plt.plot(t_event, B_merg[:, 1], label="By (RTN)")
plt.plot(t_event, B_merg[:, 2], label="Bz (RTN)")
plt.xlabel("Temps")
plt.ylabel("Champ Magnétique (nT)")
plt.title("Composantes du Champ Magnétique en RTN")
plt.legend()
plt.grid(True)

# Rotation des labels pour une meilleure lisibilité
plt.gcf().autofmt_xdate()
plt.show()
```

4.2 Tracer la norme du champ magnétique

```
# Calcul de la norme du champ magnétique
B_norm = np.sqrt(B_merg[:, 0]**2 + B_merg[:, 1]**2 + B_merg[:, 2]**2)

# Tracer la norme
plt.figure(figsize=(10, 6))
plt.plot(t_event, B_norm, label="|B| (RTN)", color='purple')
plt.xlabel("Temps")
plt.ylabel("Norme du Champ Magnétique (nT)")
plt.title("Norme du Champ Magnétique en RTN")
plt.legend()
plt.grid(True)

# Rotation des labels pour une meilleure lisibilité
plt.gcf().autofmt_xdate()
plt.show()
```