

```
"""
```

Script for formatting observation data.

The script allows you to:

- rename the labels so that the format of the input file names is consistent
- associate each trace labelled by an observer with the nearest well
- Optional: Put the point closest to the raw well in first position.
- Optional: Update the well's position based on the traces.

The raw data must be organised as follows:

raw/

```
|— pleiades
|   |— 2024
|       |— obs1
|       ...
|       |— obs5
|       |— well
|
...
|— spot_6
|   |— 2015_01_30
|       |— obs1
|       |— ...
|       |— obs4
|       |— well
|   |— 2018_01_26
|       |— obs1
|       |— ...
|       |— obs4
|       |— well
```

```
"""
```

```
import os
from dataclasses import dataclass
```

```
import geopandas as gpd
import shapely.geometry
```

```
@dataclass
class Sensor:
    name: str
    swap_ends: bool = False # Put the point closest to the raw well in
                             first position.
```

```

    update_well: bool = False # Update the well's position based on the
traces.
    date: str = ""

def rename_repository(sensor: Sensor, observers: list[str]):
    """Rename the raw files so that the input file names are all in the same
format."""
    sensor_path = os.path.join("raw", sensor.name, sensor.date)

    if not os.path.isdir(sensor_path):
        return

    well_path = os.path.join(sensor_path, "well")

    if os.path.isdir(well_path):
        for file in os.listdir(well_path):
            old_name = os.path.join(well_path, file)
            ext = os.path.splitext(old_name)[1]
            new_name = os.path.join(
                well_path, f"puits_{sensor.name}_{sensor.date}{ext}"
            )
            if new_name != old_name:
                os.rename(old_name, new_name)
                print(f"Renamed : {old_name} → {new_name}")

    for obs in observers:
        obs_path = os.path.join(sensor_path, obs)

        if not os.path.isdir(obs_path):
            continue

        for file in os.listdir(obs_path):
            if "lignes" in file.lower():
                old_name = os.path.join(obs_path, file)
                ext = os.path.splitext(old_name)[1]
                new_name = os.path.join(
                    obs_path, f"lignes_{sensor.name}_{sensor.date}_{obs}
{ext}"
                )
                if new_name != old_name:
                    os.rename(old_name, new_name)
                    print(f"Renamed : {old_name} → {new_name}")

def prepare(sensor: Sensor, obs: str):

```

```

"""
    Associates each trace with the nearest well. If necessary, it can update
    the position of the wells based on the traces and reorder the points in a
    trace so that the point on the well is the first. """
# Read
sensor_path = os.path.join("raw", sensor.name, sensor.date)
well_filename = os.path.join(
    sensor_path, "well", f"puits_{sensor.name}_{sensor.date}.shp"
)
line_filename = os.path.join(
    sensor_path, obs, f"lignes_{sensor.name}_{sensor.date}_{obs}.shp"
)

if not os.path.isfile(line_filename):
    return

print(f"Preparing: {sensor.name}-{sensor.date} for obs: {obs}")

points = gpd.read_file(well_filename)
lines = gpd.read_file(line_filename)
points = points.to_crs(lines.crs)

def distance_l2(c1, c2):
    return (c1[0] - c2[0]) ** 2 + (c1[1] - c2[1]) ** 2

for idx_line, line_geometry in enumerate(lines.geometry):
    if line_geometry is None:
        continue

    best_point = None
    best_distance = 1000
    switch = False

    for idx_point, point in points.iterrows():
        d1 = distance_l2(line_geometry.coords[0],
point["geometry"].coords[0])
        d2 = distance_l2(line_geometry.coords[1],
point["geometry"].coords[0])
        distance = min(d1, d2)

        if distance < best_distance:
            best_point = idx_point
            best_distance = distance
            switch = d2 < d1
            if distance < 1:
                break

```

```

    if switch and sensor.swap_ends:
        lines.at[idx_line, "geometry"] = shapely.geometry.LineString(
            list(line_geometry.coords[::-1])
        )

    if sensor.update_well:
        points.at[best_point, "geometry"] = shapely.geometry.Point(
            line_geometry.coords[0]
        )

    lines.at[idx_line, "puits"] = points.at[best_point, "Field1"]
    lines.at[idx_line, "date_acq"] = sensor.date

# Save
out_path = os.path.join("final", sensor.name, sensor.date, obs)
os.makedirs(out_path, exist_ok=True)

    if sensor.update_well:
        points.to_file(
            os.path.join(out_path,
                f"puits_{sensor.name}_{sensor.date}_{obs}.shp")
        )
        lines.to_file(
            os.path.join(out_path,
                f"lignes_{sensor.name}_{sensor.date}_{obs}.shp")
        )

spot_6_2015 = Sensor("spot_6", swap_ends=True, date="2015_01_30")
spot_6_2018 = Sensor("spot_6", swap_ends=True, date="2018_01_26")
pleiades = Sensor("pleiades", update_well=True, date="2024")
pleiades_neo = Sensor("pleiades_neo", update_well=True, date="2025_02_17")

observers = ["obs1", "obs2", "obs3", "obs4", "obs5"]
sensors = [spot_6_2015, spot_6_2018, pleiades, pleiades_neo]
for sensor in sensors:
    rename_repository(sensor, observers)
    for obs in observers:
        prepare(sensor, obs)

```