一、参数规范：数据字典（建模参数、决策变量、约束条件）参考申报书和建模文档，各阶段统一，避免后期返工，各自定义的参数不一致

二、接口规范（明确三模块输入 / 输出格式）

（一）模块一：光伏面板切割及分区规划

1. 输入格式

字段名称	类型	说明	示例
地形网格数据	.npy 数组	维度：[网格行数，网格列数，4]，4 列分别为 x 坐标、y 坐标、坡度、是否可建设（0/1）	见算例库标准文件
设备参数配置	.csv 文件	包含标准面板长度、逆变器容量、负载率等	面板长度：13 列等效长度；逆变器容量：320kW；负载率：0.85
切割规格需求	字典	key 为规格 ID，value 为该规格所需切割长度（m）	{"11":2.0, "12":4.0}（对应 2 列、4 列切割）

2. 输出格式（JSON 文件 / 字典）

json

```
{
  "分区列表": [
    {
      "分区 ID": "area_001",
      "面板数量": 22,   // 满足 1826 块约束
      "切割方案": [
        {"原材料 ID": "m1", "规格 ID": "l2", "切割数量": 5},
        {"原材料 ID": "m2", "规格 ID": "l3", "切割数量": 3}
      ],
      "分区边界坐标": [
        [x1, y1], [x2, y2], [x3, y3], [x4, y4]   // 多边形边界，按顺时针排序
      ],
      "分区周长": 120.5   // 单位：m
    },
```

```
    // 其他分区...
  ],
  "全局统计": {
    "总覆盖面积": 1500.2,  // 单位：m²
    "总面板采购数量": 120,  // 单位：块
    "总切割浪费率": 3.2  // 单位：%
  }
}
```

（二）模块二：电气设备选型选址及电缆共沟

1. 输入格式

字段名称	类型	说明	来源
模块一输出结果	JSON 文件	包含分区 ID、面板数量、分区边界坐标	模块一输出
道路数据	.shp 文件	山地可通行道路坐标及宽度	算例库
设备成本参数	.csv 文件	箱变、电缆的购置 / 安装 / 开挖成本	算例库
共沟配置参数	字典	含单沟最大电缆数量、挖沟成本系数	{"max_cable_per_trench": 4, "trench_cost_factor": 1.3}

2. 输出格式（JSON 文件 / 字典）

json

```
{
  "设备配置列表": [
    {
      "逆变器 ID": "inv_001",
      "所属分区 ID": "area_001",
      "安装坐标": [x, y],
      "接入箱变 ID": "box_001"
    },
    {
```

```
      "箱变 ID": "box_001",
      "容量规格": 3200,   // 单位：kVA
      "安装坐标": [x, y],
      "购置成本": 50.0,   // 单位：万元
      "安装成本": 3.0   // 单位：万元
    }
    // 其他设备...
  ],
  "电缆路由规划": [
    {
      "路由 ID": "route_001",
      "起点设备 ID": "inv_001",
      "终点设备 ID": "box_001",
      "路径坐标": [[x1,y1], [x2,y2], ...],
      "电缆规格": "3×50mm² ",
      "电缆长度": 150.3,   // 单位：m
      "是否共沟": true,
      "所属管沟 ID": "trench_001"
    }
    // 其他路由...
  ],
  "成本统计": {
    "设备总购置成本": 280.0,   // 单位：万元
    "电缆总购置成本": 45.0,   // 单位：万元
    "挖沟总成本": 30.0,   // 单位：万元
    "总土建及设备成本": 355.0   // 单位：万元
  }
}
```

（三）模块三：集成优化（建设成本 + 电力损耗）

1. 输入格式

字段名称	类型	说明	来源
模块一输出结果	JSON 文件	分区及面板配置信息	模块一输出
模块二输出结果	JSON 文件	设备配置及电缆路由信息	模块二输出
损耗相关参数	.csv 文件	电缆电阻率、折现率、上网电价等	算例库

2. 输出格式（JSON 文件 / 字典）

json

```
{
  "集成优化方案": {
    "设备优化调整": [
      {"箱变 ID": "box_001", "原容量": 3200, "优化后容量": 1600, "调整原因": "降低长期损耗"}
      // 其他调整...
    ],
    "电缆优化调整": [
      {"路由 ID": "route_001", "原截面半径": 0.02, "优化后截面半径": 0.03, "损耗降低比例": 15.2}
      // 其他调整...
    ]
  },
  "成本与损耗统计": {
    "建设总成本": 340.0,  // 单位：万元（优化后）
    "25 年总电力损耗": 120.5,  // 单位：万 kWh
    "25 年损耗等效成本": 48.2,  // 单位：元（按上网电价计算）
    "全生命周期总成本": 388.2,  // 单位：万元
    "与独立优化方案对比": {
      "建设成本降低比例": 4.3,  // 单位：%
      "电力损耗降低比例": 18.5,  // 单位：%
      "全生命周期成本降低比例": 6.8  // 单位：%
```

```
    }
  }
}
```

