Project 9 实验与工程进展报告（代码框架版）

状态概览（截至当前代码）

本报告总结目前仓库中已经实现和跑通的内容，并对照《Project 9 执行计划》（v3.0 Final）说明哪些目标已经满足，哪些仍然是后续工作的重点。最后给出一份带有操作步骤的 checklist，方便后续按条逐项完成。

---

1. 项目目标回顾（简要）

Project 9 的总体目标可以概括为三点：

1. 在多智能体设定下，严格复现并"解剖" MA-LfL，明确其在假设成立与失配时的边界。
2. 设计并实现一个基于策略梯度学习者的逆向算法 I-LOLA，作为 LOGEL 的多智能体扩展版本。
3. 在三个等级环境上进行系统对比：
    - T1：GridWorld（两智能体，离散动作，MA-SPI 匹配设定）。
    - T2：MPE simple_spread（多智能体，离散动作，PPO 学习者）。
    - T3：MultiWalker（多智能体，连续动作，PPO 学习者，MA-LFL 被视为计算上不可行）。

核心评估指标按执行计划分为两个层次：

- 指标 1：策略预测误差，进一步拆分为
  1a）"固有基线误差"，1b）"使用恢复奖励后的误差"。
- 指标 2：诱导策略性能，使用恢复的奖励从头训练新策略，并与随机策略和专家策略对比。

---

2. 已完成的工程模块与实验闭环

这一部分总结目前代码已经构成的闭环，按 T1/T2/T3 分别说明。

2.1 T1 GridWorld：MA-SPI + MA-LFL + I-LOGEL / I-LOLA

当前状态：

- `data_gen.ma_spi.gridworld_sampler` 可以在 3×3 GridWorld 上生成多阶段 MA-SPI 轨迹，并保存到 `outputs/data/t1/...pkl`。
- `runners.run_ma_lfl` 在 T1 上运行完 MA-LFL，输出
    - 1a 和 1b 的 KL，数量级约为 9×10^{-6}；
    - 特征维度信息和奖励权重诊断指标（PCC 为负、RMSE 在 0.26 左右）。
- `scripts/run_t1_ilogel_sanity.py` 已在合成数据和真实 MA-SPI 数据上验证了 I-LOGEL：
    - 合成数据上，恢复的奖励方向与真值余弦相似度接近 1；
    - 实际 T1 数据上可以得到合理的权重范数和损失曲线。
- `scripts/run_t1_ilola_eval.py`（或等价逻辑）已在 T1 上测试 I-LOLA 的一小步版本，并输出 1a/1b 指标。

- `evaluation.metrics` 和 `evaluation.plots` 支持：
  - T1 上的 KL 直方图；
  - T1 上“真实奖励 vs 恢复奖励”的热力图。
- `evaluation.report` 会自动读取 T1 的指标和图表，生成 `outputs/reports/report_t1_seed0.md`。

