**DLQ Step 2**

**Overview**

Methane **Event Detection, Localization and Quantification** is a Python-based code designed for detecting and analyzing methane emission events using sensor data and simulation results. Methane detection is crucial in environmental monitoring and industrial safety, as it helps identify leaks, track emissions, and ensure compliance with environmental standards.

This system processes raw sensor readings by removing background noise, detecting spikes in methane concentrations (events), aligning the detected events with simulation data, and performing a quantitative analysis of these events, including emission source localization and rate estimation.

**Key Features**

The system offers the following core features:

* **Spike/Event Detection**: Automatically identifies significant increases in methane concentration (spikes) in time-series data, helping to pinpoint potential methane release events.
* **Background Noise Removal**: Smooths out the methane concentration data by interpolating gaps and removing long-term background levels, ensuring only relevant fluctuations are analyzed.
* **Event Classification and Quantification**: Classifies spikes into events and calculates key statistics such as peak methane concentration and event duration.
* **Alignment with Simulations**: Matches detected events with simulation data to ensure consistency, and computes performance metrics for comparison.
* **Source Localization**: Estimates the location of methane sources based on detected spikes and available simulation data.
* **Emission Rate Calculation**: Quantifies methane emission rates for each detected event and provides uncertainty estimations to account for variations in data or measurement noise.

**System Requirements**

The system is built on Python and requires several libraries for data processing and visualization. Make sure the following dependencies are installed:

* **Python version**: 3.x
* **Required Python libraries**:
  + pandas: Data manipulation and analysis
  + numpy: Numerical computing
  + scipy: Scientific computing, including signal processing
  + matplotlib: Data visualization
  + pickle: Serialization and deserialization of Python objects
  + logging: Systematic logging for monitoring program execution

These libraries can be installed using pip:

pip install pandas numpy scipy matplotlib pickle

**Core Components**

**1. Spike Detection (find\_spikes)**

* **Purpose**: Identifies significant spikes in methane concentration data, representing potential emission events.
* **Methodology**: The algorithm identifies sudden rises and falls in concentration levels, using thresholds for the rate of increase and the amplitude of the spike to ensure only valid methane releases are detected.
* **Parameters**:
  + going\_up\_threshold: Defines the minimum rate of increase required to classify a spike (default: 0.25).
  + return\_threshold: Threshold to determine the end of the spike when concentrations drop (default: 5).
  + amp\_threshold: Minimum amplitude a spike must reach to be considered valid (default: 1).
  + cont\_diff\_threshold: Threshold for continuous changes in concentration to account for small fluctuations (default: 0.25).
  + cont\_diff\_num: Number of continuous differences to check for a valid spike (default: 10).

**2. Background Removal (remove\_background)**

* **Purpose**: Removes background methane concentrations, which represent long-term steady levels, leaving only the short-term fluctuations (spikes) that represent potential methane emission events.
* **Methodology**:
  + Missing values in the data are interpolated to maintain continuity.
  + Spikes are identified, and events are grouped if they occur close to each other within a specified time frame.
  + The algorithm estimates the background levels and removes them to isolate the relevant methane fluctuations.

**3. Event Detection (detect\_events)**

* **Purpose**: Detects and groups spikes into distinct methane events, where each event represents a possible methane release.
* **Parameters**:
  + gap\_time: Defines the maximum time gap between two consecutive spikes to consider them part of the same event (default: 5).
  + length\_threshold: Minimum duration for a spike or series of spikes to be considered a valid event (default: 2).
* **Output**: A Data Frame containing event classifications, including timestamps, maximum concentrations, and event durations.

**4. Event Alignment and Quantification**

This is the heart of the system, providing detailed analysis and comparison of detected methane emission events.

* **Event Alignment**: Compares detected methane events against simulation data, using metrics such as correlation, mean squared error, or other performance measures to ensure that the detected events align with expected simulation outcomes.
* **Localization and Quantification**:
  + **Source Localization**: Using event data and spatial simulation information, the system estimates the possible source of methane emissions.
  + **Emission Rate Calculation**: Estimates methane emission rates for each event, based on concentration data, wind conditions, and other relevant environmental variables. Uncertainty in these estimates is calculated to provide confidence bounds for the emission rates.

**File Structure**

The system processes and saves multiple data files at different stages. The files are structured as follows:

output directory/

├── step1\_raw\_observations.csv # Raw sensor data with timestamps

├── step1\_timestamps\_new.csv # Processed timestamp information

├── step2\_background\_removed\_obs\_new.csv # Cleaned data after background removal

├── step3\_event\_detection\_new.csv # Detected events with timestamps

├── step4\_alignment\_metrics\_py.csv # Metrics for alignment with simulation data

├── step5\_event\_details\_new.csv # Detailed analysis of each detected event

└── step5\_results\_new.pkl # Pickle file with all final results

**Detailed Steps in the Pipeline**

**Step 1: Raw Data Processing**

* **Input**: Raw sensor data (CSV or binary) and simulation data (Pickle file).
* **Output**: Raw observations saved in step1\_raw\_observations.csv and timestamps in step1\_timestamps\_new.csv.

