# 甲烷浓度增强反演仓库技术文档

## 目录

* [项目概览](#overview)
  + [整体架构](#system)
* [甲烷反演算法](#Methaneretrieval)
  + [匹配滤波算法](./matchedfilter.md)
  + [多层匹配滤波算法](./columnwiseprocessing.md)
  + [列处理](./columnwiseprocessing.md)
* [卫星数据预处理](#卫星数据预处理)
  + [支持的卫星平台]
  + [数据校正和预处理]
* [单位吸收谱构建]
  + [辐亮度查找表]
* [影像模拟和测试]
* [批处理](#批处理)
* [排放速率估算](#排放速率估算)
  + [综合质量增强法]
  + [风速计算]

# MF 仓库概况

## 目的与范围

MF (Matched Filter，匹配滤波) 仓库是一个专门用于从高光谱卫星数据中提取甲烷浓度增强信息的代码库。它实现了多种匹配滤波算法变体，特别关注多层匹配滤波 (Multi-Layer Matched Filter, MLMF) 方法，该方法可提高高甲烷浓度区域的提取精度。该系统支持多种高光谱卫星平台，并包含模拟甲烷羽流、生成单位吸收光谱和估算排放速率的工具。

有关特定甲烷提取算法的详细信息，请参阅 [甲烷反演算法](./Methaneretrieval.md)， 有关卫星数据处理的信息，请参阅 卫星数据预处理。

## 系统架构

MF 仓库采用模块化架构组织，包含几个关键组件协同工作以执行甲烷浓度提取。

### 高层架构

下方是 MF 仓库的高层架构图，展示了主要组件及其交互：

### 数据处理流程

MF 仓库实现了从输入数据到甲烷提取结果的完整数据处理流程：

## 核心功能

### 甲烷提取算法

该仓库实现了几种复杂度递增的甲烷提取算法：

Algorithm FeaturesCore ComponentsMethane Retrieval AlgorithmsStandard Matched FilterColumnwise Matched FilterMulti-Layer Matched FilterColumnwise Multi-Layer  
Matched FilterKalman Filter  
(Placeholder)Lognormal  
(Placeholder)Unit Absorption SpectrumRadiance Lookup TablesCovariance MatrixTransmittance SpectraIterative RefinementAlbedo AdjustmentSparsity AdjustmentDynamic Threshold  
Adjustment

####单位吸收光谱生成

单位吸收光谱 (UAS) 是匹配滤波算法的关键组件。它表示甲烷在不同浓度下的光谱特征。系统使用辐射查找表生成 UAS 和透射光谱：

### 卫星数据支持

该系统支持多种高光谱卫星平台，提供专门的数据读取器和处理函数。

| 平台 | 数据格式 | 支持算法 |
| --- | --- | --- |
| PRISMA | HDF5 | MF, MLMF |
| EnMAP | GeoTIFF | MF, MLMF |
| EMIT | NetCDF | MF, MLMF |
| ZY1 | DAT/HDR | MF, MLMF |
| GF5B-AHSI | TIFF | MF, MLMF |

每个卫星平台在 utils/satellites\_data 目录下都有自己的数据读取器。

### 图像模拟

该系统包含模拟带有甲烷羽流的卫星图像的功能，这对于测试和验证提取算法非常有用：

## 仓库结构

MF 仓库按以下主要目录组织：

| 目录 | 目的 |
| --- | --- |
| methane\_retrieval\_algorithms/ | 核心甲烷提取算法实现 |
| utils/ | 用于数据处理、UAS 生成和模拟的工具函数 |
| utils/satellites\_data/ | 卫星数据读取器和处理工具 |
| utils/emission\_estimate/ | 用于估算甲烷排放速率的工具 |
| data/ | 输入数据存储（查找表、卫星通道等） |
| results/ | 提取结果输出存储 |
| figures/ | 可视化输出存储 |
| tasks/ | 执行特定任务或工作流程的脚本 |
| docs/ | 文档文件 |

