参赛密码 _____

（由组委会填写）

第十二届“中关村青联杯”全国研究生数学建模竞赛

学　　校		后勤工程学院
参赛队号		90025016
队员姓名	1.	张柱柱
	2.	邓　群
	3.	袁广强

参赛密码 _____

（由组委会填写）




第十二届“中关村青联杯”全国研究生数学建模竞赛

题 目 水面舰艇编队防空和信息化战争评估模型

摘 要

本文以最优化理论为基础，针对水面舰艇编队防空问题，分别建立了舰艇编队优化模型、来袭导弹运动轨迹模型、防空导弹拦截模型和基于 Multi-Agent 的火力分配模型，并通过 Matlab 进行建模与仿真，给出了问题一、问题二与问题三的方案和计算结果；运用模糊数学理论，针对可疑空中目标意图识别问题，建立了模糊识别模型，通过蒙特·卡罗方法寻找到较优的模型参数，在对模型可靠性进行验证的基础上，给出了问题四的识别结果；在兰彻斯特方程基础上，引入战场感知系数和信息优势系数，建立了战略级信息化战争评估模型，并通过海湾战争的数据，对模型进行了验证。

针对问题一，以使来袭导弹被发现时刻的位置与指挥舰之间的最短距离最长为目标，在以我指挥舰为原点的 20° 至 220° 扇面内，以护卫舰警戒幕内环与扇

面边界相切、护卫舰之间警戒幕内环相切等为约束条件，构建优化模型。解得编队的最佳队形为：**以指挥舰为原点，4 艘护卫舰相对于指挥舰的方位和距离分别为{45°, 47.324km}、{95°, 47.324km}、{145°, 47.324km}和{195°, 47.324km}。**

针对问题二，在考虑编队航速情况下，以指挥舰为参考系，建立来袭导弹运动轨迹模型；其次，建立防空导弹拦截模型；再次，在上述两个模型的基础上，构建基于 Multi-Agent 的火力分配模型；最后，通过 Matlab 建立防空仿真模型并运行程序，分析仿真结果，得出结论：编队在仅使用防空导弹防御敌来袭导弹对我指挥舰进行饱和攻击时，**在最危险的方向上，最多能够拦截 8 批来袭导弹。**

针对问题三，在修改仿真模型控制参数的基础上，使用问题二所述诸模型对问题三进行求解，并得出结论：与问题二相比，编队在仅使用防空导弹防御敌来袭导弹对我指挥舰进行饱和攻击时，如果能够得到空中预警机的信息支援，则**在最危险的方向上，最多能够拦截 18 批来袭导弹，指挥舰抗饱和攻击能力提高至 2.25 倍。**

针对问题四，首先对数据进行预处理，然后分析已知意图的空中目标样本数据的特征属性，并建立模糊识别模型，而后通过蒙特·卡罗方法寻找较优的模型参数。最后，通过模糊识别模型，对各待识别意图的空中目标样本数据依次进行识别，并得出结论（见表 5-8）。

针对问题五，采用定性与定量相结合的思想，**在兰彻斯特方程基础上，引入战场感知系数和信息优势系数**，建立了战略级信息化战争评估模型，使用 Matlab 进行仿真与计算，并通过海湾战争的数据，对模型进行了验证，计算出了在信息化战争中，信息对于战争双方的影响。

关键词：水面舰艇编队，Multi-Agent 系统，模糊识别模型，战争评估模型

目　录

1. 问题的重述

我海军由1艘导弹驱逐舰和4艘导弹护卫舰组成水面舰艇编队在我南海某开阔海域巡逻，其中导弹驱逐舰为指挥舰，重要性最大。某一时刻t我指挥舰位置位于北纬15度41分7秒，东经112度42分10秒，编队航向200度（以正北为0度，顺时针方向），航速16节（即每小时16海里）。编队各舰上防空导弹型号相同，数量充足，水平最小射程为10千米，最大射程为80千米，高度影响不必考虑（因敌方导弹超低空来袭），平均速度2.4马赫（即音速340米/秒的2.4倍）。编队仅依靠自身雷达对空中目标进行探测，但有数据链，所以编队中任意一艘舰发现目标，其余舰都可以共享信息，并由指挥舰统一指挥各舰进行防御。

以我指挥舰为原点的20度至220度扇面内，等可能的有导弹来袭。来袭导弹的飞行速度0.9马赫，射程230千米，航程近似为直线，一般在离目标30千米时来袭导弹启动末制导雷达，其探测距离为30千米，搜索扇面为30度（即来袭导弹飞行方向向左和向右各15度的扇面内，若指挥舰在扇形内，则认为来袭导弹自动捕捉的目标就是指挥舰），且具有“二次捕捉”能力（即第一个目标丢失后可继续向前飞行，假设来袭导弹接近舰艇时受到电子干扰丢失目标的概率为85%，并搜索和攻击下一个目标，“二次捕捉”的范围是从第一个目标估计位置算起，向前飞行10千米，若仍然没有找到目标，则自动坠海）。每批来袭导弹的数量小于等于4枚。

由于来袭导弹一般采用超低空飞行和地球曲率的原因，各舰发现来袭导弹的随机变量都服从均匀分布，均匀分布的范围是导弹与该舰之间距离在20~30千米。可以根据发现来袭导弹时的航向航速推算其不同时刻的位置，故不考虑雷达发现目标后可能的目标“丢失”。编队发现来袭导弹时由指挥舰统一指挥编队内任一舰发射防空导弹进行拦截，进行拦截的准备时间（含发射）均为7秒，拦截的路径为最快相遇。各舰在一次拦截任务中，不能接受对另一批来袭导弹的拦截任务，只有在本次拦截任务完成后，才可以执行下一个拦截任务。指挥舰对拦截任务的分配原则是，对每批来袭导弹只使用一艘舰进行拦截，且无论该次拦截成功与否，不对该批来袭导弹进行第二次拦截。不考虑每次拦截使用的防空导弹数量。

本文根据题目要求，针对水面舰艇编队防空问题，依次建立了水面舰艇编队优化模型、来袭导弹运动轨迹模型、防空导弹拦截模型和基于Multi-Agent的火力分配模型，并通过Matlab进行仿真与计算，给出了水面舰艇编队的最佳编队方案，分别求得在有无预警机提供信息支援情况下，水面舰艇编队仅依靠防空导弹拦截来袭导弹条件下的抗饱和攻击能力；运用模糊数学理论，针对可疑空中目标意图识别问题，建立了模糊识别模型，通过蒙特·卡罗方法寻找到较优的模型参数，在对模型可靠性进行验证的基础上，对可疑空中目标意图进行识别；在兰彻斯特方程基础上，引入描述信息化特征的战场感知系数和信息优势系数，建立了战略级信息化战争评估模型，并通过海湾战争的数据，对模型进行了验证。

2. 模型假设

1. 认为指挥舰最重要，只考虑来袭导弹对指挥舰的攻击。

2. 不考虑防空导弹、预警雷达等武器装备失效的情况。

3. 认为各舰发现来袭导弹在 20~30km 服从均匀分布，即警戒幕内环为 20km，警戒幕外环为 30km，且警戒幕内环以内一定可以发现来袭导弹，警戒幕外环以外一定不能发现来袭导弹。

4. 认为指挥舰也具有防空能力，且其防空战技指标与护卫舰相同。

5. 认为指挥舰的抗饱和攻击能力是指在一个方向上的抗饱和攻击能力。

6. 认为防空导弹与来袭导弹相遇，此次拦截任务才算完成，才可以准备下一个拦截任务。

7. 认为舰艇执行一次拦截任务的时间，包括拦截准备时间（7s，含发射）和防空导弹飞行时间。

8. 若空中目标发生转向，则将其航迹在转点处进行分段，并分别作为独立的样本识别其意图。

3. 符号说明

符号	定义
l_1	表示指挥舰可能受攻击扇面 20° 边界线
l_2	表示指挥舰可能受攻击扇面 220° 边界线
r	表示护卫舰警戒幕内环为 $20\,km$
R	表示护卫舰警戒幕外环为 $30\,km$
A,B,C,D	分别表示 4 艘护卫舰
Q_1	表示护卫舰 A 警戒幕内环与 l_1 切点
Q_2	表示护卫舰 A 警戒幕内环与表示护卫舰 B 警戒幕内环切点
Q_3	表示护卫舰 B 警戒幕内环与表示护卫舰 C 警戒幕内环切点
Q_4	表示护卫舰 C 警戒幕内环与表示护卫舰 D 警戒幕内环切点
Q_5	表示护卫舰 D 警戒幕内环与 l_2 切点
a	表示 Q_1 到指挥舰 O 的距离
b	表示 Q_3 到指挥舰 O 的距离
c	表示 Q_2 到指挥舰 O 的距离
v_0	表示舰艇编队航行的速度（16 节）
v_1	表示来袭导弹的速度（0.9 马赫）
v_2	表示舰上防空导弹的速度（2.4 马赫）
X	表示待识别意图的空中目标特征属性数据集
x_i	表示待识别意图的空中目标第 i 项特征属性数据的值
U	表示模糊识别模型的指标集
V	表示模糊识别模型的评价集
$\mu(x)$	表示隶属度函数
α	表示隶属度函数参数向量
W	表示模糊评价矩阵
u_r	表示红方的战场感知系数（$0<u_r<1$）
u_b	表示蓝方的战场感知系数（$0<u_b<1$）

4. 问题的分析

4.1 问题一的分析

以指挥舰为原点（正北为 0°，顺时针方向，下同）的 20°到 220°扇面内，等可能的有导弹来袭。为保护好指挥舰，使其尽可能免遭敌导弹攻击。应尽可能提高防御纵深，实现分层防御，早期发现目标[1]，使指挥舰位于内侧，护卫舰位于外侧以拱卫指挥舰。护卫舰发现来袭导弹服从均匀分布，均匀分布的范围在 20~30km，即护卫舰警戒幕内环为 20km，警戒幕外环为 30km，且警戒幕内环以内一定可以发现来袭导弹，警戒幕外环以外一定不能发现来袭导弹。为确保能够尽早发现来袭导弹，必须要求护卫舰警戒范围覆盖住 20°到 220°扇面，且各护卫舰警戒幕内环必须相切。

通过上述分析得到一般编队队形，通过进一步分析可知，护卫舰警戒幕内环与 20°、220°扇面边界切点及护卫舰警戒幕内环之间的切点到指挥舰的距离最短。这五个切点就是来袭导弹在被发现时刻可能距离指挥舰最短的位置。因此，本文**以使来袭导弹在被发现时刻的位置与指挥舰的最短距离最长为目标**，以 20°至 220°扇面边界、护卫舰警戒幕内环相切等为约束条件，构建编队优化模型，进而得到水面舰艇编队的最佳队形。

4.2 问题二的分析

问题二以问题一优化模型求得的最佳队形为前提，要求计算出在最危险的方向上，编队的抗饱和攻击能力。根据问题二所述，可认为来袭导弹的目标均为指挥舰，而最危险的方向就是来袭导弹被指挥舰发现最晚和编队火力覆盖最薄弱的方向。通过编队最佳队形分析可知，最危险的方向就是以指挥舰为原点的 20°和 220°方向。由题，编队航向 200°，航速 16 节。考虑到若导弹沿 220°逆向飞来，则来袭导弹与编队相向而行，这种情况比来袭导弹沿 20°同向方向飞来更危险。即对于编队而言，在问题二的条件下，220°为最危险方向。

求解问题二的关键因素是：**来袭导弹的运动轨迹、防空导弹的拦截过程、舰艇编队的火力分配。**因此，需要建立来袭导弹运动轨迹模型、防空导弹拦截模型以及火力分配模型。其中，可以将水面舰艇编队的防空火力，看作是一个 Multi-Agent 系统[2]，进而建立基于 Multi-Agent 的火力分配模型。

4.3 问题三的分析

从来袭导弹到指挥舰的距离和编队火力覆盖范围等角度考虑，对于问题三而言，220°方向仍然是最危险方向。问题三与问题二的区别在于，在问题三中，编队得到了空中预警机的信息支援，这就使得编队在最危险的方向上，发现来袭导弹的位置变为距离指挥舰 200km 处。因此，可以认为除发现来袭导弹的初始位置不同外，**问题三与问题二在求解算法上没有本质差别。**在修改仿真模型控制参

数的基础上，可以使用问题二所述诸模型对问题三进行求解。

4.4 问题四的分析

首先，需要对题目附件 1 中的空中目标样本数据进行预处理，处理无效和错误数据[3]，并按题目中表 2 内容，计算样本方位角、距离、水平速度、航向角等特征属性数据。由于已知意图的空中目标数据的样本只有 15 组，即**可用于学习和训练的样本数量太少，不适合采用基于神经网络或支持向量机的识别模型。**因此，对于问题四，可以根据已知意图的空中目标数据的样本，通过分析不同意图的空中目标的各项特征属性，建立模糊识别模型进行求解。

4.5 问题五的分析

由于经典兰彻斯特方程没有考虑信息对现代战争进程的影响，若直接用兰彻斯特模型来预测信息化战争的结果，必然会出现重大误差。因此，可以采用定性与定量相结合的思想，**在兰彻斯特方程基础上，引入描述信息化特征的战场感知系数和信息优势系数，**建立战略级信息化战争评估模型，使用 Matlab 进行仿真与计算，并通过海湾战争的数据，对模型进行了验证，进而可以计算出在信息化战争中，信息对于战争双方的影响。

5. 模型的建立与求解

5.1 问题一模型的建立与求解

未发现敌方目标时，为应对所有可能的突发事件，应尽可能提高防御纵深，实现分层防御，早期发现目标。队形应使指挥舰位于内侧，护卫舰位于外侧，起拱卫作用。为保证一定能够发现来袭导弹，护卫舰警戒幕内环要覆盖住 20°到 220°扇面。同时为提高来袭导弹被发现的最短距离，则护卫舰警戒幕内环应相切。

