多波束声纳系统视野覆盖模型的建立与实际测线设计应用

摘要

本文基于多波束测深系统的工作原理，运用了**解析计算**和**微分近似思想**进行了机理建模，建立了在不同的海底地形情况下的最优测线模型。并且运用了**微分法，迭代公式法，K 维树搜索算法**来计算在给定真实情景下，设计测线的总长度，漏测海域占总待测海域面积的百分比以及测线间的重叠率。

在问题一中，只考虑探测船在垂直于坡面法向的方向运动。首先，以海域中心为原点，计算海水深度与测线距离中心点处的距离之间的关系。其次，我们可以建立一个数学模型，利用多波束换能器的开角，海底坡度求出探测器的覆盖宽度；同时我们推广了重叠率的定义，使其在曲面上也能得以计算；最后，利用，MATLAB 进行编程分析，计算结果见 result1-1.xlsx 。

在问题二中，同样是求解探测船的覆盖宽度，不过此时是建立在三维的空间模型中。首先，我们抓住影响其覆盖宽度的主要决定因素是在测线与高度组成的平面，截坡面而形成的三角形的顶角以及其探测面截海底斜坡而形成的三角形的顶角；其次，通过找出这两个角与我们已知量之间的关系，我们便能令模型回归至问题一中的情况；最后我们利用 MATLAB 进行编程分析，计算结果见 result1-2.xlsx 。

在问题三中，在问题三中，其模型与问题二类似，不过其求解的问题有所不同，要求在底面坡度一定的情况下，在一片固定大小的海域，满足每两条测线间的重叠率都在 10% 至 20% 的条件下，寻找到一组能够完全覆盖整片海域的最短的测线。首先我们先利用微元法计算出当探测船在走一个一个面积微元时的局部最优策略，来确定我们大致的方向；之后再利用截面法，从机理上论证了在该模型的条件下的最优策略。最后，在给定**相邻不等间距平行线的重叠率为 10% 情况下**，再利用几何关系求得测线垂直水深进行迭代的，计算出此时的**测线总长度为 66 海里**。

在问题四中，首先通过对数据分析得到曲面的截面可以用二次函数拟合，通过对不同 x 下的截面构建模型，利用化曲为直的思想，得到在不同 x 下的最佳测点分布，而后通过对不同 x 下的测点利用 k 维树算法搜索最近测点并统计测点间的距离，通过不断减小 dx 来提高测线的精度，最后将测线微分后计算得到**测线长度为 189.74 海里**。同时，由于利用化曲为直的思想构建的模型，在部分地方会有漏测面积的现象，我们联立经过测点的直线系（斜率与 θ 相关）与拟合后的二次函数，得到交点，通过计算交点之间的距离得到在每一 x 上漏测长度，在对漏测长度积分得到我们的**漏测面积占比为 9.2%**，由于转化直线情况下建模，此时在直线条件下的重叠率为 0，因为研究的截面函数曲线是凸函数，在这种情况下**曲线的重叠率会小于直线情况，因此重叠率为 0**。

关键字：几何　微分　近似　k 维树算法

一、 问题重述

1.1 问题背景

多波束测深系统是集成不同组件并具有多项功能的综合性探测系统，与传统的单波束测深相比较，其测量性能得到很大提升，可通过接收换能器接收从海底传回的声波，测量出具有一定宽度的水深条带，是目前海底地形勘察、海道测量以及海底浅层资源调查的重要手段之一。

1.2 问题的提出

由于多波束测深条带的覆盖宽度 W 与接收换能器的开角 θ 和水深 D 的具体取值有关，在海底地貌起伏变化的的真实情况下，若测线间距设置不合理，则可能在水深较浅处出现测漏，或在水深较深处出现较多的数据冗余。基于以上背景信息和附件内容，需建立相应的数学模型解决以下问题。

问题一：仅考虑海底地形为一规则坡面，坡度为 α，建立当探测船垂直于坡面抬升方向前进时，多波束测深的覆盖宽度以及相邻条带之间的重叠率模型。

问题二：在问题一模型建立的基础上，改变探测船前进方向，使测线方向与海底坡面的法向在水平面上投影夹角呈 β，更新多波束测深的覆盖宽度的数学模型并应用于一矩形海域之中。

问题三：在一给定海域中心海水深度的矩形斜坡海域内，在待测海域被完全覆盖、满足相邻条带之间的重叠率控制在 10% 至 20% 的前提下，应用问题 2 的计算结果设计一组测线使得总测量长度最短。

问题四：应用附件中给定的矩形海域海水深度数据，尽可能控制测线扫描条带覆盖率、相邻条带间重叠率、测线总长度，给出所设计测线的总长度，漏测海区占比、重叠率超 20% 的总长度。

二、 问题分析

2.1 问题一的分析

问题一本质上是一个平面几何问题，通过已知多波束张角 θ 和测线 i 处所在水深 D_i，可求得坡度为 α 的规则坡面被张角所截的长度，即为所求多波束测深的覆盖宽度。

本问题中通过计算测线距离中心点的不同距离，来计算其海水的深度与覆盖的宽度。然后推广题设中，重叠率的定义，得出在曲面上重叠率的公式。并且计算出距离中心点位置不同处，它们与前一条测线的重叠率分别为多少。

2.2 问题二的分析

问题二是问题一的延续，更进一步之处在于测线的方向与海底坡面的法向在水平上的夹角不再是特殊的角度 $\frac{\pi}{2}$，而是一个任意的夹角 β。分别通过几何法与解析法，找出变量 β,l 与覆盖宽度 W 之间的关系，并且相互印证了其正确性。通过建立的数学模型，分别求出了探测船在海面沿着不同的方向，行驶不同的距离时，多波束测线的覆盖宽度。

2.3 问题三的分析

问题三是建立在问题二的基础之上的，可沿用其模型，但是所求解的对象有所不同，题设要求在满足：(1) 探测面积能覆盖整片海域；(2) 相邻条带之间的重叠率为 10% 至 20% 之间的条件下，设计出一组总长度最短的测线。通过问题二给出的模型，我们可以计算出当探测船在走一个一个面积微元时的局部最优策略，以此确定我们大致的研究方向，再取截面，通过截面上的最优点之间的几何关系，得出全局状态下最佳的结果。最后利用重叠率的大小的限制，逐步迭代出在我们选择的最优策略下，其最短的测线组的长度。

2.4 问题四的分析

问题四是问题一、二、三在实际应用中的综合体现，由于题目中所给定的曲面较为特殊，我们可以发现在 y-z 平面上形成的截面曲线，我们可以用二次函数较好的拟合原本的曲线，通过文献查阅 [1]，获知实际应用中的测线常为沿等深线方向平行设计，因此我们选定 x 为前进方向，通过计算在不同时刻的 y-z 截面，将原本的三维问题，转化到二维，并通过计算在一个二次函数上测量点最优分布，最后汇总所有的测量点，统计最短距离时形成的测线，利用微分计算测线长度，最后由于我们在部分地区采用了化曲为直的想法，需要考虑近似对重叠率和漏测面积的影响。

三、 模型假设

1. 为了计算方便与驾驶员的行驶方便，此处所规划的测线尽可能为直线。
2. 海水介质对声波的传播无影响。
3. 在海洋地貌倾角较小时，可以将曲线近似为斜线进行测线设计
4. 有效探测角度不受海底散射影响