三、编码规范（统一 Python 代码风格）

（一）变量命名规范

基本原则：采用蛇形命名法（小写字母 + 下划线），见名知义，避免缩写（必要缩写需统一注释）；

分类命名：

参数变量：param_+具体含义，如 param_cable_resistivity（电缆电阻率）；

决策变量：var_+变量类型+具体含义，如 var_binary_inverter_access（逆变器接入零一变量）；

约束变量：constraint_+约束类型，如 constraint_cable_capacity（电缆容量约束）；

结果变量：result_+统计维度，如 result_total_construction_cost（建设总成本）。

（二）目录结构规范

```
mountain_pv_optimization/     项目根目录
├──── data/     数据文件夹（只读）
│     ├──── raw/     原始数据（地形、设备参数）
│     ├──── processed/     预处理后的数据（标准化网格、算例库）
│     └──── results/     输出结果（各模块 JSON 文件）
├──── model/     模型定义文件夹
│     ├──── model_cutting_partition.py     模块一：切割及分区模型
│     ├──── model_equipment_cable.py     模块二：设备及电缆模型
│     └──── model_integration.py     模块三：集成优化模型
├──── algorithm/     算法实现文件夹
│     ├──── benders_decomposition.py     Benders 分解算法
│     ├──── branch_and_price.py     分支定价算法
│     └──── reinforcement_learning.py     强化学习（DQN）实现
```

├── utils/　工具函数文件夹

│　　├── data_preprocess.py　数据预处理工具

│　　├── visualization.py　结果可视化工具

│　　└── metric_calculation.py　指标计算工具（成本、损耗）

├── tests/　测试文件夹

│　　├── test_model1.py　模块一单元测试

│　　├── test_model2.py　模块二单元测试

│　　└── test_integration.py　模块三集成测试

├── main.py　主程序入口（调用各模块）

└── README.md　项目说明文档（环境配置、运行步骤）

（四）代码提交规范

提交频率：每人定期提交 1 次代码，标注修改内容（如 “修复模块一连通性约束编码错误”）；

分支管理：主分支（main）保持可运行状态，开发分支按 “模块名 + 开发者” 命名（如 “cutting_hexianwen”）；

四、开发环境搭建规范

（一）统一工具版本

工具 / 库	版本要求	安装命令（pip）
Python	3.9 - 3.10	（系统预装 / 官网下载）
Gurobi	10.0	需从官网获取许可证后安装
PyTorch	2.0+	pip install torch==2.0.1 torchvision torchaudio
GeoPandas	0.14+	pip install geopandas==0.14.1
NumPy	1.24+	pip install numpy==1.24.3
Pandas	2.0+	pip install pandas==2.0.2
Matplotlib	3.7+	pip install matplotlib==3.7.1
Plotly	5.15+	pip install plotly==5.15.0
Git	2.30+	（官网下载）

（二）共享资源整理（Notion 目录）

工具资源：

Gurobi 许可证文件（团队共享版）；

算例库下载链接（含原始数据、预处理脚本）；

代码仓库地址（GitHub/Gitee）及访问权限说明。

文档资源：本核心文档（数据字典、接口规范、编码规范）；

申报书建模公式汇总；

每周进度看板（成员任务、完成状态、卡点问题）。

开源代码框架复用 + 算例规划方案（贴合项目需求）

一、核心开源代码推荐

（一）研究内容二优先复用：电缆路由 + 分支定价算法（直接贴合）

1. 开源项目：Luo 等人 2021 年电缆路由问题代码

地址：https://github.com/solardesign/instance

贴合点：

核心算法：分支定价割平面（BPC），完美匹配研究内容二的 “ArcFlow 混合整数规划 + 分支定价 + Kmeans 聚类列管理”；

问题场景：光伏电站电缆路由、汇流箱选址、容量约束，与 “设备选型 + 电缆共沟” 高度契合；

可复用部分：ArcFlow 模型编码、分支定价核心逻辑、Kmeans 列管理框架（无需从零开发）。

使用建议：

直接复用 branch_and_price.py 核心函数，修改 “共沟约束” 模块（原代码无共沟，新增 trench_constraint()函数，添加单沟最大电缆数量限制）；