这意味着：对于 T1，**从数据生成 → 奖励恢复 → 指标计算 → 图表与报告** 的完整代码路径已经跑通。

2.2 T2 MPE simple_spread：PPO + I-LOLA + induced 训练

当前状态：

- `runners.gen_data` 在 `configs/t2_mpe_simple_spread.yaml` 下，可以通过自实现的 PPO 训练 MPE simple_spread，并把多个更新阶段的参数和轨迹打包到 `outputs/data/t2/...pkl`。
- `models.feature_maps` 中针对 T2 的特征映射已经修复，维度为 23，数值范围合理。
- `scripts/run_t2_ilogel_eval.py` 实现了：
  - 读取 T2 数据；
  - 使用 I-LOLA 的阶段 B（CMA-ES）优化，得到一组奖励权重 `w_hat`；
  - 计算 T2 对应的 1a 和 1b：
    - 1a 原始值约 1.0×10^{-6}；
    - 1b 原始值约 1.1×10^{-6}（最新版本）；
  - 输出额外诊断信息，例如 $\|w_{\text{true}}\|$、$\|w_{\text{hat}}\|$、$\|\theta_{\text{hat,true}} - \theta_{\text{hat,pred}}\|$。
- `evaluation.metrics` 中 T2 的 KL 计算已经集成到 `run_t2_ilogel_eval.py`，并把结果写入 `outputs/metrics/t2_ilola_seed0.json`。
- `evaluation.induced_train` 中已经实现了 T2 的最小诱导训练版本：
  - 使用 `w_hat` 构造奖励网络；
  - 根据现有配置训练一个新策略；
  - 随机策略回报 `R_random` 会被估计；
  - 诱导策略回报曲线 `R_induced_curve`（训练步数 → 平均回报）已经写入 `outputs/induced/t2_induced_seed0.json`。
- `evaluation.plots` 可以绘制：
  - T2 上 I-LOLA 1a/1b KL 比较的柱状图；
  - T3 的 induced 曲线也复用同一接口。
- `evaluation.report` 已经支持生成 `report_t2_seed0.md`，自动插入 KL 图和关键信息。

这一部分说明：在 T2 上，**I-LOLA 的阶段 B + 指标 1 + 指标 2（诱导训练）** 已有一个可运行闭环。
目前缺少的是：**MA-LFL 在 T2 上的完整基线实现和 I-LOGEL 阶段 A 与阶段 B 的系统对比。**

2.3 T3 MultiWalker：PPO + I-LOLA + induced 训练（连续动作）

当前状态：

- `runners.gen_data` 在 `configs/t3_multiwalker.yaml` 下，可以通过自实现 PPO 训练 MultiWalker，并保存多阶段参数到 `outputs/data/t3/...pkl`。
- `scripts/run_t3_ilola_eval.py` 已经实现：
  - 从 T3 PPO 轨迹中提取数据；
  - 使用 I-LOLA 阶段 B（CMA-ES）优化奖励权重；
  - 使用采样的方式估计连续动作上的 KL（指标 1b 的 Monte Carlo 版本）：
    - 最新版本输出中，1b 约为 5×10^{-4}，同时打印了某个状态下的样本 KL（约 0.0035）；
  - 把结果写入 `outputs/metrics/t3_ilola_seed0.json`。
- 在 T3 上不会尝试 MA-LFL，这与执行计划中"视为计算不可行，不作为实现目标"的设定一致。
- `evaluation.induced_train` 已经支持 T3：
  - 使用 T3 的 `w_hat` 训练一个新策略；
  - 估计随机策略回报 `R_random` 和诱导策略回报序列 `R_induced_curve`；
  - 结果写入 `outputs/induced/t3_induced_seed0.json`。
- `evaluation.plots` 已可以绘制 T3 的 induced 曲线，例如当前的曲线大致在随机基线附近小幅波动。
- `evaluation.report` 已经生成 `report_t3_seed0.md`，包含 KL 和 induced 曲线的信息。

综合来看，在 T3 上，**"PPO 轨道 + I-LOLA + 连续动作 KL（Monte Carlo）+ 诱导训练"** 这条路径已经跑通，足以支撑"可行性示范"和"MA-LFL 在此设定下计算不可行"的结论。

---

3. 当前结果的初步解读（对照执行计划）

这一部分不是最终论文结论，只是对目前单个 seed 的结果做一个"方向性"对照。

3.1 T1：匹配设定下 MA-LFL 的表现

在 T1 上：

- 1a 和 1b 的 KL 都在 10^{-5} 量级且非常接近，这符合执行计划中"匹配设定下 MA-LFL 能恢复奖励"的预期。
- 奖励热力图表明：真实奖励只在一个格子非零，而恢复奖励呈现一条从某一侧向目标格子逐渐递增的形状。这体现了**奖励整形等价类**的现象：策略行为几乎一致，但奖励权重不一定点对点重合。
- 权重 PCC 为负，RMSE 不小，但对比 1a/1b 接近这一事实，可以在报告中解释为"形状不同但策略等价"。