四、 符号说明

符号	说明	单位
s_1	船身左侧截线与测线间距的投影长度	m
s_2	船身右侧截线与测线间距的投影长度	m
s	左右截线间距的投影长度	m
h	海水深度	m
θ	多波束换能器的开角	rad
α	海底的坡度	rad
L	船行的距离	海里
W	多波束条带的覆盖宽度	m
d	相邻测线间距	m
η	重叠率	/
β	测线方向与海底坡面的法向在水平面上投影的夹角	rad
α_1	测线与 h_0 组成的平面截海底斜坡形成的截面的顶角	rad
α_2	两条截线组成的平面截海底斜坡形成的截面的顶角	ms
$y'1$	在以测线为 x' 轴的 $x'Oy'$ 坐标系中，截线 s_1 的方程	m
$y'2$	在以测线为 x' 轴的 $x'Oy'$ 坐标系中，截线 s_2 的方程	m
$y1$	在以海底坡面的法向在水平面上投影为 x 轴的 xOy 坐标系中，截线 s_1 的方程	m
$y2$	在以海底坡面的法向在水平面上投影为 x 轴的 xOy 坐标系中，截线 s_2 的方程	m
S	海域总面积	海里 ^{2}

五、 模型建立与求解

5.1 问题一模型建立与求解

5.1.1 问题一几何模型的建立

对于一条离水底高度为 h 的航线，在坡度为 α，多波束换能器的开角为 θ 的情况下，如图 1(a) 所示，可以根据三角关系列出两个方程：

$$\begin{cases} s_1 = (h + s_1 \cdot tan(\alpha)) \cdot tan(\frac{\theta}{2}) \\ s_2 = (h - s_2 \cdot tan(\alpha)) \cdot tan(\frac{\theta}{2}) \end{cases} \tag{1}$$

同时，假设向右为正方向，可以观察到，受到坡度的影响，海水深度 h 与水平移动距离 L 存在着以下关系（其中 h_0 为海域中心点处的水深）：

$$h = h_0 - L \cdot tan(\alpha) \tag{2}$$

最后代入化简，可以得到在处于 L 的位置下，左、右探测距离（即截线离测线的距离向底面的投影）分别为：

$$s_1 = \frac{(h_0 - L \cdot tan(\alpha)) \cdot tan(\frac{\theta}{2})}{1 + tan(\alpha) \cdot tan(\frac{\theta}{2})} \tag{3}$$

$$s_2 = \frac{(h_0 - L \cdot tan(\alpha)) \cdot tan(\frac{\theta}{2})}{1 + tan(\alpha) \cdot tan(\frac{\theta}{2})} \tag{4}$$

此时的总覆盖宽度 W 可表示为：

$$W = (s_1 + s_2) \cdot tan(\alpha) \tag{5}$$

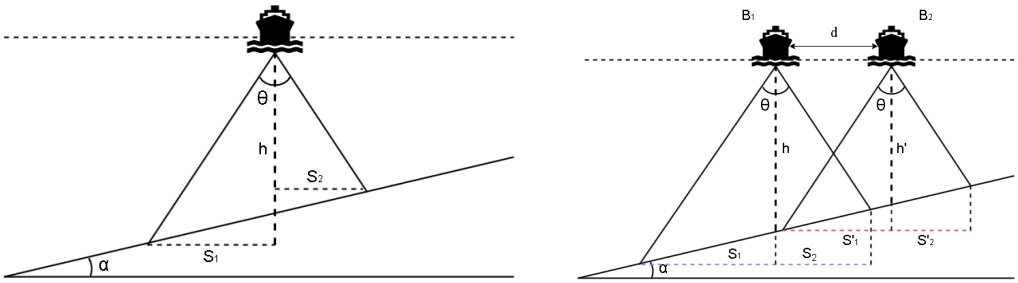
此外，在海底地形平坦时，重叠率定义为：$\eta = 1 - \frac{d}{w}$，其中，d 是相邻两条测线之间的间距，W 是条带的覆盖宽度。在此处可以将其推广至海底为任意曲面时的情况。此时定义探测船的左探测距离 s_1 与右探测距离 s_2。如图 1(b) 所示，存在两条测线 B_1, B_2，它们之间的距离为 d，B_1 的左右探测距离分别为 s_1,s_2；B_2 的左右探测距离分别为 $s_1^{'}$,$s_2^{'}$，据此可以定义其在曲面上重叠率为。

$$\eta = \frac{(s_2 + s_1^{'} - d)}{s_1 + s_2} \tag{6}$$

：

5.1.2 问题一模型的求解

接着便代入题设参数：$h_0 = 70$，$\alpha = 1.5°$，$\theta = 120°$ 分别求出在不同的测线距中心点处距离时，海水深度、覆盖宽度、与前一条测线的重叠率：



(a) 单探测船模型　　(b) 相邻探测船模型

图 1

表 1　问题 1 的计算结果

测线距中心点处的距离/m	-800	-600	-400	-200	0	200	400	600	800
海水深度/m	90.95	85.71	80.47	75.24	70.00	64.76	59.53	54.29	49.05
覆盖宽度/m	315.81	297.63	279.44	261.26	243.07	224.88	206.70	188.51	170.33
与前一条测线的重叠率/%	—	0.3364	0.2959	0.2500	0.1978	0.1378	0.0681	-0.0139	-0.1117

5.2 问题二模型建立与求解

5.2.1 问题二模型的建立

问题二仍是考虑在海底为坡面的水域上进行探测，但是此时测线方向与坡面的法向之间存在一个角度 β，如图 2 所示。该问题是对上一题的拓展与延申，所以可以沿用问题一的模型，当知道行船的高度 h 与该斜面坡度 α 时，能根据问题一的模型直接得出结论。在此处的思路是：在沿 β 的方向上，分别研究 h 与 α 是如何随着航行距离 L 变化的。

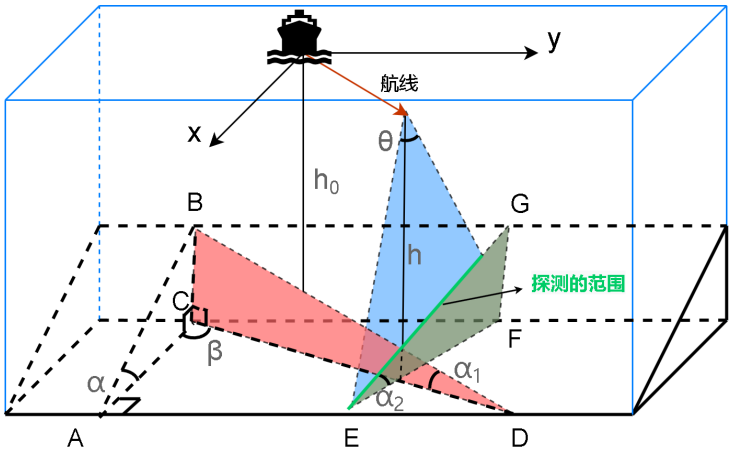


图 2　三维测线模型

首先，在沿 β 方向，建立高度 h 随着航行距离 L 变化的的模型。建立如图 2 所示的直角坐标系，先确定其航线方向，并且绘制出过该航线，并且垂直于规定的 xoy 面的平面，截海底的斜坡，得到了截面 BCD，该探测船在沿着该方向行驶的过程中，其发生的高度变化，是由截面三角形的顶角 α_1 所引起的，并且有公式：

