基于机械学习的团簇能量预测及结构全局寻优方法

摘要

团簇是由多个分子或原子聚集在一起的微观结构，研究团簇的全局最优结构 (即能量最低) 对于发现新型材料的结构和性能具有重要意义。传统的理论研究方法存在计算时间长、计算效果差等问题，而机械学习作为一个极具前景的多学科交叉领域，能够有效提高模型学习与计算的效率。因此本文对三维团簇的能量预测和结构寻优问题，采用了多种机械学习方法进行研究。

针对问题一：本文首先对数据进行预处理，对表单 2 中的无效数据采取剔除处理，并将化学成分含量数据转化为完美成分数据 (定和为 1)，再采用中心化对数比变换方法对成分数据进行转换用于清除定和限制对后续分析的影响，并对表单 1 四类别进行量化，运用量化后的数据来分析相关性；对于第一小问，题目要求分别研究文物样品表面风化与颜色、玻璃类型以及纹饰间的关系

针对问题二：物流场地 DC5 的关停将会给临近线路和场地带来压力，为了及时疏解均衡负载，我们围绕减少货量波动线路数、平衡工作负荷、减少异常流转件等多项优化目标，结合量子旋转门机制，改进了 NSGA-II 多目标遗传优化算法，使物流正常流转，将货量变动线路数减少至 1683 条，同时选用场地的溢仓货量来反映物流场地的负载情况，确保网络负荷均匀受控，详见表 2 及支撑材料。

针对问题三：为了契合实际的物流紧急情况，在题目允许通过调整线路的开闭，以减少线路运输能力的浪费。在新条件的约束下，同样在确保货量变动线路数与异常流转间数尽量少、线路工作负荷尽量均衡的条件下，对场地 DC9 关停引起物流异动进行应急调度。通过引入罚函数机制，在问题二量子 NSGA-II 优化模型的基础上进行改进，最终得出：DC9 的关停不会对线路货量产生较大影响，仍正常流转，没有线路货量变化。

针对问题四：为了给新增货运场地和线路的决策提供依据，我们基于先前研究，分别就场地和线路的重要程度指标展开讨论，发现物流场地新建需要考虑：发送货物总量、发送线路数等共 9 项指标，线路新建需考虑途径货物总量、工作频率等 4 项指标。据此分别运用投影寻踪法并结合遗传算法建立了综合评价模型，客观评判了所有物流场地和线路的重要性。随后，我们根据评价排名为新建场地和线路提出了建设的建议，认为应该新建 DC82、DC83 两个物流场所，从 DC82 出发有 DC82→DC14、DC82→DC10 等 5 条线路，以 DC82 结束有 7 条线路，5 条从 DC83 出发，7 条以 DC83 结束。同时，DC82、DC83 处理能力应设置为 176022632，DC82→DC14 线路运输能力设为 4620.57…最后，我们通过统计投影寻踪评价模型的全局影响因子、交叉检验误差等验证了模型具有优秀的鲁棒性，其余结果请详见表 7。

在文章最后，从物流调度问题的实际着手，我们对运用的模型进行评价，并给出有实际意义的参考改进方向。

关键词: **TimeNet　Inception 预测　NSGA-II 量子旋转门　投影寻踪法**

1 问题重述

1.1 问题背景

随着科学技术的不断发展，无人机广泛应用于各个领域。当无人机在空中执行定点投放任务时，优化其投放精度成为了目前的主要问题之一。其投放精度在不仅仅依赖于无人机的操作技术，还与无人机执行任务时所处状态和周围环境相关。例如执行任务时无人机的高度、速度，无人机所处位置的风速、投放点周围地理环境等因素。在这里我们仅考虑喷气式无人机执行任务时的问题。喷气式无人机如图 1。

图 1: 喷气式无人机

1.2 问题提出

问题一：假设无人机以平行于水平面的方式飞行，在空中投放一个球形物资到达地面指定位置，需建立无人机投放距离与无人机飞行高度、飞行速度、空气阻力等之间的关系的数学模型。并根据给出的条件，利用已建立的模型分别给出无人机飞行方向与风向相同、相反、垂直情况下无人机的投放距离。

问题二：现有一处河流被冰块堆积阻断，需要使用无人机发射爆炸物对目标进行爆破。在环境的影响下，无人机必须俯冲发射。无人机先水平飞行接近目标，再通过俯冲找准时机发射爆炸物。在此基础上，需建立无人机发射距离与无人机的飞行高度、飞行速度、俯冲角度及发射速度等因素之间的关系的数学模型。利用已知量以及约束条件，给出无人机击中目标的发射策略。

问题三：无人机发射爆炸物命中目标的精度与无人机飞行的稳定性有很大关系。相同条件下，无人机发射爆炸物时越稳定，命中目标的精度越高。开始俯冲后，无人机操控员需要不断调整无人机的飞行姿态以修正风向、风速对无人机的影响。确定飞行速度以及发射速度，在综合考虑各种因素的影响下，建立可量化无人机飞行的稳定性并求出稳定性与命中精度之间的关系的数学模型，此外需利用数值仿真等方法对无人机的稳定性进行分析验证。给定影响无人机飞行的各类因素的大小与范围，在此基础上，探讨出为尽量保持无人机稳定而采取的飞行姿态最优调整策略。

2 模型的假设

1. 假设不存在第三维度的风速或其他因素的影响，可转化为二维平面进行求解。
2. 假设可通过人为调整无人机的推力来控制其俯冲过程。
3. 假设无人机发射爆炸物时的发射方向与无人机的飞行方向一致。
4. 爆炸物和物资的质量分布均匀，并且没有其他外部干扰因素影响其飞行。
5. 无人机和发射筒之间的物理接触和磨损可以忽略不计。
6. 假设无人机在俯冲飞行时处于连续稳定状态

3 符号说明

符号	意义	单位
L	无人机投放距离	m
h_1	俯冲的竖直位移	m
d_1	俯冲的水平位移	m
h_2	爆炸物发射时离地面高度	m
d_2	爆炸物的水平位移	m
D	无人机开始俯冲距目标的水平距离	m
L_1	无人机的发射距离	m
v	物资的速度	m/s
v_0	无人机飞行速度	m/s
v_1	无人机俯冲初速度	m/s
v_2	无人机俯冲末速度	m/s
c	风速	m/s
V	爆炸物的发射速度	m/s
V_x	无人机横向速度	m/s
V_y	无人机纵向速度	m/s
V_z	无人机垂直速度	m/s
m	物资质量	kg
m_1	目标质量	kg
m_2	无人机质量	kg
F	空气阻力大小	N
F_d	空气阻力水平方向上的分力	N
F_h	空气阻力竖直方向上的分力	N
F_f	风速总力	N
F_t	无人机的推力	N
ρ	空气密度	kg/m^2
r	物资的半径	m
d	水平位移	m
d_0	无人机距离投放点的水平距离	m
h	物资任意时刻的高度	m
θ	物资速度与水平方向的夹角	1°

α	俯冲角度	$1°$
γ	仰角	$1°$
phi	横滚角	$1°$
φ	偏航角	$1°$
τ	俯仰角	$1°$
g	重力加速度	m/s^2
a	俯冲加速度	m/s^2
t	俯冲时间	s
A	球体的横截面积	m^2
S_0	无人机的侧面积	m^2
δ	命中精度	
C_D	物资的阻力系数	
C_d	物资的风阻系数	
K	阻力常数	
G	万有引力常数	$N*m^2/kg^2$

4 问题分析

4.1 将三维定点投放物资问题转化为二维平面上的问题

假设无人机以平行于水平面的方式飞行，物资投放之后，物资的运动轨迹实际是一个三维面上的抛体运动，在第三维度上，可能会出现不同方向上的风速对其产生影响，为更加精确的对物资进行定位，我们将其转化为二维平面上的抛体运动。

第一小问：假设物资投放时的初始水平速度与无人机的速度相等，不考虑风速的影响，则物资运动轨迹可以看成一个曲线运动。通过对物资的受力分析与牛顿运动定律，我们可列出相关的微分方程，并利用龙格-库塔法对微分方程进行数值求解。最后得到无人机投放距离与无人机飞行高度、飞行速度、空气阻力等之间的关系。

第二小问：考虑实际情况存在风速的影响，通过风速因素对对第一小问建立的数学模型进行修正，针对风速方向与无人机方向相同、相反、垂直分别建立微分方程，并利用已知的数据通过龙格-库塔法对其进行数值求解，最终得到风速方向不同时无人机的投放距离。

4.2 人为调整无人机的推力使其俯冲运动在控制之下

无人机通过水平运动接近目标，再接着俯冲发射爆炸物。在此过程中，通过人为调整无人机推力大小来控制无人机俯冲过程，为更好的调整无人机的方向以及爆炸物的发射速度，假设无人机的俯冲过程是一个匀加速直线运动，且加速度可人为控制。

第一小问: 由假设可确定出无人机俯冲过程的末速度，即可以得到俯冲过程的运动轨迹以及位移，通过俯冲角度、无人机的飞行高度以及几何关系可得，发射爆炸物时的高度。此时我们可结合第一问的模型，改变第一问模型中初速度 (发射速度) 的方向，对发射距离与无人机的飞行高度、飞行速度、俯冲角度及发射速度等因素之间的关系进行求解。

第二小问：假设风速大小为 $6m/s$ 且方向为逆风，无人机接近目标时的飞行高度为 $800m$、飞行速度为 $300km/h$，爆炸物的相对于无人机的发射速度为 $600km/h$。要求发射爆炸物时无人机与目标

的距离在 $1000m-3000m$ 之间，且无人机的高度不低于 $300m$，利用约束条件并结合第一小问建立的数学模型，通过遍历算法来调整无人机俯冲的时间、爆炸物的发射速度以及发射角度，给出精确打击目标的发射策略。

4.3 三维平面上对无人机的飞行姿态进行稳定性分析以及调整

在前两问的基础上，我们需对无人机飞行的具体过程来进行分析与调整，故需对二维平面进行升维，考虑各个方向上各项因素的影响。对此我们需建立一个在三维平面上的数学模型。

第一小问：无人机飞行的稳定性可以通过其飞行姿态的偏差来衡量，一般来说，偏差越小，其稳定性越高。针对无人机的飞行姿态受到多种因素的影响，可选择在三维平面上来进行分析：在飞行速度、发射速度一定的情况下，无人机的稳定性越好，飞行姿态的偏差越小，命中精度越高。因此，可以将无人机的稳定性与飞行姿态的偏差之间的关系量化。此后可采用数值方法进行仿真分析。可先确定无人机的初始各项参数，并利用建立的模型不断更新目标的运动状态，同时可根据发射方程，得到爆炸物的运动轨迹，最后判断是否命中目标，以及命中的精度。

第二小问：在确定各项参数之后，结合第一问所建立的模型，应用 MATLAB 仿真程序，对无人机的飞行过程进行描述，每一时刻都需更新无人机的位置，并对发射角度以及高度进行遍历，在迭代中得到遍历的结果，并根据稳定性与命中精度的关系，对遍历结果进行筛选，最终得到尽量保持无人机稳定而采取的飞行姿态最优调整策略。

