mbe-tools (v0.1.3)

`mbe-tools` 覆盖 Many-Body Expansion (MBE) 工作流：

- 簇与片段：读取 `.xyz`，水启发式或连接性+标签拆分，随机/空间抽样，支持保留离子。
- 作业准备：生成子集几何，渲染 Q-Chem/ORCA 输入，产出 PBS/Slurm 脚本（可分批提交，内置 run-control）。
- 解析：读取 ORCA/Q-Chem 输出，自动识别程序，基于路径或伴随输入推断 method/basis/grid，写出 JSONL。
- 分析：包含-排除 MBE(k)，汇总，CSV/Excel 导出与简单绘图。

当前状态：**v0.1.3 (MVP)**，站点相关的 ghost 原子等语法可按需在 backend 中调整。许可证：**MIT**。

安装（开发模式）

```
cd mbe-tools
python -m pip install -e .[analysis,cli]
```

全局设置优先级（P0）

优先级（低→高）：1) 环境变量 → 2) `~/.config/mbe-tools/config.toml` → 3) `./mbe.toml` → 4) `load_settings(path=...)`。

支持键：`qchem_command`, `orca_command`, `qchem_module`, `orca_module`, `scratch_dir`, `scheduler_queue`, `scheduler_partition`, `scheduler_account`。

环境变量：`MBE_QCHEM_CMD`, `MBE_ORCA_CMD`, `MBE_QCHEM_MODULE`, `MBE_ORCA_MODULE`, `MBE_SCRATCH`, `MBE_SCHED_QUEUE`, `MBE_SCHED_PARTITION`, `MBE_SCHED_ACCOUNT`。

最小 `mbe.toml` 示例：

```
qchem_command = "/opt/qchem/bin/qchem"
orca_command  = "/opt/orca/bin/orca"
qchem_module  = "qchem/5.2.2"
orca_module   = "orca/5.0.3"
scratch_dir   = "/scratch/${USER}"
scheduler_queue = "normal"
scheduler_partition = "work"
scheduler_account = "proj123"
```

快速上手（Python API）

1. 片段化 XYZ

```
from mbe_tools.cluster import read_xyz,
fragment_by_water_heuristic, fragment_by_connectivity

xyz = read_xyz("Water20.xyz")
frags = fragment_by_water_heuristic(xyz, oh_cutoff=1.25)
frags_conn = fragment_by_connectivity(xyz, scale=1.2)
```

2. 抽样并写回 XYZ

```
from mbe_tools.cluster import sample_fragments, write_xyz

picked = sample_fragments(frags, n=10, seed=42)
write_xyz("Water10_sample.xyz", picked)
```

3. 生成子集几何

```
from mbe_tools.mbe import MBEParams, generate_subsets_xyz

params = MBEParams(max_order=3, cp_correction=True,
backend="qchem")
subset_jobs = list(generate_subsets_xyz(frags, params))
```

4. 构建输入

```
mbe build-input water.geom --backend qchem --method
wb97m-v --basis def2-ma-qzvpp --out water_qchem.inp
mbe build-input water.geom --backend orca  --method
wb97m-v --basis def2-ma-qzvpp --out water_orca.inp
```

5. 生成 PBS/Slurm 模板（含 run-control）

```
mbe template --scheduler pbs   --backend qchem --job-name
mbe-qchem --chunk-size 20 --out qchem.pbs
mbe template --scheduler slurm --backend orca  --job-name
mbe-orca  --partition work --chunk-size 10 --out
orca.sbatch
```

6. 解析输出为 JSONL

```
mbe parse ./Output --program auto --glob "*.out" --out
parsed.jsonl
```

7. 分析 JSONL

```
mbe analyze parsed.jsonl --to-csv results.csv --to-xlsx
results.xlsx --plot mbe.png
```