总体上，T1 已经可以支撑执行计划里的第一步结论：**在假设完全匹配的小环境下，MA-LFL 和 I-LOLA 都可以达到很低的策略预测误差。**

3.2 T2：失配设定下 I-LOLA 的表现

在 T2 上：

- I-LOLA 的 1a/1b 都在 10^{-6} 量级，且差距不大。这说明在当前设置下，基于策略梯度的 I-LOLA 可以在 PPO 数据上达到与"理想基线"接近的预测误差。
- 诱导训练的结果显示，使用恢复奖励重新训练得到的策略，其回报大致略优于随机策略，曲线有一定波动，说明训练仍然受超参数和样本数影响。
- 由于 MA-LFL 基线在 T2 上尚未系统实现，目前还不能展示"MA-LFL 失配 → 1b 明显劣于 I-LOLA"的对比图，这是后续实验的重要部分。

从执行计划视角看，T2 已经完成了"在失配设定下跑通 I-LOLA 的代码和指标"，但**尚未完成"与 MA-LFL 的系统对比"这一关键实验任务。**

3.3 T3：连续控制下 I-LOLA 的可行性

在 T3 上：

- I-LOLA 已经可以在 MultiWalker 上稳定运行，并输出有限的 KL 和诱导训练曲线，这表明在计算上是可行的。
- 目前 1b 约为 5×10^{-4}。由于缺少 T3 上的"理想基线"（1a），这一数值主要作为绝对参考，而非相对差距。
- 诱导训练的曲线与随机基线非常接近，略有起伏，说明目前的特征设计和训练预算下，恢复的奖励对最终行为的影响还比较有限。

这一部分基本实现了执行计划中的要求：**展示 I-LOLA 在连续控制环境下的工程可行性，同时保留 MA-LFL 在该设定下"计算不可行"的负面结果。**

---

4. 与执行计划的差距与后续方向概览

对照《Project 9 执行计划》和《工程实施计划》，当前代码框架已经满足：

1. T1：MA-SPI + MA-LFL + I-LOGEL + I-LOLA 的完整闭环；
2. T2/T3：基于 PPO 的数据生成和 I-LOLA 的 Stage B 优化；
3. 三个环境的指标 1（1a/1b）、部分指标 2（诱导训练）、以及自动图表和 Markdown 报告生成。

仍然有几个重要方向尚未完成或只做了最小版本：

1. **T2 上 MA-LFL 与 I-LOLA 的系统对比**：需要在同一数据集上跑 MA-LFL，并将 1a/1b 与 I-LOLA 的结果并列展示。
2. **Stage A（I-LOGEL）与 Stage B（I-LOLA）的对比**：特别是在 T2 上，需要验证"联合优化（Stage B）优于独立近似（Stage A）"这一关键主张。
3. **多 seed 实验与统计结果**：当前基本都是 seed 0，需要多 seed 重复和均值 ± 置信区间。
4. **T2.5 扩展实验（可选）**：执行计划中提出的简单扩展版本，目前尚未在代码中体现。
5. **文档与论文写作**：需要把上述结果整理成正式论文式结构，并补充理论分析与图表说明。

下面给出一份 checklist，分条列出这些后续工作，并给出建议的具体操作步骤。

---

5. 后续工作 checklist 与操作步骤（重写版：可验收、可对照）

本节把"后半程必须完成的关键工作"改写成**可一步一步照做**的 checklist。每一项都包含：

- **目的**：为什么要做
- **实施步骤**：你应该改哪里、跑什么命令
- **验收输出**：必须生成哪些文件（json / png / md）
- **完成判定（预期现象）**：出现什么结果说明这一步顺利完成；出现什么现象说明还没完成/实现有问题