首先得到一般编队队形，通过对一般编队队形分析，发现护卫舰警戒幕内环与 20°、220°扇面边界切点及护卫舰警戒幕内环之间的切点到指挥舰的距离最短。这五个切点就是来袭导弹被发现到的距离指挥舰最短的位置。因此，可以以来袭导弹被发现到的距离指挥舰最短的位置最长为目标，以 20°到 220°扇面边界，护卫舰警戒幕内环相切等为约束条件，构建优化模型，进而得到编队最佳队形。

5.1.1 确定优化目标

根据问题一的分析，可得一般编队队形，如图 5-1 所示。

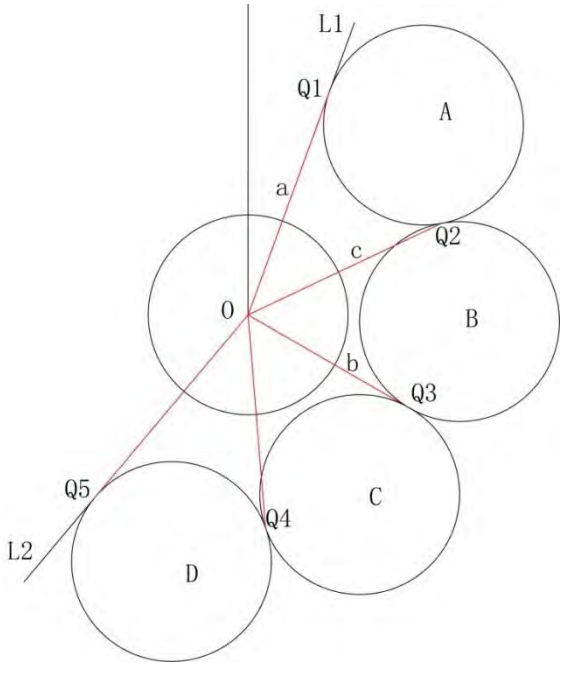

图 5-1 一般编队队形

图 5-1 中原点O为指挥舰所在位置，A,B,C,D为护卫舰的位置，l_1为指挥舰受导弹来袭的 20°边界，l_2为指挥舰受导弹来袭的 220°边界，Q_1是护卫舰A警戒幕内环与l_1切点，Q_2是护卫舰A和护卫舰B警戒幕内环切点，Q_3是护卫舰B和

护卫舰 C 警戒幕内环切点， Q_4 是护卫舰 C 和护卫舰 D 警戒幕内环切点， Q_5 是护卫舰 A 警戒幕内环与 l_2 切点， a 是切点 Q_1 到指挥舰 O 的距离， b 是切点 Q_3 到指挥舰 O 的距离， c 是切点 Q_2 到指挥舰 O 的距离。

分析图 5-1 可知，各护卫舰警戒幕内环到指挥舰的最短距离就是各个切点到指挥舰的距离，也就是可能的最危险方向。换而言之，**在最危险的方向上，使编队发现来袭导弹的初始位置与指挥舰之间的最短距离最长，即为问题一模型的优化目标。**

5.1.2 水面舰艇编队优化模型

如图 5-1 所示，由于护卫舰对称地分布在 20° 到 220° 的扇面内，因此有 $a=\left|OQ_5\right|$ 和 $c=\left|OQ_4\right|$。同时由几何学原理可知，总有 $c\geq\min\{a,b\}$ 。

令 $P=\min\left\{a,b\right\}$，则如图 5-2 所示，在 a 取得最小值与最大值的两种极限情况之间，水面舰艇编队必然存在一种队形，使得 P 值最大，则该队形即为水面舰艇编队的最优队形。

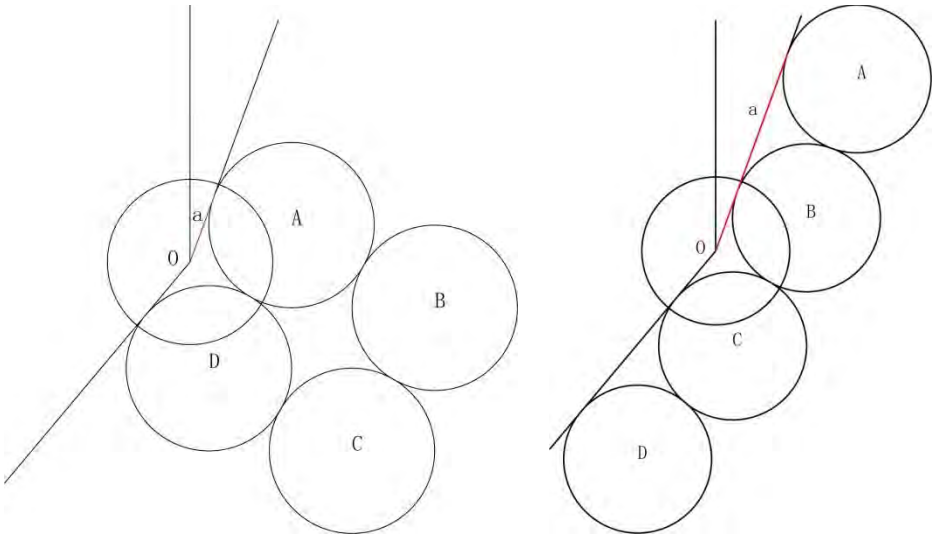

图 5-2 a 取最值的编队队形图（左： a 值最小；右： a 值最大）

根据图 5-2 得到

$$\begin{cases} a_{\min}=\dfrac{20}{\tan 50^\circ}=16.782 \\ a_{\max}=\dfrac{20}{\tan 50^\circ}+2\cdot r=56.782 \end{cases} \tag{5-1}$$

为便于计算，将图 5-1 逆时针旋转 120° ，如图 5-3 所示。

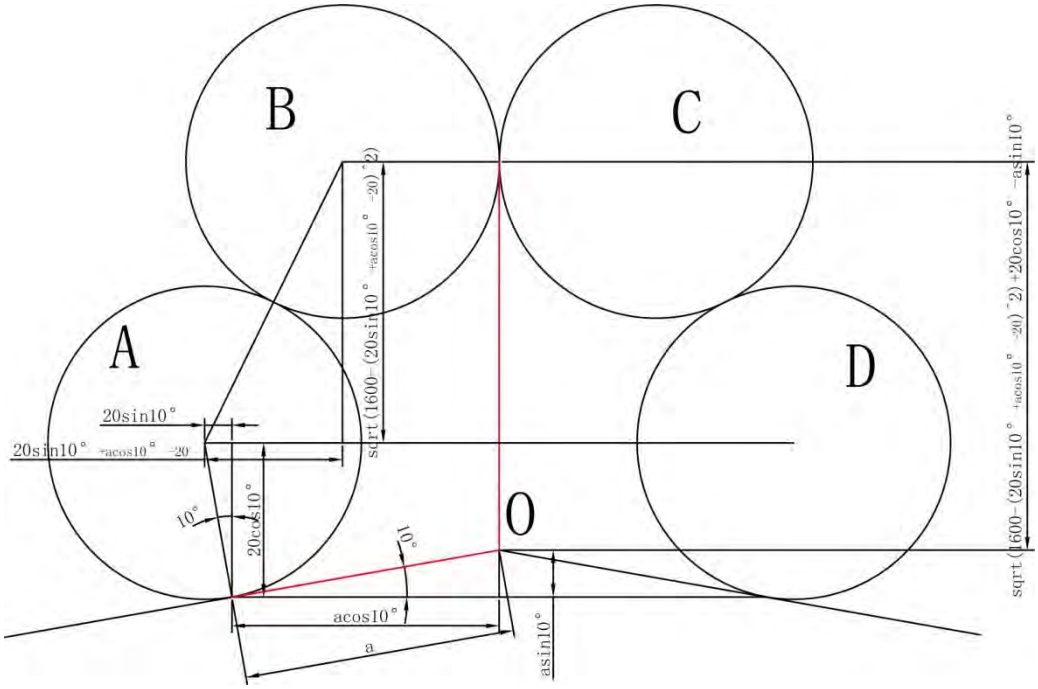

图 5-3 逆时针旋转的一般编队队形

根据图 5-3 中所示的几何关系，可得

$$b=\sqrt{40^2-(a\cdot\cos 10°+20\cdot\sin 10°-20)^2}+20\cdot\cos 10°-a\cdot\sin 10° \quad (5\text{-}2)$$

化简得

$$b=\sqrt{-0.9702a^2+32.5444a+1327.0896}+19.7-0.174a \quad (5\text{-}3)$$

综上所述，将水面舰艇编队优化模型定义如下：

目标函数：

$$\max\{P\} \quad (5\text{-}4)$$

约束条件 *s.t.*：

$$16.782<a<56.782 \quad (5\text{-}5)$$

5.1.3 问题一模型的求解

根据目标函数及约束条件，利用 Matlab 编程求解得出不同 a 值时的目标函数值，结果如图 5-4 所示。（相关 Matlab 源代码，见附录 1）

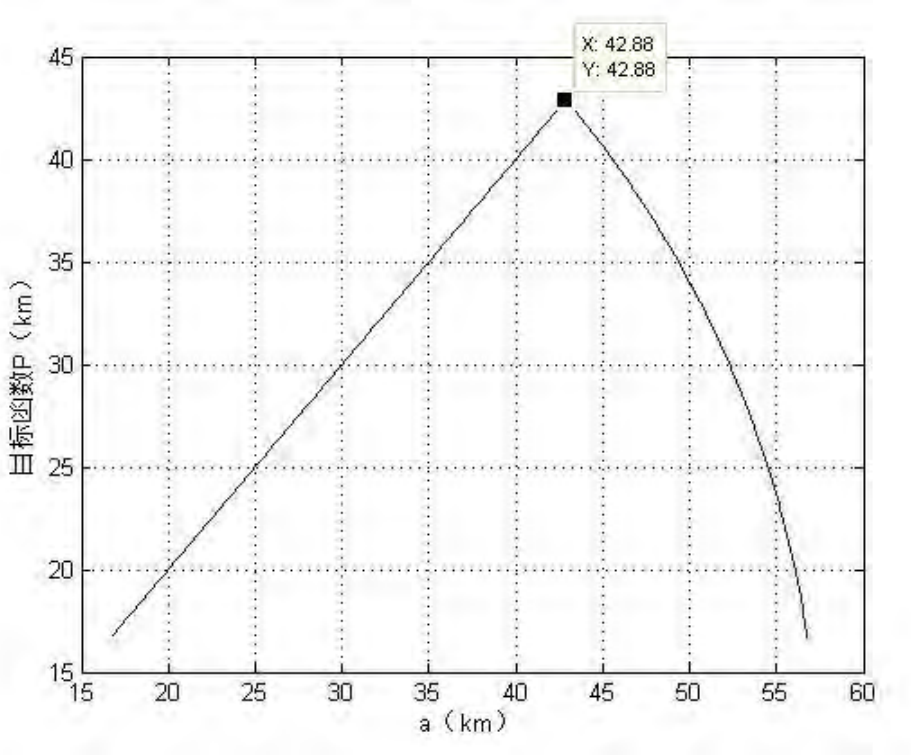

图 5-4 不同 a 值时的目标函数值

如图5-4所示，通过计算可得，当 $a=42.878$ 时，目标函数 P 值最大为42.878，此时 $P_{\max}=a=b=42.878$。此时，编队最佳队形如图 5-5 所示。

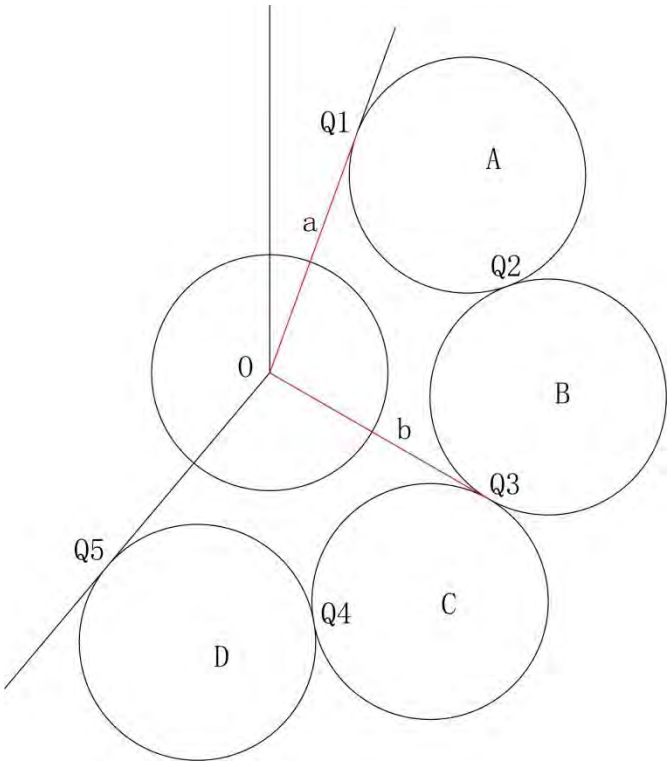

图 5-5 水面舰艇编队最优队形的示意图

如图 5-5 所示，水面舰艇编队的最优队形为：**指挥舰位于原点，各护卫舰相对指挥舰的方位和距离分别为**{45°，47.324km}、{95°，47.324km}、{145°，47.324km}和{195°，47.324km}。

5.2 问题二模型的建立与求解

求解问题二的关键因素是：**来袭导弹的运动轨迹、防空导弹的拦截过程、舰艇编队的火力分配。**因此，需要建立来袭导弹运动轨迹模型、防空导弹拦截模型以及火力分配模型。其中，可以将水面舰艇编队的防空火力，看作是一个 Multi-Agent 系统，进而建立基于 Multi-Agent 的火力分配模型。

5.2.1 来袭导弹运动轨迹模型

通过编队最佳队形分析可知，最危险的方向就是以指挥舰为原点的 20°和 220°方向。由题，编队航向 200°，航速 16 节。考虑到若导弹沿 220°逆向飞来，则来袭导弹与编队相向而行，这种情况比来袭导弹沿 20°同向方向飞来更危险。即对于编队而言，在问题二的条件下，220°为最危险方向，如图 5-6 所示。