适配项目数据字典：将原代码参数（如汇流箱容量、电缆成本）替换为我们定义的标准化参数（Q_box、c3 等）；

保留其可视化模块，修改为标注共沟段的电缆路由图。

（二）研究内容一复用：Benders 分解 + DQN 框架

方案 1：Pyomo 官方 Benders 分解通用框架（最易改造）

仓库：https://github.com/Pyomo/pyomo/tree/main/examples/pyomo/opt

核心文件：benders_decomposition.py（通用 Benders 分解实现，支持 Gurobi 求解）

核心算法：原生支持 “主问题 + 子问题” 迭代、割平面生成，完美匹配 Benders 分解逻辑；

工具适配：支持 Gurobi 求解器（与项目技术栈一致），可直接接入地形网格数据（以矩阵形式输入）。

复用步骤

改造主问题：定义光伏分区的决策变量（如 x[i,j,k]表示网格(i,j)是否属于分区 k）、目标函数（最小化切割浪费 + 分区周长）；

添加光伏约束：在主问题中加入 “切割长度整数约束”“分区连通性约束”（用 Pyomo 的逻辑约束编码）；

子问题适配：将子问题定义为 “验证分区的地形可行性（坡度≤阈值）”，生成可行性割平面反馈给主问题；

数据对接：将地形网格数据（GeoPandas 处理后的矩阵）作为输入，替换原示例的通用数据。

方案 2：能源系统优化开源项目（含光伏分区逻辑）

仓库：https://github.com/NREL/REopt_API（美国国家可再生能源实验室 NREL 的能源系统优化工具）

核心模块：reopt/optimization/models/pv_layout.py（光伏布局规划模块）

场景适配：原生支持光伏阵列的分区、选址约束（含地形坡度、可建设区域限制）；

算法支持：内置 “分区优化 + 整数规划” 逻辑，可扩展 Benders 分解加速求解。

复用步骤

提取光伏分区模块：复用 pv_layout.py 中的 “网格地形解析”“分区容量约束” 代码；

集成 Benders 分解：基于 REopt 的光伏参数（面板尺寸、切割规则），在其整数规划模型基础上，添加 Benders 主 / 子问题迭代逻辑；

适配约束：补充 “切割长度整数约束”“分区连通性约束”（参考 REopt 的现有约束框架编码）。

补充 DQN 加速：复用 PyTorch 实现的 DQN 框架：https://github.com/ShangtongZhang/DeepRL（轻量易改）；

修改方向：将 DQN 的状态空间定义为 “地形网格 + 已切割区域”，动作空间为 “切割规格选择”，奖励函数为 “切割浪费率 + 分区周长”。

（三）研究内容三复用：集成优化模块化接口

1. 开源项目：运筹优化集成框架（模块化设计）

地址：https://github.com/matthewjwoodruff/optimizationframework

贴合点：

结构适配：支持多模块接口对接（分区模块→设备模块→集成模块），完美匹配项目的 “模块衔接规范”；

算法支持：内置线性化处理工具，可直接复用用于研究内容三的 “电力损耗分段线性化”。

使用建议：

复用其 integrate.py 中的接口适配逻辑，按我们定义的 JSON 输出格式修改数据传递函数；

新增 “成本 损耗协同优化” 目标函数，整合前两个模块的输出参数。

二、算例规划（先简后繁，适配研究进度）

（一）研究内容一 / 二 / 三：用公开简化算例（保证算法实现 + 集成）

1. 算例来源：

基础算例：直接使用 Luo 开源项目中的 17 个标准算例（https://github.com/solardesign/instance），包含 108 块光伏板、汇流箱容量 6/8 路，适配中小型场景；

简化修改：

地形数据：保留网格结构，将坡度数据简化为 “0（平坦）/1（起伏）” 二元值，降低计算复杂度；

约束简化：研究内容一暂不考虑复杂地形切割，仅保留 “整数切割 + 连通性约束”；研究内容二暂用 2 种箱变规格（1600/3200kVA），共沟数量限制为 4 根。

2. 验证指标：

研究内容一：覆盖面积利用率≥90%、切割浪费率≤5%、分区连通性 100%；

研究内容二：共沟成本比非共沟降低≥10%、电缆总长度误差≤8%；

研究内容三：全生命周期成本比独立优化降低≥8%。

（二）研究内容四：接入甘肃实际算例（深度验证）

1. 数据适配步骤：

数据预处理：用 utils/data_preprocess.py 将甘肃算例的地形 DEM 数据（.tif 格式）转换为标准化网格矩阵（参考项目 data/processed/格式）；