CLI 速查

- `mbe fragment <xyz>`：水启发式拆分+抽样 → XYZ。参数：`--out-xyz [sample.xyz]`，`--n`，`--seed`，`--require-ion`，`--mode [random|spatial]`，空间模式额外 `--prefer-special`，`--k-neighbors`，`--start-index`，`--oh-cutoff`。
- `mbe gen <xyz>`：生成子集几何。参数：`--out-dir [mbe_geoms]`，`--max-order [2]`，`--order/--orders`，`--cp/--no-cp`，`--scheme`，`--backend [qchem|orca]`，`--oh-cutoff`。
- `mbe build-input <geom>`：渲染 Q-Chem/ORCA 输入。参数：后端、必填 `--method`/`--basis`、电荷/多重度，Q-Chem (`--thresh`/`--tole`/`--scf-convergence`/`--rem-extra`)，ORCA (`--grid`/`--scf-convergence`/`--keyword-line-extra`)，`--out`；批量：让 `geom` 指向目录并加 `--glob "*.geom" --out-dir 输出目录` 可一次生成多份。
- `mbe template`：PBS/Slurm 脚本（含 run-control）。通用：`--scheduler`，`--backend`，`--job-name`，`--walltime`，`--mem-gb`，`--chunk-size`，`--module`，`--command`，`--out`；PBS+qchem 另有 `--ncpus`，`--queue`，`--project`；Slurm+orca 另有 `--ncpus`(cpus-per-task)，`--ntasks`，`--partition`，`--project`(account)，`--qos`；`--wrapper` 会生成可直接 `bash job.sh` 的提交脚本，内部写入隐藏的 `._*.pbs/.sbatch` 并调用 qsub/sbatch。
- `mbe parse <root>`：解析输出 → JSONL。参数：`--program [auto|qchem|orca]`，`--glob-pattern`，`--out`，`--infer-metadata`。
- `mbe analyze <parsed.jsonl>`：汇总/导出。参数：`--to-csv`，`--to-xlsx`，`--plot`，`--scheme [simple|strict]`，`--max-order`。

使用 `mbe <command> --help` 查看完整参数。

定义汇总（CLI & API）

| 范围 | 条目 | 功能 | 关键参数/参数 | 备注 | 实现 | | CLI | `mbe fragment <xyz>` | 水启发拆分并抽样 → XYZ | `--n`, `--seed`, `--mode random | spatial`, `--require-ion`, `--prefer-special`, `--k-neighbors`, `--start-index`, `--oh-cutoff` | 空间模式可强制特殊片段；写出抽样 XYZ | src/mbe_tools/cli.py | | CLI | `mbe gen <xyz>` | 生成指定阶的子集几何 | `--max-order` 或可重复的 `--order/--orders`, `--cp/--no-cp`, `--scheme`, `--backend qchem | orca`, `--oh-cutoff`, `--out-dir` | 阶数可给列表；CP 控制 ghost | src/mbe_tools/cli.py | | CLI | `mbe build-input <geom>` | 从 .geom 渲染 Q-Chem/ORCA 输入 | 必填 `--method` / `--basis`; Q-Chem: `--thresh` / `--tole` / `--scf-convergence` / `--rem-extra`; ORCA: `--grid` / `--scf-convergence` / `--keyword-line-extra`; `--out`; 批量: `--glob`, `--out-dir` | `--glob` 时 `geom` 必须是目录；输出按 stem 命名 | src/mbe_tools/cli.py | | CLI | `mbe template` | 生成 PBS/Slurm 脚本（含 run-control） | 公共: `--scheduler pbs | slurm`, `--backend qchem | orca`, `--job-name`, `--walltime`, `--mem-gb`, `--chunk-size`, `--module`, `--command`, `--out`; PBS: `--ncpus`, `--queue`, `--project`; Slurm: `--ncpus` (cpus-per-task), `--ntasks`, `--partition`, `--project` (account), `--qos`; `--wrapper` | `--wrapper` 产出可直接 `bash job.sh` 的提交器，内部写隐藏 `._*.pbs/.sbatch` 再 qsub/sbatch | src/mbe_tools/cli.py → src/mbe_tools/hpc_templates.py | | CLI | `mbe parse <root>` | 解析 Q-Chem/ORCA 输出为 JSONL | `--program auto | qchem | orca`, `--glob-pattern`, `--out`, `--infer-metadata` | 从文件名与伴随输入推断 method/basis/grid | src/mbe_tools/cli.py → src/mbe_tools/parsers/io.py | | CLI | `mbe analyze <parsed.jsonl>` | 汇总/导出/绘图 | `--to-csv`, `--to-xlsx`, `--plot`, `--scheme simple | strict`, `--max-order` | `strict` 用包含-排除; `simple` 计算相对单体均值的 ΔE | src/mbe_tools/cli.py → src/mbe_tools/analysis.py | | API | 簇与片段 | `read_xyz`, `write_xyz`, `fragment_by_water_heuristic`, `fragment_by_connectivity`, `sample_fragments`, `spatial_sample_fragments` | 参见函数参数：切距、缩放、seed 等 | 支持保留离子和特殊片段优先 | src/mbe_tools/cluster.py | | API | MBE 生成 | `MBEParams`, `generate_subsets_xyz` | 参数: `max_order`, `orders`, `cp_correction`, `backend`, `scheme` | 产出 `(job_id, subset_indices, geom_text)` | src/mbe_tools/mbe.py | | API | 输入构建 | `render_qchem_input`, `render_orca_input`, `build_input_from_geom` | 必填方法/基组；可选 thresh/tole/scf/grid/附加 rem | CLI `build-input` 复用 | src/mbe_tools/input_builder.py | | API | 模板 | `render_pbs_qchem`, `render_slurm_orca` | 资源、分批、run-control 包装 | `wrapper` 参数同 CLI | src/mbe_tools/hpc_templates.py | | API | 解析 | `detect_program`, `parse_files`, `infer_metadata_from_path`, `glob_paths` | 程序自动识别; 从文件名/输入推断元数据 | 伴随输入辅助 method/basis/grid |