## 多层匹配滤波算法 (MLMF)

该仓库的关键创新是多层匹配滤波 (MLMF) 算法，它通过处理高甲烷浓度下的非线性吸收效应，改进了标准匹配滤波。

MLMF 算法的工作原理如下：

1. 使用标准匹配滤波器计算初始甲烷浓度估算值。
2. 对于高浓度像素，动态调整处理过程。
3. 对不同浓度范围使用多层单位吸收光谱和透射光谱。
4. 根据当前估算值迭代优化结果。

MLMF 算法的核心实现在 methane\_retrieval\_algorithms/ml\_matchedfilter.py 中。

主要函数 ml\_matched\_filter() 接收以下关键参数：

* data\_cube: 卫星数据的 3D 数组
* initial\_unit\_absorption\_spectrum: 用于第一遍处理的初始 UAS
* uas\_list: 不同浓度范围的 UAS 数组
* transmittance\_list: 不同浓度范围的透射光谱数组
* iterate: 迭代优化标志
* albedoadjust: 反照率调整标志
* dynamic\_adjust: 动态阈值调整标志

## 工作流程示例

使用 MF 仓库的典型甲烷提取工作流程包括以下步骤：

多层匹配滤波算法的示例参数：

* 波长范围：2150-2500 nm (短波红外)
* 动态调整阈值：5000-50000 ppm·m
* 太阳天顶角：0-90 度
* 地面高度：0-5 km

# 系统架构

本文详细概述了甲烷提取 (MF) 系统的架构，描述了核心组件、它们之间的交互以及数据流。该架构旨在利用各种匹配滤波算法高效处理高光谱卫星数据，以提取甲烷浓度。

## 系统概述

MF 系统旨在利用匹配滤波算法从卫星图像中检测和量化甲烷排放。它支持多种卫星平台，并包含用于数据预处理、甲烷浓度提取和排放速率估算的组件。

## 核心算法架构

系统实现了几种复杂度和功能逐步增加的甲烷提取算法。这些算法使用单位吸收光谱 (UAS) 处理卫星数据，以检测甲烷羽流。

核心匹配滤波算法有几种变体：

1. **标准匹配滤波 (Standard Matched Filter)**：通过与背景光谱比较来检测甲烷的基本实现。
2. **列式匹配滤波 (Columnwise Matched Filter)**：按列组处理图像数据，以更好地管理内存。
3. **多层匹配滤波 (Multi-Layer Matched Filter)**：对不同甲烷浓度范围使用多个单位吸收光谱。
4. **列式多层匹配滤波 (Columnwise Multi-Layer Matched Filter)**：结合了列式处理和多层方法。

## 数据流架构

下图展示了系统完整的数据流，从数据输入到最终的分析输出：

## 卫星数据处理子系统

系统支持多种卫星平台，并为每种平台提供了专门的数据读取器。处理流程处理每种卫星平台的独特数据格式和特性。

系统通过以下方式处理卫星数据：

1. **数据读取器 (Data Readers)**：特定于平台用于读取和解释原始卫星数据的模块。
2. **元数据提取 (Metadata Extraction)**：提取太阳天顶角 (SZA) 和地面高度等关键参数。
3. **波段选择 (Band Selection)**：过滤到相关的短波红外 (SWIR) 波段（通常为 2150-2500 nm）。
4. **辐射定标 (Radiometric Calibration)**：将原始值转换为定标后的辐射度。
5. **UAS 生成 (UAS Generation)**：为特定条件创建单位吸收光谱。

## 单位吸收光谱生成

单位吸收光谱 (UAS) 是甲烷提取的关键组件。系统根据基于 MODTRAN 模拟的辐射查找表 (LUT) 生成 UAS。

UAS 生成过程中的关键函数包括：

1. **LUT 加载 (LUT Loading)**：加载预计算的特定卫星的辐射查找表。
2. **插值 (Interpolation)**：对特定参数在 LUT 值之间进行插值。
3. **UAS 计算 (UAS Calculation)**：计算不同浓度范围的单位吸收光谱。
4. **透射率计算 (Transmittance Calculation)**：计算在多层方法中使用的透射光谱。