参数映射：将实际算例中的设备参数（如电缆型号、箱变价格）映射到项目数据字典中的 param_参数；

场景扩展：在实际算例中新增 “山地坡度分级”“复杂共沟路径” 场景，验证算法鲁棒性。

2. 验证重点：

与人工设计方案对比：成本降低≥15%、求解时间≤3 小时；

约束满足度：电缆过载率 = 0、共沟数量合规率 100%、电力损耗误差≤5%。

三、快速行动步骤

第 12 天：下载 3 个核心开源项目，按项目目录结构（model//algorithm//utils/）拆分复用代码，删除冗余功能；

第 34 天：修改研究内容二的开源代码，新增共沟约束模块，用公开算例跑通 “设备选型 + 电缆路由” 核心逻辑；

第 56 天：复用 Benders 分解 + DQN 框架，实现研究内容一的 “切割 分区”

协同优化，验证约束满足度；

第 7 天：对接前两个模块输出，用集成框架实现研究内容三的基础版本（暂不考虑电力损耗，先保证接口通畅）；

同步: 将甘肃实际算例进行预处理，生成标准化数据集，为研究内容四预留接口。

一、数据可用性核心依据

1. 格式完全匹配开源项目规范

数据结构: 共 108 个 PVA（ID 0107）+ 1 个逆变器，每行格式为「PVA_ID x 坐标 y 坐标」（单位：毫米，符合 Luo 开源项目的真实光伏算例格式），最后一行明确标注逆变器位置，无格式错误；

数据完整性：无缺失记录、无坐标异常值（数值合理，无负数/超范围数据），无需剔除无效数据，直接满足“数据清洗”的基础要求。

2. 适配项目算例处理规范（仅需 3 步简单预处理）

预处理环节	现状	需执行操作	可行性
单位统一	开源数据为毫米，项目规范为米	将 x/y 坐标 ÷1000 转换为米	简单（1 行代码即可实现）
标准化补充	缺少地形/设备/约束字段	按数据字典默认值补充： • 地形: grid_size=10m，slope_matrix 全 0（平地简化），buildable_matrix 全 true； • 设备: q=320kW，Q_box=1600/3200kVA，c3=200 元/m； • 约束: 标注“整数切割”“分区 1826 块”“共沟≤4 根”	无需额外数据，直接按规范填充
网格化处理	无网格索引	按坐标映射网格: row = y / grid_size，col = x / grid_size（转换后米坐标 ÷10），生成二维矩阵	自动计算，无手动操作成本
约束显性化	无约束标注	添加 constraint_info 字段，明确汇流箱容量（6/8 路）、分区连通性等约束	按开源项目 + 项目规范直接标注

3. 完美适配三大模块需求

模块一（切割及分区）：108 个 PVA 可分为 46 个分区（每个分区 1826 块，符合逆变器负载约束），PVA 坐标呈规则排列（x/y 递增），适配“2×整数列”切割规则，可直接用于切割算法测试；

模块二（设备选型+电缆共沟）：108 个 PVA 对应 46 个逆变器，可搭配 1600kVA（接 5 个逆变器）或 3200kVA（接 10 个逆变器）箱变，逆变器位置固定，可作为分区中心计算电缆路由，适配分支定价算法的电缆共沟优化；

模块三（集成优化）：转换后的坐标可生成距离矩阵，补充损耗参数（r_c=0.015、lambda=0.4 等）后，可直接用于建设成本+电力损耗的协同优化。

二、数据优势与扩展潜力

1. 核心优势

来源权威：来自 Luo 等人 2021 年开源的真实光伏算例，已通过工程验证，无数据矛盾；

规模适配：108 个 PVA 规模适中，既不会因过小导致算法测试不充分，也不会因过大导致计算超时，适合模块开发初期的调试；

场景灵活：默认是平地场景（slope_matrix 全 0），可手动修改部分网格的坡度值（如设置部分区域坡度=20°），快速扩展为山地场景，适配模块三的地形复杂度测试。

2. 扩展方向

模拟山地地形：修改 slope_matrix，设置部分网格坡度>25°（不可建设），生成 buildable_matrix 的局部 false 值，测试模块一的切割分区对不可建设区域的适配能力；

增加约束复杂度：在 constraint_info 中添加“坡度成本系数”（陡坡电缆成本×1.5），测试模块二的电缆路由成本优化；

批量生成算例：基于该数据的坐标规律，修改 x/y 偏移量，批量生成多个算例，用于算法鲁棒性测试。

三、快速使用步骤（按算例处理规范执行）

1. 格式转换与单位统一：

将 r1.txt 的 x、y 坐标（毫米）转换为米（÷1000），生成标准化 PVA 坐标列表；

最后一行逆变器坐标同样转换为米，记录为 inverter_coord。

2. 标准化补充与约束标注：

按数据字典补全 instance_info（instance_id=“public_easy_r1”，type=“public”，difficulty=“easy”，n_nodes=108）；

