

一、所用 AI 工具名称和版本

ChatGPT-5

二、具体目的和环节

2.1 环节一：绘制模拟退火算法流程图

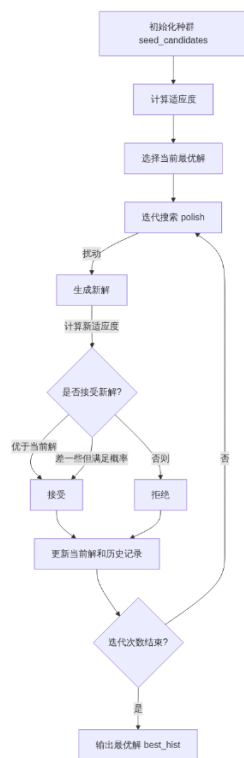
2.2 环节二：完善第六大题灵敏度代码

2.3 环节三：完善论文第七部分模型的评价、改进与推广

三、关键交互记录

3.1 环节一：

提示词：模拟退火算法、流程图



回复：

3.2 环节二：

关键词：problem2.py、灵敏度分析

回复：import numpy as np

import matplotlib.pyplot as plt

import random

```
# ===== 可显示中文（可选） =====
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['axes.unicode_minus'] = False
```

```

# ===== 导入你的问题二代码 =====
import test2p as m # 你提供的主体求解代码（同目录下）

# ----- 工具：在基准参数下求一套最佳策略 -----
def find_baseline_best(seed_n=260, top_k=36, iters=420):
    """
    复用 test2p.py 的搜索与 polish 流程，求基准参数下的一套最优 plan
    """
    # 为了复现性，给个随机种子（可按需注释掉）
    np.random.seed(42)
    random.seed(42)

    seeds = m.seed_candidates(seed_n)
    scored = []
    for s in seeds:
        u, _, _ = m.evaluate_union(s, dt=0.04)
        scored.append((u, s))
    scored.sort(key=lambda x: x[0], reverse=True)
    pool = [x[1] for x in scored[:top_k]]

    best_plan, best_union = None, -1.0
    for p in pool:
        cand, uni, _, _ = m.polish(p, iters=iters, dt=0.02)
        if uni > best_union:
            best_plan, best_union = cand, uni

    # 再细一点的评估
    final_u, _, _ = m.evaluate_union(best_plan, dt=0.01)
    if final_u > best_union:
        best_union = final_u
    return best_plan, best_union

# ----- 工具：单参数灵敏度计算 -----
def sensitivity_curve(best_plan, which_param, base_value, pct_span=0.05, n_pts=21):
    """
    在固定 best_plan 下，让 which_param 在 [1-5%, 1+5%] 连续扰动
    返回：x(百分比%), y(有效遮蔽时间 s)
    which_param ∈ {'R_s', 'v_s', 'v_m'}
    """
    # 百分比刻度（例如 -5% 到 +5%）
    percents = np.linspace(-pct_span, +pct_span, n_pts)
    times = []

```

```

# 记录原始值，分析后要还原
orig_radius = m.radius
orig_sink_v = m.sink_v
orig_v_m     = m.v_m

for p in percents:
    # 设置当前被扰动的参数
    if which_param == 'R_s':
        m.radius = base_value * (1.0 + p)
    elif which_param == 'v_s':
        m.sink_v = base_value * (1.0 + p)
    elif which_param == 'v_m':
        m.v_m = base_value * (1.0 + p)
    # 这个量在 test2p 中用于限制时间上界，需要同步更新
    m.t_hit_fake = np.linalg.norm(m.M0) / m.v_m

    # 评估这套固定策略在当前环境下的有效遮蔽时间（并集）
    union_time, _, _ = m.evaluate_union(best_plan, dt=0.01)
    times.append(union_time)

# 还原全局参数
m.radius = orig_radius
m.sink_v = orig_sink_v
m.v_m     = orig_v_m
m.t_hit_fake = np.linalg.norm(m.M0) / m.v_m

return percents * 100.0, np.array(times)

# ===== 主流程 =====
if __name__ == "__main__":
    # 基准环境参数
    R_s0 = 10.0    # 烟幕半径
    v_s0 = 3.0     # 下沉速度
    v_m0 = 300.0   # 导弹速度

    # 先确保基准环境写入到模块全局
    m.radius = R_s0
    m.sink_v = v_s0
    m.v_m     = v_m0
    m.t_hit_fake = np.linalg.norm(m.M0) / m.v_m

    # 1) 基准下求一套最优策略（固定住）
    best_plan, base_union = find_baseline_best()
    print("基准环境下的并集遮蔽时间（s）: ", round(base_union, 3))