$$h = h_0 + L \cdot tan(\alpha_1) \tag{7}$$

通过四面体 $ABCD$ 的几何关系，可以列出三个关系式：

$$AC = DCcos(\beta) \quad BC = ACtan(\alpha) \quad BC = CDtan(\alpha_1)$$

并据此能够推导出 α_1 与 α、β 之间的关系

$$\alpha_1 = arctan(cos\beta \cdot tan\alpha) \tag{8}$$

联立 7、 8两式可以求得，当探测船沿着 β 方向前进时，海水深度 h 是如何随着 L 变化所变化的：

$$h = h_0 + L \cdot cos(\beta) \cdot tan(\alpha) \tag{9}$$

下一步探究的是在沿 β 方向前进距离 L 时，探测范围的变化。定义探测面（蓝色）截海底的斜坡的截面是 EFG，在水深 h 与多波束换能器的夹角 θ 已知的情况下，只需要求得截面产生的等效坡度角 α_2 即可得知其探测范围的覆盖宽度。由几何关系，在底面 ABC 存在 $CD \perp EF$，由此可知，EF 与 x 方向的夹角为 $\frac{\pi}{2} + \beta$。可以利用 7式，将其中 β 替换为 $\frac{\pi}{2} + \beta$，即可得到 α_2 与坡度角 α、航行方向 β 的关系。

$$\alpha_2 = arctan(cos(\frac{\pi}{2} + \beta) \cdot tan\alpha) = arctan(sin\beta \cdot tan\alpha) \tag{10}$$

以 α_1 作为引起高度变化的角度，以 α_2 作为测量时坡面的倾角，代入 3、 4式，可以得到：

$$s_1 = \frac{(h_0 + Lcos\beta tan\alpha)tan\frac{\theta}{2}}{1 - tan\frac{\theta}{2} \cdot sin\beta \cdot tan\alpha} \tag{11}$$

$$s_2 = \frac{(h_0 + Lcos\beta tan\alpha)tan\frac{\theta}{2}}{1 + tan\frac{\theta}{2} \cdot sin\beta \cdot tan\alpha} \tag{12}$$

覆盖宽度 W 即为

$$W = \frac{s_1 + s_2}{cos\alpha_2} = \frac{(h_0 + Lcos\beta tan\alpha)tan\frac{\theta}{2}}{cos(arctan(sin\beta \cdot tan\alpha))} \cdot (\frac{2}{1 - tan^2\frac{\theta}{2} \cdot sin^2\beta \cdot tan^2\alpha}) \tag{13}$$

5.2.2 问题二模型的求解

根据模型，在给定海域中心点处海水深度 h_0、海底坡面倾角 α、多波束换能器的张角 θ 的情况下，可以求出探测船只沿着某个方向 β 前进 L 距离时，其覆盖宽度的大小，计算结果如下表

覆盖宽度/m		测量船距海域中心点处的距离/海里							
		0	0.3	0.6	0.9	1.2	1.5	1.8	2.1
测线方向夹角/°	0	415.7	466.1	516.5	566.9	617.3	667.7	718.1	768.5
	45	416.2	451.9	487.6	523.2	559.0	594.6	630.3	666.0
	90	416.7	416.7	416.7	416.7	416.7	416.7	416.7	416.7
	135	416.2	380.5	344.8	309.2	273.5	237.8	202.1	166.4
	180	415.7	365.3	314.9	264.5	214.1	163.7	113.3	63.0
	225	416.2	380.5	344.8	309.2	273.5	237.8	202.1	166.4
	270	416.7	416.7	416.7	416.7	416.7	416.7	416.7	416.7
	315	416.2	451.9	487.6	523.2	558.9	594.6	630.3	666.0

同时，可以绘制出他们的关系，用三维曲面为示意：

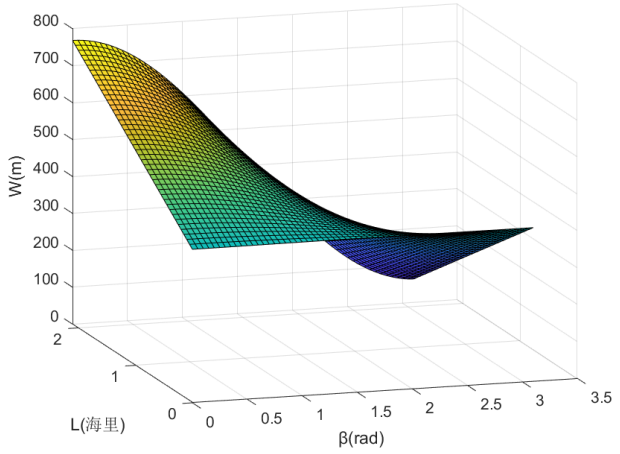


图 3　向 β 方向前进距离 L，此时的探测宽度 W

观察到由于坡角 α 较小，在 $L=0$ 附近的时候，β 角对探测宽度的影响较小，但是 β 角的影响会随着航行距离的增加而变得更加显著。

5.3 问题三模型建立与求解

5.3.1 问题三模型的建立

该问题要求在底面坡度一定的情况下，在一片固定大小的海域，满足每两条测线间的重叠率都在 10% 至 20% 的条件下，寻找到一组能够完全覆盖整片海域的，最短的测线。首先对前进方向 β 进行分 d 析，探测船在沿不同的方向前进时，其左右截线满足问题二的模型：

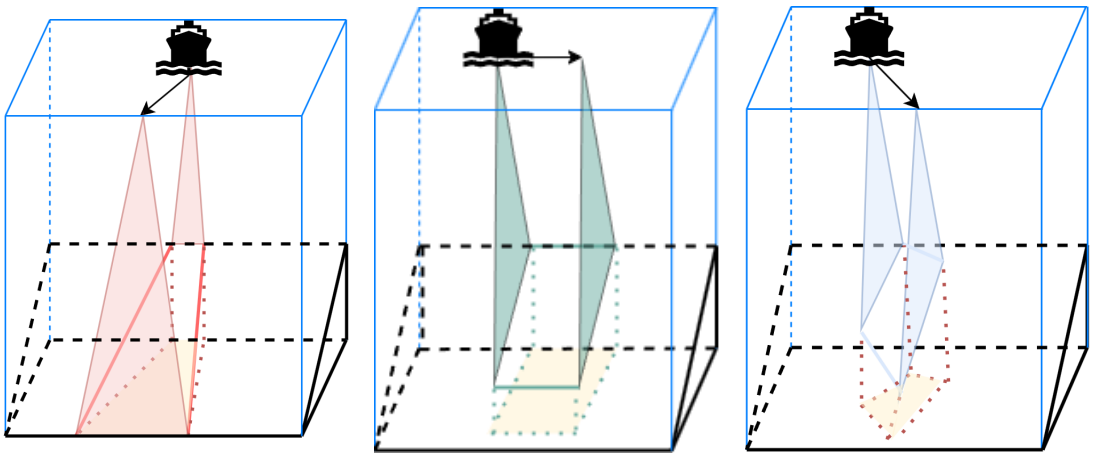
$$s_1 = \frac{(h_0 + Lcos\beta tan\alpha)tan\frac{\theta}{2}}{1 - tan\frac{\theta}{2} \cdot sin\beta \cdot tan\alpha} \qquad s_2 = \frac{(h_0 + Lcos\beta tan\alpha)tan\frac{\theta}{2}}{1 + tan\frac{\theta}{2} \cdot sin\beta \cdot tan\alpha}$$