> 说明
> 1）本 checklist 默认你已经完成 Phase 0–3 的"主链路跑通"，并且 T1/T2/T3 的 I-LOLA（Stage B）、metrics、plots、reports 都能生成。
> 2）本 checklist 的核心是把"单次跑通"提升为"可写成论文结论的最终结果"。
> 3）优先级顺序：5.1（T2 MA-LfL 错配基线）最重要，其次 5.2（StageA vs StageB），再到 5.3（多 seed 统计），最后是文档交付。

5.1 T2：实现 MA-LfL 的"错配基线"并与 I-LOLA 对比（最关键）

目的
补齐执行计划的核心论证：

- 在 T1（匹配 MA-SPI）里 MA-LfL 能工作；
- 在 T2（PPO 学习者）里 MA-LfL 属于**模型错配**，应该表现更差；
- I-LOLA 是匹配模型，在 T2 上应更贴近自己的基线误差（1a）。

> 注意：这里的 MA-LfL-on-PPO 不是追求"MA-LfL 最强实现"，而是追求"按 MA-LfL 的 MA-SPI 规则去做 lookahead"，从而得到一个自洽的错配对照。

5.1.1 实施步骤

Step A：新增一个 runner（或扩展现有 runner）用于 T2 的 MA-LfL 错配评估

1. 新增脚本：`scripts/run_t2_ma_lfl_mismatch.py`（推荐独立脚本，避免影响 T1）。
   - 输入：T2 的 PPO 数据 pkl（例如 `outputs/data/t2/...pkl`）
   - 输出：`outputs/metrics/t2_ma_lfl_seed{seed}.json`
   - 同时输出：一份权重 `outputs/weights/t2_ma_lfl_seed{seed}.npy`（或 json）
2. 在脚本内实现/调用以下逻辑（最小可行版本）：
   - 从 PPO 数据里取一批 state（建议用你现有 metrics 的采样 batch，保持一致）
   - 构造 MA-LfL 的 lookahead：
     - 使用 MA-LfL 的 "soft 改进"形式（按执行计划的公式精神）从 π_t 推出 $\hat{\pi}_{t+1}$
     - 允许 Q 的估计使用 Monte Carlo / 近似（但必须是"同一个 lookahead 流程"用于 1a 和 1b）
   - 计算：

- `err_1a`：用真实奖励 W^*（或你定义的"真奖励"基准）跑 lookahead 得到 KL
- `err_1b`：用恢复奖励 $\hat{W}$ 跑 lookahead 得到 KL

> 如果你当前没有 T2 的 "W_true"，也可以把 `err_1a` 定义成：
> 使用"PPO 自带的 reward（环境 reward）"映射到你的 feature space 得到一个固定 W_{proxy}。
> 关键是：它是固定的、可复现的，并且作为"同一错配模型的天花板"存在。

Step B：把 T2 的对比图补齐

3. 在 `evaluation/plots.py` 增加一个函数，例如 `plot_t2_compare_lfl_ilola(...)`：
   - 读 `outputs/metrics/t2_ma_lfl_seed0.json`
   - 读 `outputs/metrics/t2_ilola_seed0.json`
   - 画一张柱状图（四根柱子）：
     - `MA-LfL 1a`, `MA-LfL 1b`, `I-LOLA 1a`, `I-LOLA 1b`
   - 保存为：`outputs/plots/t2_kl_compare_mismatch_seed0.png`
4. 在 `evaluation/report.py` 的 T2 报告生成里：
   - 插入这张新图
   - 增加一段简短说明：这是错配实验（MA-SPI 模型拟合 PPO 学习者）

5.1.2 运行命令（建议顺序）

```
# 1) 生成/确认 T2 数据存在（如果已有可跳过）
python -m runners.gen_data --config configs/t2_mpe_simple_spread.yaml --seed 0

# 2) 先跑 I-LOLA（你已有）
python scripts/run_t2_ilogel_eval.py

# 3) 再跑 MA-LfL 错配基线（新增）
python scripts/run_t2_ma_lfl_mismatch.py --seed 0

# 4) 生成 plots + reports
python scripts/run_phase3_plots_reports.py
```