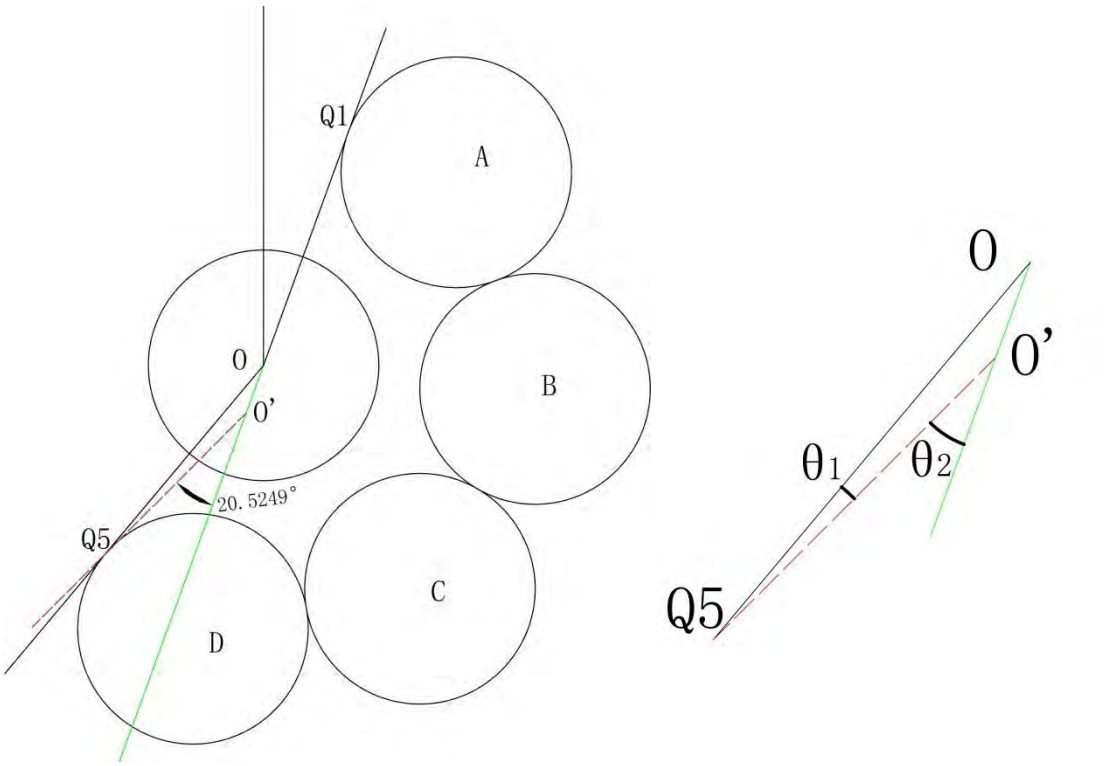

图 5-6 来袭导弹运动轨迹示意图（左：全局；右：局部）

如图 5-6 所示，来袭导弹在Q_5点被发现，目标为指挥舰航向上的某一位置O'。由于指挥舰以 200°方向航行，航速 16 节，若不对来袭导弹进行拦截，则会在O'点与之相遇。

Q_5点到原点O的距离为 42.878km，设经过t时间，来袭导弹与指挥舰相遇，

由三角形余弦定理可得

$$(v_1 \cdot t)^2 = 42878^2 + (v_0 \cdot t)^2 - 2 \cdot 42878 \cdot v_0 \cdot t \cdot \cos 20° \tag{5-6}$$

由已知

$$v_0 = 8.231 m/s，\ v_1 = 306 m/s \tag{5-7}$$

解式 5-6 得$t_1 = 136.674$，$t_2 = -143.765 < 0$(舍)。说明舰队若不进行拦截，来袭导弹在 136.674s 时，将与指挥舰相遇。

根据计算结果，利用三角形余弦定理可得

$$\theta_1 = 0.5249°，\ \theta_2 = 20.5249° \tag{5-8}$$

以指挥舰为原点，0° 方向为 y 轴正方向，90° 方向为 x 轴正方向，建立平面指标坐标系 xOy。进而，以原点O（即指挥舰）为参考系，即假设指挥舰静止，将来袭导弹相对于地面的运动，转化为相对于原点O的相对运动。

考虑舰队航向 200°，航速 16 节。由于来袭导弹飞行路径近似为直线，且目标为指挥舰，因此认为在以指挥舰为原点的平面直角坐标系中，来袭导弹的飞行路径为一条经过一、三象限和原点的直线，该直线的斜率即为来袭导弹相对于原点O，速度在 y 轴方向与 x 轴方向分量之比，即

$$k = \frac{v_y}{v_x} = \frac{v_1 \cdot \cos(220.5249° - 180°) - v_0 \cdot \cos 220°}{v_1 \cdot \sin(220.5249° - 180°) - v_0 \cdot \sin 220°} = 0.8775 \tag{5-9}$$

进而，得到来袭导弹运动路径所在直线表达式为

$$y = kx = 0.8775x \tag{5-10}$$

根据式 5-10 所示直线表达式可知，在以指挥舰为原点 O 的平面直角坐标系中，来袭导弹运动方向与 y 轴正方向夹角为 228.7342°。虽然参考系的不同，会导致来袭导弹相对于指挥舰方向的改变，但是发现来袭导弹的初始位置与指挥舰所在位置的相对距离不会发生变化，即

$$OQ_5 = OQ_5^{'} \tag{5-11}$$

综上所述，建立来袭导弹运动轨迹模型如图 5-7 所示。

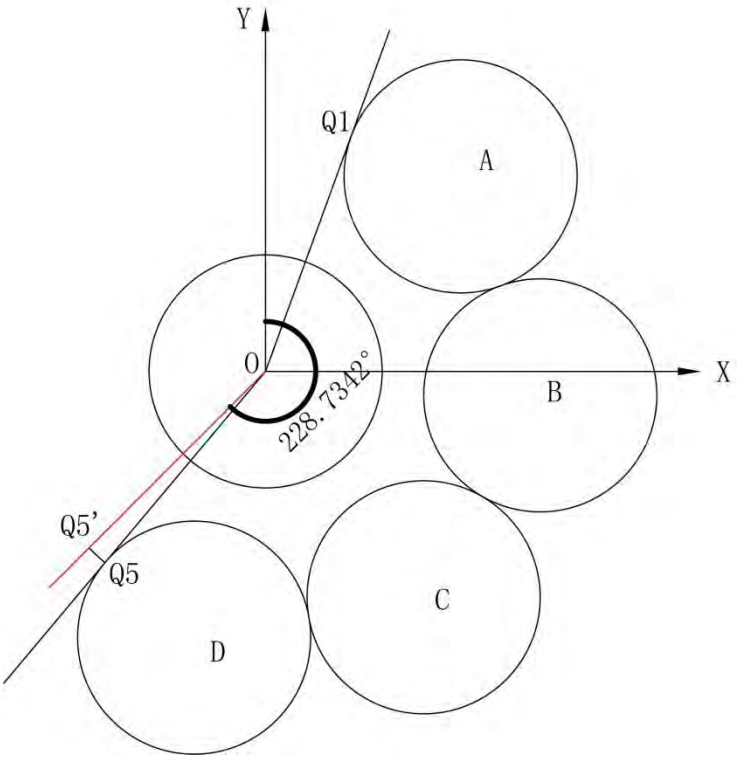

图 5-7 来袭导弹运动轨迹模型示意图

5.2.2 防空导弹拦截模型

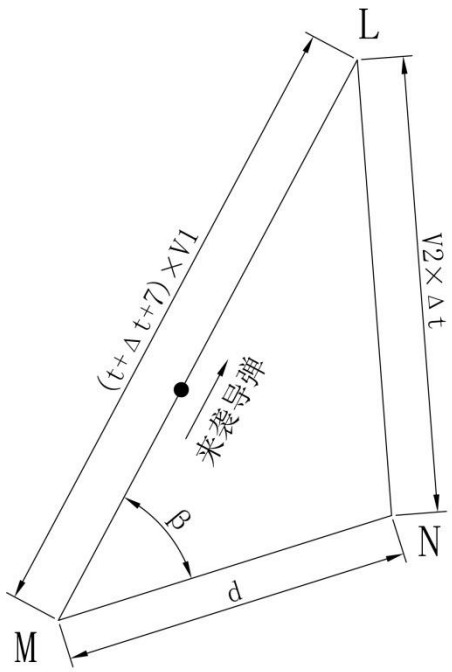

图 5-8 防空导弹拦截模型示意图

如图 5-8 所示，已知在 t 时刻，某舰船所在位置为 N，由舰船发射防空导弹

对来袭导弹进行拦截，发现来袭导弹的初始位置为M，拦截地点为L。来袭导弹速度为v_1，防空导弹速度为v_2，舰船速度忽略不计。舰船进行拦截的准备时间7s，防空导弹飞行时间为Δt，M与N之间距离为d，且$\angle LMN = \beta$。

根据三角形余弦定理，可得

$$\cos\beta = \frac{d^2 + [(t+\Delta t+7)\cdot v_1]^2 - (\Delta t \cdot v_2)^2}{2\cdot d\cdot[(t+\Delta t+7)\cdot v_1]} \tag{5-12}$$

化简得

$$(v_1^2 - v_2^2)\Delta t^2 + 2v_1[(t+7)v_1 - d\cos\beta]\Delta t + [d^2 + (t+7)^2 v_1^2 - 2v_1 d\cos\beta(t+7)] = 0 \tag{5-13}$$

由模型已知条件分析可知，由于防空导弹拦截模型为追击问题且$v_2 > v_1$，因此方程式5-13必有解。根据一元二次方程求根公式，可计算出式5-13所示方程两个解Δt_1和Δt_2。当Δt_1和Δt_2中存在负解时，则舍掉负解，保留正解；若Δt_1和Δt_2同时为正解，根据题述“最快相遇”原则，取二者中的较小解。

最后，利用计算得到的防空导弹飞行时间Δt，对拦截模型进行验证。由题，防空导弹射程为10~80km，即

$$10 \le v_2 \cdot \Delta t \le 80 \tag{5-14}$$

若Δt符合上述条件，则允许该舰船根据上述模型发射防空导弹进行拦截；反之，不进行拦截。

需要说明的是，**防空导弹拦截模型以拦截位置（即防空导弹与来袭导弹相遇位置）与发射防空导弹的舰艇之间距离作为判断依据。**换而言之，即使来袭导弹当前位置在防空导弹射程之外，只要保证拦截位置在防空导弹射程之内，模型即认为舰艇对于来袭导弹是可以进行拦截的。

5.2.3 基于 Multi-Agent 的火力分配模型

在舰艇防空导弹拦截模型的基础上，构建基于Multi-Agent火力分配优化模型。将舰艇抽象为具有适应性的演化Agent，编队构成一个Multi-Agent系统。指挥舰是负责全局任务分配的$Agent_0$，称作管理者（Manager）[4]。

当舰艇编队遭遇单批次来袭导弹攻击时，首先，由$Agent_0$向全部$Agent_i$（$i = 0,1,...,4$）广播任务（$Task$）；然后，$Agent_i$通过防空导弹拦截模型对$Task$进行分析，并将分析结果以投标（Bid）的方式反馈给$Agent_0$；而后，$Agent_0$根据各$Agent_i$的投标，选出并指定最优$Agent_{best}$发射防空导弹对来袭导弹进行拦截。

当舰艇编队遭遇多批次来袭导弹攻击时，将多批次来袭导弹分解成多个单批次拦截任务，利用单批次Multi-Agent火力分配模型，在各Agent相互协同下不

断调整火力分配策略，最终达到拦截来袭导弹批数最大。

综上所述，针对遭遇单批次来袭导弹攻击的情况，建立基于 Multi-Agent 的火力分配模型如下：

假设当前时刻为 t，指挥舰的位置为 $O=(0,0)$，发现来袭导弹的初始位置为 $M_0=(x_{M0},y_{M0})$，来袭导弹当前位置为 $M=(x_M,y_M)$，

Step 1：计算来袭导弹与指挥舰的距离

$$s=\left|\overrightarrow{OM}\right| \tag{5-15}$$

Step 2：由 $Agent_0$ 向全部 $Agent_i$（$i=0,1,...,4$）广播任务

$$Task=\{O,M_0,M,t\} \tag{5-16}$$

Step 3： $Agent_i$ 在接收到 $Task$ 后，根据防空导弹拦截模型，计算各自拦截结果，并向 $Agent_0$ 反馈投标

$$Bid_i=\{flag_i,\Delta t_i\} \tag{5-17}$$

其中，

$$flag_i=\begin{cases}0 & ，无法进行拦截\\ 1 & ，可以进行拦截\end{cases} \tag{5-18}$$

Δt_i 表示，如果 $Agent_i$ 对来袭导弹进行拦截，防空导弹在拦截过程中的飞行时间。

Step 4： $Agent_0$ 在收到各 $Agent_i$ 反馈的 Bid_i 后，根据“最快相遇”原则，寻找火力最优的 $Agent_{best}$，其判断规则如下：

$$flag_i=1 \text{ 且 } \Delta t_i=\min\{\Delta t_i\}（i=0,1,...,4） \tag{5-19}$$

满足式 5-16 条件的 $Agent_i$，即为 $Agent_{best}$。

Step 5： $Agent_0$ 向 $Agent_{best}$ 下达拦截指令，由 $Agent_{best}$ 向来袭导弹发射防空导弹进行拦截。

5.2.4 Matlab 仿真模型及求解

在来袭导弹运动轨迹模型、防空导弹拦截模型和基于 Multi-Agent 的火力分配模型的基础上，通过 Matlab 软件建立仿真模型，对问题二进行求解。仿真模

型如下：

Step 1：设定仿真模型参数，如表 5-1 所示。

表 5-1 问题二仿真模型控制参数

控制变量	含义	初始值	单位
t	仿真时钟	0	s
det	仿真时钟步长	0.1	s
$t_{\max}$	最大仿真时钟	140	s
flag	仿真终止标志位	0	-
beat	拦截批数	0	批
$l_{\min}$	拦截地点与指挥舰之间距离阈值下限	10000	m
l_0	来袭导弹初始位置与指挥舰的距离	42878	m