在 β 给定的情况下，其左右测距 s_1 和 s_2 都与航行距离 L 呈线性关系，并且斜率与初始高度 h_0 无关。此时，若以测线为 x' 轴建立坐标系 $x'Oy'$，则两条截线分别是 $y_1' = s_1$、$y_2' = -s_2$，函数关系式可以写作：

$$y_1' = \frac{cos\beta \cdot tan\alpha \cdot tan\frac{\theta}{2}}{1 - tan\frac{\theta}{2} \cdot sin\beta \cdot tan\alpha}x' + \frac{tan\frac{\theta}{2}}{1 - tan\frac{\theta}{2} \cdot sin\beta \cdot tan\alpha}h_0 \tag{14}$$

$$y_2' = -\frac{cos\beta \cdot tan\alpha \cdot tan\frac{\theta}{2}}{1 + tan\frac{\theta}{2} \cdot sin\beta \cdot tan\alpha}x' - \frac{tan\frac{\theta}{2}}{1 + tan\frac{\theta}{2} \cdot sin\beta \cdot tan\alpha}h_0 \tag{15}$$

绘制出在不同 β 下，测线 s_1 与 s_2 的位置。如图 4，沿不同方向航行，其探测宽度在底面的投影始终是矩形或梯形，如图 4(a)，沿着坡面的梯度前进的过程中，其测线在底面的投影将是一个等腰梯形；图 4(b) 中所显示的，若探测船沿着等高线方向前进，其测线在底面的投影将是一个矩形；而如图 4(c)，探测船沿着任意方向前进，测线在底面的投影将是一个梯形。



(a) $\beta = 0$ **(b)** $\beta = \frac{\pi}{2}$ **(c)** $0 < \beta < \frac{\pi}{2}$

图 4　沿不同方向行驶对应的测线形状

在 $\beta = \frac{\pi}{2}$ 时，其投影为一个矩形的情况下，各对边相互平行。此时能够严格控制每两条测线间的重叠率都在 10%；而在其它的 β 之下，其投影为梯形，应该保证其最小

重叠率，即上底的其重叠率为 10%，而其他部分的重叠率必然会大于 10%。综上，沿着 $\beta = \frac{\pi}{2}$ 方向行驶，所扫过的面积的利用率最大。

同时，当探测船在某一位置，在不同的航行方向，即不同的 β 的情况下，其探测宽度 s 也有所不同

$$s = s_1 + s_2 = \frac{h_0 tan\frac{\theta}{2}}{1 - tan\frac{\theta}{2} \cdot sin\beta \cdot tan\alpha} + \frac{h_0 tan\frac{\theta}{2}}{1 + tan\frac{\theta}{2} \cdot sin\beta \cdot tan\alpha} \tag{16}$$

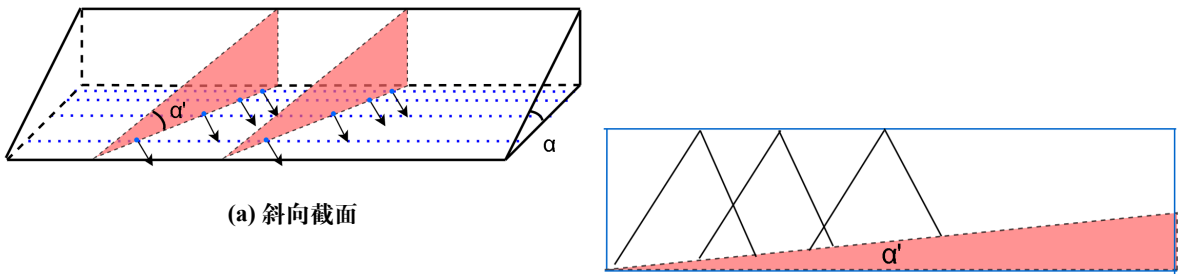
令 $\frac{ds}{d\beta} = 0$，在 β 达到 $\frac{\pi}{2}$ 时，其探测宽度有极大值 $s_s max = s\frac{\pi}{2})$。先选定探测角 β 令其往该方向前进一个距离微元 dL，则在此过程中被探测的面积微元为

$$dS = s_{L+dl} \cdot dl = (s_1 + s_2) \cdot dl + dl^2(\frac{cos\beta \cdot tan\alpha \cdot tan\frac{\theta}{2}}{1 + tan\frac{\theta}{2} \cdot sin\beta \cdot tan\alpha})$$

忽略二阶小量，可以看出在任意一个位置，往 $\beta = \frac{\pi}{2}$ 侧行驶探测得到的的面积大于往其他方向行驶产生的面积。记海域总面积为 S，可将题设改写为较为宽松的形式：

$$\int s \cdot dl > S$$

其中 $\int s \cdot dl$ 为探测的面积，由于存在着重叠率，其会稍大于海域总面积。在 $\beta = \frac{\pi}{2}$ 的情况下，覆盖宽度 s 较长，走相同面积的情况下需要的距离 l 较短，且因为其面积利用率较大，故沿着等高线前进在每一块小微元内都是优于向其他方向前进的。



(a) 斜向截面

(b) 等效坡度图

图 5　截面法求解最优路径

下面采用截面法进行具体分析：如图 5(a) 沿着某一个方向 β 航行，其探测面与海底斜坡所截的面是一个夹角呈 α' 的坡，且总有 $\alpha' \leq \alpha$。此时，若对截面图进行分析，则面的最优探测点实际上是在一条条 $\beta = \frac{\pi}{2}$ 平行线上的，如图 5(a)。若一艘探测船在某一时刻的前进方向为 β，且它在与其航行方向垂直的截面上的最优探测点上；但由于是沿着 β 方向前进，所以紧接着，其将偏离在下一时刻的最优探测点。当且仅当 $\beta = \frac{\pi}{2}$ 时，能保证每一时刻都能行驶在最优探测点上。当我们考虑斜向前进，前进距离 dx 极小时，可以通过计算的 $arctan$ 值，此时我们得到的 $\Delta\alpha$ 也极小，$\alpha + \Delta\alpha$ 趋近于 α，可知平行线较好。

5.3.2 问题三模型的求解

为了方便模型的求解，将以固定坐标系进行模型的计算。定义 $\beta = 0$ 为 x 轴，$\beta = \frac{\pi}{2}$ 为 y 轴，代入坐标变换公式

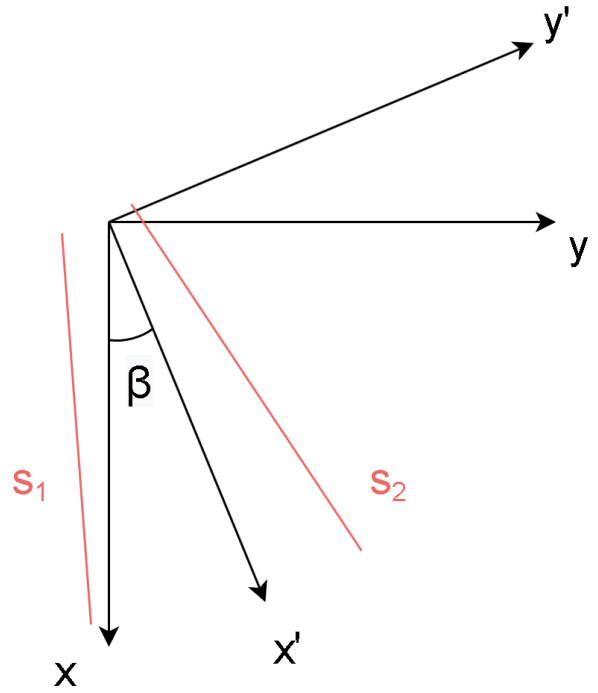


图 6　s_1，s_2 在旋转坐标系下的形式

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} cos\beta & sin\beta \\ -sin\beta & cos\beta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{17}$$