5.1.3 验收输出（必须生成）

- `outputs/metrics/t2_ma_lfl_seed0.json`
- `outputs/plots/t2_kl_compare_mismatch_seed0.png`
- `outputs/reports/report_t2_seed0.md`（更新后应包含该图和文字解释）

5.1.4 完成判定（预期现象）

满足下面两条，认为 5.1 完成：

1. **自洽性检查**：
   - `t2_ma_lfl_seed0.json` 中必须包含数值型 `err_1a` 与 `err_1b`（非 NaN，非缺失）
   - `err_1a` 与 `err_1b` 不应恒等于 0 或恒等于同一个常数（除非你明确解释原因）
2. **错配对比"方向性"出现**（不要求一次就非常强，但至少合理）：
   - 常见的预期是：MA-LfL 的 `err_1a` 本身会比 T1 大很多（因为模型错了），并且 `err_1b` 往往不贴近 `err_1a`（恢复奖励不能弥补模型错配）。
   - I-LOLA 通常会表现为：`err_1b` 更贴近自己的 `err_1a`（即恢复误差贴近模型天花板）。
   - 如果你跑出来反过来（MA-LfL 更好），通常意味着：
     - MA-LfL lookahead 里不小心用了 PPO 的梯度信息（变成"对模型"了）
     - 或者 MA-LfL 的 1a/1b 计算没有用同一套采样 batch（导致差异被噪声淹没）

5.2 Stage A vs Stage B：I-LOGEL 与 I-LOLA 的系统对比（T2 为主）

目的
证明 I-LOLA 的 Stage B（耦合动态 + CMA-ES）不是"随便拟合"，而是相对 Stage A（独立近似）的一个清晰扩展。至少要给出一张图说明两者的差异。

5.2.1 实施步骤

1. 新增脚本：`scripts/run_t2_stageA_ilogel_eval.py`
   - 读取 `outputs/data/t2/...pkl`
   - 调用 Stage A（I-LOGEL）得到 `w_hat_A`
   - 计算同样定义的 `err_1a` / `err_1b`（用同一批 state batch）
   - 保存：`outputs/metrics/t2_ilogel_stageA_seed0.json`
   - 保存权重：`outputs/weights/t2_ilogel_stageA_seed0.npy`
2. 扩展 `evaluation/plots.py`：
   - 画一张柱状图对比：`StageA 1b` vs `StageB 1b`（可加 `MA-LfL 1b`）
   - 保存：`outputs/plots/t2_stageA_vs_stageB_seed0.png`
3. 更新 `report_t2_seed0.md`：
   - 增加一小段文字解释 Stage A 与 Stage B 的区别

5.2.2 运行命令

```
python scripts/run_t2_stageA_ilogel_eval.py --seed 0
python scripts/run_t2_ilogel_eval.py  # Stage B（你已有）
python scripts/run_phase3_plots_reports.py
```

5.2.3 验收输出

- `outputs/metrics/t2_ilogel_stageA_seed0.json`
- `outputs/plots/t2_stageA_vs_stageB_seed0.png`
- `outputs/reports/report_t2_seed0.md`（应包含该图）

5.2.4 完成判定（预期现象）

- Stage A、Stage B 都能输出非 NaN 的 `err_1a/err_1b`
- 你至少能观察到一种稳定差异（满足其一即可）：
  - Stage B 的 `err_1b` 更小；或
  - Stage B 的 `err_1b` 更接近 `err_1a`；或
  - Stage B 的结果在不同 seed 下方差更小（见 5.3）

> 若 Stage B 没有显著优于 Stage A，也不算失败。你可以在讨论里写成：
> “Stage B 在当前预算和特征下未显著胜出，但它提供了对耦合动态建模的通用框架，且在某些设置更稳/更可控。”