## 图像模拟与测试

系统包含模拟带有甲烷羽流的卫星图像的功能，用于测试和验证。

图像模拟过程包括：

1. **羽流生成 (Plume Generation)**：使用高斯模型或真实模板创建甲烷羽流浓度图。
2. **基础辐射模拟 (Base Radiance Simulation)**：从查找表生成背景辐射光谱。
3. **羽流效应添加 (Plume Effect Addition)**：根据甲烷浓度应用透射率效应。
4. **噪声添加 (Noise Addition)**：添加逼真的传感器噪声。

## 排放速率估算

甲烷提取后，系统可以估算检测到的羽流的排放速率。

排放速率估算过程包括：

1. **羽流分割 (Plume Segmentation)**：将羽流从背景噪声中分离出来。
2. **质量计算 (Mass Calculation)**：将浓度转换为甲烷总质量。
3. **风速调整 (Wind Speed Adjustment)**：结合风速计算排放速率。
4. **不确定性估算 (Uncertainty Estimation)**：计算排放估算的置信区间。

## 批处理

系统支持对多个卫星图像进行批处理，以供日常操作使用。

批处理系统提供：

1. **文件管理 (File Management)**：用于过滤和选择合适文件的工具。
2. **ROI 处理 (ROI Processing)**：专注于特定感兴趣区域。
3. **结果管理 (Result Management)**：对结果进行有组织的存储和比较。
4. **操作自动化 (Operational Automation)**：用于执行常规处理任务的脚本。

## 技术栈和依赖

MF 系统使用以下关键技术构建：

| 组件 | 技术 |
| --- | --- |
| 核心算法 | NumPy, SciPy |
| 图像处理 | GDAL, Rasterio |
| 可视化 | Matplotlib, Seaborn |
| GIS 处理 | GeoPandas, Shapely |
| 数据格式 | HDF5, NetCDF, GeoTIFF |
| 科学计算 | SciPy, Numerical Python |

## 系统配置与部署

系统通过环境变量和配置文件进行配置：

| 配置项目 | 目的 |
| --- | --- |
| Conda 环境 | 包含所有依赖的 Python 环境 (mf\_environment.yml) |
| 数据路径 | 在工具模块中配置，用于查找表和卫星通道信息 |
| 算法参数 | 通过函数参数设置，提供适用于大多数场景的默认值 |
| 输出设置 | 在批处理脚本中配置 |

## 结论

MF 系统的模块化架构允许灵活处理不同的卫星平台、应用各种甲烷提取算法以及扩展功能。系统的核心组件——数据读取器、提取算法和排放估算——协同工作，为基于卫星的甲烷监测提供全面的解决方案。

有关甲烷提取算法的具体信息，请参阅 [甲烷反演算法](./Methaneretrieval.md)。

有关支持的卫星平台的详细信息，请参阅 支持的卫星平台。

# 甲烷反演算法

## 目的与范围

本文提供了 MF 仓库中实现的甲烷反演算法的技术文档。这些算法用于从高光谱卫星图像中检测和量化甲烷浓度。有关卫星数据处理的信息，请参阅 [Satellite Data Processing](https://deepwiki.com/Emeric53/MF/3-satellite-data-processing)；有关单位吸收光谱生成的详细信息，请参阅 [Unit Absorption Spectrum Generation](https://deepwiki.com/Emeric53/MF/4-unit-absorption-spectrum-generation)。