注：$t=0$表示来袭导弹被编队发现的时刻；$t_{\max}=\dfrac{l_0}{v_1}$

Step 2：建立来袭导弹运动轨迹模型。

Step 3：针对当前 t 时刻来袭导弹信息和舰艇编队状态，建立防空导弹拦截模型和基于 Multi-Agent 的火力分配模型。若有舰艇可执行拦截任务，则向该舰艇下达拦截指令，且 *beat* 值增加 1，并进行 Step 4；反之，则跳转至 Step 5。

Step 4：对执行拦截指令的 $Agent_i$ 进行约束，即在接下来的 $(7+\Delta t_i)$ 时间段内，$Agent_i$ 不能被分配火力拦截任务；

Step 5：若仿真时钟 $t<t_{\max}$，则 t 的值增加 det，并重复 Step 3；反之，则进行 Step 6。

Step 6：仿真终止标志位 *flag* 置 1，仿真结束，输出仿真结果。

运行 Matlab 仿真模型，对问题二进行求解，得到仿真结果如表 5-2 所示。（相关 Matlab 源代码，见附录 2）

表 5-2 问题二仿真结果

t	来袭导弹与指挥舰的距离（km）	仿真模型状态	*beat*
0	42.878	指挥舰准备发起拦截	1
0	42.878	护卫舰 A 准备发起拦截	2
0	42.878	护卫舰 B 准备发起拦截	3
0	42.878	护卫舰 C 准备发起拦截	4
0	42.878	护卫舰 D 准备发起拦截	5
41.6	29.8508	护卫舰 D 拦截完成	5

41.6	29.8508	护卫舰 D 准备发起拦截	6
43.4	29.2872	指挥舰拦截完成	6
43.4	29.2872	指挥舰准备发起拦截	7
68.6	21.3968	护卫舰 C 拦截完成	7
74.9	19.4244	指挥舰拦截完成	7
74.9	19.4244	指挥舰准备发起拦截	7
82.0	17.2018	护卫舰 B 拦截完成	8
85.5	16.1063	护卫舰 A 拦截完成	8
93.3	13.6652	护卫舰 D 拦截完成	8
97.8	12.2571	指挥舰拦截完成	8

从仿真结果可以看出，当不考虑使用电子干扰和近程火炮（包括密集阵火炮）等拦截手段，仅使用防空导弹拦截来袭导弹，上述水面舰艇编队防御敌来袭导弹对我指挥舰进行饱和攻击时，**在最危险的方向上，编队最多能够拦截 8 批来袭导弹。**

5.3 问题三模型的建立与求解

如 5.2 所述，求解问题二时主要考虑的关键因素是：来袭导弹的运动轨迹、防空导弹的拦截过程、舰艇编队的火力分配。如图 5-9 所示，与问题二相比，在问题三中，编队得到了空中预警机的信息支援，这就使得编队在最危险的方向上，发现来袭导弹的位置变为距离指挥舰 200km 处。但是，从火力覆盖角度考虑，220° 方向仍然是最危险方向。因此，可以认为除发现来袭导弹的初始位置不同外，问题三与问题二在求解算法上没有本质差别。在修改仿真模型控制参数的基础上，可以使用问题二所述诸模型对问题三进行求解。

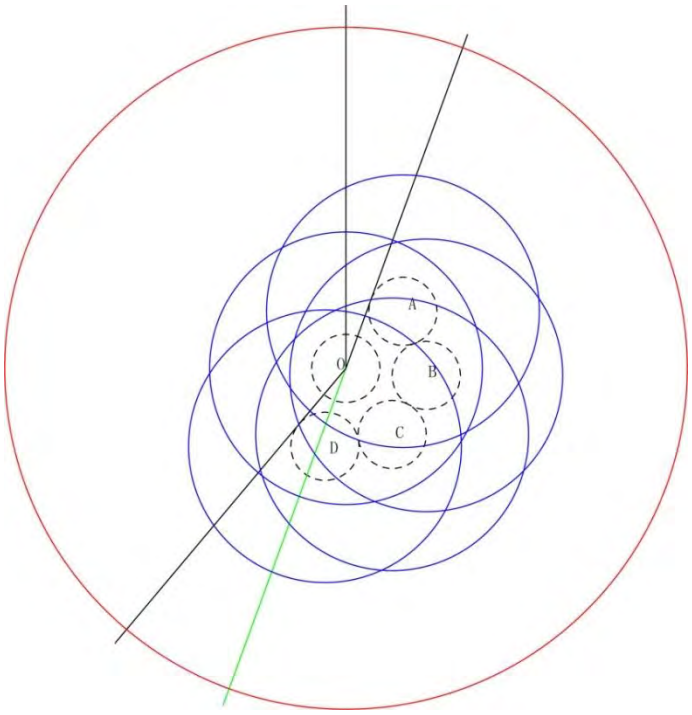

图 5-9 水面舰艇编队防空预警与火力覆盖范围示意图

Matlab 仿真模型如下：

Step 1：设定仿真模型参数，如表 5-3 所示。

表 5-3 问题三仿真模型控制参数

控制变量	含义	初始值	单位
t	仿真时钟	0	s
det	仿真时钟步长	0.1	s
$t_{\max}$	最大仿真时钟	654	s
flag	仿真终止标志位	0	-
beat	拦截批数	0	批
$l_{\min}$	拦截地点与指挥舰之间距离阈值下限	10000	m
l_0	来袭导弹初始位置与指挥舰的距离	200000	m

注：$t=0$表示来袭导弹被编队发现的时刻；$t_{\max}=\dfrac{l_0}{v_1}$

Step 2：建立来袭导弹运动轨迹模型。

Step 3：针对当前 t 时刻来袭导弹信息和舰艇编队状态，建立防空导弹拦截模型和基于 Multi-Agent 的火力分配模型。若有舰艇可执行拦截任务，则向该舰艇下达拦截指令，且 *beat* 值增加 1，并进行 Step 4；反之，则跳转至 Step 5。

Step 4：对执行拦截指令的 $Agent_i$ 进行约束，即在接下来的 $(7+\Delta t_i)$ 时间段内，$Agent_i$ 不能被分配火力拦截任务；

Step 5：若仿真时钟 $t<t_{\max}$，则 t 的值增加 det，并重复 Step 3；反之，则进行 Step 6。

Step 6：仿真终止标志位 *flag* 置 1，仿真结束，输出仿真结果。

运行 Matlab 仿真模型，对问题三进行求解，得到仿真结果如表 5-4 所示。（相关 Matlab 源代码，见附录 3）

表 5-4 问题三仿真结果

t	来袭导弹与指挥舰的距离（km）	仿真模型状态	*beat*
174.3	145.417	护卫舰 D 准备发起拦截	1
279.4	112.5044	护卫舰 D 拦截完成	1
279.4	112.5044	护卫舰 D 准备发起拦截	2
287.2	110.0618	指挥舰准备发起拦截	3
321.8	99.2266	护卫舰 C 准备发起拦截	4
357.5	88.047	护卫舰 D 拦截完成	4

357.5	88.047	护卫舰 D 准备发起拦截	5
392.3	77.1492	指挥舰拦截完成	5
392.3	77.1492	指挥舰准备发起拦截	6
416.9	69.4456	护卫舰 D 拦截完成	6
416.9	69.4456	护卫舰 D 准备发起拦截	7
419.7	68.5687	护卫舰 B 准备发起拦截	8
426.9	66.314	护卫舰 C 拦截完成	8
426.9	66.314	护卫舰 C 准备发起拦截	9
441.6	61.7106	护卫舰 A 准备发起拦截	10
464	54.6959	护卫舰 D 拦截完成	10
464	54.6959	护卫舰 D 准备发起拦截	11
468.7	53.2241	指挥舰拦截完成	11
468.7	53.2241	指挥舰准备发起拦截	12
504.6	41.9818	护卫舰 D 拦截完成	12
504.6	41.9818	护卫舰 D 准备发起拦截	13
509.2	40.5413	护卫舰 C 拦截完成	13
509.2	40.5413	护卫舰 C 准备发起拦截	14
524.3	35.8127	指挥舰拦截完成	14
524.3	35.8127	指挥舰准备发起拦截	15
524.8	35.6561	护卫舰 B 拦截完成	15
524.8	35.6561	护卫舰 B 准备发起拦截	16
545.1	29.299	护卫舰 D 拦截完成	16
545.1	29.299	护卫舰 D 准备发起拦截	17
546.7	28.798	护卫舰 A 拦截完成	17
564.7	23.1612	指挥舰拦截完成	17
564.7	23.1612	指挥舰准备发起拦截	18
578.2	18.9336	护卫舰 C 拦截完成	18
593.5	14.1423	护卫舰 D 拦截完成	18
594.1	13.9544	指挥舰拦截完成	18
604.1	10.8229	护卫舰 B 拦截完成	18

从仿真结果可以看出，与问题二相比，当不考虑使用电子干扰和近程火炮（包括密集阵火炮）等拦截手段，如果能够得到空中预警机的信息支援，水面舰艇编队仅使用防空导弹拦截敌来袭导弹对我指挥舰进行饱和攻击时，**拦截来袭导弹数由 8 批增至 18 批，指挥舰抗饱和攻击能力提高至 2.25 倍，**水面舰艇编队对来袭导弹防御能力明显提高。

5.4 问题四模型的建立与求解

首先，需要对题目附件 1 中的空中目标样本数据进行预处理，处理无效和错误数据，并按题目中表 2 内容，计算样本方位角、距离、水平速度、航向角等特征属性数据。然后，可根据已知意图的空中目标数据的样本，通过分析不同意图的空中目标的各项特征属性，建立模糊识别模型进行求解。

5.4.1 数据预处理

数据预处理就是分析数据前，发现并纠正数据文件中可识别的错误，包括检查数据一致性，处理无效值和缺失值等。

利用 Excel将题目附件 1 中的数据按目标 ID 分离出来，得到 12 批空中目标各自的位置信息。在目标的位置信息中，发现同一目标同一时间有不同的坐标，与实际情况不符。为计算方便，把作战时间相同信息的去掉，只保留其中一组。

发现 ID 号为 41006851 的空中目标，作战时间由 1471428004s 变为 1471428006s 时，高度由 2600m 变到 1200m，跟现实情况不符，但其经度、纬度仍按趋势变化，为保证数据一致性，将高度 1200m 改为 2600m。

题目附件 1 中，只给出了空中目标的时间、纬度、经度和高度信息，没有给出表 2 中的方位角、距离、水平速度、航向角、目标属性信息。

首先，将题目附件 1 中以弧度为单位的经纬度转换为以度为单位的经纬度，其转换方法是：弧度值乘以 180° 除以 II 得到度数值。然后利用 COORD GM 软件将指挥舰经纬度和计算到的经纬度批量转换成大地坐标。大地坐标模型是椭球 WGS84 模型，平面坐标模型是椭球北京 54 模型，中央子午线设为东经 111°。软件界面如图 5-10 所示。

图 5-10 COORD GM 软件界面图

下面以空中目标 41006830 为例，叙述水平速度、航向角、距离、方位角和目标属性的计算过程。

（1）计算水平速度

水平速度是指空中目标在水平面上的速度。根据转换得到的数据，可得空中

目标 41006830 在平面坐标系的散点图如图 5-11 所示。

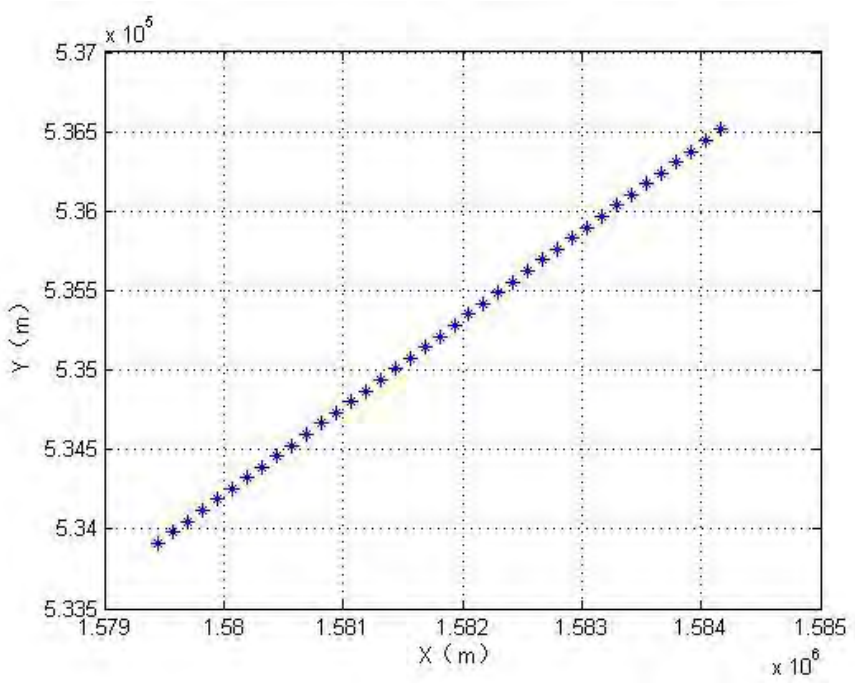

图 5-11 空中目标 41006830 在平面坐标系的散点图

由图 5-11 可知，空中目标 41006830 飞行过程中，航向近似不变。以空中目标飞行的总距离除以总时间，得到平均的水平速度。其中总距离为各时间段内距离之和。

$$\begin{cases} \overline{V} = \dfrac{S_{总}}{T_{总}} \\ S_{总} = \sum S_i \end{cases} \tag{5-20}$$