```

```

print("最优策略参数 (theta, v_f, t1, f1, t2, f2, t3, f3): ")
print([round(x, 4) for x in best_plan])

# 2) 分别对 R_s, v_s, v_m 做 ±5% 扰动, 计算曲线
x_Rs, y_Rs = sensitivity_curve(best_plan, 'R_s', R_s0, pct_span=0.05, n_pts=21)
x_vs, y_vs = sensitivity_curve(best_plan, 'v_s', v_s0, pct_span=0.05, n_pts=21)
x_vm, y_vm = sensitivity_curve(best_plan, 'v_m', v_m0, pct_span=0.05, n_pts=21)

# 3) 画图
plt.figure(figsize=(10, 6))
plt.plot(x_Rs, y_Rs, marker='o', label='烟幕半径 $R_s$')
plt.plot(x_vs, y_vs, marker='s', label='下沉速度 $v_s$')
plt.plot(x_vm, y_vm, marker='^', label='导弹速度 $v_m$')

plt.xlabel('参数变化百分比 (%)', fontsize=12)
plt.ylabel('有效遮蔽时间 (s)', fontsize=12)
plt.title('问题二非线性优化模型的灵敏度分析 (策略固定)', fontsize=14)
plt.grid(True, alpha=0.35)
plt.legend()
plt.tight_layout()
plt.show()

```

3.3 环节三:

关键词: A 题.pdf、公式(32)、公式(51)、评价、改进、推广

回复:

(1) 模型层次清晰, 递进关系明确。五个问题逐步扩展模型复杂度, 从单架无人机单枚烟幕干扰弹到多架无人机、多导弹、多次投放, 形成了由简单到复杂的层次化建模体系。

(2) 考虑了实际物理约束。模型中考虑了导弹、无人机、烟幕干扰弹和烟幕的动力学特性, 尤其引入了烟幕下沉速度、爆炸后扩散范围、投放时间间隔等实际约束, 保证模型与真实作战环境的契合度。

优化算法选取合理, 计算效率高。遗传算法在搜索空间大、目标函数复杂、局部极值多的问题上表现优异, 保证了全局最优解的寻找能力; 模拟退火算法在问题三中的应用, 解决了多枚干扰弹投放时的局部搜索与全局跳出问题, 优化效果明显。两者结合, 使得模型兼顾全局搜索和局部优化, 提升了计算效率和结果可靠性。

(1) 忽略了空气阻力和风场影响。模型中假设无人机、导弹及烟幕干扰弹的运动不受空气阻力和风力影响, 但在实际环境中风速变化会显著影响烟幕扩散和位置, 可能导致模型结果与真实情况产生偏差。

(2) 缺乏多导弹协同攻击情况下的适应性。模型假设导弹独立飞行且无规避策略, 未考虑导弹在受干扰后可能采取的机动调整行为, 导致模型在更复杂的对抗场景中适应性不足。

(3) 引入空气动力学模型。在烟幕干扰弹及云团运动中加入空气阻力和风场模型, 例如利用流体力学方程或经验公式修正下沉速度与扩散半径, 使计算结果更加符合实际。

(4) 考虑烟幕的非线性叠加效应。建立烟幕浓度场模型, 通过偏微分方程描述多个烟

幕云团在空间叠加时的动态变化，进而更精确地计算遮蔽效果。

- (5) 优化算法并行化与智能化。将遗传算法与模拟退火算法结合粒子群优化 (PSO) 或强化学习方法，引入并行计算技术，以降低高维优化问题的计算复杂度，加快收敛速度。
- (6) 引入对抗博弈模型。考虑导弹具备智能规避与路径调整功能，将问题扩展为无人机与导弹之间的博弈优化模型，从单向遮蔽问题升级为双向对抗优化问题，提升模型的实战价值。

四、 采纳和人工修改情况

4.1 环节一：无修改，全采纳

4.2 环节二：代码框架采纳，人工修改部分较多，详见 `problem2_anlysis.py`

4.3 环节三：部分采纳，人工修改情况详见论文正文第七大题