## 算法概述

仓库实现了四种主要的反演算法，其复杂性逐步增加：

1. 标准匹配滤波 (Standard Matched Filter, MF)
2. 多层匹配滤波 (Multi-Layer Matched Filter, MLMF)
3. 列式匹配滤波 (Columnwise Matched Filter, CMF)
4. 列式多层匹配滤波 (Columnwise Multi-Layer Matched Filter, CMLMF)

这些算法利用甲烷在短波红外 (SWIR) 区域（通常为 2100-2500 nm）独特的吸收特征。

### 甲烷反演算法层级结构

## 标准匹配滤波 (MF)

标准匹配滤波是系统的基础算法，实现在 matchedfilter.py 中。

### 算法流程图

### 实现细节

matched\_filter 函数接受卫星数据立方体和单位吸收光谱作为输入，并输出一个甲烷浓度值的二维数组：

def matched\_filter(  
 data\_cube: np.ndarray,  
 unit\_absorption\_spectrum: np.ndarray,  
 iterate: bool = False,  
 albedoadjust: bool = False,  
 sparsity: bool = False,  
) -> np.ndarray

算法的关键步骤：

1. **背景计算 (Background Calculation)**：计算所有像素的平均光谱。
2. **目标光谱创建 (Target Spectrum Creation)**：将背景光谱乘以单位吸收光谱。
3. **协方差计算 (Covariance Calculation)**：从辐射度差计算协方差矩阵。
4. **反照率调整 (Albedo Adjustment)**：可选地调整地表反射率的变化。
5. **浓度计算 (Concentration Computation)**：应用匹配滤波方程计算浓度。
6. **迭代优化 (Iterative Refinement)**：可选地通过多次迭代提高精度。

## 多层匹配滤波 (MLMF)

多层匹配滤波在标准方法的基础上进行了增强，以更好地处理高甲烷浓度，实现在 ml\_matchedfilter.py 中。

### 实现细节

ml\_matched\_filter 函数通过多层处理扩展了标准算法：

def ml\_matched\_filter(  
 data\_cube: np.ndarray,  
 initial\_unit\_absorption\_spectrum: np.ndarray,  
 uas\_list: np.ndarray,  
 transmittance\_list: np.ndarray,  
 iterate: bool = False,  
 albedoadjust: bool = False,  
 sparsity: bool = False,  
 dynamic\_adjust: bool = True,  
 threshold: float = 5000,  
 threshold\_step: float = 5000,  
 max\_threshold: float = 50000,  
) -> np.ndarray

主要增强功能：

1. **多层 UAS (Multiple UAS Layers)**：对不同的浓度范围使用不同的单位吸收光谱。
2. **透射率建模 (Transmittance Modeling)**：考虑高浓度下的非线性效应。
3. **动态阈值调整 (Dynamic Threshold Adjustment)**：逐步处理更高浓度的像素。
4. **分层处理 (Layer-by-Layer Processing)**：对每个浓度范围应用适当的 UAS。

函数 generate\_uas\_transmittance\_list 生成不同浓度层所需的 UAS 和透射率数组：

def generate\_uas\_transmittance\_list(satelitetype, sza, altitude)

## 逐列处理方法

两种算法的逐列变体旨在通过按列组处理数据，更有效地处理大型图像。

### 处理策略

### 实现细节

#### 逐列匹配滤波 (Columnwise Matched Filter)

def columnwise\_matched\_filter(  
 data\_cube: np.ndarray,  
 unit\_absorption\_spectrum: np.ndarray,  
 iterate: bool = False,  
 albedoadjust: bool = False,  
 sparsity: bool = False,  
 group\_size: int = 5,  
) -> np.ndarray

#### 逐列多层匹配滤波 (Columnwise Multi-Layer Matched Filter)

def columnwise\_ml\_matched\_filter(  
 data\_cube: np.ndarray,  
 initial\_unit\_absorption\_spectrum: np.ndarray,  
 uas\_list: np.ndarray,  
 transmittance\_list: np.ndarray,  
 iterate: bool = False,  
 albedoadjust: bool = False,  
 sparsity: bool = False,  
 group\_size: int = 5,  
 dynamic\_adjust: bool = True,  
 threshold: float = 5000,  
 threshold\_step: float = 5000,  
 max\_threshold: float = 50000,  
) -> np.ndarray