5 问题一的模型

5.1 忽略风速下的微分方程关系模型

5.1.1 微分方程关系的确定

忽略风速对物资投放后运动的影响，假设物资投放速度等于无人机的飞行速度，且投放时方向于无人机运动方向相同。假设无人机飞行高度为 h_0, 飞行速度为 v_0, 无人机投放距离为 L, 无人机离投放点的水平距离 d_0。物资在任意时刻 t 的高度为 h，水平位移为 d, 速度为 v, 受到的空气阻力为 F, 速度与水平方向上的夹角为 θ。物资的质量为 m, 物资为球形, 我们可设其半径为 r, 则物资的横截面积为 $A(A=\pi r^2)$。

根据球类运动空气阻力的计算和分析**?**，阻力 F 公式为

$$F=\frac{1}{2}\rho C_D A v^2 \tag{1}$$

其中，ρ 为空气密度，C_D 为物资的阻力系数。我们可以定义一个常量 K 来作为物资的阻力常数。

$$K=\frac{1}{2}\frac{\rho A}{m} \tag{2}$$

规定竖直向上为正方向，把物资受到的阻力进行分解，物资水平方向上受到的阻力为 F_d, 竖直方向上受到的阻力为 F_h, 我们对物资投放下落过程进行受力分析，如图 2。

根据牛顿运动学定律, 我们可以得到物资在任意时刻 t 水平方向上和竖直方向上的微分方程如

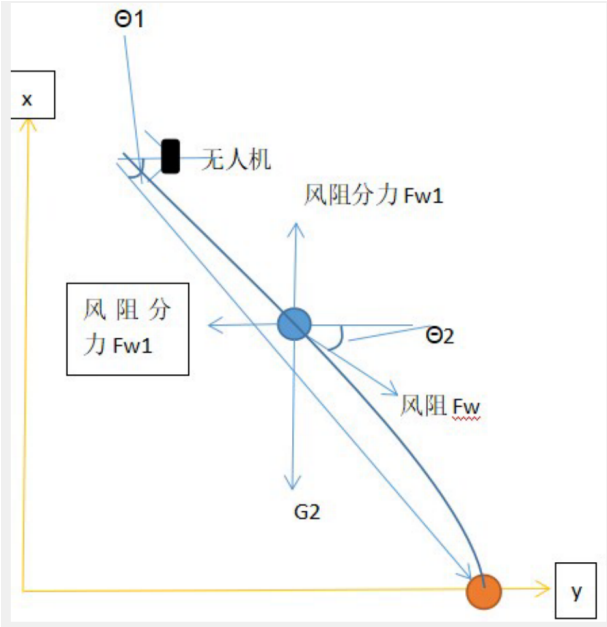

图 2: 物资投放时的受力分析

下：

$$\begin{cases} m\frac{\mathrm{d}^2h}{\mathrm{d}t^2} = -mg + F_d \\ m\frac{\mathrm{d}^2d}{\mathrm{d}t^2} = F_h \\ F_d = mKC_D\sin\theta \\ F_h = mKC_D\cos\theta \end{cases} \tag{3}$$

由图 2 可知，存在几何关系 $\frac{v_y}{v_x} = \frac{\frac{\mathrm{d}h}{\mathrm{d}t}}{\frac{\mathrm{d}d}{\mathrm{d}t}} = \tan\theta$，则可推导出以下微分方程组：

当 $\theta \neq 0$ 时：

$$\begin{cases} \frac{\mathrm{d}^2h}{\mathrm{d}t^2} = -g + \frac{KC_D}{\sin\theta}(\frac{\mathrm{d}h}{\mathrm{d}t})^2 \\ \frac{\mathrm{d}^2d}{\mathrm{d}t^2} = -\frac{KC_D}{\cos\theta}(\frac{\mathrm{d}d}{\mathrm{d}t})^2 \\ \frac{\frac{\mathrm{d}h}{\mathrm{d}t}}{\frac{\mathrm{d}d}{\mathrm{d}t}} = \tan\theta \end{cases} \tag{4}$$

当 $\theta = 0$ 时：

$$\begin{cases} \frac{\mathrm{d}^2h}{\mathrm{d}t^2} = -g + KC_D(\frac{\mathrm{d}h}{\mathrm{d}t})^2 \\ \frac{\mathrm{d}^2d}{\mathrm{d}t^2} = -\frac{KC_D}{\cos\theta}(\frac{\mathrm{d}d}{\mathrm{d}t})^2 \end{cases} \tag{5}$$

5.1.2 引入龙格——库塔法对微分方程进行求解

龙格——库塔算法源于欧拉公式：

$$y_{n+1} = y_n + hf(x_n, y_n) \tag{6}$$

其中，h 为选取步长，(x_n, y_n) 为函数 $f(x, y)$ 上一点。利用该公式使用一次 $f(x_n, y_n)$ 可以根据 y_n 求得 y_{n+1}。

基于该思想而产生的龙格——库塔法的计算步骤如下：通过计算函数函数某些点上的函数值，经过对函数值作线性组合，构造出一组近似计算公式，最后把近似公式与解的泰勒展开式进行比较，使

得前面若干项相同得到一定精度的数值计算公式。? 其中，工程上用途广泛的典型四阶龙格——库塔计算公式如下：

$$\begin{cases} y_{n+1} = y_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \\ k_1 = f(x_n, y_n) \\ k_2 = f(x_n + \frac{h}{2}, y_n + \frac{h}{2}k_1) \\ k_3 = f(x_n + \frac{h}{2}, y_n + \frac{h}{2}k_2) \\ k_4 = f(x_n + h, y_n + hk_3) \end{cases} \tag{7}$$

物资初始水平速度 $v_{d0} = v_0$, 物资初始竖直速度 $v_{h0} = 0$, 初始速度与水平方向下的夹角为 0。应用四阶龙格-库塔法对上述微分方程组进行积分求解无人机离投放点的水平距离的流程图如下：

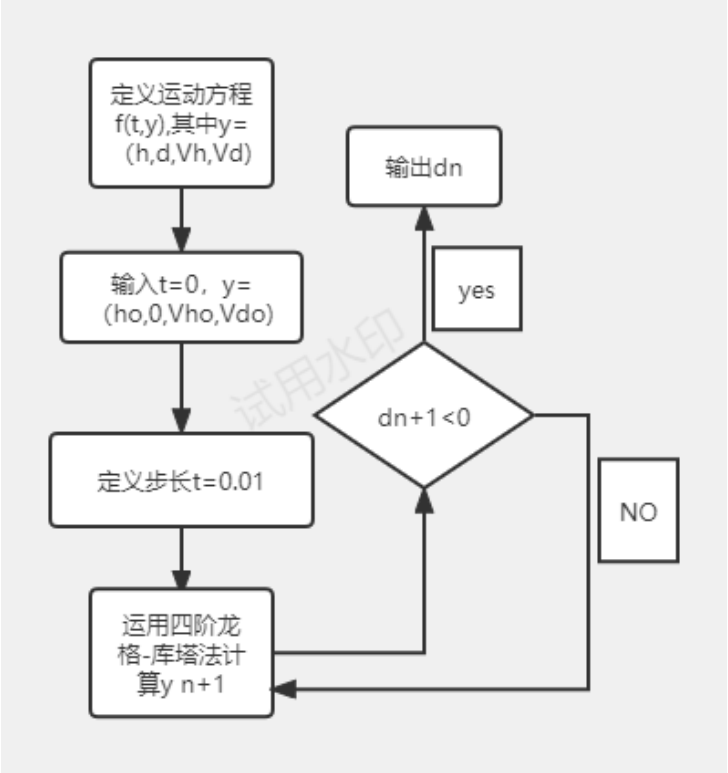

图 3: 物资投放时的受力分析

对于第一小问所确定的微分方程组, 利用龙格-库塔法进行数值求解。最后得到了给定无人机飞行高度为 h_0, 飞行速度为 v, 能求出无人机离投放点的水平距离 d_0。然后根据勾股定理, 可以得到无人机投放距离 $L = \sqrt{h_0^2 + d_0^2}$。

最终得到了无人机投放距离与无人机飞行高度 h_0、飞行速度 v、空气阻力 F 之间的数学关系。

5.2 考虑风速因素时对微分方程关系模型的改进与求解

5.2.1 考虑风速以及方向对第一小问的模型进行改进

在无人机执行任务时，风速是一个不可避免的影响因素，对此假设风速为 c, 球体对于风速的阻力系数为 C_d，风速的方向对于物资所受阻力有着明显影响，已知无人机的飞行高度 h 为 300m, 飞行速度 v 为 300km/h，风速 c 为 5m/s，风向与水平面平行。定义风速对物资的力公式如下:

$$F_f = mKC_dc^2 \tag{8}$$

当无人机飞行方向与风向相同时，对第一小问的关系模型改进得到微分方程组如下：

当 $\theta \neq 0$ 时：

$$\begin{cases} \frac{\mathrm{d}^2h}{\mathrm{d}t^2} = -g + \frac{KC_D}{\sin\theta}(\frac{\mathrm{d}h}{\mathrm{d}t})^2 \\ \frac{\mathrm{d}^2d}{\mathrm{d}t^2} = -\frac{KC_D}{\cos\theta}(\frac{\mathrm{d}d}{\mathrm{d}t})^2 + KC_dc^2 \\ \frac{\frac{\mathrm{d}h}{\mathrm{d}t}}{\frac{\mathrm{d}d}{\mathrm{d}t}} = \tan\theta \end{cases} \tag{9}$$

当 $\theta = 0$ 时：

$$\begin{cases} \frac{\mathrm{d}^2h}{\mathrm{d}t^2} = -g + KC_D(\frac{\mathrm{d}h}{\mathrm{d}t})^2 \\ \frac{\mathrm{d}^2d}{\mathrm{d}t^2} = -\frac{KC_D}{\cos\theta}(\frac{\mathrm{d}d}{\mathrm{d}t})^2 + KC_dc^2 \end{cases} \tag{10}$$

当无人机飞行方向与风向相逆时，对第一小问的关系模型改进得到微分方程组如下：

当 $\theta \neq 0$ 时：

$$\begin{cases} \frac{\mathrm{d}^2h}{\mathrm{d}t^2} = -g + \frac{KC_D}{\sin\theta}(\frac{\mathrm{d}h}{\mathrm{d}t})^2 \\ \frac{\mathrm{d}^2d}{\mathrm{d}t^2} = -\frac{KC_D}{\cos\theta}(\frac{\mathrm{d}d}{\mathrm{d}t})^2 - KC_dc^2 \\ \frac{\frac{\mathrm{d}h}{\mathrm{d}t}}{\frac{\mathrm{d}d}{\mathrm{d}t}} = \tan\theta \end{cases} \tag{11}$$

当 $\theta = 0$ 时：

$$\begin{cases} \frac{\mathrm{d}^2h}{\mathrm{d}t^2} = -g + KC_D(\frac{\mathrm{d}h}{\mathrm{d}t})^2 \\ \frac{\mathrm{d}^2d}{\mathrm{d}t^2} = -\frac{KC_D}{\cos\theta}(\frac{\mathrm{d}d}{\mathrm{d}t})^2 - KC_dc^2 \end{cases} \tag{12}$$