在固定坐标系下，航线解析式为 $y = tan\beta \cdot x$，测线方程可以写作：

$$y_1 = \frac{sin\beta + cos\beta M_1}{cos\beta - sin\beta M_1} x + \frac{M_1}{cos\beta tan\alpha (cos\beta - sin\beta M_1)} h_0 \tag{18}$$

$$y_2 = \frac{sin\beta - cos\beta M_2}{cos\beta + sin\beta M_2} x + \frac{M_2}{cos\beta tan\alpha (cos\beta + sin\beta M_2)} h_0 \tag{19}$$

其中

$$M_1 = \frac{cos\beta tan\alpha tan\frac{\theta}{2}}{1 - sin\beta tan\alpha tan\frac{\theta}{2}} \quad M_2 = \frac{cos\beta tan\alpha tan\frac{\theta}{2}}{1 + sin\beta tan\alpha tan\frac{\theta}{2}}$$

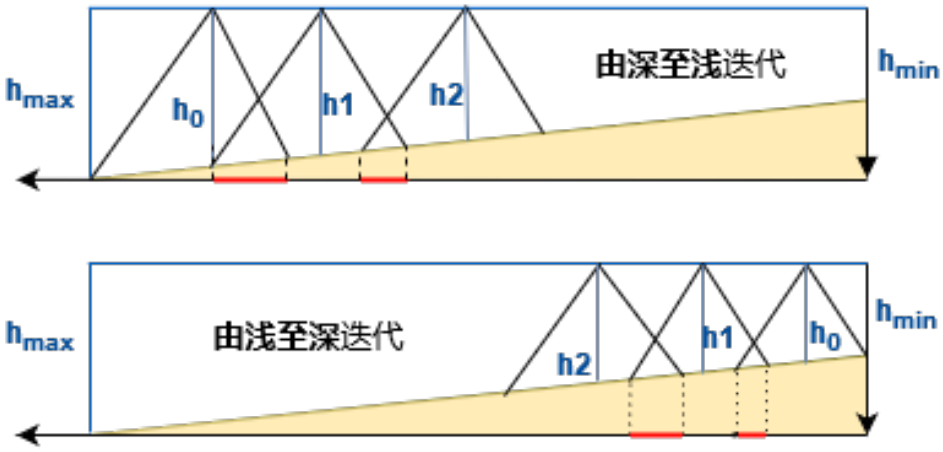


图 7　测线垂直处水深迭代法

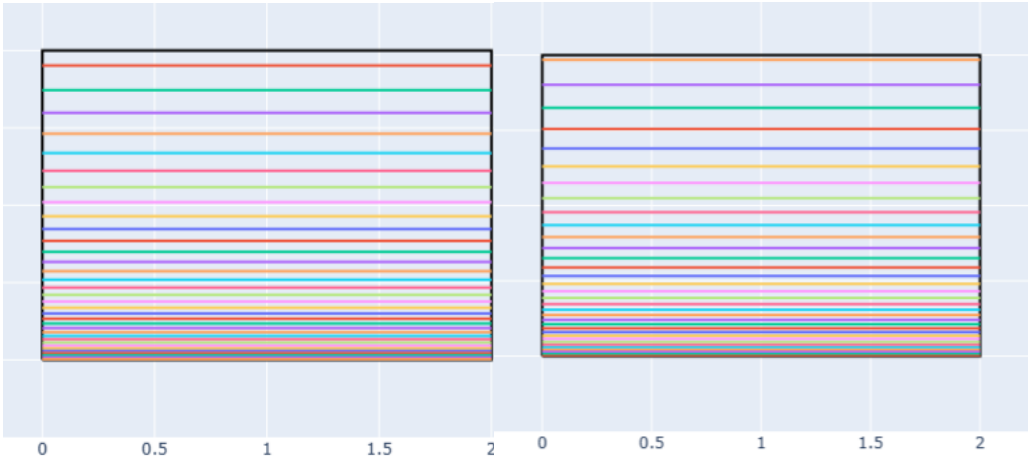
利用平面几何知识，可通过构造约束条件使第一条测线的对应截线于矩形区域边缘处重合，得到初始的测线垂直处水深。在限定平行测线间的**逐次重叠率 η 恒为 10%** 的情况下，可通过方程 18、 19 自深到浅处的侧线垂直处水深迭代公式为

$$h_{i+1} = \frac{h_i\left(\cot\alpha + (2\eta - 1)\tan\frac{\theta}{2}\right)}{\cot\alpha + \tan\frac{\theta}{2}} \tag{20}$$

自浅到深处的侧线垂直处水深迭代公式为

$$h_{i+1} = \frac{h_i\left(\cot\alpha + (1 - 2\eta)\tan\frac{\theta}{2}\right)}{\cot\alpha - \tan\frac{\theta}{2}} \tag{21}$$

下图分别于左右两侧展示自深到浅和自浅到深的测线位置迭代结果，其测线具体的放置位置存在些许不同，但方法的选择对平行测线的实际条数不存在绝对影响。在问题所给条件下，南北长 2 海里，东西宽 4 海里，海域中心深度为 110m 的矩形海域在逐次重叠率 η 恒为 10% 的前提下，获得的**最少东西方向测线的条数为 33 条**，测量长度最短为 **66 海里**。此时设计的**不等距平行测线**可达到对整个海域的完全覆盖。