src/mbe_tools/parsers/io.py | | API | 分析 | `read_jsonl`, `to_dataframe`, `summarize_by_order`, `compute_delta_energy`, `strict_mbe_orders`, `assemble_mbe_energy`, `order_totals_as_rows` | 生成 MBE 表与图的工具函数 | `strict_mbe_orders` 生成包含-排除行 | src/mbe_tools/analysis.py |

CLI 详表与示例

命令	参数	说明	示例
`mbe fragment <xyz>`	`--out-xyz PATH`	输出抽样 XYZ 路径	`mbe fragment water3.xyz --out-xyz demo/sample.xyz`
	`--n INT`	抽样片段数	`--n 2`
	`--seed INT`	随机种子	`--seed 42`
	`--mode random | spatial`	抽样模式	`--mode spatial`
	`--require-ion`	若存在离子，强制包含	`--require-ion`
	`--prefer-special`	空间模式优先选特殊片段	`--prefer-special`
	`--k-neighbors INT`	空间模式候选邻居数	`--k-neighbors 4`
	`--start-index INT`	空间模式起始片段索引	`--start-index 0`
	`--oh-cutoff FLOAT`	水启发 O-H 切距	`--oh-cutoff 1.25`
`mbe gen <xyz>`	`--out-dir DIR`	输出目录	`--out-dir geoms`
	`--max-order INT`	生成至该最大阶	`--max-order 3`
	`--order/--orders INT (repeat)`	指定阶列表	`--order 1 --order 3`
	`--cp/--no-cp`	是否加 CP ghost	`--cp`
	`--scheme STR`	MBE 方案标签	`--scheme mbe`

命令	参数	说明	示例
	`--backend qchem | orca`	后端格式	`--backend qchem`
	`--oh-cutoff FLOAT`	水启发切距	`--oh-cutoff 1.25`
`mbe build-input <geom>`	`--backend qchem | orca`	选择后端	`--backend qchem`
	`--method STR`	方法（必填）	`--method wb97m-v`
	`--basis STR`	基组（必填）	`--basis def2-svpd`
	`--charge INT`	总电荷	`--charge 0`
	`--multiplicity INT`	自旋多重度	`--multiplicity 1`
	`--thresh FLOAT`	Q-Chem THRESH	`--thresh 14`
	`--tole FLOAT`	Q-Chem TolE	`--tole 8`
	`--scf-convergence STR`	SCF 收敛键	`--scf-convergence 8` (Q-Chem) / `--scf-convergence TightSCF` (ORCA)
	`--grid STR`	ORCA 网格	`--grid Grid5`
	`--rem-extra STR`	Q-Chem 额外 rem 行（\n 分隔）	`--rem-extra "sym_ignore 1"`
	`--keyword-line-extra STR`	ORCA 头部附加关键字	`--keyword-line-extra "SlowConv"`
	`--out PATH`	输出文件	`--out job.inp`

命令	参数	说明	示例
	`--glob PATTERN`	批量匹配目录下 geom	`--glob "*.geom"`
	`--out-dir DIR`	批量输出目录	`--out-dir inputs_batch`
`mbe template`	`--scheduler pbs | slurm`	调度器类型	`--scheduler pbs`
	`--backend qchem | orca`	后端	`--backend qchem`
	`--job-name STR`	作业名	`--job-name mbe-qchem`
	`--walltime HH:MM:SS`	壁钟时间	`--walltime 24:00:00`
	`--mem-gb FLOAT`	内存 GB	`--mem-gb 64`
	`--chunk-size INT`	每子作业输入数	`--chunk-size 10`
	`--module STR`	module load 名	`--module qchem/5.2.2`
	`--command STR`	可执行覆盖	`--command /opt/qchem/bin/qchem`
	`--out PATH`	输出脚本	`--out job.pbs`
	`--ncpus INT`	PBS ncpus / Slurm cpus-per-task	`--ncpus 32`
	`--ntasks INT`	Slurm ntasks	`--ntasks 1`
	`--queue STR`	PBS 队列	`--queue normal`
	`--project STR`	PBS project / Slurm account	`--project proj123`