当无人机飞行方向与风向垂直时，对第一小问的关系模型改进得到微分方程组如下：

当 $\theta \neq 0$ 时：

$$\begin{cases} \frac{\mathrm{d}^2h}{\mathrm{d}t^2} = -g + \frac{KC_D}{\sin\theta}(\frac{\mathrm{d}h}{\mathrm{d}t})^2 + KC_dc^2 \\ \frac{\mathrm{d}^2d}{\mathrm{d}t^2} = -\frac{KC_D}{\cos\theta}(\frac{\mathrm{d}d}{\mathrm{d}t})^2 \\ \frac{\frac{\mathrm{d}h}{\mathrm{d}t}}{\frac{\mathrm{d}d}{\mathrm{d}t}} = \tan\theta \end{cases} \tag{13}$$

当 $\theta = 0$ 时：

$$\begin{cases} \frac{\mathrm{d}^2h}{\mathrm{d}t^2} = -g + KC_D(\frac{\mathrm{d}h}{\mathrm{d}t})^2 + KC_dc^2 \\ \frac{\mathrm{d}^2d}{\mathrm{d}t^2} = -\frac{KC_D}{\cos\theta}(\frac{\mathrm{d}d}{\mathrm{d}t})^2 \end{cases} \tag{14}$$

5.2.2 确定模型中的各类参数解得不同风速下的物资投放距离

通过查阅资料以及计算，我们确定 $\rho = 1.29kg/m^2$、$C_D = 0.5$、$C_d = 0.3$、$m = 50kg$、$g = 9.81m/s^2$、$A = 0.04\pi m^2$ 来对上述微分方程组进行求解。

通过龙格-库塔法, 我们可以计算出物体物资运动的轨迹以及无人机飞行方向与风向相同、相反、垂直时物资的投放距离。

无人机飞行方向与风向相同、相反、垂直下物资的运动轨迹如图 4、5、6：

无人机飞行方向与风向相同、相反、垂直时物资的投放距离如下表：

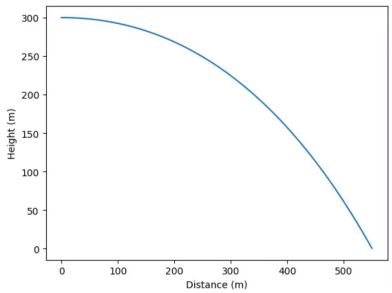

图 4: 相同时物资的运动轨迹

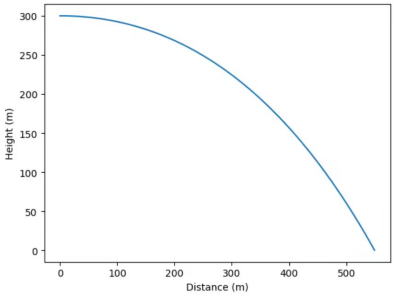

图 5: 相反时物资的运动轨迹

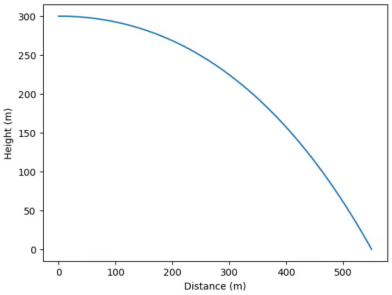

图 6: 垂直时物资的运动轨迹

风向	物资的投放距离 (单位 m)
相同	626.4997297402875
相反	625.9162883752786
垂直	626.1897780146544

6 问题二模型

6.1 自定义无人机俯冲过程改进第一小问的模型

6.1.1 对俯冲过程的自定义

如图 6 所示，无人机俯冲过程可分为三个阶段，进入俯冲阶段、沿直线下降阶段和退出俯冲阶段。**?** 在本问题中，为简化运算，需假设进入俯冲阶段时的路程可以视为一条直线，也就是将无人机的俯冲过程定义为匀加速直线运动。

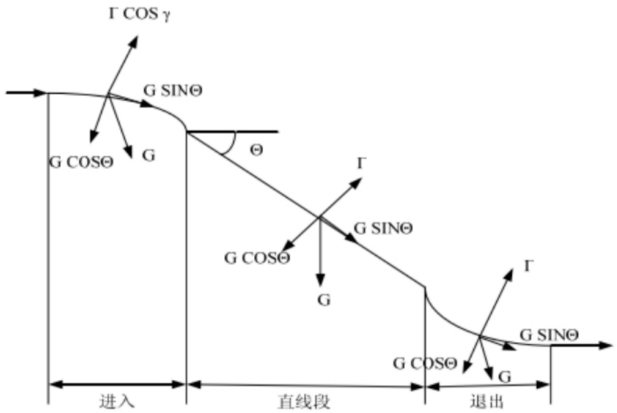

图 7: 无人机俯冲的运动过程

6.1.2 对无人机全过程的分析并通过爆炸物的初速度改进第一小问的模型

假设无人机的俯冲角度为 α, 无人机俯冲的竖直位移为 h_1, 无人机俯冲初末状态的飞行速度分别为 v_1 和 v_2, 其加速度为 a, 俯冲时间为 t。爆炸物发射时距离地面的高度为 h_2, 其发射速度为 V, 发射水平速度为 V_d, 发射竖直速度 V_h, 发射爆炸物时无人机与目标的距离为 L_1, 爆炸物的水平位移为 d_2, 爆炸物为球形, 其质量为 5kg, 半径为 8cm。针对无人机发射爆炸物的整个过程, 如图 7。

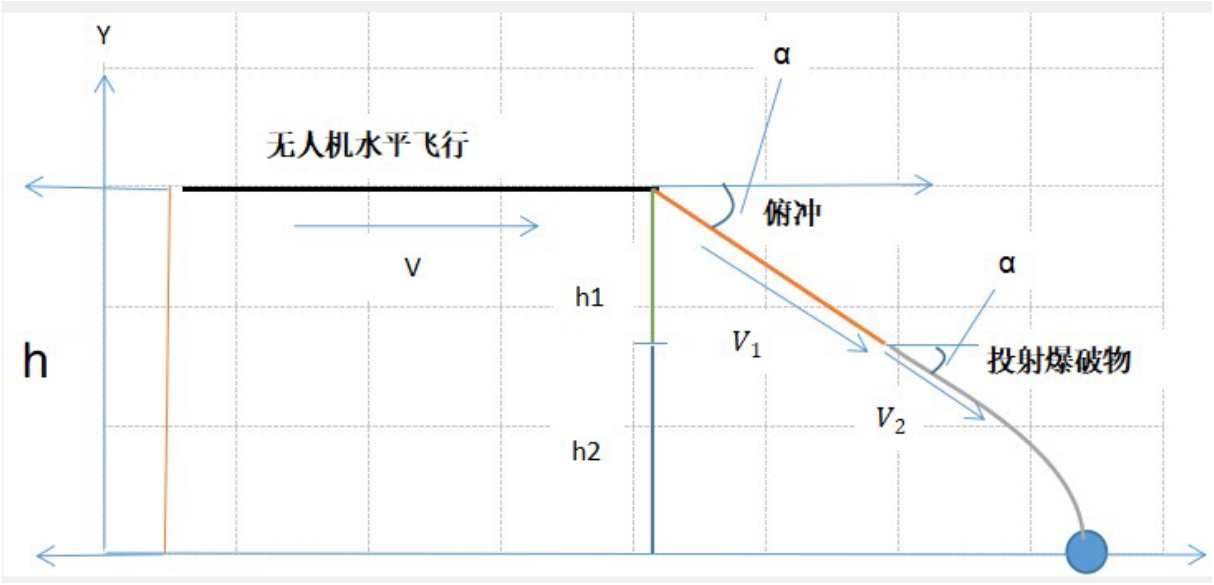

图 8: 无人机发射爆炸物的整个过程

无人机先水平飞行到接近目标, 再进行匀加速直线的俯冲运动, 对俯冲过程进行分析：控制无人机推力使得无人机在发射角度为 α 下进行匀加速直线运动, 其俯冲的初速度为 v_1, 且加速度为 a, 俯冲时间 t 可计算出其俯冲的竖直位移 h_1、无人机发射爆炸物时的高度 h_2、无人机俯冲的水平位移 d_1。

$$\begin{cases} h_1 = (v_1 t + \frac{at^2}{2}) \sin\alpha \\ d_1 = (v_1 t + \frac{at^2}{2}) \cos\alpha \\ h_2 = h - h_1 \end{cases} \tag{15}$$

无人机俯冲后找准时机发射爆炸物, 此时爆炸物的发射角度与无人机的运动方向相同即 α, 爆炸物的飞行速度为 V。然后对第一问中忽略风速的模型的初始条件进行修正, 爆炸物的水平初速度为 $V_{d0} = Vcosa$, 竖直初速度 $V_{h0} = V\sin\alpha$。对于给定的俯冲角度 、爆炸物发射速度 V 、发射高度 h_2, 运用改进后的算法能够得到爆炸物的水平位移 d_2。最后通过勾股定理, 可以得到 $L_1 = \sqrt{d_2^2 + h_2^2}$。

最终确定了无人机发射距离 $L1$ 与无人机的飞行高度 h、飞行速度 v_1、俯冲加速度 a、俯冲时间 t、俯冲角度 α 及爆炸物发射速度 V 之间数学关系。

6.2 假以逆风模型遍历发射角度以及发射速度的发射策略

6.2.1 逆风微分方程关系模型的改进

假设风速与无人机水平运动的方向相反, 风速为 $c = 6m/s$。无人机接近目标时的飞行高度为 $h = 800m$、飞行速度为 $v_1 = 300km/h$, 爆炸物相对于无人机的发射速度为 $600km/h$，则爆炸物发射速度 $V = v_2 + 600km/h$。结合第一问第二小问的逆风模型, 可知逆风时微分方程组如下：

当 $\theta \neq 0$ 时：

$$\begin{cases} \frac{\mathrm{d}^2 h}{\mathrm{d}t^2} = -g + \frac{KC_D}{\sin\theta}(\frac{\mathrm{d}h}{\mathrm{d}t})^2 \\ \frac{\mathrm{d}^2 d}{\mathrm{d}t^2} = -\frac{KC_D}{\cos\theta}(\frac{\mathrm{d}d}{\mathrm{d}t})^2 - KC_d c^2 \\ \frac{\frac{\mathrm{d}h}{\mathrm{d}t}}{\frac{\mathrm{d}d}{\mathrm{d}t}} = \tan\theta \end{cases} \tag{16}$$

当 $\theta = 0$ 时：

$$\begin{cases} \frac{\mathrm{d}^2 h}{\mathrm{d}t^2} = -g + KC_D(\frac{\mathrm{d}h}{\mathrm{d}t})^2 \\ \frac{\mathrm{d}^2 d}{\mathrm{d}t^2} = -\frac{KC_D}{\cos\theta}(\frac{\mathrm{d}d}{\mathrm{d}t})^2 - KC_d c^2 \end{cases} \tag{17}$$