补全 terrain_data（grid_size=10，slope_matrix 全 0，buildable_matrix 全 true）；

补全 equipment_params（q=320，Q_box=[1600,3200]，c3=200）；

添加 constraint_info：[{"type":"切割约束","value":"2×整数列","priority":"高"},{"type":"汇流箱容量","value":"6/8 路","priority":"高"},{"type":"分区面板数","value":"1826","priority":"高"},{"type":"电缆共沟","value":"≤4 根","priority":"高"}]。

3. 网格化处理与文件生成：

按 grid_size=10，将 PVA 的米坐标映射为网格索引（row=round(y/10)，col=round(x/10)）；

生成 JSON/Excel 双格式文件，按命名规则保存至 `instance_lib/public/easy/public_easy_r1.json`；

调用`load_instance.py`验证字段完整性（确保无缺失必选字段）。

4. 模块调用测试：

模块一：加载该算例，测试切割算法是否生成 2×整数列规格，分区是否满足 1826 块/区；

模块二：基于分区结果，测试箱变选型（1600/3200kVA）与电缆共沟路由的成本优化效果；

模块三：补充 loss_params，测试全生命周期成本（建设成本+损耗）的优化

结果。

四、总结

该数据是“格式合规、适配性强、可扩展”的优质公开算例，无需复杂处理即可接入项目的模块开发流程，既能满足基础功能测试，也能通过简单修改扩展为山地场景，完全符合项目对公开简化算例的需求。

一、核心可复用算例/数据集（优先开源项目+文档配套数据）

（一）Luo 等人 2021 年开源项目（https://github.com/solardesign/instance）——直接贴合研究内容二

这是最核心的资源，项目本身包含两类可直接复用的算例，且格式与你的数据字典、模块接口协议高度兼容：

1. 真实光伏算例（优先选用**[已下载好]**

数据格式：完全匹配你提供的规范，每行含`PVA_ID x 坐标 y 坐标`（单位：毫米），最后一行为逆变器位置（如`inverter 5000 3000`）。

核心参数：汇流箱容量默认 6 路/8 路（与`Q_box`对应），电缆路由基于曼哈顿距离（适配`d_ij`定义），电缆成本计算逻辑明确（PVA 到汇流箱单价 0.0039365 USD/毫米，汇流箱到逆变器成本为 2.77 倍）。

可复用规模：17 个真实场景算例，每个算例含 108 个 PVA，对应山地光伏“分区设计”需求（可按区域拆分/合并算例）。

获取方式：直接从 GitHub 仓库下载`instance`目录下的`.txt`格式文件，无需格式转换即可接入模块。

2. CMST 基准算例（用于算法验证）

数据格式：遵循 ORLibrary 的 Capacitated Minimum Spanning Tree（CMST）标准格式（参考文档 id=1），含节点数、边数、容量约束、成本矩阵等。

适配场景：可用于验证“电缆共沟+容量约束”模块，通过修改边成本参数（模拟山地地形复杂度），转化为山地光伏场景。

规模覆盖：节点数 40160 个，容量 520，涵盖`tc`（根节点居中）、`te`（根

节点角落）、`td`（根节点外部）三类拓扑，可模拟山地逆变器不同选址场景。

（二）文档配套的扩展算例（来自文档 id=2、4、3）

1. pCTP/pCTPC 算例（文档 id=2、4）

来源：基于 ORLibrary 的 pmedian 数据集扩展（节点数 100900），含节点坐标、弧成本、容量约束、覆盖半径等参数。

适配价值：可通过修改“覆盖半径”模拟山地光伏的地形遮挡（如缩小覆盖范围），“容量约束”对应汇流箱/逆变器负载限制，“弧成本”可叠加山地施工难度系数（如陡坡路段成本倍增）。

获取方式：文档 id=2 的 Section 5.1 提供了数据生成方法，可基于 ORLibrary（http://people.brunel.ac.uk/~mastjjb/jeb/orlib/capmstinfo.html）下载原始数据，按文档方法生成含山地特征的算例。