5.3 多 seed 实验与统计汇总（把 demo 变成结论）

目的
避免“只跑 seed0 的偶然性”。最终交付必须给出多 seed 的均值/方差，才能形成可写入论文的结论。

5.3.1 实施步骤

Step A：写一个统一的多 seed 批量运行脚本

1. 新增脚本 `scripts/run_all_seeds.py`（或分别 t1/t2/t3）：
   - seeds 设为 `[0,1,2,3,4]`（最小 5 个）
   - 对每个 seed 依次执行（按你现有脚本）：
     - T1：`gen_data` + `run_ma_lfl`
     - T2：`gen_data` + `run_t2_ilogel_eval` + `run_t2_ma_lfl_mismatch` + `run_t2_stageA_ilogel_eval` + `run_t2_induced`
     - T3：`gen_data` + `run_t3_ilola_eval` + `run_t3_induced`
     - 最后跑一次 `run_phase3_plots_reports.py`（或每个 seed 一次）

Step B：写一个聚合脚本汇总结果成表格

2. 新增 `scripts/aggregate_results.py`：
    - 扫描 `outputs/metrics/*.json` 与 `outputs/induced/*.json`
    - 汇总为：
        - `outputs/summary/metrics_summary.csv`
        - `outputs/summary/metrics_summary.md`（方便直接贴到报告里）
    - 至少包含列：`task`, `algo`, `seed`, `err_1a`, `err_1b`, `final_loss`, `omega_norm`
    - induced 至少包含：`R_random`, `R_expert`（若有）, `R_induced_last`（或曲线均值）

Step C：增加统计图（误差条 / 均值±std）

3. 扩展 `evaluation/plots.py`：
    - 读取 summary，画 `err_1b` 的均值±std 柱状图（按 T1/T2/T3 分开）
    - 保存：`outputs/plots/summary_err1b.png`

5.3.2 运行命令

```
python scripts/run_all_seeds.py
python scripts/aggregate_results.py
```

5.3.3 验收输出

- `outputs/metrics/` 中出现多个 seed 的 json（例如 `*_seed1.json ... *_seed4.json`）
- `outputs/summary/metrics_summary.csv`
- `outputs/summary/metrics_summary.md`
- `outputs/plots/summary_err1b.png`

5.3.4 完成判定（预期现象）

- 汇总表里每个任务/算法都有 5 条记录（5 seeds）
- `err_1b` 的方差不是 0（除非你真的是完全确定性）
- 关键对比结论在统计层面仍然成立（例如在 T2 上 I-LOLA 的 1b 更贴近其 1a，或 MA-LfL 错配的 1b 更不稳定）

5.4 T2.5 扩展实验（可选，但有助于"扩展性"叙事）

目的
提供一个比 T2 稍复杂但仍可控的环境变体，展示方法在轻微变化下仍可运行。

5.4.1 实施步骤

1. 新增 config：`configs/t25_mpe_extended.yaml`
    - 变更一项即可：更多 agent / 更大 world / 不同 landmark 数
2. 复用 T2 脚本运行：
    - `gen_data`
    - `run_t2_stageA_ilogel_eval`（可复用成通用脚本）
    - `run_t2_ilogel_eval`
    - `run_t2_ma_lfl_mismatch`（如果可运行）
    - `run_t2_induced`
3. 把输出命名为 `t25_*`，并扩展 report 生成一份 `report_t25_seed0.md`

5.4.2 验收输出

- `outputs/metrics/t25_*.json`
- `outputs/plots/t25_*.png`
- `outputs/reports/report_t25_seed0.md`