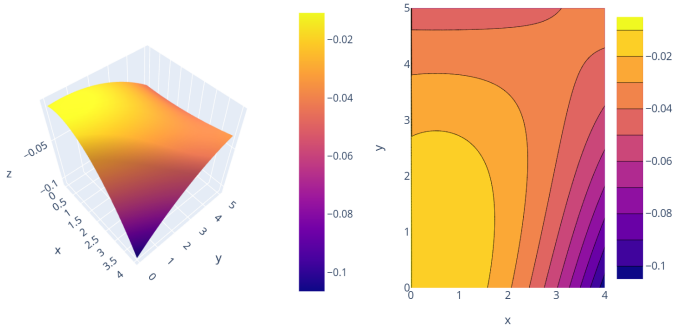


(a) 自深到浅　　**(b) 自浅到深**

5.4 问题四模型建立与求解

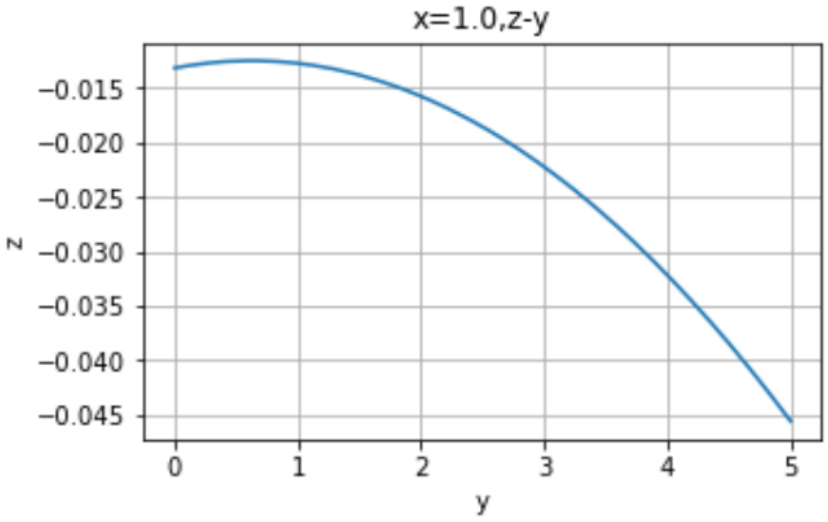
5.4.1 问题四模型建立

首先通过对题目中的数据构建出对应的曲面，并绘制出对应的等高线图，

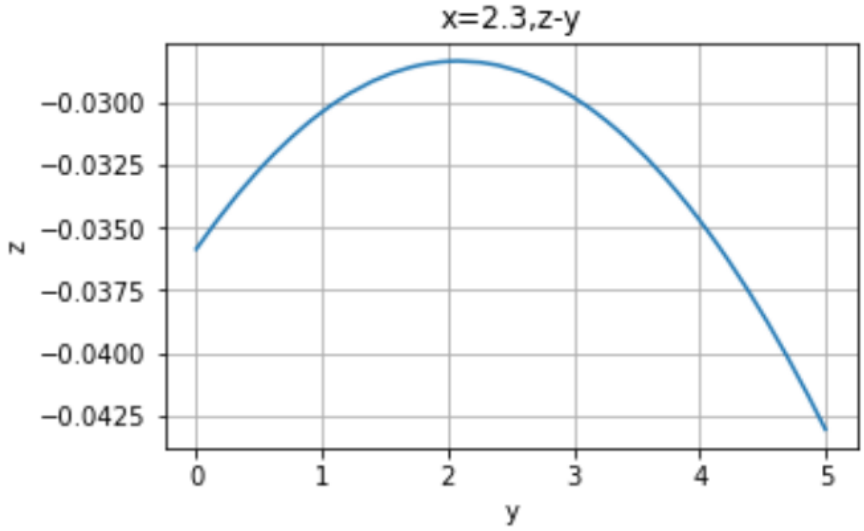


(a) 斜面示意图　　**(b) 等高线图**

可以发现题目中给定的斜面整体比较光滑，沿 x 轴方向对斜面进行微分切割，观察不同 x 坐标下的系统截面的特征。



(a) 当 $x = 1.0$ 海里时，$z - y$ 方向截面



(b) 当 $x = 2.3$ 海里时，$z - y$ 方向截面

图 10　在不同 x 坐标下系统的截面

通过观察可以发现对应的斜率较小，可以利用函数对于给定数据的在每一 x 下的最高点与最低点之间的斜率进行计算，可以得到曲面的**最大斜率倾角**约为 0.85°，

在斜率较小的情况下，可以在每一平面上近似将曲线拟合为对应的直线来计算斜面的最佳探测点位，在前几题中已经得到了对于一个规整的斜面在采用平行于等深线方向的测线可以使得整体的重叠率较为稳定的情况下，总体的测线长度也相对较短。在这里，考虑对于斜面进行微分切割将原本的曲线通过转化为直线来进行分析

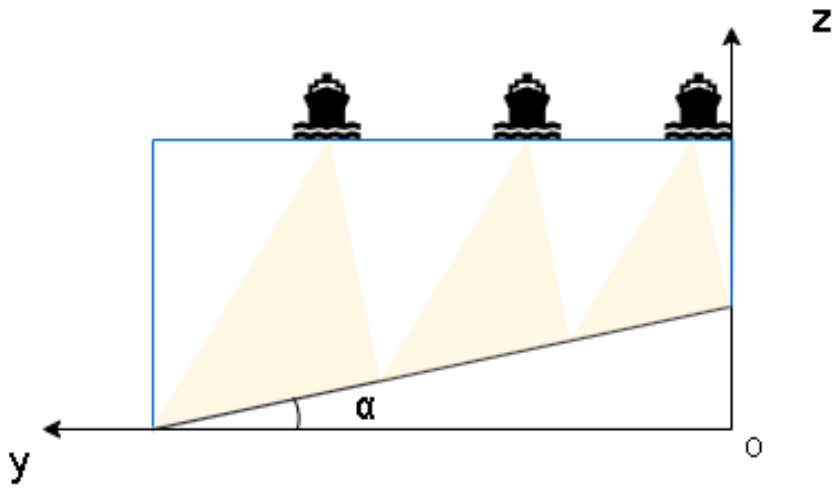


图 11　将曲线拟合为直线进行分析

5.4.2 问题四模型的求解

对于一个规则的斜角分析在上题中可以得到，通过不同的 dx, 来分割曲面扫描得到不同的点，每一个点列均代表在当前 x 下对于斜坡的最少测量数目，当以 dx 扫描完整个曲面后，会得到如下的点阵，

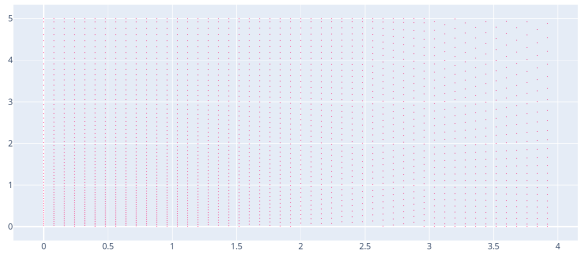
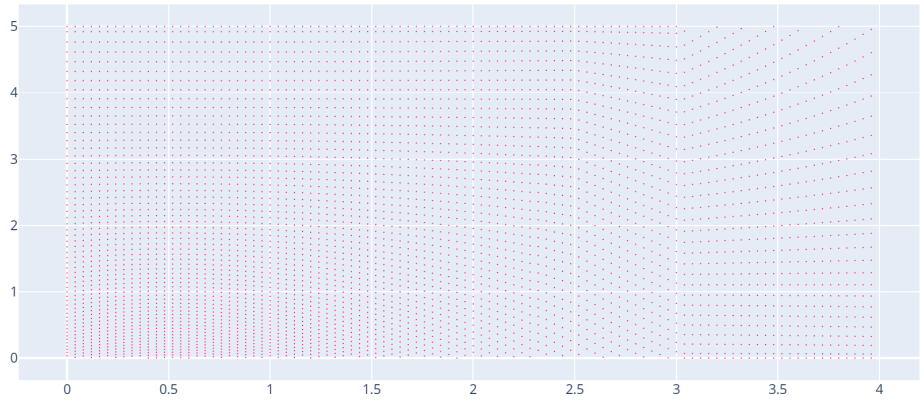
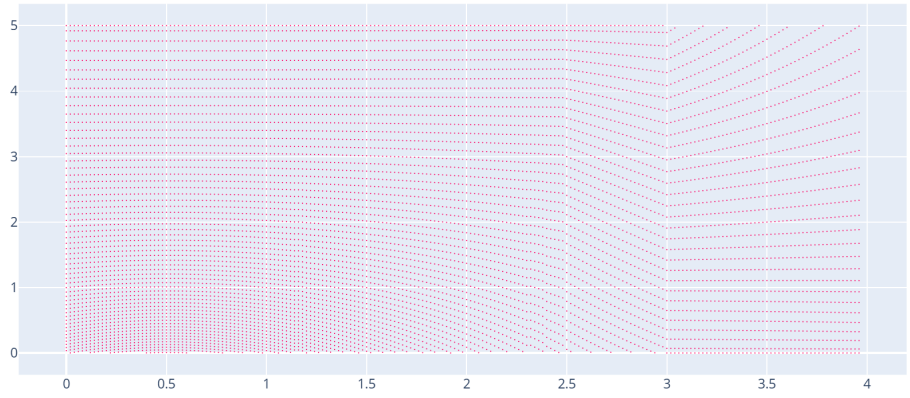