对上面的逆风模型的初速度进行修正, 爆炸物的水平初速度为 $V_{d0} = V\cos\alpha$, 竖直初速度 $V_{h0} = V\sin\alpha$。对于给定的俯冲角度 、爆炸物发射速度 V 、发射高度 h_2, 能够得到爆炸物的水平位移 d_1。最后通过勾股定理, 可以得到 $L_1 = \sqrt{d_1^2 + h_2^2}$。

6.2.2 在一定约束条件下遍历发射角度和速度得到精确打击的发射策略

查阅资料以及数据了解到，无人机俯冲的加速度取决于无人机的设计和性能，以及俯冲的角度和速度等因素。在实际操作中，无人机俯冲时需要遵循安全性和操作性能的要求，因此加速度需在可承受的范围内。无人机俯冲的加速度应该符合飞机的设计规范和安全要求，在本问题中确定加速度 a 为 $20m/s^2$, 俯冲时间 t 为 $3s$。为保证发射角度能够保证弹药以较大的速度和准确度命中目标，同时避免无人机本身受到弹药反冲的风险，此外，无人机俯冲时角度不宜过大，否则弹药可能会产生“偏离现象”，导致偏离目标，降低打击效果。同时也不宜过小，否则会增加无人机受到抗风能力的要求，加大飞行难度和风险。综上所述，我们确定发射角度 α 的范围为 $[10°, 50°]$。

结合公式 (15), 我们采用遍历算法对角度 α 以每一度为一次遍历的单位, 可算出无人机发射爆炸物时的高度 h, 以 $h \geq 300m$ 为约束条件, 可以得到新的 α 的范围。

再结合公式 (16)、(17), 利用采用龙格——库格法, 对其进行数值求解, 以对新的角度 α 每一度为一次遍历的单位, 可以算出发射爆炸物时无人机与目标的距离 L_1 和无人机开始俯冲距目标的水平距离 $D = d_1 + d_2$。以 $1000m \leq L_1 \leq 3000m$ 和 $D \leq 10000m$ 为约束条件, 可以得到最终的角度 α 范围, 以及每个角度 α 所确定的发射爆炸物时无人机与目标的距离 L_1 和无人机开始俯冲距目标的水平距离 D。最后得到发射角度的局部遍历图如下。

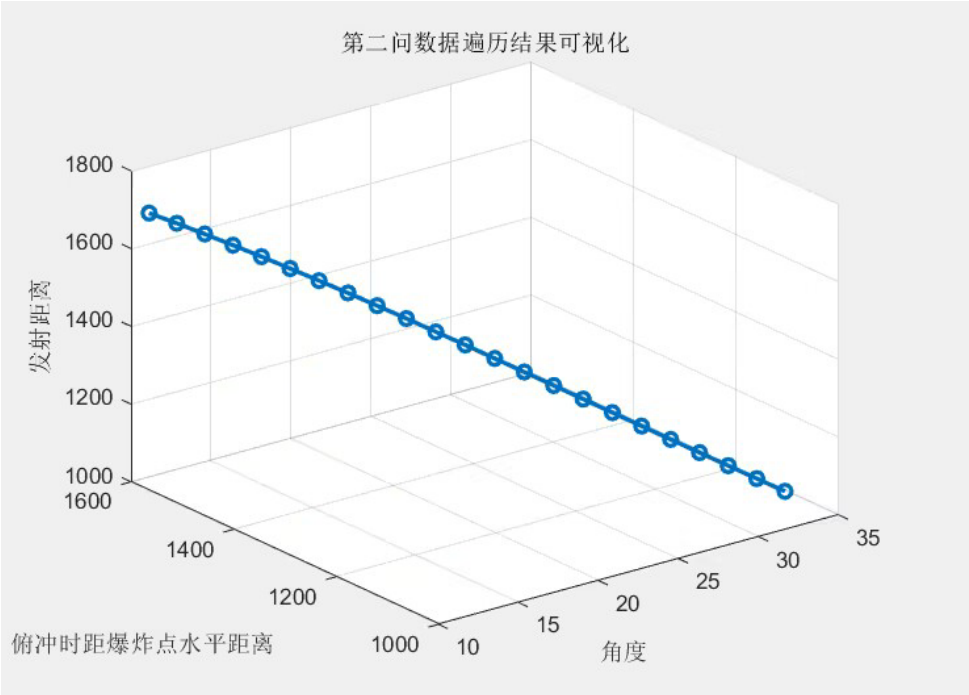

图 9: 发射角度的局部遍历图

首先，我们可以看到，随着发射角度的增加，发射距离逐渐减小。这是因为在较大的角度下，无人机飞行的轨迹更加弯曲，导致发射距离减小。

其次，我们可以看到，当发射角度为 10 度时，发射距离以及俯冲时距爆炸点水平距离最远。这是因为在 10 度左右的角度下，无人机飞行的轨迹较为平直。

最后，我们还可以看到，当发射角度为 32 度时，发射距离以及俯冲时距爆炸点水平距离最近。这是因为在较大的角度下，无人机飞行的轨迹更加弯曲，导致发射距离减小。

综上所述，对于需要在发射爆炸物时保持一定高度的无人机，我们可以选择在发射角度为 20 度左右的情况下进行发射，以达到最远的发射距离和较好的水平距离折中效果。当然，在实际应用中，我们还需要考虑无人机的性能和环境条件等因素，以确定最优的发射角度。

最终得到无人机击中目标的发射策略如下表：

发射角度	发射距离 (单位 m)	俯冲时距爆炸点水平距离 (单位 m)
10	1564.60359417169	1712.86339081383
11	1541.95133331364	1689.18868332338
12	1518.64870727308	1664.63654271622
13	1494.69601658843	1639.20013552174
14	1470.5018345182	1613.33949265539
15	1445.68273931874	1586.60888823598
16	1420.67234132279	1559.49835764619
17	1395.50889756245	1532.04612437374
18	1369.77588604461	1503.76270808183
19	1343.94524967773	1475.18552157607
20	1317.57840261934	1445.79437405981
21	1291.6648184929	1416.73710686069
22	1265.27629178441	1386.91779804867
23	1238.93634119333	1356.94111583851
24	1212.68956098763	1326.85171227506
25	1186.581114374	1296.69520230166
26	1160.65660429722	1266.51811573234
27	1134.96191735998	1236.36783031012
28	1109.54303845335	1206.2924828977
29	1084.44583421273	1176.34085590938
30	1059.71580410931	1146.5622362153
31	1035.39779887105	1117.00624394439
32	1011.53570700755	1087.72262889421

以上所给出的发射策略是在俯冲加速度为 $a = 20m/s$，俯冲时间 $t = 3s$ 所确定的，发射策略可通过改变加速度以及俯冲时间来进行调整从而迭代出不同的发射角度得到各种不同的发射策略。

7 问题三模型

7.1 确定无人机飞行的稳定性以及命中精度的关系

7.1.1 在三维平面具体量化无人机飞行的稳定性

无人机在飞行过程中会受到来自各个方向的风速影响，因此我们将此问题升维成三维。建立三维坐标系，对无人机在三维平面上的飞行过程进行分析如图 8。在考虑无人机飞行稳定性时，需要

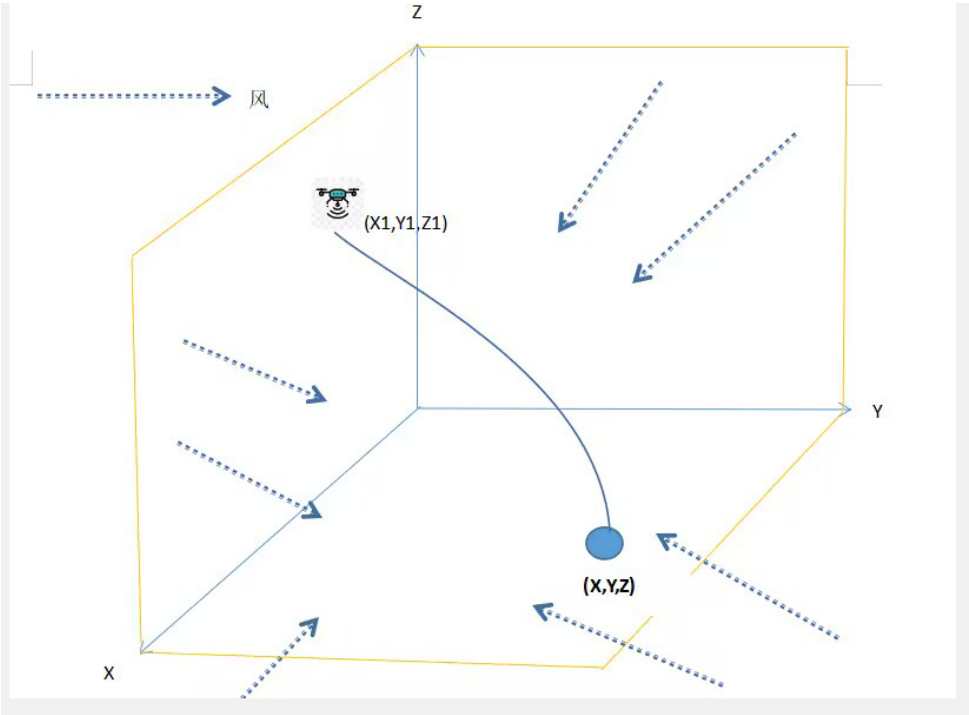

图 10: 无人机发射爆炸物的整个过程

综合考虑以下因素：

风速和风向对无人机的影响；无人机的质量和重心位置；无人机的气动特性，包括升力和阻力的变化；无人机的控制系统，包括姿态控制和飞行控制。为具体量化无人机飞行的稳定性，我们引入飞行姿态的偏差对其进行衡量。假设无人机飞行的姿态角分别为俯仰角 τ、横滚角 ϕ、偏航角 φ，假设无人机初始位置为 (x_0, y_0, z_0), 无人机的位置坐标为 (x', y', z')。定义命中精度为 δ。

$$\delta = \sqrt{(x - x')^2 + (y - y')^2 + (z - z')^2} \tag{18}$$

规定横向风速为 V_x，纵向风速为 V_y，垂直风速为 V_z。根据各角度的关系可定义：

$$\begin{cases} \tau'' = (gsin(\phi) - \frac{C_D S_0 V_x^2}{2m} - \frac{F_t sin(\tau)}{V} \\ \phi'' = -gcos(\phi)sin(\varphi) - \frac{C_D S_0 V_y^2}{2m} + \frac{F_t cos(\tau)sin(\varphi)}{V sin(\tau)} \\ \varphi'' = -gcos(\phi)cos(\varphi) - \frac{C_D S_0 V z^2}{2m} + \frac{F_t cos(\tau)cos(\varphi)}{V sin(\tau)} \end{cases} \tag{19}$$

其中，S_0 为无人机的侧面积，F_t 为无人机发动机产生的推力，V 为无人机的速度。

我们可以采用飞行物体运动学公式建立无人机运动模型，假设无人机初始位置为 (x_0, y_0, z_0)，初

始速度为 (v_{x0}, v_{y0}, v_{z0})，则无人机在 t 时间后的位置为：

$$\begin{cases} x = x_0 + v_{x0}t \\ y = y_0 + v_{y0}t \\ z = z_0 + v_{z0}t - \frac{1}{2}gt^2 \end{cases} \tag{20}$$