（2）计算航向角

航向角是指空中目标飞行的方向（正北为 0°，顺时针方向一周分为 360°）。若目标散点图近似为直线，则用 Matlab 线性拟合散点。由于正北方向即为 Y 轴正向，拟合得到的直线与 Y 轴的夹角就是航向角。计算过程中发现存在空中目标轨迹为折线的情况，需要分别计算两段航迹的航向角。

使用 Matlab 软件对空中目标 41006830 的散点图进行线性拟合，结果如图 5-12 所示。

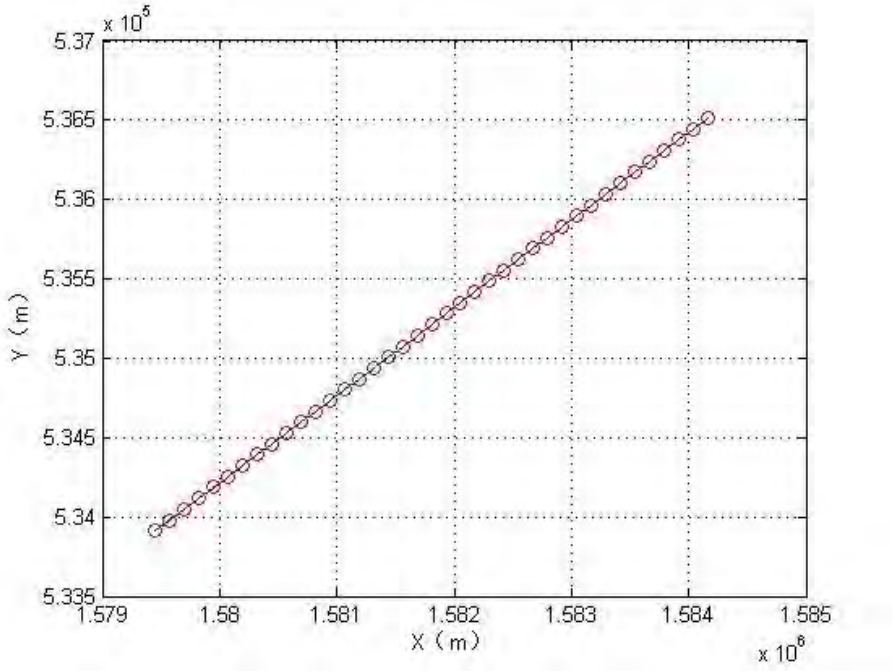

图 5-12 目标 41006830 的线性拟合图

（3）计算距离

距离是指指挥舰位置到空中目标的距离。虽然空中目标处于不断运动中，与指挥舰的距离也在不断变化，但是相应时间段内距离变化量相对于距离很小，可以忽略。所以，将距离定义为目标的平均坐标点到指挥舰的距离。对于轨迹为折线的目标，采用同样的方法分别计算目标在每段轨迹上的距离。

（4）计算方位角

方位角是指从指挥舰到空中目标方向的方位角，正北时为 0mil，顺时针方向一周分为 6400mil。方位角是指从指挥舰。以空中目标的平均坐标点相对于指挥舰的方位角作为空中目标的方位角。已知空中目标的平均坐标和指挥舰坐标，可以得到空中目标相对于指挥舰的角度，经过换算可得到空中目标方位角。对于轨迹为折线的目标，采用同样的方法分别计算目标在每段航迹上的方位角。

（5）计算目标属性

目标属性是根据目标在雷达上反射面积$\sigma(\mathrm{m}^2)$的大小确定的。

$$\begin{cases} 0 \le \sigma < 2, \text{小目标} \\ 2 \le \sigma < 4, \text{中目标} \\ \sigma \ge 4, \text{大目标} \end{cases} \tag{5-21}$$

根据题目表 1 中空中目标的雷达反射面积及式 5-21，可以得到各个目标的属性。

根据上述过程可以计算出未知意图的 12 批空中目标数据，格式与题目中表 2 相同，结果如表 5-5 所示。

表 5-5 未知意图的 12 批空中目标数据

目标 ID	方位角	距离	水平速度	航向角	高度	雷达反射面积	目标属性
	$\beta(mil)$	$D(km)$	$V(m/s)$	$\theta(°)$	$H(km)$	$\sigma(m^2)$	
41006830	4021	213	142	61	7.0	3.5	中目标
41006831	2830	206	269	13	9.2	5.7	大目标
41006836	3622	221	230	10	4.6	1.9	小目标
41006837	2032	220	195	307	5.2	4.3	大目标
41006839	353	232	195	261	5.2	5.5	大目标
41006842	1170	160	266	231	3.4	2.6	中目标
41006860	1907	208	283	170	9.4	6.2	大目标
41006885	4952	205	301	100	1.4	1.1	小目标
41006851	6393	168	301	144	2.6	5.5	大目标
	6	163	301	174	2.6	5.5	大目标
41006872	4738	236	177	99	6.0	1.7	小目标
	4748	235	177	358	6.0	1.7	小目标
41006891	888	229	195	232	4.8	3.6	中目标
	877	227	195	114	4.8	3.6	中目标
41006893	99	243	213	155	8.6	3.1	中目标
	92	241	213	278	8.6	3.1	中目标

注：表 5-5 中 ID 号为 41006851、41006872、41006891 和 41006893 的空中目标，其运动轨迹为折线。因此，将上述四个空中目标的运动轨迹在折点处分为两段，分别作为单独的样本。

5.4.2 模糊识别模型

首先，将已知意图的飞行样本数据归一化，并计算不同意图空中目标的各项特征属性数据的平均值 m_{ij} 与标准差 n_{ij} 。其中，m_{ij} 表示已知意图的空中目标样本数据中，第 i 项特征属性数据对于第 j 项意图的平均值；n_{ij} 表示已知意图的空中目标样本数据中，第 i 项特征属性数据对于第 j 项意图的标准差。

令待识别空中目标 X 的特征属性数据集为

$$X=\{x_i\} \quad (i=1,2,...,6) \tag{5-22}$$

其中，x_i 表示待识别空中目标第 i 项特征属性数据的值。

5.4.2.1 构建指标集

$$U=\{u_i\}\quad (i=1,2,...,6) \tag{5-23}$$

其中，u_1为方位角，u_2为距离，u_3为水平速度，u_4为航向角，u_5为高度，u_6为雷达反射面积。

5.4.2.2 构建评价集

$$V=\{v_j\}\quad (j=1,2,...,5) \tag{5-24}$$

其中，v_1为侦查，v_2为攻击，v_3为掩护，v_4为监视，v_5为其他。

5.4.2.3 构建隶属度函数

设隶属度函数参数向量为

$$\alpha=[\alpha_1,\alpha_2,\alpha_3,\alpha_4,\alpha_5] \tag{5-25}$$

（1）指标$u_i\ (i=1,2,...,5)$对意图v_1的隶属度函数：

$$\mu_{i1}(x_i)=e^{-\frac{\alpha_1\cdot(x_i-m_{i1})^2}{n_{i1}^2}}\quad (i=1,2,...,5) \tag{5-26}$$

（2）指标$u_i\ (i=1,2,...,5)$对意图v_2的隶属度函数：

$$\mu_{i2}(x_i)=e^{-\frac{\alpha_2\cdot(x_i-m_{i2})^2}{n_{i2}^2}}\quad (i=1,2,...,5) \tag{5-27}$$

（3）指标$u_i\ (i=1,2,...,5)$对意图v_3的隶属度函数：

$$\mu_{i3}(x_i)=e^{-\frac{\alpha_3\cdot(x_i-m_{i3})^2}{n_{i3}^2}}\quad (i=1,2,...,5) \tag{5-28}$$

（4）指标$u_i\ (i=1,2,...,5)$对意图v_4的隶属度函数：

$$\mu_{i4}(x_i)=e^{-\frac{\alpha_4\cdot(x_i-m_{i4})^2}{m_{i4}^2}}\quad (i=1,2,...,5) \tag{5-29}$$

（5）指标$u_i\ (i=1,2,...,5)$对意图v_5的隶属度函数：

$$\mu_{i5}(x_i)=e^{-\frac{\alpha_5\cdot(x_i-m_{i5})^2}{n_{i5}^2}}\quad(i=1,2,...,5)\tag{5-30}$$

（6）指标u_6对意图v_j的隶属度函数：

①根据题目附件 A 所述，具有侦查意图的空中目标一般为中、小型机，建立指标u_6对意图v_1的隶属度函数：

$$\mu_{61}(x_6)=\begin{cases}0, & x_6\geq 4\\ 1, & \text{其它}\end{cases}\tag{5-31}$$

②根据题目附件 A 所述，具有攻击意图的空中目标可为大、中、小型机，建立指标u_6对意图v_2的隶属度函数：

$$\mu_{62}(x_6)=1\tag{5-32}$$

③根据题目附件 A 所述，具有掩护意图的空中目标一般为中、小型机，建立指标u_6对意图v_3的隶属度函数：

$$\mu_{63}(x_6)=\begin{cases}0, & x_6\geq 4\\ 1, & \text{其它}\end{cases}\tag{5-33}$$

④根据题目附件 A 所述，具有监视意图的空中目标一般为大、中型机，建立指标u_6对意图v_4的隶属度函数：

$$\mu_{64}(x_6)=\begin{cases}0, & \text{其它}\\ 1, & x_6\geq 2\end{cases}\tag{5-34}$$

⑤根据题目附件 A 所述，具有其它意图的空中目标可为大、中、小型机，建立指标u_6对意图v_5的隶属度函数：

$$\mu_{65}(x_6)=1\tag{5-35}$$

5.4.2.4 模糊评价矩阵

根据待识别空中目标的特征属性数据集 X 和各评价指标的隶属度函数 μ_{ij}，可计算模糊评价矩阵得

$$W=(w_{ji})_{5\times 6}\tag{5-36}$$

其中， w_{ji} 表示指标u_i对意图v_i的隶属度，即

$$w_{ji} = \mu_{ij}(x_i) \tag{5-37}$$

令

$$W_j(x_i) = \frac{1}{6}\sum_{j=1}^{6} W_{ji} \tag{5-38}$$

根据最大隶属度原则，令

$$W_{j\max} = \max\{W_j(x_i)\} \tag{5-39}$$

则$W_{j\max}$对应的v_j即为该待识别空中目标X的意图。

5.4.2.5 可靠性检验

模糊识别模型建立后，首先采用蒙特·卡罗方法，通过计算机进行大量的模拟试算，进而确定相对较优的隶属度函数参数α，并利用已知意图的 15 组空中目标样本数据，对模糊识别模型的可靠性进行验证。

通过蒙特·卡罗方法确定的模糊识别模型隶属度函数参数α的值，如表 5-6 所示。

表 5-6 隶属度函数参数α的值

α	α_1	α_2	α_3	α_4	α_5
参数值	1.8938	0.6209	0.0008	22.8997	2.8169

模糊识别模型可靠性检验结果如表 5-7 所示。

表 5-7 模糊识别模型可靠性的检验结果

样本组数	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
实际意图	1	2	1	2	1	1	2	3	4	5	3	5	5	5	2
识别结果	1	2	1	2	1	1	2	3	4	3	3	5	4	2	4

在表 5-7 中，第 2、3 行中的 1、2、3、4 和 5，依次分别对应“侦查”、“攻击”、“掩护”、“监视”和“其它”。

分析检验结果可知，上述模糊识别模型对问题四中已知意图的空中目标样本数据进行识别时，**准确率可达到 73.33%**，即能够正确识别出大部分空中目标的真实意图，具有一定的可靠性。

5.4.3 问题四模型的求解

通过 5.4.2 中建立模糊识别模型，对问题四中待识别空中目标意图进行分析识别。由于 ID 号为 41006851、41006872、41006891 和 41006893 的空中目标，其运动轨迹为折线，因此将上述四个空中目标的运动轨迹在折点处分为两段，分别作为单独的样本进行意图识别。模型识别结果如表 5-8 所示。（相关 Matlab 源代码，见附录 4）

表 5-8 未知意图的空中目标意图识别结果

目标 ID	目标属性	识别结果	意图
41006830	中目标	3	掩护
41006831	大目标	4	监视
41006836	小目标	5	其他
41006837	大目标	5	其他
41006839	大目标	2	攻击
41006842	中目标	2	攻击
41006860	大目标	5	其他
41006885	小目标	5	其他
41006851（第 1 段）	大目标	5	其他
41006851（第 2 段）	大目标	3	掩护
41006872（第 1 段）	小目标	5	其他
41006872（第 2 段）	小目标	5	其他
41006891（第 1 段）	中目标	4	监视
41006891（第 2 段）	中目标	5	其他
41006893（第 1 段）	中目标	5	其他
41006893（第 2 段）	中目标	2	攻击

注：表 5-8 中 ID 号为 41006851、41006872、41006891 和 41006893 的空中目标，其运动轨迹为折线。因此，将上述四个空中目标的运动轨迹在折点处分为两段，分别作为单独的样本。

5.5 问题五模型的建立与求解

信息化条件下作战对传统的作战评估模型和作战结果已经产生重要的甚至某种程度上是决定性的影响。由于经典兰彻斯特方程没有考虑信息对现代战争进程的影响，若直接采用兰彻斯特方程这种传统的消耗战模型去预测信息化战争的结果，必然会出现重大误差。

文献[5]提出了一种广义兰彻斯特作战模型，该模型**通过将战场感知系数和信息优势系数引入兰彻斯特方程，描述信息对于信息化战争的影响。**本文参考了这种模型，并在此基础上，建立了战略级信息化战争评估模型。

5.5.1 战略级信息化战争评估模型

经典兰彻斯特方程根据两种战术情况分为线性律和平方律两种。

兰彻斯特线性律方程如下

$$\begin{cases} \dfrac{dr}{dt} = -U_l \cdot r \cdot b \\ \dfrac{db}{dt} = -T_l \cdot r \cdot b \end{cases} \tag{5-40}$$