两个函数都执行以下操作：

1. 将数据立方体分割成大小为 group\_size 的列组。
2. 使用相应的非逐列原始算法独立处理每个组。
3. 将处理后的组合并成完整的浓度图。

这种方法减少了内存需求，并允许潜在的并行化。

## 单位吸收光谱生成

单位吸收光谱 (UAS) 是一个关键输入，表示甲烷在不同浓度下的光谱特征。

### UAS 生成过程

### 实现细节

generate\_radiance\_lut\_and\_uas.py 中的 UAS 生成函数包括：

def generate\_satellite\_uas\_for\_specific\_range\_from\_lut(  
 satellite\_name: str,  
 start\_enhancement: float,  
 end\_enhancement: float,  
 lower\_wavelength: float,  
 upper\_wavelength: float,  
 sza: float,  
 altitude: float,  
)

此函数执行以下操作：

1. 创建一个甲烷浓度范围。
2. 从查找表中检索每个浓度的辐射度值。
3. 计算辐射度值的自然对数。
4. 对对数辐射度与浓度进行线性拟合。
5. 将此斜率作为单位吸收光谱返回。

查找表是根据 MODTRAN 辐射传输模型的结果生成的。

## 算法参数和性能特性

### 算法通用参数

| 参数 | 类型 | 描述 | 影响 |
| --- | --- | --- | --- |
| data\_cube | np.ndarray | 3D 高光谱图像数据 | 主要输入数据 |
| unit\_absorption\_spectrum | np.ndarray | 甲烷的光谱特征 | 定义检测目标 |
| iterate | bool | 启用迭代优化 | 提高精度，但计算成本更高 |
| albedoadjust | bool | 考虑地表反射率变化 | 在非均匀地表上获得更好的结果 |
| sparsity | bool | 调整稀疏羽流特征 | 增强点源检测能力 |

### 多层特有参数

| 参数 | 类型 | 描述 | 影响 |
| --- | --- | --- | --- |
| uas\_list | np.ndarray | 不同浓度范围的多层 UAS | 更好地处理浓度变化 |
| transmittance\_list | np.ndarray | 不同浓度的透射率数据 | 考虑非线性效应 |
| dynamic\_adjust | bool | 启用多层处理 | 提高高浓度检测能力 |
| threshold | float | 初始浓度阈值 | 分层处理的起始点 |
| threshold\_step | float | 阈值增量 | 控制分层处理的进展 |
| max\_threshold | float | 最大浓度阈值 | 多层处理的上限 |

### 列式处理参数

| 参数 | 类型 | 描述 | 影响 |
| --- | --- | --- | --- |
| group\_size | int | 一起处理的列数 | 内存使用与组边界空间信息的权衡 |

## 算法选择指南

根据你的具体需求选择合适的算法：

| 算法 | 何时使用 | 优点 | 局限性 |
| --- | --- | --- | --- |
| 标准 MF | 初始检测、小图像、均匀低浓度 | 速度更快、实现更简单 | 对高浓度区域精度较低 |
| 多层 MF | 量化分析、宽浓度范围 | 在不同浓度范围内精度更高 | 计算需求更高 |
| 列式 MF | 大图像、内存受限 | 内存使用量低 | 可能在组边界失去一些空间上下文 |
| 列式 MLMF | 具有不同浓度的大图像 | 结合了内存效率和精度 | 计算量最大 |

为获得最佳结果：

1. 使用标准 MF 进行初始检测。
2. 如果存在高浓度，切换到 MLMF。
3. 对于超出内存限制的大图像，使用列式变体。
4. 根据场景特征和精度要求调整参数。

## 算法与支持组件之间的数据流