其中，g 为重力加速度，可以取 $9.8m/s^2$ 等常值。根据牛顿运动学公式，退得目标在三维平面中的运动方程为：

$$\begin{cases} x_1 = x_{10} + v_{x1}t \\ y_1 = y_{10} + v_{y1}t \\ z_1 = z_{10} + v_{z1}t - \frac{1}{2}gt^2 \end{cases} \tag{21}$$

其中，(x_{10}, y_{10}, z_{10}) 表示目标的初始位置，(v_{x1}, v_{y1}, v_{z1}) 表示目标各个方向上的初始速度。

7.1.2 无人机与目标运动之间的相互影响

无人机对目标的引力大小与距离的平方成反比，因此可以根据万有引力定律建立无人机与目标之间的运动方程如下：

$$\begin{cases} m_1\frac{d^2x_1}{dt^2} = -\frac{Gm_1m_2(x_1-x_2)}{((x_1-x_2)^2+(y_1-y_2)^2+(z_1-z_2)^2)^{\frac{3}{2}}} \\ m_1\frac{d^2y_1}{dt^2} = -\frac{Gm_1m_2(y_1-y_2)}{((x_1-x_2)^2+(y_1-y_2)^2+(z_1-z_2)^2)^{\frac{3}{2}}} \\ m_1\frac{d^2z_1}{dt^2} = -\frac{Gm_1m_2(z_1-z_2)}{((x_1-x_2)^2+(y_1-y_2)^2+(z_1-z_2)^2)^{\frac{3}{2}}} \\ m_2\frac{d^2x_2}{dt^2} = -\frac{Gm_1m_2(x_1-x_2)}{((x_1-x_2)^2+(y_1-y_2)^2+(z_1-z_2)^2)^{\frac{3}{2}}} \\ m_2\frac{d^2y_2}{dt^2} = -\frac{Gm_1m_2(y_1-y_2)}{((x_1-x_2)^2+(y_1-y_2)^2+(z_1-z_2)^2)^{\frac{3}{2}}} \\ m_2\frac{d^2z_2}{dt^2} = -\frac{Gm_1m_2(z_1-z_2)}{((x_1-x_2)^2+(y_1-y_2)^2+(z_1-z_2)^2)^{\frac{3}{2}}} \end{cases} \tag{22}$$

其中，m_1 和 m_2 分别是目标和无人机的质量，G 为万有引力常数。

7.1.3 无人机发射爆炸物的轨迹方程

在三维坐标系中，我们可以根据极坐标系下的物理公式建立无人机发射爆炸物模型，假设一枚爆炸物的质量为 m_b，发射速度为 v_b，仰角为 γ，发射时刻为 t_0，则其轨迹可以表示为：

$$\begin{cases} x_b = (v_b\cos\gamma)\cdot(t-t_0) \\ y_b = (v_b\sin\gamma)\cdot(t-t_0) \\ z_b = (v_{z0} + v_b\sin\gamma - g(t-t_0))\cdot(t-t_0) - \frac{1}{2}gt'^2 \end{cases} \tag{23}$$

其中，t' 表示发射爆炸物至命中目标所需的时间。结合上述所建立的模型，可得到稳定性与命中精度之间的关系。

7.1.4 采用 MATLAB 对上述模型进行数值仿真以及验证分析

在上述模型的基础上，可以采用数值方法进行仿真分析。首先确定无人机和目标的初始位置、速度等参数，然后根据公式 19、20、21，计算无人机和目标在 t 时间内的位置和速度。在得到位置以及速度之后，利用万有引力定律，通过公式 22 计算出无人机对目标的引力大小，再根据质量和距离的

关系，求出无人机对目标施加的加速度，从而更新目标的运动状态。相应地，根据发射方程 23，计算出爆炸物在 t' 时间内的运动轨迹，最后判断爆炸物是否命中目标，并通过公式 18 计算爆炸物命中目标的精度。同时为了建立量化关系，可以通过统计分析，计算出在不同的参数条件下，无人机发射爆炸物命中目标的概率和精度。具体步骤如下：

Step1: 首先采用计算机仿真方法，模拟各种参数情况下的运动过程，得到大量的数据样本。

Step2: 利用样本数据，采用数据挖掘、机器学习等方法，建立预测模型，对无人机的稳定性和命中精度进行预测。

Step3: 最终，可以对模型进行优化，确定最佳的参数组合，以实现无人机的最优性能。

7.2 确定各类参数并采用遍历算法得到最优策略

7.2.1 根据题设对模型进行参数的调整以及约束

题设中给出风速 c 为 $6m/s$，无人机的飞行速度范围为 $300km/h-400km/h$，爆炸物的发射速度为 $500km/h$（相对于无人机的速度）。无人机在 $800m$ 高度开始俯冲，初始俯冲角度为 45°，发射爆炸物时的飞行高度不低于 300m。则有以下初始条件：

$$\begin{cases} c = 6m/s \\ 300km/h \leq v_0 \leq 400km/h \\ V - v_2 = 500km/h \\ h_1 + h_2 = 800m \\ \gamma = 45^\circ \\ h_2 \leq 300m \end{cases} \tag{24}$$

7.2.2 采用 MATLAB 程序对上述模型进行遍历

首先，为减少 MATLAB 程序的运行时间，我们采用三元组顺序表法? 来储存相同的数据元素。以下为运用 MATLAB 程序求解飞行姿态调整策略的步骤：

Step1: 定义速度、高度、质量、半径等参数以及仿真步长。

Step2: 定义需要使用的公式并对题设条件的角度进行转换。

Step3: 在三维坐标面上进行速度以及各方向上阻力的计算。

Step4: 对无人机的运动过程公式进行描写，并采用三元组法存储数据。

Step5: 在无人机飞行高度的范围内进行遍历，并同时更新飞机位置。

Step6: 遍历合适的发射角度，更新发射位置、发射速度并对各方向上的风阻以及阻力进行计算。

Step7: 对爆炸物的发射过程进行分析与计算，对时间取一定步长进行遍历，每一步长的迭代都需更新记录，最终计算出位移的绝对值，并通过各个方向上的分速度计算出爆炸物的合速度。同时对风阻力也进行更新，代入发射过程中计算。

Step8: 最终运行并输出符合条件下无人机高度、速度、俯冲角度。

7.2.3 无人机命中精度关于发射高度以及速度的全局遍历图

7.2.4 无人机飞行姿态的最终调整策略

我们通过利用 MATLAB 进行数值仿真后，得到了速度与高度在不同精度下的关系，稳定性与精度的关系 $\delta = \sqrt{(x-x')^2 + (y-y')^2 + (z-z')^2}$，经过分析验证再由图 11 可知，最适合的无人机

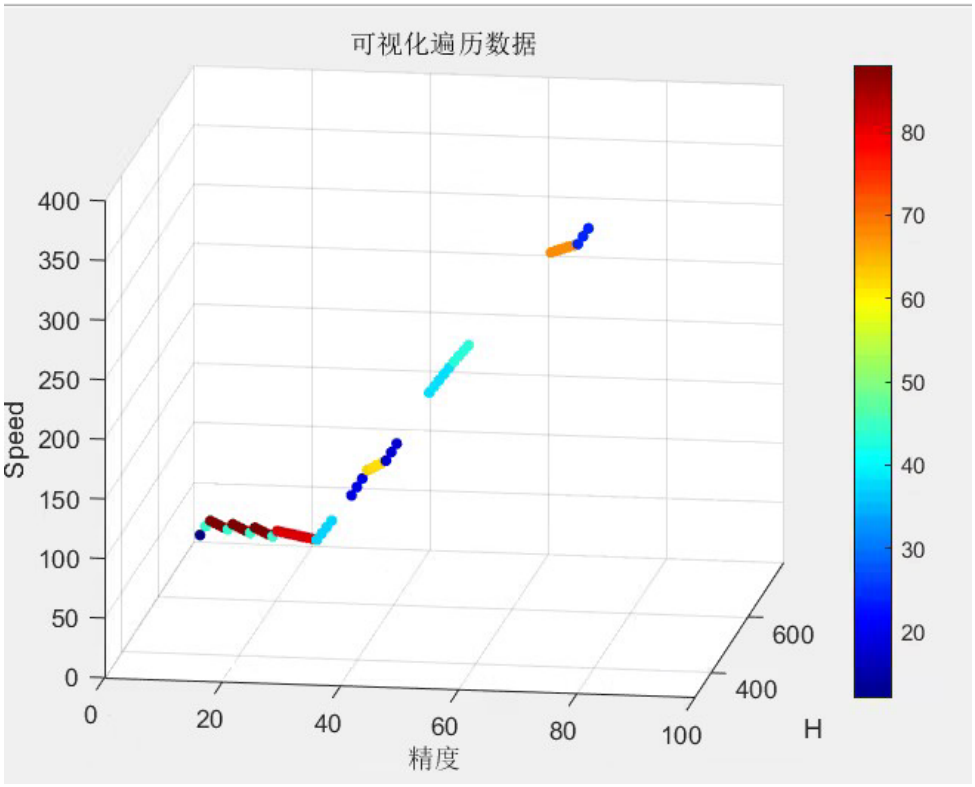

图 11: 命中精度的全局遍历图

发射飞行速度在 325 到 395 之间，最佳精度为 70，发射俯冲角度在 70 度到 80 度之间，首先，我们需要确定一个合适的初始俯冲角度，以确保无人机可以逐渐降低高度并保持稳定。一般来说，我们可以选择一个较小的俯冲角度，例如 45°，然后改变俯冲角度，以使无人机缓慢降低高度并保持稳定。

当无人机高度降低到一定程度时，例如 500m 左右，我们需要开始逐渐调整飞行姿态，使无人机的速度在发射爆炸物时不低于要求的速度，并保持稳定。在调整飞行姿态时，我们需要考虑无人机的动力性能、气流等因素，无人机的动力性能包括飞行速度、最大飞行高度、续航能力，并逐步调整飞行姿态，以使无人机达到要求的速度和稳定性。

当无人机高度降低到 300m 左右时，我们需要进一步调整飞行姿态，以确保无人机在发射爆炸物时的飞行高度不低于要求的高度，并保持稳定同时调整俯冲角度。在调整飞行姿态时，我们需要考虑无人机的高度、姿态、速度等因素，并逐步调整飞行姿态，以使无人机达到要求的高度和稳定性。

在发射爆炸物后，我们需要逐渐调整飞行姿态，使无人机恢复稳定飞行状态，并逐渐增加高度，以避免撞地或其他障碍物。在调整飞行姿态时，我们需要考虑无人机的动力性能、高度、速度等因素，并逐步调整飞行姿态，以使无人机恢复稳定飞行状态。

8 模型的评价和改进

8.1 模型的优点和不足

8.1.1 模型的优点

1、模型的原创性高，本文中模型均为自主建模。

2、问题一：第一小问：考虑了阻力对物资运动状态影响，并对每一时刻的物资受到的阻力进行了修正。对模型进行了合理简化，把三维空间的运动巧妙转换为二维平面的运动。第二小问：定义了风速对物资运动的作用力，而不是简单的把风速与物资水平分速度相加。