**Step 2: Background Removal**

* **Input**: Raw observations.
* **Process**: Background removal algorithm processes the data to remove steady-state methane levels.
* **Output**: Cleaned methane concentrations are saved in step2\_background\_removed\_obs\_new.csv.

**Step 3: Event Detection**

* **Input**: Cleaned sensor data.
* **Process**: Spike detection algorithm groups significant spikes into distinct events based on the defined thresholds and gaps between spikes.
* **Output**: Detected events and their characteristics are saved in step3\_event\_detection\_new.csv.

**Step 4: Event Alignment**

* **Input**: Detected events and simulation data.
* **Process**: The system compares the detected events with simulation data to calculate alignment metrics.
* **Output**: Metrics are saved in step4\_alignment\_metrics\_py.csv.

**Step 5: Localization and Quantification**

* **Input**: Detected events and simulation data.
* **Process**: The system estimates source locations, calculates emission rates, and provides uncertainty bounds for each event.
* **Output**: Detailed event analysis is saved in step5\_event\_details\_new.csv, and the final results are serialized in step5\_results\_new.pkl.

**Usage Instructions**

**Step 1: Set Up Configuration**

Edit the configuration file to adjust parameters such as file paths, thresholds for event detection, and simulation data location. Example:

config = {

'gap\_time': 5,

'length\_threshold': 2,

'simulation\_data\_path': "/path/to/simulation\_output\_new.pkl",

'output\_file\_path': "/path/to/output/",

}

**Step 2: Run the Analysis**

To execute the methane event detection system, run the Python script as follows:

python step2\_DLQ.py

This will process the data and generate output files in the specified output directory.

**Output Files Description**

1. **Raw Observations** (step1\_raw\_observations.csv): Contains the raw sensor readings and timestamp information.
2. **Background Removed Data** (step2\_background\_removed\_obs\_new.csv): Processed data after background removal, focusing on short-term fluctuations in methane concentration.
3. **Event Detection Results** (step3\_event\_detection\_new.csv): A table of detected events with associated timestamps and maximum methane concentrations.
4. **Alignment Metrics** (step4\_alignment\_metrics\_py.csv): Metrics quantifying how well detected events align with simulation data, such as correlation coefficients.
5. **Event Details** (step5\_event\_details\_new.csv): A detailed breakdown of each event, including estimated source locations, emission rates, and uncertainty estimations.

**How It Works**

The methane event detection system follows a stepwise approach:

1. **Initial Data Loading**: Loads raw sensor data and simulation outputs.
2. **Background Removal**: Cleans up the data by removing long-term background methane levels.
3. **Event Detection**: Automatically detects and classifies events based on concentration spikes.
4. **Event-Simulation Alignment**: Compares detected events with simulation data to ensure consistency and calculate performance metrics.
5. **Source Localization and Emission Rate Estimation**: Estimates the location of methane sources and quantifies emission rates for each detected event.

**Conclusion**

The DLQ System is a powerful tool for analyzing methane emissions from sensor data. It offers automated event detection, source localization, and emission rate quantification, making it an essential tool for environmental monitoring, industrial safety, and regulatory compliance.

**Details on plots (extra information)**

 **Event Timeline Plot (event\_timeline.png):**

* Shows the concentration measurements over time
* Detected events are highlighted in different colors
* Helps identify the temporal pattern of emission events
* Higher peaks indicate stronger concentration measurements

 **Enhanced Wind Rose (wind\_rose\_enhanced.png):**

* Shows wind direction and speed distribution
* Color-coded by wind speed ranges
* Longer bars indicate more frequent wind from that direction
* Helps understand predominant wind patterns during measurements

 **Emission Rate Estimates (emission\_rates.png):**

* Shows estimated emission rates for each detected event
* Error bars indicate uncertainty in the estimates
* Source locations are labeled
* Helps assess the magnitude and reliability of emission estimates

 **Concentration Heatmap (concentration\_heatmap.png):**

* Shows concentration readings across all sensors over time
* Darker colors indicate higher concentrations
* Helps identify spatial patterns in the measurements
* Useful for seeing which sensors detected the events most strongly

 **Wind Speed vs Concentration (wind\_speed\_concentration.png):**

* Shows relationship between wind speed and measured concentrations
* Helps understand how wind conditions affect measurements
* Clusters or patterns might indicate optimal detection conditions

 **Event Durations (event\_durations.png):**

* Shows how long each detected event lasts
* Helps understand typical event timeframes
* Useful for identifying brief vs prolonged emission events