2. HEEC 真实项目算例（文档 id=3）

场景：湖北某 40MW 光伏电站，分为 29 个分区（17 个独立算例），含地形分区、PVA 布局、汇流箱/逆变器位置真实数据。

核心参数：PVA 数量 108/分区，汇流箱容量 6/8 路，电缆成本占总投资 1%2%，与山地光伏项目成本结构一致。

获取方式：文档 id=3 的 GitHub 链接（与你提供的开源项目同源）可下载，含原始坐标、成本参数、手动设计方案（用于算法对比验证）。

二、其他开源仓库补充资源（适配山地场景）

（一）Pyomo 开源示例库（https://github.com/Pyomo/pyomo/tree/main/examples/pyomo/opt）

可获取算例：网络设计类（含 CMST、设施选址、弧流模型），如`capacitated_min_spanning_tree.py`对应的算例，含节点容量、边成本约束。

适配方法：修改算例中的“节点坐标”为山地经纬度投影坐标，“边成本”叠加地形坡度系数（如坡度>25°时成本×1.5），“容量约束”映射为汇流箱负载上限。

用途：用于算法模块的单元测试，验证 ArcFlow 模型编码的正确性。

（二）REopt® API（可再生能源优化模型）

可获取数据：含光伏电站设备选型、电缆路由、容量配置的真实场景数据集（美国 NREL 提供），支持自定义地形复杂度、设备参数。

适配价值：提供“山地光伏+储能”混合场景数据，可提取电缆路由的距离矩阵、成本系数，补充高海拔/复杂地形算例。

获取方式：通过 NREL Developer Network 申请 API 调用权限，或下载示例数据集（https://github.com/NREL/REoptAnalysisScripts）。

三、山地场景适配改造方法（关键步骤）

1. 地形特征注入：对平面算例的边成本进行修正，基于坡度数据（可假设或导入 GIS 地形数据），公式为`山地边成本 = 原始成本 × (1 + 0.05×坡度)`（坡度单位：°）。

2. 分区算例扩展：将开源项目的 17 个算例按“逆变器位置+地形复杂度”分组，合并为节点数 200500 的大规模算例（模拟山地电站多分区联动）。

3. 约束参数对齐：

汇流箱容量：沿用 6/8 路，或按项目需求修改为 12/16 路（开源项目支持自定义`num1`/`num2`参数）。

电缆共沟约束：参考文档 id=2 的`CpCTPC`模型，在算例中添加“同一弧段最多允许 3 条电缆共沟”的约束参数。

四、优质补充数据集（外部来源）

1. ORLibrary（经典优化算例库）

地址：http://people.brunel.ac.uk/~mastjjb/jeb/orlib/capmstinfo.html

资源：Capacitated Minimum Spanning Tree（CMST）算例（节点数 1001000），含容量、成本、节点坐标，可直接用于算法性能测试。

2. IEEE DataPort（光伏专项数据集）

地址：https://ieeedataport.org/search?q=solar+photovoltaic+cable+routing

资源：含山地光伏电站的 PVA 布局、电缆路径、地形坡度数据，部分为中

国西南山区案例（如云南、四川），格式为 CSV/JSON，可直接导入你的模块。

3. NREL 光伏数据集

地址：https://www.nrel.gov/grid/solarresourcedata.html

资源：提供全球光伏资源分布、地形复杂度数据，可结合 GIS 工具提取山地坐标、距离矩阵，生成自定义算例。

五、复用优先级建议

1. 第一优先级：Luo 开源项目的 17 个真实光伏算例（直接接入，无需大规模修改）；

2. 第二优先级：文档配套的 pCTPC 算例（含覆盖半径/容量约束，适配山地遮挡）；

3. 第三优先级：Pyomo/REopt 的扩展算例（用于算法验证和场景扩展）；

4. 补充优先级：IEEE DataPort 的山地专项数据（提升场景真实性）。

建议先基于 Luo 开源项目的算例完成“分支定价算法+电缆路由”模块的联调，再通过 CMST 基准算例验证算法鲁棒性，最后用山地专项数据优化地形适配参数。

对照《算例处理规范与模块接口协议》《数据字典与输入输出格式规范》及项目申报书的要求，从以下维度逐一验证，均满足合规性：

一、核心合规性验证（全达标）

1. 结构完整性（符合算例存储规范）

文件包含所有必选核心字段，无缺失 / 冗余：

基础信息：instance_info（算例标识、类型、难度等）

核心数据：pva_list（面板坐标 + 网格映射）、terrain_data（地形参数）

参数配置：pva_params（面板参数）、equipment_params（设备参数）、loss_params（损耗参数）

辅助数据：dist_matrix（距离矩阵）、constraint_info（约束标注）

完全匹配 “双格式存储” 中 JSON 文件的字段要求。

2. 数据规范性（符合预处理流程）

验证项	具体情况
单位统一	所有长度（x/y/grid_size）均为 “米（m）”，功率为 “千瓦（kW）”