3、问题二：自定义俯冲过程，使得俯冲过程可控。并运用遍历算法解出所有满足条件的发射策略。

4、问题三：采用三维平面建模的方法，较全面地考虑无人机发射爆炸物命中目标的精度与无人机飞行的稳定性之间的关系。

8.2 模型的缺点与改进

8.2.1 模型的缺点

1、对于各项参数均为查阅资料所得，可能与实际情况略有偏差。

2、问题二中为简化模型，将无人机从水平面到俯冲之间的这一曲线运动简化为直线运动。并假设风速为逆风，而在风速方向不确定时会有不同的发射策略。

3、问题三中，无人机在实际应用中可能会遇到更加复杂的情况；模型中存在一些不确定性因素，如风速和风等参数的变化可能会对模型的精度产生一定影响；模型中没有考虑无人机与环境中其他物体的相互影响，如建筑物、地形等。

8.2.2 模型的改进

1、问题一：

考虑投放精度和稳定性的要求时，需要考虑以下因素：目标投放区域的大小和形状针对不同的投放区域大小和形状，需要设计不同的飞行策略和投放点选择算法，以最大程度地提高投放精度和稳定性。在投放过程中，风速和气压等环境因素的变化可能会对无人机和物资的运动轨迹产生影响，因此需要在投放过程中进行实时监测和适时调整以保证投放精度和稳定性。综上所述，需要在模型中引入相关的约束条件和优化目标，并基于实时数据进行动态调整，以实现更优化的无人机物资投放效果。

2、问题二：在问题二模型的基础上，引入足够的参数如无人机的结构、质量、引擎和控制器等；可采用数值仿真方法。利用计算机软件，将所建立的数学模型进行求解和模拟，获得无人机俯冲过程中机体和控制器的状态变化，验证预测结果的准确性和实验结果的可行性；在实验室进行验证，可以使用实验数据对模型进行修正和改进，提高模型的精度和准确性，进一步验证模型的实用性。

3、问题三：通过增加模型的复杂度，引入更多的因素和参数，提高模型的适用性和准确度；对于不确定性因素，可以采用蒙特卡罗模拟等方法进行模拟分析，得到更加可靠的结果；对于无人机与环境中其他物体的相互影响，可以采用计算流体力学等方法进行分析，得到更加精确的结果。此外，可以考虑利用机器学习、深度学习等方法来优化模型，提高模型的测准确度和自适应性。

参考文献

周雨青, 叶兆宁, 吴宗汉. 球类运动中空气阻力的计算和分析 [J]. 物理与工程, 2002, 12(1):5.

潘艇, 杨福彪, 朱勇, 等. 基于龙格-库塔的弹道微分方程解算的 FPGA 实现 [J]. 计算机测量与控制, 2015, 23(12):4.

陈挚. 目标的识别与快速准确攻击一体化设计研究 [D]. 南京航空航天大学, 2011.

仲一. 矩阵的三种存储方式-三元组法行逻辑链接法十字链表法.https://zhuanlan.zhihu.com/p/340424465.2023.5.17.

创建三维绘图.https://ww2.mathworks.cn/help/matlab/visualize/creating-3-d-plots.html.2023.5.17

小林 up. 空间坐标系坐标变换及 matlab 代码实现.https://blog.csdn.net/subtitle/*article*/*details*/123301279.2023.5.