图 12　$dx = 0.08$ 时，扫描曲面得到的点阵

当不断减小 dx 提高系统的分辨率，由于点的数目不断增多，可以发现会形成较为清晰的数条测线，从此直观感受入手分析。



(a) $dx = 0.04$ **时，扫描曲面得到的点阵**



(b) $dx = 0.02$ **时，扫描曲面得到的点阵**

图 13　不同 dx **时，扫描曲面得到的点阵**

当利用足够小的 dx 构架了足够多的点之后，的想法就是如何在不同的点之间描绘出最佳路径，在这里采用的是 K 维树算法（KD-Tree，是一种用于高效处理多维空间数据的数据结构）。用于对在不同 x 之间的数据点群进行分割和组织，来快速的搜索最近邻点和计算相应的距离，并且可以提高计算机的计算小效率。通过对于位于 x 的点列和位于 $(x + dx)$ 的点列做最近邻的点搜索，计算对应点之间的距离，由此可以得到一条相对最短的路径，同时由于斜面在每一时刻的切面得到的点数不同，可能存在多个点最后连到相同点的情况，这种情况下虽然会影响路径的长度，但是由于在沿斜向直线走的过程中，会增大原本的探测面积，有助于降低漏测率，因此，考虑保留相应部分。最后得到的测线如下：

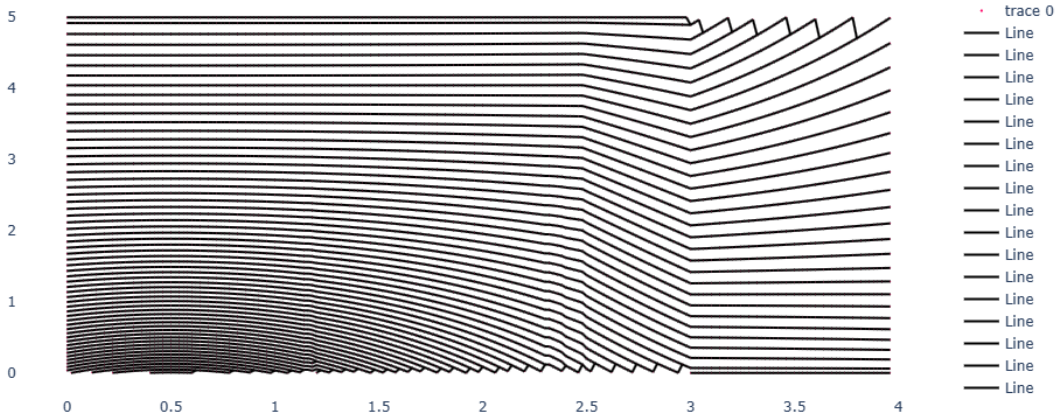


图 14　通过计算最短距离得到的理论测线

对不同 x 之间点列的最短距离累加可以得到**最后的测线总长度为:189.7356271394432 海里**

虽然通过将曲面切片为曲线，再在每一个截面上将曲线拟合为直线，通过直线的理论解来得到的最短测线长度，但是由于曲线的特殊性质与的直线情况下的点并不完全匹配，会在局部区域出现漏测的情况，而且当对给定曲面进行分析，会发现它在垂直于 x 方向的截面曲线具有凸函数的性质如图所示

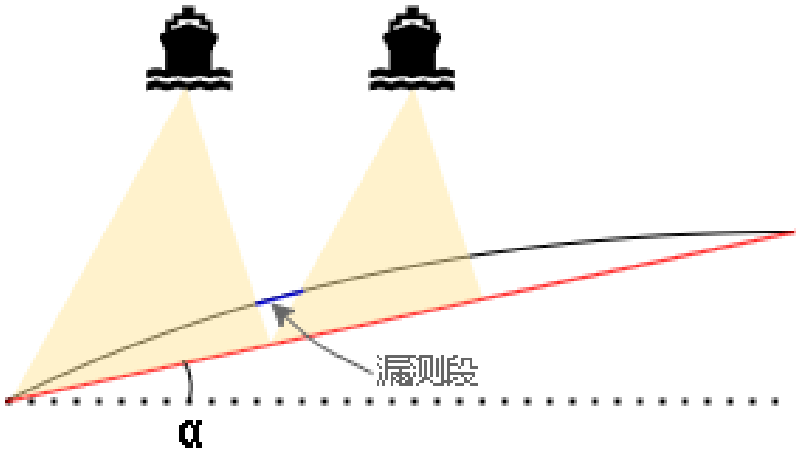


图 15　由于曲线形成的漏测部分

这时对原本的切面进行分析，对给定的切面数据进行多项式拟合，可以发现给定的切面数据可以很好的拟合成二次函数，如图所示

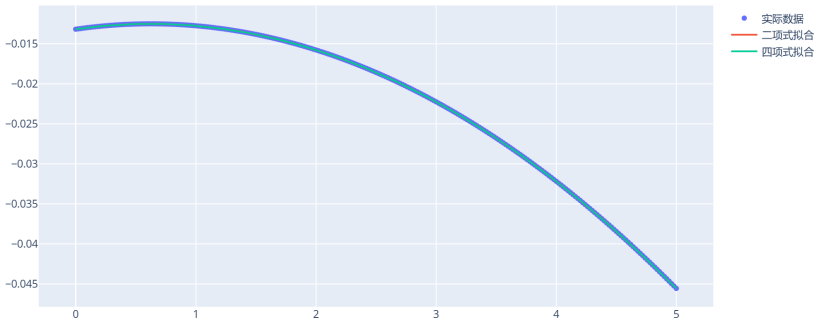


图 16　当 $x = 1.0$ 时的曲线拟合

由于我们用二次函数就可以很好的描述在不同 x 时的曲线性质，我们可以以每一 x 下的不同的最佳点为起始点建立直线系，代表我们的探测范围，由于斜率是相同的，我们可以与二次函数联立求解交点，有

$$k_1 = cot(\frac{\theta}{2}) \tag{22}$$

$$k_2 = -cot(\frac{\theta}{2}) \tag{23}$$

$$z_1 = k_1(y - D) \tag{24}$$

$$z_2 = k_2(y - D) \tag{25}$$

其中 D 为我们在指定 x 下得到的最优点列矩阵，由于我们在两种不同的斜率下会得到两种不同的交点，如图所示

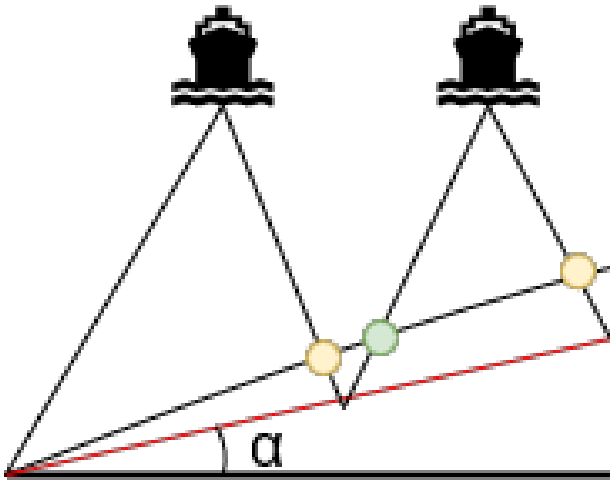


图 17　直线与拟合后的二次函数相交形成两种不同的交点

我们的漏测部分可以利用该不同两点的距离表示，考虑到曲线拟合斜率较小，我们可以将原本的弧长简化为投影到 y 轴上的投影长度，当我们从 0 点出发，还应注意**黄点和蓝点应对应不同的** D_i, 这样才反应漏测长度，我们可以用 l_i 表示在不同 x 上的漏测长度由于在 x 轴上微分取截面, 我们的理论漏测面积为:

$$S = \int l \, \mathrm{d}x \tag{26}$$

在我们 dx 取得较小的情况下可以简化为:

$$S = \sum l_i x_i \tag{27}$$

在这种情况下，经过理论计算可以得到：**漏测面积占比为：9.2%** 重叠率：由于考虑的是简化后的斜坡模型（类似图二），并且在斜坡模型上得到的理论最优解，因此我们的**重叠率接近于 0**。