验证项	具体情况
网格映射	网格尺寸 grid_size = 10 m，grid_coord 计算逻辑：row = round(y / 10)、col = round(x / 10)，与坐标映映射规则一致
数据完整性	108 个 PVA 面板（ID 0–107）无重复 / 缺失，逆变器坐标（172.46 m, 481.0 m）解析正确
约束显性化	constraint_info 包含 9 项约束，覆盖三大模块的高 / 中优先级约束（如切割约束、共沟约束、损耗约束），标注清晰

3. 参数一致性（符合数据字典）

所有参数取值与《数据字典》完全匹配，无偏离：

面板参数：D=12.0m（标准长度）、b=3.0m（宽度）、t_l_options（2.0/4.0...12.0m，整数列切割）

设备参数：逆变器 q=320kW、箱变 Q_box_options=[1600,3200]kVA、电缆 c3=200 元/m

损耗参数：lambda=0.4、K_segments=3、I_max=200A（符合非线性损耗拟合要求）

约束参数：分区面板数[18,26]、单沟最大电缆数 4、分区周长[60.0,90.0]m

4. 命名与存储（符合目录 / 命名规则）

文件名：public_easy_r1.json，遵循[类型缩写]_[难度]_[序号]规则（public = 公开算例、easy = 简单难度、r1 = 原文件序号）

目录适配：存储路径 data/processed/PV/public/easy，完全匹配算例库目录结构

编码格式：UTF-8（无中文乱码风险）

二、模块适配性验证（可直接调用）

1. 模块一（光伏面板切割及分区）

满足“整数切割约束”：t_l_options 仅包含 2.0 的整数倍（2.0/4.0...12.0m）

满足“分区约束”：108 个 PVA 可分为 5 个分区（22 块 / 区左右），符合[18,26]块 / 区的逆变器容量约束

网格坐标完整：grid_coord 为整数索引，可直接用于分区连通性判断

2. 模块二（电气设备选型 + 电缆共沟）

设备参数齐全：箱变容量、电缆成本、共沟约束（trench_max_cables=4）均已明确

距离矩阵可用：dist_matrix 包含 108 个 PVA+1 个逆变器的曼哈顿距离，可直接用于电缆路由成本计算

逆变器坐标固定：inverter_coord 为分区中心参考点，适配设备选址逻辑

3. 模块三（集成优化）

损耗参数完整：loss_params 包含电阻、电流分段、折现率等所有必选参数，可直接用于 I^2R 非线性损耗拟合

耦合约束适配：equipment_params 与 loss_params 参数一致（如电缆 r_c=0.015m），满足“建设成本 + 电力损耗”协同优化要求

Luo 等人 2021 年电缆路由开源项目（solardesign/instance）复用指南

一、开源项目核心信息（基于 readme.doc 提取）

（一）算例数据格式（关键适配基础）

1. 真实光伏算例格式（研究内容二直接复用）

输入文件：每行记录 1 个光伏阵列（PVA）的位置信息，含 3 个字段：PVA_ID x 坐标 y 坐标（单位：毫米），最后一行固定为逆变器位置（ID 可自定义，坐标格式一致）。

示例：

plaintext

1 1000 2000

2 1200 2000

…

inverter 5000 3000　　最后一行固定为逆变器

核心约束：默认汇流箱（combiner box）容量为 6 路或 8 路（与项目数据字典中 Q_box 对应），电缆路由基于曼哈顿距离计算（适配项目 d_ij 定义）。

2. CMST 算例格式（可选扩展）

参考 ORLibrary 标准格式（http://people.brunel.ac.uk/~mastjjb/jeb/orlib/capmstinfo.html），支持容量约束下的最小生成树问题，可用于研究内容二的算法验证。

3. 结果文件格式（复用输出框架）

第一行：最优解总成本（电缆成本）；

后续每行：1 个汇流箱子树信息，格式为子树节点数 节点 ID 列表 父节点位置列表；

示例：8 16 10 4 17 11 5 23 22 1 1 2 1 4 5 4 1

（含义：8 个节点的子树，节点 16 父节点为 1（汇流箱），节点 10 父节点为 1，节点 4 父节点为 2，以此类推）。

（二）电缆成本计算逻辑（必须适配项目参数）

以算例 Region 1 为例，原代码成本公式需适配项目数据字典：

距离计算：总距离 = PVA 到汇流箱距离之和 + 2.77 × 汇流箱到逆变器距离之和

2.77：汇流箱 逆变器电缆单位成本系数（项目可调整为 c1/c2，即逆变器电缆成本/汇流箱电缆成本）；

单位成本：PVA 汇流箱电缆成本为 0.0039365 USD/毫米（项目需替换为 c2，注意单位转换：1 米 = 1000 毫米，即 c2 = 项目定义的元/米 ÷ 1000）；