式 5-40 中，r 为红方兵力数量，b 为蓝方兵力数量，U_l 为蓝方作战效能，T_l 为红方作战效能。

兰彻斯特律平方律方程如下

$$\begin{cases} \dfrac{dr}{dt} = -U_s b \\ \dfrac{db}{dt} = -T_s r \end{cases} \tag{5-41}$$

式 5-41 中，r 为红方兵力数量，b 为蓝方兵力数量，T_s、U_s 分别是红、蓝方在单位时间内平均毁伤对方兵力数量。

兰彻斯特线性律和平方律分别描述的是作战双方在完全信息支援和无信息支援条件下的作战效能。将红、蓝双方战场感知系数 u_r 和 u_b 引入线性律和平方律，得到方程如下

$$\begin{cases} \dfrac{dr}{dt} = -u_b \cdot U_e \cdot b - (1-u_b) \cdot U_c \cdot r \cdot b \\ \dfrac{db}{dt} = -u_r \cdot T_e \cdot r - (1-u_r) \cdot T_c \cdot r \cdot b \end{cases} \tag{5-42}$$

式 5-42 中，r 为红方兵力数量，b 为蓝方兵力数量，u_r、u_b 分别是红、蓝方的战场感知系数（$0 < u_r < 1$，$0 < u_b < 1$），U_e 是蓝方对暴露的红方兵力毁伤系数，U_c 是蓝方对隐蔽的红方兵力毁伤系数，T_e 是红方对暴露的蓝方兵力毁伤系数，T_c 是红方对隐蔽的蓝方兵力毁伤系数。

引入信息优势系数 $k = \dfrac{u_r}{u_b}$，作战双方在完全信息支援和无信息支援条件下的毁伤系数可以修正为

$$\begin{cases} U_e = \dfrac{U_s}{k} \\ T_e = T_s \cdot k \\ U_c = \dfrac{U_l}{k^2} \\ T_c = T_l \cdot k^2 \end{cases} \tag{5-43}$$

将式 5-43 代入式 5-42，可得到战略级信息化战争评估模型，即

$$\begin{cases} \dfrac{dr}{dt} = -\dfrac{U_s}{k} \cdot b \cdot u_b - (1-u_b) \cdot \dfrac{U_l}{k^2} \cdot r \cdot b \\ \dfrac{db}{dt} = -k \cdot T_s \cdot r \cdot u_r - (1-u_r) \cdot T_l \cdot r \cdot b \cdot k^2 \end{cases} \tag{5-44}$$

将评估模型写成差分形式，得到

$$\begin{cases} r(t+1) = r(t) - \dfrac{U_s}{k} \cdot b(t) \cdot u_b - (1-u_b) \cdot \dfrac{U_l}{k^2} \cdot r(t) \cdot b(t) \\ b(t+1) = b(t) - k \cdot T_s \cdot r(t) \cdot u_r - (1-u_r) \cdot T_l \cdot r(t) \cdot b(t) \cdot k^2 \end{cases} \tag{5-45}$$

5.5.2 战略级信息化战争评估模型的验证

根据题目附件 B 海湾战争相关资料，得到双方交战兵力数据，其中伊军总兵力 120 万，美国及多国部队总兵力 76 万。战争中美国及多国部队充分利用其在信息系统、指挥对抗、系统稳定性上的优势，掌握了制信息权，从而拥有了战争主动权。

为了检验模型中信息化条件对于战争的影响程度，分别测试两组感知系数，使用 Matlab 软件对模型进行仿真，并对结果进行比较。（相关 Matlab 源代码，见附录 5）

（1）美国及多国部队感知系数 $u_r = 0.2$，伊军感知系数 $u_b = 0.1$，$k = 2$。美国及多国部队兵力数量 $r = 760000$，伊军兵力数量 $b = 1200000$。$T_l = U_l = 0.0000001$，$T_s = U_s = 0.1$，仿真结果如图 5-13 所示。

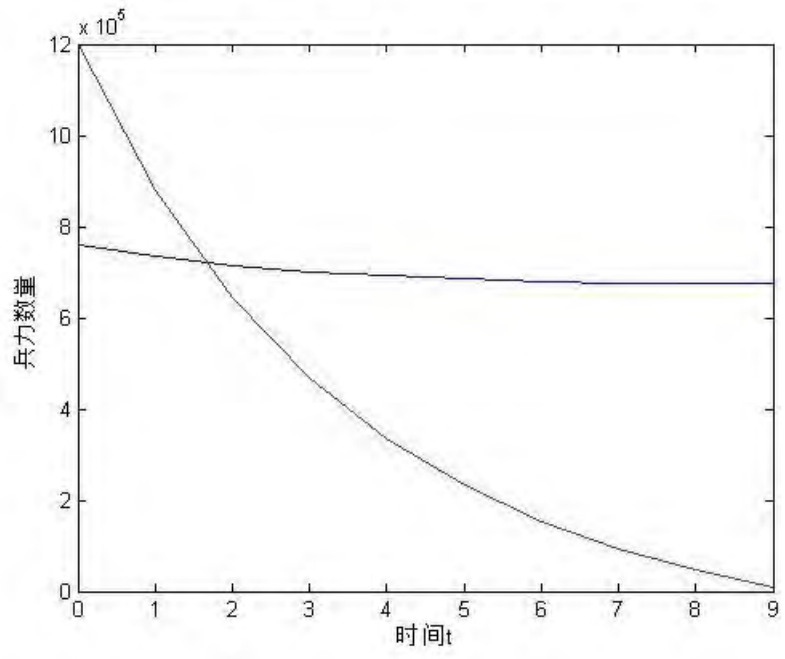

图 5-13 $k=2$ 仿真结果图

（2）美国及多国部队感知系数 $u_r=0.3$，伊军感知系数 $u_b=0.1$，$k=3$。美国及多国部队兵力数量 $r=760000$，伊军兵力数量 $b=1200000$。$T_l=U_l=0.0000001$，$T_s=U_s=0.1$，仿真结果如图 5-14 所示。

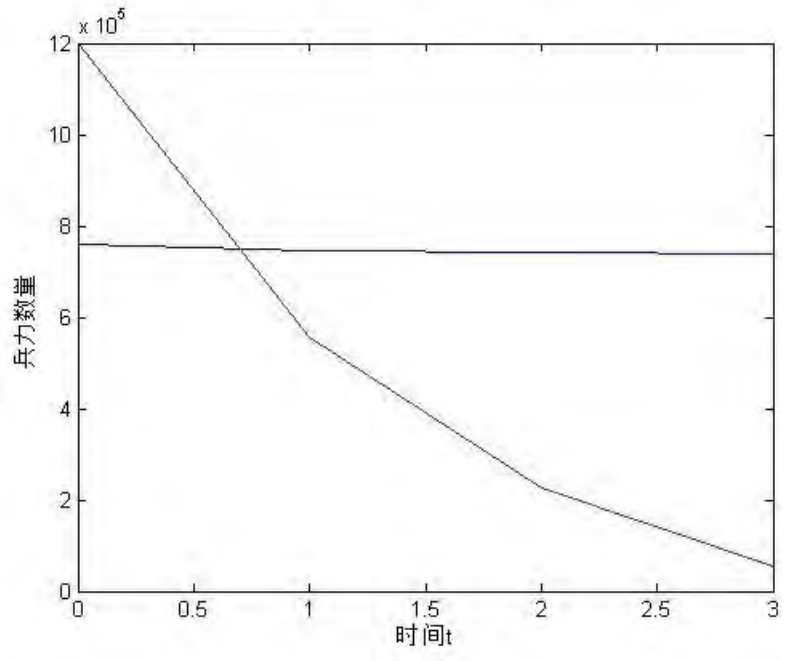

图 5-14 $k=3$ 仿真结果图

对比分析图 5-13 和图 5-14 可知，当 $k=2$ 时，作战时间大约为 9 个时间单位，美国及多国部队兵力损失 86200 个单位；当 $k=3$ 时，作战时间大约为 3 个时间

单位，美国及多国部队兵力损失 21583 个单位。信息优势从 $k=2$ 提高到 $k=3$ 时，作战时间减少至原来的 $1/3$，美国及多国部队兵力损失减少到原来的 $1/4$。

综上所述，**在双方兵力、武器性能相当的情况下，获得信息优势的一方能迅速击败敌方，而且随着信息优势的提高，己方伤亡大幅减小，作战进程大幅加快。**

6. 模型的评价

6.1 模型的优点

对于问题一，指挥舰位于内侧，护卫舰位于外侧，提高了防御纵深。水上舰艇编队优化模型以使来袭导弹在被发现时刻的位置与指挥舰的最短距离最长为目标，而不是以警戒幕面积最大为优化目标，简化了模型计算的复杂程度，提高了模型的可行性。

对于问题二和问题三，从来袭导弹的运动轨迹、防空导弹的拦截过程、舰艇编队的火力分配等三个角度进行问题的分析与求解。尤其是构建了基于 Multi-Agent 的火力分配优化模型，不仅适用于单批次或多批次单方向来袭导弹拦截任务的火力分配，还适用于多批次多方向来袭导弹拦截任务的火力分配，具有较强的实际意义。

对于问题四，建立了适用于只有少量训练样本的模糊识别模型，并通过蒙特·卡罗方法确定了较优的参数，模型识别的准确率达到 73.33%，具有一定可靠性。

对于问题五，采用定性与定量相结合的思想，在兰彻斯特方程基础上，引入战场感知系数和信息优势系数，建立了战略级信息化战争评估模型，计算出了在信息化战争中，信息对于战争双方的影响。

6.2 模型的缺点

对于问题一，水上舰艇编队优化模型未考虑电子干扰、近程火炮（包括密集阵火炮）等拦截手段以及来袭导弹“二次捕捉”能力等因素对编队队形的影响。

对于问题五，模型的计算是基于作战双方的实力进行的。本文由于数据不全，只考虑了作战双方的兵力数量，并没有考虑能够影响作战双方实力的武器装备等因素。

参考文献

[1] 闫国玉, 郭万海, 李国建. 水面舰艇编队防空体系构建及其效能研究[J]. 火力与指挥控制, 2005, 30(增刊).

[2] 周晶, 宋辉, 王秀森, 等. 基于 Multi-Agent 的编队对空防御方法[J]. 兵工自动化, 2011, 30(9).

[3] 汪天飞, 邹进, 张军. 数学建模与数学实验[M]. 北京: 科学出版社, 2013.

[4] 李志强. 复杂系统与战争模拟研究[M]. 北京: 国防大学出版社, 2011.

[5] 吴俊, 杨峰, 梁彦, 等. 面向信息化战争的广义兰切斯特作战模型[J]. 火力与指挥控制, 2010, 35(2).

附录 1

（1）No_1_result.m 源代码

```
clc;
clear all;
x=42.878;
% - A 点距离
L_A=sqrt((20*sind(10)+x*cosd(10))^2+(20*cosd(10)-x*sind(10))^2)
% - A 点角度
T_A=atand((20*cosd(10)-x*sind(10))/(20*sind(10)+x*cosd(10)))+30
% - B 点距离
L_B=sqrt((20)^2+(sqrt(1600-(20*sind(10)+x*cosd(10)-20)^2)+20*cosd(10)-x*sind(10))^2)
% - B 点角度
T_B=atand((sqrt(1600-(20*sind(10)+x*cosd(10)-20)^2)+20*cosd(10)-x*sind(10))/20)+30
% - C 点距离
L_C=L_B
% - C 点角度
T_C=T_B+2*atand(20/(sqrt(1600-(20*sind(10)+x*cosd(10)-20)^2)+20*cosd(10)-x*sind(10)))
% - D 点距离
L_D=L_A
% - D 点角度
T_D=T_A+2*atand((20*sind(10)+x*cosd(10))/(20*cosd(10)-x*sind(10)))
```

（2）No_1_figure.m 源代码

```
clc;
clear all;
x_min=16.782;
x_max=56.782;
x_equal=42.878;
x1=x_min:0.05:x_equal;
x2=x_equal:0.05:x_max;
y1=x1;
y2=sqrt(-0.9702*x2.^2 + 32.5444*x2 + 1327.0896)-0.174*x2+19.7;
figure;
plot(x1,y1);
hold on;
plot(x2,y2);
grid on;
```

附录 2

（1）No_2_Init.m 源代码

```
clc;
clear all;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
% - 设定运动参数
% - 单位（m/s）
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
v0=8.231;      %指挥舰运动速度
v1=306;         %来袭导弹运动速度
v2=816;         %防空导弹运动速度
Pmax=42878; %导弹群初始位置与指挥舰距离，单位 m

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
% - 设定仿真参数
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
t=0;        %仿真时钟
step=0.1; %设定时钟步长，单位 s
flag=0; %仿真控制标志位，为 1 时仿真结束（来袭导弹群距离指挥舰距离≤10 公里）
beat=0; %拦截次数
eggs=No_2_Location_t(Pmax,v0,v1,0); %来袭导弹群初始位置
dmin=10000; %仿真结束判断阈值，10000m

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
% - 以指挥舰为远点，正北方向为 y 轴正方向，建立指标坐标系
% - 单位（m）!!!!!!!
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
% - 变换单位（m-km）
Pmax=Pmax/1000;
% - A 点距离
L_A=sqrt((20*sind(10)+Pmax*cosd(10))^2+(20*cosd(10)-Pmax*sind(10))^2);
% - A 点角度
T_A=atand((20*cosd(10)-Pmax*sind(10))/(20*sind(10)+Pmax*cosd(10)))+30;
% - B 点距离
L_B=sqrt((20)^2+(sqrt(1600-(20*sind(10)+Pmax*cosd(10)-20)^2)+20*cosd(10)-Pmax*sind(10))
^2);
```

```
%-B 点角度
T_B=atand((sqrt(1600-(20*sind(10)+Pmax*cosd(10)-20)^2)+20*cosd(10)-Pmax*sind(10))/20)+3
0;
%-C 点距离
L_C=L_B;
%-C 点角度
T_C=T_B+2*atand(20/(sqrt(1600-(20*sind(10)+Pmax*cosd(10)-20)^2)+20*cosd(10)-Pmax*sind
(10)));
%-D 点距离
L_D=L_A;
%-D 点角度
T_D=T_A+2*atand((20*sind(10)+Pmax*cosd(10))/(20*cosd(10)-Pmax*sind(10)));
%- 变换单位（km-m）
Pmax=Pmax*1000;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
%- 构建舰队模型 [ 编号 , x 坐标 , y 坐标 , beta , l , 状态标志 ]
%  -  指挥舰编号为 0，护卫舰编号为 1-4
%  -  beta 为【舰船与来袭导弹初始位置连线】与【来袭导弹飞行方向】的夹角
%  -  l 为舰船与来袭导弹初始位置距离
%  -  状态标志=0 表示可以发射防空导弹
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
ships=[ 0 ,          0          ,           0          , 0 , 0 , 0 ;
        1 , L_A*sind(T_A)*1000 , L_A*cosd(T_A)*1000 , 0 , 0 , 0 ;
        2 , L_B*sind(T_B)*1000 , L_B*cosd(T_B)*1000 , 0 , 0 , 0 ;
        3 , L_C*sind(T_C)*1000 , L_C*cosd(T_C)*1000 , 0 , 0 , 0 ;
        4 , L_D*sind(T_D)*1000 , L_D*cosd(T_D)*1000 , 0 , 0 , 0 ];
%- 计算 beta，beta
for i=1:1:5
    ships(i,4)=No_2_Beta(ships(i,2:3),eggs);
    ships(i,5)=sqrt(sum((ships(i,2:3)-eggs).^2));
end
```