命令	参数	说明	示例
	`--partition STR`	Slurm 分区	`--partition work`
	`--qos STR`	Slurm QoS	`--qos normal`
	`--wrapper`	生成 bash 提交器	`--wrapper`
`mbe parse <root>`	`--program auto | qchem | orca`	程序选择或自动	`--program auto`
	`--glob-pattern PAT`	匹配输出	`--glob-pattern "*.out"`
	`--out PATH`	JSONL 输出	`--out parsed.jsonl`
	`--infer-metadata/--no-infer-metadata`	推断元数据开关	`--infer-metadata`
`mbe analyze <jsonl>`	`--to-csv PATH`	导出 CSV	`--to-csv results.csv`
	`--to-xlsx PATH`	导出 Excel	`--to-xlsx results.xlsx`
	`--plot PATH`	生成图	`--plot mbe.png`
	`--scheme simple | strict`	汇总方案	`--scheme simple`
	`--max-order INT`	严格模式最大阶	`--max-order 2`

Run-control（模板）

- 查找顺序：`<input>.mbe.control.toml` 优先，其次 `mbe.control.toml`，否则视为未启用。

- 尝试日志：先写 `job._try.out`，失败重命名为 `job.attemptN.out`，成功重命名为 `job.out`；`confirm.log_path` 可改临时日志路径。
- 确认：`confirm.regex_any` 必须命中且 `confirm.regex_none` 不命中，并且退出码为 0 才视为成功。
- 重试：`retry.enabled`，`max_attempts`，`sleep_seconds`，`cleanup_globs`，`write_failed_last`（将最后一次复制到 `failed_last_path`）。
- 删除保护：`delete.enabled` 且 `allow_delete_outputs=true` 才会删输出；输入仅在命中 `delete_inputs_globs` 时删除。
- 状态：`.mbe_state.json` 记录结果/次数/匹配/日志；`skip_if_done` 为真时已完成则跳过。

子集命名

- 推荐：`{backend}_k{order}_f{i1}-{i2}-{i3}_{cp|nocp}_{hash}`，**0-based** 片段索引（可零填充），如 `qchem_k2_f000-003_cp_deadbeef.out`。
- 兼容旧式：`{backend}_k{order}_{i1}.{i2}..._{hash}` 视为名字里的 1-based，解析时转 0-based。JSON 输出中的 `subset_indices` 始终为 0-based。

JSONL 模式（解析输出）


```
[

  {
    "job_id": "qchem_k2_f000-003_cp_deadbeef",
    "program": "qchem",
    "program_detected": "qchem",
    "status": "ok",
    "error_reason": null,
    "path": ".../job.out",
    "energy_hartree": -458.7018184,
    "cpu_seconds": 1234.5,
    "wall_seconds": 1234.5,
    "method": "wB97M-V",
    "basis": "def2-ma-QZVPP",
    "grid": "SG-2",
    "subset_size": 2,
    "subset_indices": [0, 2],
    "cp_correction": true,
    "extra": {}
  }

]
```

API 速览

- 簇与片段（src/mbe_tools/cluster.py）：`read_xyz`，`write_xyz`，`fragment_by_water_heuristic`，`fragment_by_connectivity`，`sample_fragments`，`spatial_sample_fragments`。
- MBE 生成（src/mbe_tools/mbe.py）：`MBEParams`，`generate_subsets_xyz`，`qchem_molecule_block`，`orca_xyz_block`。
- 输入构建（src/mbe_tools/input_builder.py）：`render_qchem_input`，`render_orca_input`，`build_input_from_geom`。
- HPC 模板（src/mbe_tools/hpc_templates.py）：`render_pbs_qchem`，`render_slurm_orca`（均包含 run-control 包装）。
- 解析（src/mbe_tools/parsers/io.py）：`detect_program`，`parse_files`，`infer_metadata_from_path`，`glob_paths`。
- 分析（src/mbe_tools/analysis.py）：`read_jsonl`，`summarize_by_order`，`compute_delta_energy`，`strict_mbe_orders`。

Notebook

`notebooks/sample_walkthrough.ipynb` 展示端到端示例：构建输入、生成模板、用合成数据组装 MBE(k)。

许可证

MIT