5.4.3 完成判定

- 跑通即可；不要求一定优于 T2
- 报告中能说明：轻微环境变化下 pipeline 仍工作

5.5 主报告写作与结构化交付（从“自动日志”升级为“正式报告”）

目的
把三段故事线（T1 匹配、T2 错配对比、T3 可扩展性）写成一个可交付的正式文档。自动报告保留为附录。

5.5.1 实施步骤

1. 新建：`outputs/reports/project9_final_report.md`（或仓库根目录 `FINAL_REPORT.md`）
2. 建议结构：
    - 摘要（你做了什么，结论是什么）
    - 方法：MA-LfL 边界、I-LOGEL、I-LOLA（两阶段）
    - 实验：T1/T2/T3 设计与 1a/1b 指标定义
    - 结果：
        - T1：KL + reward heatmap + shaping 解释（PCC 不可靠）
        - T2：**三方对比图**（MA-LfL mismatch vs StageA vs StageB）+ 多 seed 统计图
        - T3：MC KL + induced（含真实/代理 expert baseline 的解释）

- 讨论与局限：MA-LfL 在 T3 的不可行性、近似误差来源、未来工作
- 附录：链接到自动报告 `report_t1_*`, `report_t2_*`, `report_t3_*`

3. 把 `outputs/summary/metrics_summary.md` 中的表直接粘进去，避免手写错误。

5.5.2 验收输出

- `project9_final_report.md`（必须存在）
- 文档中必须引用：
  - `t1_reward_heatmap.png`
  - T2 的对比图（`t2_kl_compare_mismatch_seed0.png` + `t2_stageA_vs_stageB_seed0.png` + `summary_err1b.png`）
  - `t3_kl_seed0.png` 与 `t3_induced_returns_seed0.png`

5.5.3 完成判定（预期现象）

- 读者只看这一份主报告，就能理解：
  1）为什么 T1 权重不对但 KL 对（shaping）；
  2）为什么 T2 要做错配 MA-LfL；
  3）I-LOLA 的贡献点在哪里（StageA vs StageB）；
  4）T3 表示可扩展性与工程可行性。

5.6 复现性固化：一键复现脚本 + 环境锁定

目的
保证“别人拉下仓库就能复现”。这是课程项目交付时非常加分的部分，也能避免答辩时被追问。

5.6.1 实施步骤

1. 新建脚本：`scripts/reproduce_minimal.ps1`（Windows）
   - 顺序执行 seed0 的全链路（T1/T2/T3）
2. 新建脚本：`scripts/reproduce_full.ps1`
   - 执行 5 seeds 的全链路 + aggregate + summary plots + final report（如果你愿意）
3. 在 `README.md` 中加入：
   - 环境安装步骤
   - 运行 minimal / full 脚本的方法
4. 检查所有脚本不依赖绝对路径；所有输出都进入 `outputs/`。

5.6.2 验收输出

- `scripts/reproduce_minimal.ps1` 能在一台新机器上跑到：
  - `outputs/reports/report_t1_seed0.md`
  - `outputs/reports/report_t2_seed0.md`
  - `outputs/reports/report_t3_seed0.md`
- `scripts/reproduce_full.ps1` 能生成：
  - `outputs/summary/metrics_summary.csv`
  - `outputs/plots/summary_err1b.png`

5.6.3 完成判定

- 任何人按 README 操作，能得到同样结构的 outputs 目录（数值可略有差异，但文件应齐全）
- `check_env.py` 通过，且不会出现缺库或路径错误

---

6. 小结

从工程角度看，当前仓库已经完成了 Project 9 的**核心代码框架和单 seed 实验闭环**：

- T1：完全匹配设定下的 MA-LFL 和 I-LOLA；
- T2：PPO 轨道上的 I-LOLA + 诱导训练；
- T3：连续控制环境上的 I-LOLA 可行性示范。

接下来最重要的工作，不再是“能不能跑起来”，而是**系统地对比不同算法、在多个 seed 上收集稳定统计结果，并把这些结果写成清晰的理论和实验故事。**上面的 checklist 可以作为后半程的路线图，你可以按照自己的时间安排，先从 T2 的 MA-LFL 基线和 Stage A vs Stage B 对比开始，一步一步补齐整套故事。