（2）No_2_Main.m 源代码

```
%- 初始化
No_2_Init;

%- 开始仿真
disp(['--> -----------------------------------------------------']);
disp(['-->']);
disp(['-->                              【开始仿真】']);
disp(['-->']);
```

```
disp(['--> ----------------------------------------------------']);
disp(['-->']);
tic;
while flag==0
    % - step1:计算导弹群距离指挥舰距离
    eggs_now=No_2_Location_t(Pmax,v0,v1,t); %导弹群当前位置
    d=sqrt(sum(eggs_now.^2));    %导弹群与指挥舰当前距离
    %disp(['--> t= ',num2str(t),' s ， 来 袭 导 弹 距 离 指 挥 舰 ： ',num2str(d/1000),'
km']);%pause(0.05);

    % - step2:判断仿真是否终止
    % - 仿真继续
    if d>999    %仿真继续
        % - step3:对 Multi-Agent 进行任务分配
        for i=1:1:5
            % - step4:进行状态轮询
            if ships(i,6)==0
                % - step:5 计算攻击条件

attack_flag=No_2_Attack_Or_Not(Pmax,ships(i,2:5),v0,v1,v2,t);    %attack_flag=[flag,det_t];
                    if attack_flag(1)==1    %可以发射防空导弹
                        ships(i,6)=7+attack_flag(2);
                        %ships(i,6)=7;
                        disp(['--> t= ',num2str(t),' s ， 来袭导弹距离指挥舰: ',num2str(d/1000),'
km , ',num2str(i),' 号舰船发起拦截']);%pause(0.05);
                    end
                else
                    ships(i,6)=ships(i,6)-step;
                    if ships(i,6)<=0
                        ships(i,6)=0;
                        beat=beat+1;
                        disp(['--> t= ',num2str(t),' s ， 来袭导弹距离指挥舰: ',num2str(d/1000),'
km , ',num2str(i),' 号舰船拦截完成，总拦截数：',num2str(beat)]);%pause(0.05);
                        % - 重复 step:5 计算攻击条件

attack_flag=No_2_Attack_Or_Not(Pmax,ships(i,2:5),v0,v1,v2,t);    %attack_flag=[flag,det_t];
                        if attack_flag(1)==1    %可以发射防空导弹
                            ships(i,6)=7+attack_flag(2);
                            %ships(i,6)=7;
                        disp(['--> t= ',num2str(t),' s ， 来袭导弹距离指挥舰: ',num2str(d/1000),'
km , ',num2str(i),' 号舰船发起拦截']);%pause(0.05);
                        end
                    end
                end
```

```
        end
        % - 更新仿真时钟
        t=t+step;

    % - 仿真终止
    else
        flag=1;
    end
end
disp(['-->']);
disp(['--> ---------------------------------------------------------------']);
disp(['-->']);
disp(['-->                                    【仿真结束】']);
disp(['-->']);
disp(['-->                             总计拦截来袭导弹 ',num2str(beat),' 批']);
disp(['-->']);
disp(['-->                             仿真用时： ',num2str(toc),' s']);
disp(['--> ---------------------------------------------------------------']);
```

（3）No_2_Beta.m 源代码

```
function beta=No_2_Beta(X_ship,egg)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
% - beta:【舰船与来袭导弹初始位置连线】与【来袭导弹飞行方向】的夹角
% - X_ship=[x,y]:目标舰坐标
% - egg=[x,y]:来袭导弹初始位置坐标
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
cos_beta=(sum(egg.^2)+sum((X_ship-egg).^2)-sum(X_ship.^2))/(2*sqrt(sum(egg.^2))*sqrt(sum((
X_ship-egg).^2)));
beta=acosd(cos_beta);
```

（4）No_2_Location_t.m 源代码

```
function Location=No_2_Location_t(Pmax,v0,v1,t)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
% - Location=[xt,yt]:t 时刻导弹的位置
% - Pmax
% - v0:指挥舰速度
% - v1:来袭导弹速度
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
% - 计算导弹相对于指挥舰速度的 x 轴分量
vx=v1*cosd(220.5249-180)+v0*sind(200-180);   %235.4130
```

```
%- 计算导弹相对于指挥舰速度的 y 轴分量
vy=v1*sind(220.5249-180)+v0*cosd(200-180);    %206.5668
%- 导弹相对运动方向与 x 轴夹角
A=atand(vy/vx);
%- 计算 xt
xt=vx*t-Pmax*cosd(A);
%- 计算 yt
yt=vy*t-Pmax*sind(A);
%- 整理输出
Location=[xt,yt];
```

（5）No_2_Attack_Or_Not.m 源代码

```
function attack_flag=No_2_Attack_Or_Not(Pmax,ship,v0,v1,v2,t)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
%-attack_flag=[flag,det_t]
%    -flag=1:表示可以发射防空导弹
%    -det_t:拦截冻结时间
%-ship=[x,y,beta,l]
%-v0:指挥舰速度
%-v1:来袭导弹速度
%-v2:防空导弹速度
%-t:仿真时钟当前值
% 注：导弹发射准备时间为 7 秒
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%

%- 构建方程参数
a=v1^2-v2^2;
b=2*(v1^2)*(t+7)-2*cosd(ship(3))*v1*ship(4);
c= ship(4)^2 + ((t+7)^2)*(v1^2) - 2*cosd(ship(3))*v1*ship(4)*(t+7);

%- 构建方程
str=['(',num2str(a),')*x^2+(',num2str(b),')*x+(',num2str(c),')=0'];

%- 求解
x=solve(str);
x=double(x);

%- 分析结果
if x(1)<0
    det_t=x(2);
elseif x(2)<0
    det_t=x(1);
```

```
else
    det_t=min([x(1),x(2)]);
end

% - 判断攻击条件
flag=1;
% - 条件 1：是否超出防空导弹射程
if det_t*v2>=80000
    flag=0;
end
if det_t*v2<=10000
    flag=0;
end
% - 条件 2：拦截地点距离指挥舰是否太近
attack_Location=No_2_Location_t(Pmax,v0,v1,t+det_t);
if sqrt(sum(attack_Location.^2))<10000 || t+det_t>115
    flag=0;
end

% - 整理输出
attack_flag=[flag,det_t];
```

附录 3

（1）No_3_Init.m 源代码

```
clc;
clear all;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
% - 设定运动参数
% - 单位（m/s）
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
v0=8.231;     %指挥舰运动速度
v1=306;        %来袭导弹运动速度
v2=816;        %防空导弹运动速度
Pmax=200000; %导弹群初始位置与指挥舰距离，单位 m

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
% - 设定仿真参数
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
t=0;       %仿真时钟
step=1; %设定时钟步长，单位 s
flag=0; %仿真控制标志位，为 1 时仿真结束（来袭导弹群距离指挥舰距离≤10 公里）
beat=0; %拦截次数
eggs=No_3_Location_t(Pmax,v0,v1,0); %来袭导弹群初始位置
dmin=999; %仿真结束判断阈值，10000m

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
% - 以指挥舰为远点，正北方向为 y 轴正方向，建立指标坐标系
% - 单位（m）!!!!!!!
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
% - 变换单位（m-km）
Pmax=42878/1000;
% - A 点距离
L_A=sqrt((20*sind(10)+Pmax*cosd(10))^2+(20*cosd(10)-Pmax*sind(10))^2);
% - A 点角度
T_A=atand((20*cosd(10)-Pmax*sind(10))/(20*sind(10)+Pmax*cosd(10)))+30;
% - B 点距离
L_B=sqrt((20)^2+(sqrt(1600-(20*sind(10)+Pmax*cosd(10)-20)^2)+20*cosd(10)-Pmax*sind(10))
^2);
```

```
% - B 点角度
T_B=atand((sqrt(1600-(20*sind(10)+Pmax*cosd(10)-20)^2)+20*cosd(10)-Pmax*sind(10))/20)+3
0;
% - C 点距离
L_C=L_B;
% - C 点角度
T_C=T_B+2*atand(20/(sqrt(1600-(20*sind(10)+Pmax*cosd(10)-20)^2)+20*cosd(10)-Pmax*sind
(10)));
% - D 点距离
L_D=L_A;
% - D 点角度
T_D=T_A+2*atand((20*sind(10)+Pmax*cosd(10))/(20*cosd(10)-Pmax*sind(10)));
% - 变换单位（km-m）
Pmax=200000;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
% - 构建舰队模型 [ 编号 , x 坐标 , y 坐标 , beta , l , 状态标志 ]
%  - 指挥舰编号为 0，护卫舰编号为 1-4
%  - beta 为【舰船与来袭导弹初始位置连线】与【来袭导弹飞行方向】的夹角
%  - l 为舰船与来袭导弹初始位置距离
%  - 状态标志=0 表示可以发射防空导弹
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
ships=[ 0 ,              0              ,              0              , 0 , 0 , 0 ;
          1 , L_A*sind(T_A)*1000 , L_A*cosd(T_A)*1000 , 0 , 0 , 0 ;
          2 , L_B*sind(T_B)*1000 , L_B*cosd(T_B)*1000 , 0 , 0 , 0 ;
          3 , L_C*sind(T_C)*1000 , L_C*cosd(T_C)*1000 , 0 , 0 , 0 ;
          4 , L_D*sind(T_D)*1000 , L_D*cosd(T_D)*1000 , 0 , 0 , 0 ];
% - 计算 beta，beta
for i=1:1:5
    ships(i,4)=No_3_Beta(ships(i,2:3),eggs);
    ships(i,5)=sqrt(sum((ships(i,2:3)-eggs).^2));
end
```

（2）No_3_Main.m 源代码

```
% - 初始化
No_3_Init;

% - 开始仿真
disp(['--> ----------------------------------------------------']);
disp(['-->']);
disp(['-->                                【开始仿真】']);
disp(['-->']);
```

```
disp(['--> ----------------------------------------------------']);
disp(['-->']);
tic;
while flag==0
    % - step1:计算导弹群距离指挥舰距离
    eggs_now=No_3_Location_t(Pmax,v0,v1,t); %导弹群当前位置
    d=sqrt(sum(eggs_now.^2));    %导弹群与指挥舰当前距离
    %disp(['--> t= ',num2str(t),' s ，来袭导弹距离指挥舰: ',num2str(d/1000),'
km']);%pause(0.05);

    % - step2:判断仿真是否终止
    % - 仿真继续
    if d>dmin     %仿真继续
        % - step3:对 Multi-Agent 进行任务分配
        for i=1:1:5
            % - step4:进行状态轮询
            if ships(i,6)==0
                % - step:5 计算攻击条件

attack_flag=No_3_Attack_Or_Not(Pmax,ships(i,2:5),v0,v1,v2,t);    %attack_flag=[flag,det_t];
                    if attack_flag(1)==1    %可以发射防空导弹
                        ships(i,6)=7+attack_flag(2);
                        %ships(i,6)=7;
                        disp(['--> t= ',num2str(t),' s ，来袭导弹距离指挥舰: ',num2str(d/1000),'
km , ',num2str(i),' 号舰船发起拦截']);%pause(0.05);
                    end
                else
                    ships(i,6)=ships(i,6)-step;
                    if ships(i,6)<=0
                        ships(i,6)=0;
                        beat=beat+1;
                        disp(['--> t= ',num2str(t),' s ，来袭导弹距离指挥舰: ',num2str(d/1000),'
km , ',num2str(i),' 号舰船拦截完成，总拦截数：',num2str(beat)]);%pause(0.05);
                        % - 重复 step:5 计算攻击条件

attack_flag=No_3_Attack_Or_Not(Pmax,ships(i,2:5),v0,v1,v2,t);    %attack_flag=[flag,det_t];
                        if attack_flag(1)==1    %可以发射防空导弹
                            ships(i,6)=7+attack_flag(2);
                            %ships(i,6)=7;
                        disp(['--> t= ',num2str(t),' s ，来袭导弹距离指挥舰: ',num2str(d/1000),'
km , ',num2str(i),' 号舰船发起拦截']);%pause(0.05);
                        end
                    end
                end
```

```
            end
            % - 更新仿真时钟
            t=t+step;

        % - 仿真终止
        else
            flag=1;
        end
end
disp(['-->']);
disp(['--> ----------------------------------------------------']);
disp(['-->']);
disp(['-->                           【仿真结束】']);
disp(['-->']);
disp(['-->                      总计拦截来袭导弹 ',num2str(beat),' 批']);
disp(['-->']);
disp(['-->                      仿真用时： ',num2str(toc),' s']);
disp(['--> ----------------------------------------------------']);
```