9 附录

第一二问的 Python 源程序

```
#问题一第一小问的代码
#不考虑风速的模型的求解函数代码
#给定无人机高度h0和无人机速度v0，可求出无人机投放距离L。

#导入库
import numpy as np
import matplotlib.pyplot as plt
import math
# 定义初始条件
K = 0.5*1.29*np.pi*(0.2**2)/50
cd=0.5 #物资空气阻力系数
g = 9.81 # 重力加速度，单位m/s^2
tmax = 40.0 # 时间上限，单位s
dt = 0.01 # 步长，单位s
#定义函数
def q(h0,v0):
#h0,无人机高度，单位m。v0，无人机飞行速度，单位，m/s。
# 定义运动方程
def f1(t, y, i):
h=y[i,0]
d=y[i,1]
vh=y[i,2]
vd=y[i,3]
if(vd==0):
theta=0.5*np.pi
else:
x=vh/vd
theta = math.atan(x)
if(theta==0):
return[-vh,
vd,
-g,
- K / np.cos(theta) * vd**2]
elif(theta==0.5*np.pi):
return[-vh,
vd,
-g + K*cd/ np.sin(theta) * vh**2,
0]
else:
return [-vh,
vd,
-g + K*cd/ np.sin(theta) * vh**2,
- K*cd / np.cos(theta) * vd**2
]
# 初始化数组
nsteps = int(tmax / dt)
t = np.zeros(nsteps)
y = np.zeros((nsteps, 4))
# 初值
y[0, 0] = h0
y[0, 1] = 0
```

```
    y[0, 2] = 0
    y[0, 3] = v0
    # 利用龙格-库塔法对运动方程进行数值求解
    for i in range(nsteps - 1):
    k1 = np.multiply(np.array(f(t[i], y, i)), dt)
    k2 = np.multiply(np.array(f(t[i] + dt / 2, y + k1 / 2, i)), dt)
    k3 = np.multiply(np.array(f(t[i] + dt / 2, y + k2 / 2, i)), dt)
    k4 = np.multiply(np.array(f(t[i] + dt, y + k3, i)), dt)
    if ((y[i] + (k1 + 2*k2 + 2*k3 + k4) / 6)[0]<0):
    row_to_delete=range(i+1,nsteps)
    y=np.delete(y,row_to_delete,axis=0)
    row_to_delete=range(i+1,nsteps)
    t=np.delete(t,row_to_delete,axis=0)
    break
    else:
    y[i+1] = y[i] + (k1 + 2*k2 + 2*k3 + k4) / 6
    t[i+1] = t[i] + dt
    # 绘制轨迹图
    plt.plot(y[:, 1], y[:, 0])
    plt.xlabel('Distance (m)')
    plt.ylabel('Height (m)')
    plt.show()
    # 绘制速度图
    v = np.sqrt(y[:, 2]**2 + y[:, 3]**2)
    plt.plot(t, v)
    plt.xlabel('Time (s)')
    plt.ylabel('Velocity (m/s)')
    plt.show()
    # 绘制加速度图
    a = np.zeros(y.shape[0])
    a[0]=-np.sqrt(g**2 + v0**2)
    for i in range(1,y.shape[0]):
    a[i] = (v[i] - v[i-1]) / dt
    plt.plot(t, a)
    plt.xlabel('Time (s)')
    plt.ylabel('Acceleration (m/s^2)')
    plt.show()
    return math.sqrt(y[-1,1]**2 + h0**2)
    #问题一第二小问的代码

    ##当风速为顺风时，投放距离的模型求解代码

    # 导入库
    import numpy as np
    import matplotlib.pyplot as plt
    import math
    # 初始条件
    v0 = 300.0/3.6 # 初始速度，单位m/s
    h0 = 300.0 # 初始高度，单位m
    c = 5.0 # 风速，单位m/s # 物体与水平面的夹角，单位rad
    K = 0.5*1.29*np.pi*(0.2**2)/50
    cd=0.5 #物资空气阻力系数
    cc=0.3 #物资风速阻力系数
    g = 9.81 # 重力加速度，单位m/s^2
    tmax = 40.0 # 时间上限，单位s
```

```
dt = 0.01  # 步长，单位s
# 定义运动方程
def f(t, y, i):
h=y[i,0]
d=y[i,1]
vh=y[i,2]
vd=y[i,3]
if(vd==0):
theta=0.5*np.pi
else:
x=vh/vd
theta = math.atan(x)
if(theta==0):
return[-vh,
vd,
g,
- K / np.cos(theta) * vd**2 + K*cc*c**2]
elif(theta==0.5*np.pi):
return[-vh,
vd,
g - K*cd/ np.sin(theta) * vh**2,
K*cc*c**2]
else:
return [-vh,
vd,
g - K*cd/ np.sin(theta) * vh**2,
- K*cd / np.cos(theta) * vd**2 + K*cc*c**2
]
# 初始化数组
nsteps = int(tmax / dt)
t = np.zeros(nsteps)
y = np.zeros((nsteps, 4))

# 初值
y[0, 0] = h0
y[0, 1] = 0
y[0, 2] = 0
y[0, 3] = v0

# 运动方程求解
for i in range(nsteps - 1):
k1 = np.multiply(np.array(f(t[i], y, i)), dt)
k2 = np.multiply(np.array(f(t[i] + dt / 2, y + k1 / 2, i)), dt)
k3 = np.multiply(np.array(f(t[i] + dt / 2, y + k2 / 2, i)), dt)
k4 = np.multiply(np.array(f(t[i] + dt, y + k3, i)), dt)
if((y[i] + (k1 + 2*k2 + 2*k3 + k4) / 6)[0]<0):
row_to_delete=range(i+1,nsteps)
y=np.delete(y,row_to_delete,axis=0)
row_to_delete=range(i+1,nsteps)
t=np.delete(t,row_to_delete,axis=0)
break
else:
y[i+1] = y[i] + (k1 + 2*k2 + 2*k3 + k4) / 6
t[i+1] = t[i] + dt
# 绘制轨迹图
```

```
plt.plot(y[:, 1], y[:, 0])
plt.xlabel('Distance (m)')
plt.ylabel('Height (m)')
plt.show()

# 绘制速度图
v = np.sqrt(y[:, 2]**2 + y[:, 3]**2)
plt.plot(t, v)
plt.xlabel('Time (s)')
plt.ylabel('Velocity (m/s)')
plt.show()

# 绘制加速度图
a = np.zeros(y.shape[0])
a[0]=-np.sqrt(g**2 + v0**2)
for i in range(1,y.shape[0]):
a[i] = (v[i] - v[i-1]) / dt
plt.plot(t, a)
plt.xlabel('Time (s)')
plt.ylabel('Acceleration (m/s^2)')
plt.show()
print(math.sqrt(y[-1,1]**2 + h0**2))

##当风速为逆风时，投放距离的模型求解代码

# 导入库
import numpy as np
import matplotlib.pyplot as plt
import math
# 初始条件
v0 = 300.0/3.6  # 初始速度，单位m/s
h0 = 300.0  # 初始高度，单位m
c = 5.0  # 风速，单位m/s # 物体与水平面的夹角，单位rad
K = 0.5*1.29*np.pi*(0.2**2)/50
cd=0.5 #物资空气阻力系数
cc=0.3 #物资风速阻力系数
g = 9.81  # 重力加速度，单位m/s^2
tmax = 40.0  # 时间上限，单位s
dt = 0.01  # 步长，单位s
# 定义运动方程
def f(t, y, i):
h=y[i,0]
d=y[i,1]
vh=y[i,2]
vd=y[i,3]
if (vd==0):
theta=0.5*np.pi
else:
x=vh/vd
theta = math.atan(x)
if (theta==0):
return[-vh,
vd,
g,
- K / np.cos(theta) * vd**2 - K*cc*c**2]
```

```
 elif (theta==0.5*np.pi):
return[-vh,
vd,
g - K*cd/ np.sin(theta) * vh**2,
-K*cc*c**2]
else:
return [-vh,
vd,
g - K*cd/ np.sin(theta) * vh**2,
- K*cd / np.cos(theta) * vd**2 - K*cc*c**2
]
# 初始化数组
nsteps = int(tmax / dt)
t = np.zeros(nsteps)
y = np.zeros((nsteps, 4))

# 初值
y[0, 0] = h0
y[0, 1] = 0
y[0, 2] = 0
y[0, 3] = v0

# 运动方程求解
for i in range(nsteps - 1):
k1 = np.multiply(np.array(f(t[i], y, i)), dt)
k2 = np.multiply(np.array(f(t[i] + dt / 2, y + k1 / 2, i)), dt)
k3 = np.multiply(np.array(f(t[i] + dt / 2, y + k2 / 2, i)), dt)
k4 = np.multiply(np.array(f(t[i] + dt, y + k3, i)), dt)
if ((y[i] + (k1 + 2*k2 + 2*k3 + k4) / 6)[0]<0):
row_to_delete=range(i+1,nsteps)
y=np.delete(y,row_to_delete,axis=0)
row_to_delete=range(i+1,nsteps)
t=np.delete(t,row_to_delete,axis=0)
break
else:
y[i+1] = y[i] + (k1 + 2*k2 + 2*k3 + k4) / 6
t[i+1] = t[i] + dt
# 绘制轨迹图
plt.plot(y[:, 1], y[:, 0])
plt.xlabel('Distance (m)')
plt.ylabel('Height (m)')
plt.show()

# 绘制速度图
v = np.sqrt(y[:, 2]**2 + y[:, 3]**2)
plt.plot(t, v)
plt.xlabel('Time (s)')
plt.ylabel('Velocity (m/s)')
plt.show()

# 绘制加速度图
a = np.zeros(y.shape[0])
a[0]=-np.sqrt(g**2 + v0**2)
for i in range(1,y.shape[0]):
a[i] = (v[i] - v[i-1]) / dt
```

```
plt.plot(t, a)
plt.xlabel('Time (s)')
plt.ylabel('Acceleration (m/s^2)')
plt.show()
print(math.sqrt(y[-1,1]**2 + h0**2))

##当风速竖直向下时，投放距离的模型求解代码

# 导入库
import numpy as np
import matplotlib.pyplot as plt
import math
# 初始条件
v0 = 300.0/3.6 # 初始速度，单位m/s
h0 = 300.0 # 初始高度，单位m
c = 5.0 # 风速，单位m/s # 物体与水平面的夹角，单位rad
K = 0.5*1.29*np.pi*(0.2**2)/50
cd=0.5 #物资空气阻力系数
cc=0.3 #物资风速阻力系数
g = 9.81 # 重力加速度，单位m/s^2
tmax = 40.0 # 时间上限，单位s
dt = 0.01 # 步长，单位s
# 定义运动方程
def f(t, y, i):
h=y[i,0]
d=y[i,1]
vh=y[i,2]
vd=y[i,3]
if(vd==0):
theta=0.5*np.pi
else:
x=vh/vd
theta = math.atan(x)
if(theta==0):
return[-vh,
vd,
g,
- K / np.cos(theta) * vd**2]
elif(theta==0.5*np.pi):
return[-vh,
vd,
g - K*cd/ np.sin(theta) * vh**2+K*cc*c**2,
0]
else:
return [-vh,
vd,
g - K*cd/ np.sin(theta) * vh**2+K*cc*c**2,
- K*cd / np.cos(theta) * vd**2
]
# 初始化数组
nsteps = int(tmax / dt)
t = np.zeros(nsteps)
y = np.zeros((nsteps, 4))

# 初值
```

```
y[0, 0] = h0
y[0, 1] = 0
y[0, 2] = 0
y[0, 3] = v0

# 运动方程求解
for i in range(nsteps - 1):
k1 = np.multiply(np.array(f(t[i], y, i)), dt)
k2 = np.multiply(np.array(f(t[i] + dt / 2, y + k1 / 2, i)), dt)
k3 = np.multiply(np.array(f(t[i] + dt / 2, y + k2 / 2, i)), dt)
k4 = np.multiply(np.array(f(t[i] + dt, y + k3, i)), dt)
if ((y[i] + (k1 + 2*k2 + 2*k3 + k4) / 6)[0]<0):
row_to_delete=range(i+1,nsteps)
y=np.delete(y,row_to_delete,axis=0)
row_to_delete=range(i+1,nsteps)
t=np.delete(t,row_to_delete,axis=0)
break
else:
y[i+1] = y[i] + (k1 + 2*k2 + 2*k3 + k4) / 6
t[i+1] = t[i] + dt
# 绘制轨迹图
plt.plot(y[:, 1], y[:, 0])
plt.xlabel('Distance (m)')
plt.ylabel('Height (m)')
plt.show()

# 绘制速度图
v = np.sqrt(y[:, 2]**2 + y[:, 3]**2)
plt.plot(t, v)
plt.xlabel('Time (s)')
plt.ylabel('Velocity (m/s)')
plt.show()

# 绘制加速度图
a = np.zeros(y.shape[0])
a[0]=-np.sqrt(g**2 + v0**2)
for i in range(1,y.shape[0]):
a[i] = (v[i] - v[i-1]) / dt
plt.plot(t, a)
plt.xlabel('Time (s)')
plt.ylabel('Acceleration (m/s^2)')
plt.show()
print(math.sqrt(y[-1,1]**2 + h0**2))
```

```
#第二问第一小问的代码

#导入库
import numpy as np
import matplotlib.pyplot as plt
import math

#初始条件
K = 0.5*1.225*np.pi*(0.08**2)/5
cd=0.5 #炸弹的阻力系数
g = 9.81  # 重力加速度，单位m/s^2
```

```
    tmax = 100.0 # 时间上限，单位s
    dt = 0.01  # 步长，单位s

    #利用龙格-库塔法对模型进行求解
    def q(v,t0,a,H,v0,theta):
    #v无人机俯冲速度，t0无人机俯冲时间、a无人机俯冲加速度、H无人机高度、v0发射速度、theta俯冲角度

    # 定义运动方程
    def f1(t, y, i):
    h=y[i,0]
    d=y[i,1]
    vh=y[i,2]
    vd=y[i,3]
     if(vd==0):
    theta=0.5*np.pi
     else:
    x=vh/vd
    theta = math.atan(x)
     if(theta==0):
    return[-vh,
    vd,
    -g,
    - K / np.cos(theta) * vd**2]
     elif(theta==0.5*np.pi):
    return[-vh,
    vd,
    -g + K*cd/ np.sin(theta) * vh**2,
    0]
     else:
    return [-vh,
    vd,
    -g + K*cd/ np.sin(theta) * vh**2,
    - K*cd / np.cos(theta) * vd**2
    ]
    # 初始化数组
    nsteps = int(tmax / dt)
    t = np.zeros(nsteps)
    y = np.zeros((nsteps, 4))
    h=H-(v*t0-0.5*a*t0**2)*np.sin(theta)
    # 初值
    y[0, 0] = h
    y[0, 1] = 0
    y[0, 2] = v0*np.sin(theta)
    y[0, 3] = v0*np.cos(theta)
    # 运动方程求解
    for i in range(nsteps - 1):
    k1 = np.multiply(np.array(f(t[i], y, i)), dt)
    k2 = np.multiply(np.array(f(t[i] + dt / 2, y + k1 / 2, i)), dt)
    k3 = np.multiply(np.array(f(t[i] + dt / 2, y + k2 / 2, i)), dt)
    k4 = np.multiply(np.array(f(t[i] + dt, y + k3, i)), dt)
     if((y[i] + (k1 + 2*k2 + 2*k3 + k4) / 6)[0]<0):
    row_to_delete=range(i+1,nsteps)
    y=np.delete(y,row_to_delete,axis=0)
    row_to_delete=range(i+1,nsteps)
    t=np.delete(t,row_to_delete,axis=0)
```

```
        break
        else:
        y[i+1] = y[i] + (k1 + 2*k2 + 2*k3 + k4) / 6
        t[i+1] = t[i] + dt
        if ((y[-1,1]+(v*t0-0.5*a*t0**2)*np.cos(theta))>10000):
        return 0
        print(math.sqrt(H-(v*t0-0.5*a*t0**2)*np.sin(theta)**2+y[-1,1]**2))

        #问题二第二小问代码

        #导入库

        import numpy as np
        import matplotlib.pyplot as plt
        import math
        import pandas as pd

        #初始条件

        K = 0.5*1.225*np.pi*(0.08**2)/5
        cd=0.5 #炸弹的阻力系数
        cc=0.2 #风速的阻力系数
