# Guide des notebooks et fichiers créés

## Base de données piezos

1. Récupération des données
   1. Requête API et création du dataframe

Fonction f\_list\_piezzo(departement:list, nbre\_mesures:int=50) -> list:

Permet de requeter les propriétés de piezos qui ont au moins 50 mesures sur la liste des départements présents dans la liste

Création d'un dataframe Pandas avec la liste des piezos et leurs propriétés

1. Création de la base de données
   1. Création directe à partir du dataframe avec sqlalchemy

Création d'une base de données SQLite dans le répertoire '../data/liste\_piezos.db' si inexistante.

Création d'une table data\_piezo avec les données des piézos.

Possibilité d'écraser la base existante pour en créer une nouvelle. (Déconseillé car perte des informations d'analyses déjà réalisés).

* 1. Fichiers disponibles pour les piezos ?

Fonction permettant d'introduire dans la base de données si les chroniques piezo, ERA5 et les fichiers combinés sont présents dans les répertoires data correspondants. Cette fonction est également présente dans le notebook de récupération des fichiers ERA5.

## Import Hub'eau

Requete auprès de la base de données pour récupérer les informations des clusters

Fonction f\_data\_piezo pour telecharger auprès de l'API Hub'eau la chronique de chaque piezo si celle-ci n'existe pas déjà dans le repertoire ../data/piezo.

1. Collecte de toutes les chroniques de la liste établie à partir de la bdd

Téléchargement au format csv des chroniques. Si un code\_bss n'a pas de chronique via Hub'eau, celui-ci est supprimé de la base de données.

## Import ERA5

1. Fonction pour lancer les requêtes

Pour ne pas lancer trop de requêtes en simultanée, la fonction permet de lancer les requêtes par région.

Les critères de la requête à l'API sont au format JSON.

1. Lancer les requêtes auprès de l'API

Le programme demande de valider la requête. Attention, possibilité de lancer des requêtes en doublon. Il est préférable d'attendre que tous les fichiers soient disponible en téléchargement sur le site <https://cds.climate.copernicus.eu/cdsapp#!/yourrequests> avant de relancer des requêtes.

Le programme vérifie que les fichiers associés au code bss ne sont pas déjà présents avant de lancer la requête. Si le fichier n'existe pas, les données code\_bss et l'identifiant de requête API sont stockés dans un fichier csv (../data/hist\_requetes.csv) pour récupérer les données ultérieurement et associé le fichier de sortie avec le nom du code\_bss.

1. Récupérer automatiquement les requêtes abouties

Le programme peut être lancé à tout moment pour télécharger les fichiers mis à dispositions via l'API.

Le programme associe les identifiants de requêtes avec le code\_bss depuis le fichier csv ../data/hist\_requetes.csv et télécharge les fichiers via l'API sous le format code\_bss.nc. Les fichiers .nc sont convertis en fichiers .pkl puis supprimés.

1. Jointure era5 et piezo

Le programme créé un nouveau fichier csv dans le repertoire ../data/combined\_data/ avec une jointure sur le temps entre les fichiers piezo et les fichiers era5.

La base de données est mise à jour pour indiquer quels fichiers sont disponibles.

## Clean région

Le programme permet pour une liste de régions de :

* Récupérer les chroniques de l'ensemble des clusters dans un dataframe qui a pour index les dates de mesure
* Aggréger les données à la semaine avec comme point de départ le 1er Janvier 1970 ("epoch").
* Trouver une date de début et une date de fin espacées d'au moins 8 ans où il y a un maximum de piézos qui possèdent des données à ces dates.
* Supprimer les données aberrantes des chroniques avec la méthode IQR
* Supprimer les piézos qui ont des données manquantes consécutives de plus de 10 semaines
* Interpoler les données manquantes avec la méthode linéaire.
* Sauvegarder le dataframe nettoyé au format ../data/clusters/{region}.csv

## Clustering – région auto

1. Détermination du nombre optimal de clusters à créer

Le programme permet de :

* Sélectionner une région
* Standardisation des données
* Déterminer le nombre optimal de cluster à créer avec l'algorithme KMeans (f\_clusters\_coude()). Possibilité de renseigner la plage du nombre de clusters à évaluer et le nombre d'itération pour des positions de départ différentes des centroïdes. Plus ce nombre est important et plus la précision des résultats est élevée mais plus le temps de calcul est élevé.

1. Création des clusters

Un dictionnaire est créé avec comme clé la région et en valeur le nombre optimal de clusters.

Pour chaque région, k clusters sont créés en fonction de la valeur correspondante dans le dictionnaire.

Le profil moyen de chaque cluster est calculé. Les chroniques dont la valeur du MSE est abbérante (méthode IQR). La fonction f\_cluster\_cleaning permet d'itérer cette élimination jusqu'à ce qu'il n'y ait plus de profil aberrant.

Chaque chronique se voit attribué un code de cluster sous la forme "region\_lettre".

Le profil moyen des clusters est enregistré au format "region.csv".

Une table dans la base de données SQLite est créée pour stocker le nom du piézomètre dont la chronique possède le MSE le plus faible vis-à-vis du profil moyen du cluster associé.

## Clean – France\_flagship\_piezo

Le profil moyen de chaque cluster ne peut être utilisé pour du clustering à l'échelle nationale. Cela est dû au fait que les plages de temps optimales pour chaque région (plages de temps > 8 ans avec le maximum de piézos possédant des données) sont différentes. De ce fait la période de recouvrement entre ces profils moyens est très faible voire inexistante.

Les chroniques type de chaque cluster sont donc utilisées pour le clustering national. Le même traitement que dans le notebook "clean – région" est utilisé.

Le dataframe après nettoyage est enregistré au format csv (FRANCE.csv).

## Clustering – France

Pour le clustering à l'échelle nationale, une hiérarchisation est souhaitée afin de réaliser un découpage à plusieurs niveaux et connaitre les liens entre ces niveaux de clustering.

Le module cluster.hierarchy.linkage du package Scipy a été choisi dans ce sens. Il permet de réaliser un dendrogramme permettant de visualiser la proximité entre les données d'entrée.

Dans le cas présent, 4 niveaux de découpage ont été réalisée pour constituer 3, 5, 11 et 19 clusters.

Les clusters sont renommés depuis le niveau 1 jusqu'au niveau 4. Au niveau 1, les clusters possèdent les numéros 1, 2 et 3. Au niveau, les clusters sont nommés 11 et 12 (issus du cluster 1 du niveau précédent), 21 et 22 (issus du cluster 2) et 31 (cluster 3 qui n'a pas été divisé à ce niveau).

Chaque piézomètre se voit attribué son numéro de cluster pour les 4 niveaux de découpage créés. Ces données sont entrées dans la base de données.

Pour chaque niveau, et pour chaque cluster, une moyenne des chroniques est réalisée en fonction du temps. La piézomètre dont le MSE avec la moyenne du cluster est le plus faible est désigné comme le piézomètre type de ce cluster. Ces informations sont renseignées dans la table flagship\_piezo de la base de données.