总成本公式：总电缆成本 = 单位成本 × 总距离（需新增共沟成本 c3，原代码无此部分，需额外开发）。

二、研究内容二复用操作指南（设备选型 + 电缆共沟）

（一）可复用核心模块（直接拷贝 + 少量修改）

（二）关键开发：新增共沟约束模块（原代码无，核心修改点）

在 branch_and_price.py 中新增 trench_constraint()函数，实现单沟最大电缆数量限制（项目数据字典中 N_max，默认 4 根）：

调用位置：在 solve_branch_and_price()函数中，调用 generate_arc_flow_model()后，直接添加 model = trench_constraint(model, cable_routes, N_max)。

（三）项目参数适配（替换原代码硬编码参数）

（四）可视化修改：标注共沟段

在 visualization.py 的 plot_cable_routes()函数中新增共沟段标注：

三、快速验证：跑通最小共沟算例

步骤 1：从开源仓库下载 Region 1 算例（region1.txt），修改坐标单位为 “米”（除以 1000）；

步骤 2：在项目根目录新建 test_trench.py，调用复用模块 + 共沟约束：

python

运行

```
from branch_and_price import solve_branch_and_price, trench_constraint
from instance_reader import read_pva_coords
```

1. 读取算例

```
pva_coords, inverter_coord = read_pva_coords("region1.txt")
```

2. 初始化模型（原代码逻辑）

```
model = init_gurobi_model(pva_coords, inverter_coord)
```

3. 新增共沟约束

```
cable_routes = generate_arc_flow_routes(model)      原代码生成路径
model = trench_constraint(model, cable_routes, max_cable_per_trench=4)
```

4. 求解并可视化

```
result = solve_branch_and_price(model)
plot_cable_routes(result["cable_routes"], result["trench_assignments"])
print(f"优化后总成本：{result['total_cost']} 元")
```

步骤 3：运行代码，检查输出是否满足：共沟段标注正确、单沟电缆数≤4、成本计算包含共沟开挖成本 c3。

二、基于 ShangtongZhang/DeepRL 的 DQNBenders 混合框架

利用 DeepRL 的模块化 DQN 实现，构建“DQN 决策切割规格 + Benders 验证可行性”的协同优化流程。

1. 整体流程

加载地形网格→DQN输出切割动作→Benders分解验证/优化分区→计算奖励（浪费率+周长）→更新 DQN → 迭代收敛

2. 核心模块改造（基于 dqn_feature 模板）

（1）状态空间（State）

替换游戏像素为**结构化光伏分区特征**：

- 地形特征：每个网格的光伏潜力、坡度、朝向等（展平为向量）；
- 已切割区域：二值掩码（0/1 表示未切/已切），展平后拼接。

```
config.state_dim = grid_feature_dim + cut_area_dim
config.network_fn = lambda: VanillaNet(config.action_dim,
FCBody(config.state_dim))
```

（2）动作空间（Action）

定义离散切割规格（如 8 种预设方案）：

```
action_space = Discrete(8)
cut_spec_map = {
    0: (3, 3, 'horizontal'),
    1: (3, 3, 'vertical'),
    2: (5, 5, 'horizontal'),
     ...
}
```

（3）奖励函数（Reward）

综合惩罚浪费与鼓励紧凑分区：

```
reward = (0.7  waste_rate + 0.3  normalized_perimeter)
```

- **waste_rate**：未利用光伏面积 / 总切割面积；
- **perimeter**：分区边界长度（归一化），越小越紧凑。

（4）嵌入 Benders 分解验证

在 `execute_cut` 中调用 Benders 求解器：

```
def execute_cut(self, current_cut, spec):
    candidate = self.generate_candidate(current_cut, spec)
    feasible, optimized = self.benders_decomposition(candidate)
    return optimized if feasible else self.fallback_cut(current_cut)
```

- **主问题**：MIP 模型求解分区方案；

- **子问题**：验证连通性、地形约束等；
- **割平面**：不可行时生成 Benders cut 并迭代。

3. 工程化调优建议

组件	推荐配置
训练加速	启用 async_actor=True, async_replay=True
DQN 超参	batch_size=32, target_update_freq=500, ε greedy 从 1.0 衰减至 0.01（1e4 步）
Benders 设置	主/子问题超时限制（如 30s），割平面缓存避免重复添加
评估指标	光伏利用率、连通性合规率、平均求解时间（vs 纯 Benders）