        g = 9.81 # 重力加速度，单位m/s^2
        c=6 #风速，单位m/s
        tmax = 100.0 # 时间上限，单位s
        dt = 0.01 # 步长，单位s

        #爆炸物在空气阻力和风速影响下的运动方程函数

        def f1(t, y, i):
        h=y[i,0]
        d=y[i,1]
        vh=y[i,2]
        vd=y[i,3]
        if (vd==0):
        theta=0.5*np.pi
        else:
        x=vh/vd
        theta = math.atan(x)
        if (theta==0):
        return[-vh,
        vd,
        -g,
        - K / np.cos(theta) * vd**2]
        elif (theta==0.5*np.pi):
        return[-vh,
        vd,
        g - K*cd/ np.sin(theta) * vh**2,
        - K*cc * c**2]
        else:
        return [-vh,
        vd,
        g - K*cd/ np.sin(theta) * vh**2,
        - K*cd / np.cos(theta) * vd**2- K*cc * c**2
        ]
```

```
122
123         #用龙格-库塔法，对于给定的爆炸物高度h、爆炸物发射速度v0、爆炸物发射角度，求解爆炸物水平距离
124
125         def f2(h,v0,theta):
126         # 初始化数组
127         nsteps = int(tmax / dt)
128         t = np.zeros(nsteps)
129         y = np.zeros((nsteps, 4))
130         y[0, 0] = h
131         y[0, 1] = 0
132         y[0, 2] = v0*np.sin(theta)
133         y[0, 3] = v0*np.cos(theta)
134         for i in range(0,nsteps - 1):
135         k1 = np.multiply(np.array(f1(t[i], y, i)), dt)
136         k2 = np.multiply(np.array(f1(t[i] + dt / 2, y + k1 / 2, i)), dt)
137         k3 = np.multiply(np.array(f1(t[i] + dt / 2, y + k2 / 2, i)), dt)
138         k4 = np.multiply(np.array(f1(t[i] + dt, y + k3, i)), dt)
139         if ((y[i] + (k1 + 2*k2 + 2*k3 + k4) / 6)[0]<0):
140         row_to_delete=range(i+1,nsteps)
141         y=np.delete(y,row_to_delete,axis=0)
142         row_to_delete=range(i+1,nsteps)
143         t=np.delete(t,row_to_delete,axis=0)
144         break
145         else:
146         y[i+1] = y[i] + (k1 + 2*k2 + 2*k3 + k4) / 6
147         t[i+1] = t[i] + dt
148         return y[-1,1]
149
150         #遍历算法的函数
151
152         def q(t0,a):
153         #t0无人机俯冲时间、vm为俯冲末速度
154         v=300.0/3.6 #无人机俯冲速度
155         H=800       #高度
156         v0=600.0/3.6+v+a*t0#发射速度
157         #遍历寻找俯冲角度的值
158         theta = np.zeros(900)
159         for i in range(10,50,1):
160         x=i*np.pi/180.0
161         if ((H-(v*t0+0.5*a*t0**2)*np.sin(x))>=300):
162         theta[i]=x
163         theta=np.delete(theta,theta==0,axis=0)
164         # 初始化数组
165         thetaa = np.zeros(theta.shape[0])
166         reslut =np.zeros((1,3))
167         for j in range(0,theta.shape[0]):
168         x=theta[j]
169         h=H-(v*t0+0.5*a*t0**2)*np.sin(x)
170         l=f2(h,v0,x)
171         d=math.sqrt(h**2+l**2)
172         L=l+(v*t0+0.5*a*t0**2)*np.cos(x)
173         if ((L<=10000.0)&(L>=0.0)):
174         if ((d>=1000.0)&(d<=3000.0)):
175         res=np.array([x*180/np.pi,d,L])
176         reslut =np.vstack((reslut, res))
```

```
    reslut =np.delete( reslut ,0, axis=0)
    reslut  = pd.DataFrame(reslut)
    return  reslut

    #输入数值,导出结果

    res=q(3,20)
    res .to_excel(r"D:\user\桌面\2023.cy.a\q2\结果.xlsx")
```

```
%第二问结果遍历绘图  % 数据
    angle = [10:32];
    distance1 = [1564.603594, 1541.951333, 1518.648707, 1494.696017, 1470.501835, 1445.682
    distance2 = [1712.863391, 1689.188683, 1664.636543, 1639.200136, 1613.339493, 1586.608

    % 三维平面曲线图
    figure
    plot3(angle, distance1, zeros(size(angle)), 'LineWidth', 2, 'Color', 'b')
    hold on
    plot3(angle, distance2, zeros(size(angle)), 'LineWidth', 2, 'Color', 'r')
    grid on
    xlabel('角度')
    ylabel('发射距离')
    zlabel('俯冲时距爆炸点水平距离')
    title ('数据遍历图')
    legend('distance1', 'distance2')

    %% 问题三，三维坐标系
    clc;  close all;
    %% 仿真步长
    dt = 1e-3;
    % 顺风：626.499m
    % 逆风：625.916m
    % 无风：726.189m
    % 无空气阻力也无风：2500m
    %% 基本数据
    Vplane = 300/3.6;
    Vo = 800/3.6; % 物体初速度：900/3.6m/s
    H = 800;
    R = 0.08; % 球体半径：8cm
    M = 5; % 球体重量：5kg
    Vf = 6; % 风速的绝对值：6m/s
    fu1 = 0; % 水平风的角度，0代表顺风（沿y轴），90代表（逆x轴），180代表逆风（逆y轴
    % 物理常量
    g = 9.8; % m/s^2
    ro = 1.29; % 风阻相关
```

```
C = 0.5; % 粘度
%% 公式
S = 2*pi*R^2; % 迎风面积，球体面积的一半
fu = fu1*pi/180; % 风速方向，角度转弧度。对matlab的三角函数需要转换。
Kf = 0.5*ro*C*S; % 风阻系数
%
k = 0;
%% 角度制转换
xita1 = 45;
xita = xita1*pi/180;
%% 坐标系速度计算
% 物体状态，初始值
V = Vo; % 合成速度的绝对值
X = 0; Y = 0; Z = H; % 坐标的绝对值
Vx = 0; Vy = Vo*cos(xita); Vz = -Vo*sin(xita); % 速度的分量
dx = 0; dy = 0; dz = 0; % 坐标的步进量
% 自然风速，相对于坐标轴
Vfx = -Vf*sin(fu); Vfy = Vf*cos(fu); Vfz = 0; % 水平风，无z轴分量
% 物体迎风状态，计算物体相对于风的速度，接着计算风阻
V_fx = Vx - Vfx;
V_fy = Vy - Vfy;
V_fz = Vz - Vfz;
V_f = sqrt(V_fx^2 + V_fy^2 + V_fz^2); % 相对速度的合成，绝对值大小。用于计算风阻
Ff = Kf*(V_f^2); % 风阻力的总大小，绝对值
F_fx = -Ff*sin(fu); % 各方向上的阻力分力
F_fy = -Ff*cos(fu);
F_fz = 0;
%% 运动过程
angle = 45;
Yplane = 0; Hplane = 800;
flag = 0;
result =[];
for H = 800:-1e-6:200
flag = flag+1;
xita = angle*pi/180; % 转为弧度
% 飞机位置更新
yplane = Yplane; hplane = Hplane;
dt2 = 1e-1; % 控制步长设为千分之一秒
dYplane = Vplane*dt2*cos(xita);
dHplane = -Vplane*dt2*sin(xita);
Yplane = yplane+dYplane; Hplane = hplane+dHplane; H = Hplane;
%% 遍历合适的发射角
for k = 1:1:89
```

```
%% 开始每一次的假设发射
% 更新发射位置、发射速度
X = 0; Y = Yplane; Z = Hplane;
Vx = 0; Vy = Vo*cos(k); Vz = -Vo*sin(k); % 速度的分量
% 自然风速，相对于坐标轴
Vfx = -Vf*sin(fu); Vfy = Vf*cos(fu); Vfz = 0; % 水平风，无z轴分量
% 物体迎风状态，计算物体相对于风的速度，接着计算风阻
V_fx = Vx - Vfx;
V_fy = Vy - Vfy;
V_fz = Vz - Vfz;
V_f = sqrt(V_fx^2 + V_fy^2 + V_fz^2); % 相对速度的合成，绝对值大小。用于计算风阻
Ff = Kf*(V_f^2); % 风阻力的总大小，绝对值
F_fx = -Ff*sin(fu); % 各方向上的阻力分力
F_fy = -Ff*cos(fu);
F_fz = 0;
%% 开始发射
T = 0; % 记录运动时间。
for t = 0:dt:100  % dt=1e-6作为步长
T = T + dt;
if Z <= 0 % 落地时刻
break
end
% 位置更新并记录上一时刻的值
x = X; y = Y; z = Z; % 记录上一时刻
dx = Vx*dt;    dy = Vy*dt;    dz = Vz*dt; % 位移量更新
ds = sqrt(dx^2 + dy^2 + dz^2); % 位移的绝对值
X = x + dx;     Y = y + dy;     Z = z + dz; % 位置更新
% 物体速度更新并记录上一时刻的值
vx = Vx; vy = Vy; vz = Vz; v = V;
Vx = vx + F_fx/M*dt;  Vy = vy + F_fy/M*dt;    Vz = vz + F_fz/M*dt - g*dt;
V = sqrt(Vx^2 + Vy^2 + Vz^2); % 物体速度的绝对值
% 迎风速度更新，物体相对于风的各方向相对速度
V_fx = Vx - Vfx ; V_fy = Vy - Vfy; V_fz = Vz - Vfz;
V_f = sqrt(V_fx^2 + V_fy^2 + V_fz^2); % 总相对速度合成，绝对值大小
% 风阻力的更新
Ff = Kf*(V_f^2); % 风阻力的绝对值
F_fx = -Ff*dx/ds; F_fy = -Ff*dy/ds; F_fz = -Ff*dz/ds; % 各方向分力，阻力，大小更新
% 物体下落，dz为负，Ffz为正。
end
if abs(Y-911.881210)<10 & H>=300
angle = k;
fprintf('---------我爱cqupt---------\n');
fprintf('精度 = %d \n', flag); %精度输出
```

```
                    fprintf("无人机高度 = %.1f \n", H); %无人机高度
                    fprintf("Yplane（速度） = %.1f \n", Yplane); %无人机速度
                    fprintf("俯冲角度 = %.1f \n", angle);
                    result = cat(1, result, [flag, H, Yplane, angle]); %策略存储矩阵
                    break
                    end
                    end

                    end


第三问遍历结果可视化
% 定义数据
                    data = [1,794.107443490112,5.89255650988790,12;
                    2,792.374846066631,14.0437865160029,44;
                    3,786.586026312806,20.0382848521584,88;
                    4,778.257769420980,20.3291139913459,88;
                    5,769.929512529154,20.6199431305334,88;
                    6,761.601255637328,20.9107722697209,88;
                    7,753.272998745502,21.2016014089084,44;
                    8,747.484178991677,27.1960997450638,88;
                    9,739.155922099852,27.4869288842513,88;
                    10,730.827665208026,27.7777580234389,88;
                    11,722.499408316200,28.0685871626264,88;
                    12,714.171151424374,28.3594163018139,44;
                    13,708.382331670549,34.3539146379693,88;
                    14,700.054074778723,34.6447437771568,88;
                    15,691.725817886897,34.9355729163443,88;
                    16,683.397560995071,35.2264020555318,88;
                    17,675.069304103246,35.5172311947193,44;
                    18,669.280484349421,41.5117295308748,81;
                    19,661.049748177794,42.8153500728767,81;
                    20,652.819012006168,44.1189706148786,81;
                    21,644.588275834542,45.4225911568805,81;
                    22,636.357539662916,46.7262116988825,81;
                    23,628.126803491290,48.0298322408844,81;
                    24,619.896067319664,49.3334527828863,81;
                    25,611.665331148037,50.6370733248882,81;
                    26,603.434594976411,51.9406938668901,81;
                    27,595.203858804785,53.2443144088921,37;
                    28,590.188733611851,59.8996103259528,37;
                    29,585.173608418918,66.5549062430136,37;
                    30,580.158483225984,73.2102021600744,37;
```

```
34,560.097982454249,99.8313858283175,19;
35,557.384914500439,107.710707291645,19;
36,554.671846546630,115.590028754973,19;
37,551.958778592820,123.469350218300,63;
38,544.533724224584,127.252604382797,62;
39,537.175827617426,131.164867406012,62;
40,529.817931010268,135.077130429228,62;
41,522.460034403110,138.989393452444,18;
42,519.884892783319,146.914864421570,18;
43,517.309751163528,154.840335390696,18;
49,501.858901444781,202.393161205454,38;
50,496.728389150400,208.959917485510,38;
51,491.597876856020,215.526673765566,38;
52,486.467364561639,222.093430045622,38;
53,481.336852267259,228.660186325678,38;
54,476.206339972878,235.226942605734,43;
55,470.523020305691,241.321556785894,43;
56,464.839700638503,247.416170966054,43;
57,459.156380971316,253.510785146214,43;
74,362.539946629128,357.119226208929,68;
75,354.813414507738,360.240947820728,68;
76,347.086882386348,363.362669432528,68;
77,339.360350264958,366.484391044327,68;
78,331.633818143569,369.606112656126,68;
79,323.907286022179,372.727834267925,68;
80,316.180753900789,375.849555879725,24;
81,312.791281875157,383.462434693413,24;
82,309.401809849526,391.075313507101,24;
];

% 绘制三维图
figure;
scatter3(data(:,1), data(:,2), data(:,3), 20, data(:,4), 'filled');
xlabel('精度');
ylabel('H');
zlabel('Speed');
title('可视化遍历数据');
colormap(jet);
colorbar;
```