（3）No_3_Beta.m 源代码

```
function beta=No_3_Beta(X_ship,egg)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
% - beta:【舰船与来袭导弹初始位置连线】与【来袭导弹飞行方向】的夹角
% - X_ship=[x,y]:目标舰坐标
% - egg=[x,y]:来袭导弹初始位置坐标
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
cos_beta=(sum(egg.^2)+sum((X_ship-egg).^2)-sum(X_ship.^2))/(2*sqrt(sum(egg.^2))*sqrt(sum((
X_ship-egg).^2)));
beta=acosd(cos_beta);
```

（4）No_3_Location_t.m 源代码

```
function Location=No_3_Location_t(Pmax,v0,v1,t)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
% - Location=[xt,yt]:t 时刻导弹的位置
% - Pmax
% - v0:指挥舰速度
% - v1:来袭导弹速度
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
% - 计算导弹相对于指挥舰速度的 x 轴分量
vx=v1*cosd(220-180)+v0*sind(200-180);    %237.2248
```

```
%- 计算导弹相对于指挥舰速度的 y 轴分量
vy=v1*sind(220-180)+v0*cosd(200-180);   %204.4276
%- 导弹相对运动方向与 x 轴夹角
A=atand(vy/vx);
%- 计算 xt
xt=vx*t-Pmax*cosd(A);
%- 计算 yt
yt=vy*t-Pmax*sind(A);
%- 整理输出
Location=[xt,yt];
```

（5）No_3_Attack_Or_Not.m 源代码

```
function attack_flag=No_3_Attack_Or_Not(Pmax,ship,v0,v1,v2,t)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
%- attack_flag=[flag,det_t]
%   - flag=1:表示可以发射防空导弹
%   - det_t:拦截冻结时间
%- ship=[x,y,beta,l]
%- v0:指挥舰速度
%- v1:来袭导弹速度
%- v2:防空导弹速度
%- t:仿真时钟当前值
% 注：导弹发射准备时间为 7 秒
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%

%- 构建方程参数
a=v1^2-v2^2;
b=2*(v1^2)*(t+7)-2*cosd(ship(3))*v1*ship(4);
c= ship(4)^2 + ((t+7)^2)*(v1^2) - 2*cosd(ship(3))*v1*ship(4)*(t+7);

%- 构建方程
str=['(',num2str(a),')*x^2+(',num2str(b),')*x+(',num2str(c),')=0'];

%- 求解
x=solve(str);
x=double(x);

%- 分析结果
if x(1)<0
    det_t=x(2);
elseif x(2)<0
    det_t=x(1);
```

```
else
    det_t=min([x(1),x(2)]);
end

% - 判断攻击条件
flag=1;
% - 条件 1：是否超出防空导弹射程
if det_t*v2>=80000
    flag=0;
end
if det_t*v2<=10000
    flag=0;
end
% - 条件 2：拦截地点距离指挥舰是否太近
attack_Location=No_3_Location_t(Pmax,v0,v1,t+det_t);
if sqrt(sum(attack_Location.^2))<10000 || t+det_t>605
    flag=0;
end

% - 整理输出
attack_flag=[flag,det_t];
```

附录 4

（1）No_4_Example.mat 中数据

1.0e+003 *

0.8100	0.2810	0.2500	0.2020	0.0060	0.0030	0.0010
2.3000	0.2100	0.3000	0.3100	0.0040	0.0012	0.0020
0.8200	0.2800	0.2450	0.2010	0.0065	0.0054	0.0010
2.3250	0.2150	0.3200	0.3240	0.0042	0.0028	0.0020
0.8300	0.2820	0.2550	0.2000	0.0042	0.0047	0.0010
0.8250	0.2840	0.2500	0.2040	0.0050	0.0026	0.0010
2.2500	0.1500	0.3000	0.1550	0.0050	0.0033	0.0020
4.0000	0.1100	0.3000	0.0500	0.0034	0.0021	0.0030
2.8000	0.2600	0.2150	0.2600	0.0077	0.0068	0.0040
5.1200	0.1100	0.2100	0.0520	0.0036	0.0037	0.0050
4.0200	0.1200	0.2800	0.0520	0.0036	0.0017	0.0030
4.8000	0.1400	0.2200	0.0180	0.0096	0.0057	0.0050
0.4800	0.2950	0.2920	0.2450	0.0099	0.0069	0.0050
2.4500	0.2100	0.2300	0.2100	0.0050	0.0012	0.0050
2.9000	0.2900	0.2720	0.3500	0.0056	0.0052	0.0020

（2）No_4_Init.m 源代码

```
clc;
clear all;

% - 导入源数据(data 为 15x7 的矩阵，行为样本，列为属性)
load No_4_Example.mat;
total_std=std(No_4_Example(:,1:6),1);
new_std=total_std./sum(total_std);
new_std(2,:)=new_std(1,:);
new_std(3,:)=new_std(1,:);
new_std(4,:)=new_std(1,:);
new_std(5,:)=new_std(1,:);

% - 归一化
temp_he=sum(No_4_Example(:,1:6));
data(1:15,1)=No_4_Example(1:15,1)/temp_he(1);
data(1:15,2)=No_4_Example(1:15,2)/temp_he(2);
data(1:15,3)=No_4_Example(1:15,3)/temp_he(3);
data(1:15,4)=No_4_Example(1:15,4)/temp_he(4);
data(1:15,5)=No_4_Example(1:15,5)/temp_he(5);
data(1:15,6)=No_4_Example(1:15,6)/temp_he(6);
data(:,7)=No_4_Example(:,7);

% - 计算样本各个属性平均值和标准差
```

```
% -        u1    u2    u3    u4    u5    u6
% -    v1
% -    v2
% -    v3
% -    v4
% -    v5
m=zeros(5,6);    %平均值
n=zeros(5,6);    %标准差
for i=1:5
    temp_data=zeros(1,6);
    temp_index=0;
    for j=1:15
        if data(j,7)==i
            temp_index=temp_index+1;
            temp_data(temp_index,1:6)=data(j,1:6);
        end
    end
    m(i,:)=mean(temp_data,1);
    if i==4
        n(i,:)=mean(temp_data);
    else
        n(i,:)=std(temp_data,1);
    end
end

% - 导入待识别数据
Aim_data=[4021,213,142,61,7.0,3.5;
2830,206,269,13,9.2,5.7;
3622,221,230,10,4.6,1.9;
2032,220,195,307,5.2,4.3;
353,232,195,261,5.2,5.5;
1170,160,266,231,3.4,2.6;
1907,208,283,170,9.4,6.2;
4952,205,301,100,1.4,1.1;
6393,168,301,144,2.6,5.5;
6,163,301,174,2.6,5.5;
4738,236,177,99,6.0,1.7;
4748,235,177,358,6.0,1.7;
888,229,195,232,4.8,3.6;
877,227,195,114,4.8,3.6;
99,243,213,155,8.6,3.1;
92,241,213,278,8.6,3.1];
% - 归一化
temp_he=sum(Aim_data(:,1:6));
```

```
temp_Aim_data(1:length(Aim_data(:,1)),1)=Aim_data(1:length(Aim_data(:,1)),1)/temp_he(1);
temp_Aim_data(1:length(Aim_data(:,1)),2)=Aim_data(1:length(Aim_data(:,1)),2)/temp_he(2);
temp_Aim_data(1:length(Aim_data(:,1)),3)=Aim_data(1:length(Aim_data(:,1)),3)/temp_he(3);
temp_Aim_data(1:length(Aim_data(:,1)),4)=Aim_data(1:length(Aim_data(:,1)),4)/temp_he(4);
temp_Aim_data(1:length(Aim_data(:,1)),5)=Aim_data(1:length(Aim_data(:,1)),5)/temp_he(5);
temp_Aim_data(1:length(Aim_data(:,1)),6)=Aim_data(1:length(Aim_data(:,1)),6)/temp_he(6);
Aim_data=temp_Aim_data;
```

（3）No_4_Main.m 源代码

```
% - 程序初始化
No_4_Init;
tic;

% - 计算模糊评价矩阵
result=zeros(length(Aim_data(:,1)),1);
a=[1.8938,0.6209,0.0008,22.8997,2.8169];
for i=1:length(Aim_data(:,1))
    W=zeros(5,6);
    W(1,1:5)=No_4_Membership_1(a(1),Aim_data(i,:),m,n);
    W(2,1:5)=No_4_Membership_2(a(2),Aim_data(i,:),m,n);
    W(3,1:5)=No_4_Membership_3(a(3),Aim_data(i,:),m,n);
    W(4,1:5)=No_4_Membership_4(a(4),Aim_data(i,:),m,n);
    W(5,1:5)=No_4_Membership_5(a(5),Aim_data(i,:),m,n);

    W(1,6)=No_4_Judge_1(Aim_data(i,6),n(1,6));
    W(2,6)=No_4_Judge_2(Aim_data(i,6),n(2,6));
    W(3,6)=No_4_Judge_3(Aim_data(i,6),n(3,6));
    W(4,6)=No_4_Judge_4(Aim_data(i,6),n(4,6));
    W(5,6)=No_4_Judge_5(Aim_data(i,6),n(5,6));

    %W=sum(W')'/5;
    W=(sum((W.*new_std)'))';

    [value,index]=max(W);
    if index<1
        index=round(5*rand(1));
    end
    result(i,1)=index;
end
result
toc
```

（4）No_4_Membership_1.m 源代码

```
function Membership_j=No_4_Membership_1(a,x,m,n)
```

```
%Membership_j=exp(-10*(x(1,1:5)-m(1,1:5)).^2./n(1,1:5).^2)*No_4_Judge_1(x(1,6),n(1,6));
Membership_j=exp(-a*(x(1,1:5)-m(1,1:5)).^2./n(1,1:5).^2);
```

（5）No_4_Membership_2.m 源代码

```
function Membership_j=No_4_Membership_2(a,x,m,n)
%Membership_j=exp(-10*(x(1,1:5)-m(2,1:5)).^2./n(2,1:5).^2)*No_4_Judge_2(x(1,6),n(2,6));
Membership_j=exp(-a*(x(1,1:5)-m(2,1:5)).^2./n(2,1:5).^2);
```

（6）No_4_Membership_3.m 源代码

```
function Membership_j=No_4_Membership_3(a,x,m,n)
%Membership_j=exp(-10*(x(1,1:5)-m(3,1:5)).^2./n(3,1:5).^2)*No_4_Judge_3(x(1,6),n(3,6));
Membership_j=exp(-a*(x(1,1:5)-m(3,1:5)).^2./n(3,1:5).^2);
```

（7）No_4_Membership_4.m 源代码

```
function Membership_j=No_4_Membership_4(a,x,m,n)
%Membership_j=exp(-10*(x(1,1:5)-m(4,1:5)).^2./n(4,1:5).^2)*No_4_Judge_4(x(1,6),m(4,6));
Membership_j=exp(-a*(x(1,1:5)-m(4,1:5)).^2./n(4,1:5).^2);
```

（8）No_4_Membership_5.m 源代码

```
function Membership_j=No_4_Membership_5(a,x,m,n)
%Membership_j=exp(-10*(x(1,1:5)-m(5,1:5)).^2./n(5,1:5).^2)*No_4_Judge_5(x(1,6),n(5,6));
Membership_j=exp(-a*(x(1,1:5)-m(5,1:5)).^2./n(5,1:5).^2);
```

（9）No_4_Judge_1.m 源代码

```
function y=No_4_Judge_1(x6,n_16)
if x6>=4
    %y=exp(-10*(x6-4)^2/n_16^2);
    y=0;
else
    y=1;
end
```

（10）No_4_Judge_2.m 源代码

```
function y=No_4_Judge_2(x6,n_26)
y=1;
```

（11）No_4_Judge_3.m 源代码

```
function y=No_4_Judge_3(x6,n_36)
if x6>=4
    %y=exp(-10*(x6-4)^2/n_36^2);
    y=0;
else
    y=1;
end
```

（12）No_4_Judge_4.m 源代码

```
function y=No_4_Judge_4(x6,n_46)
if x6<=2
    %y=exp(-10*(x6-2)^2/n_46^2);
    y=0;
else
    y=1;
end
```

（13）No_4_Judge_5.m 源代码

```
function y=No_4_Judge_5(x6,n_56)
y=1;
```

附录 5

（1）No_5_Init.m 源代码

```
clc;
clear all;

%- 设置仿真参数
t=0;
step=1;
flag=0;
r=10000;
b=10000;
Ts=0.1;
Us=0.1;
Tl=0.00001;
Ul=0.00001;
Ur=0.2;
Ub=0.1;
k=Ur/Ub;
```

（2）No_5_Main.m 源代码

```
%- 程序初始化
No_5_Init;

%- 开始仿真
r_result=0;
b_result=0;
while flag==0
    if t>0
        r_result(t)=r;
        b_result(t)=b;
        temp_r=No_5_r(r,b,Us,k,Ub,Ul);
        temp_b=No_5_b(b,r,k,Ts,Ur,Tl);
        r=temp_r;
        b=temp_b;
    end
    %- 判断仿真是否终止
    if r<=0 || b<=0
        flag=1;
    else
        t=t+step;
    end
    disp(['--> t= ',num2str(t),' , r= ',num2str(r),' , b= ',num2str(b)]);
end
```

```
plot(1:t,r_result,1:t,b_result);
```

（3）No_5_r.m 源代码

```
function y=No_5_r(r,b,Us,k,Ub,Ul)
y=r-Us*b*Ub/k-(1-Ub)*Ul*r*b/(k^2);
```

（4）No_5_b.m 源代码

```
function y=No_5_b(b,r,k,Ts,Ur,Tl)
y=b-k*Ts*r*Ur-(1-Ur)*Tl*(k^2)*r*b;
```