六、 模型分析与检验

6.1 模型稳定性分析

6.1.1 测线水深递推式的稳定性分析

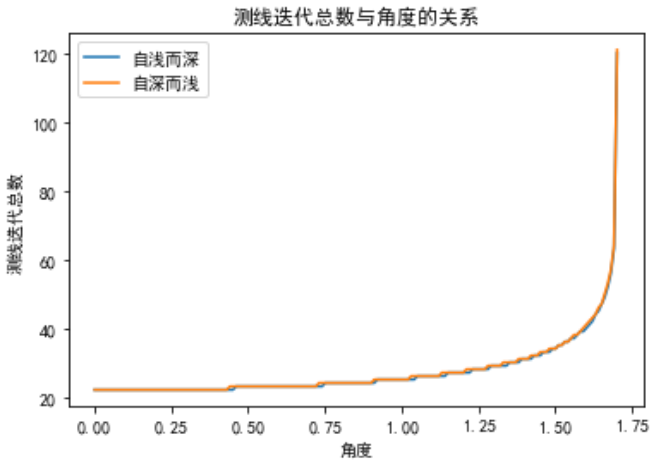


图 18　不同倾角下测线水深递推式 (平缓)

通过改变海底坡面倾角，我们观察发现当倾角增大使得 h_min 靠近 0 时，此时迭代公式会逼近零点造成测线数目激增。但此时两种方法保持近乎一致的增长速率。

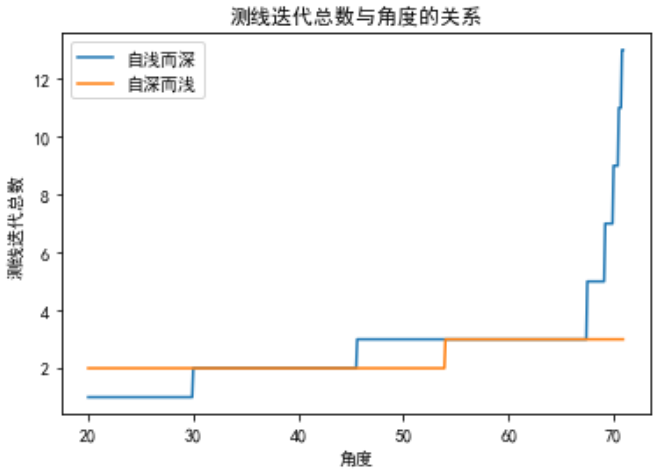


图 19　不同倾角下测线水深递推式 (陡峭)

但在倾角增大超出一定范围后，自深而浅表现出更强的鲁棒性。

七、 模型的评价

7.1 模型优点

1. 建立多波束测深在规则斜面情况下的覆盖宽度和重叠率模型，并推导可变方向测线覆盖宽度表达式，假设简洁，公式简明易用。
2. 利用最优控制方法设计矩形海域内最短测线，并通过机理建模分析假设可行性。
3. 应用截面法将原本的三维问题，转化到二维通过曲面拟合和切片处理设计测线。
4. 通过计算每个切面的最佳点分布，利用 K 维树搜索来构造测线

7.2 模型缺点

1. 对实际斜面的近似过于理想，未能考虑并处理更复杂的海底地貌。
2. 利用最优控制方法设计矩形海域内最短测线，并通过机理建模分析假设可行性。
3. 应用截面法将原本的三维问题，转化到二维通，过曲面拟合和切片处理设计测线。
4. 将二次函数拟合为直线较为理想，存在一定差异
5. 模型主要考虑测量时的测线长度和漏测面积，应均衡考虑漏测面积和重叠率的影响

7.3 模型改进与推广

进一步平衡重叠率和漏测面积的权重关系，为实际任务提供更合适的测线设计

改进曲线下最佳测量点的算法，快速高效的切片处理地图数据

考虑高阶函数曲线的地形情况下的模拟

考虑存在地形突变点时的影响

参考文献

[1] 夏伟, 刘雁春, 肖付民等. 测线布设方向的不同对海底显示的影响 [J]. 海洋测绘,2004(03):28-31.

[2] 张立华, 殷晓冬. 水深测量计划测线布设与航迹控制算法 [J]. 海洋测绘,2002(02):33-35.

附录 A　第四问求解程序——Python

```
from numba import jit
from numba import cuda
import numpy as np
import math
import matplotlib.pyplot as plt
def NcoverWidth(l,D,alpha,beta,gama,n,d):
    """
    本函数旨在通过对于指定的深度和坡度返回一个覆盖宽度
    """
    z =np.cos(beta)*np.tan(alpha)*l + D + n*d*np.tan(alpha)*np.sin(beta)
    s1 = np.sqrt(3)/(1+np.tan(gama/2)*np.sin(beta)*np.tan(alpha))*z
    s2 = np.sqrt(3)/(1-np.tan(gama/2)*np.sin(beta)*np.tan(alpha))*z
    return s1,s2

def ita(l,D,alpha,beta,gama,n,d):
    """
    定义重叠率，考虑相邻两条纹
    """
    z =np.cos(beta)*np.tan(alpha)*l + D + n*d*np.tan(alpha)*np.sin(beta)
    s1 = np.sqrt(3)/(1+np.tan(gama/2)*np.sin(beta)*np.tan(alpha))*z
    s2 = np.sqrt(3)/(1-np.tan(gama/2)*np.sin(beta)*np.tan(alpha))*z
    ita = 1 + (np.sqrt(3)*d*np.tan(alpha)
    *np.sin(beta)-d)/((1-np.sqrt(3)*np.sin(beta)*np.tan(alpha))(s1+s2))
    return ita
```

```
from numba import jit
from numba import cuda
import numpy as np
import math
import matplotlib.pyplot as plt
def AllcoverWidth(l,D,alpha,beta,gama,n1,n2,d):
    """
    本函数旨在通过对于指定的深度和坡度返回一个覆盖宽度，考虑两条线
    """
    z1 = np.cos(beta)*np.tan(alpha)*l + D + n1*d*np.tan(alpha)*np.sin(beta)
    z2 = np.cos(beta)*np.tan(alpha)*l + D + n2*d*np.tan(alpha)*np.sin(beta)
    s1a,s1b = NcoverWidth(l,D,alpha,beta,gama,n1,d)
    s2a,s2b = NcoverWidth(l,D,alpha,beta,gama,n2,d)
    width = s1a+d+s2b
    return width


import pandas as pd
from numba import jit
```

```
from numba import cuda
import numpy as np
import math
import matplotlib.pyplot as plt


#导入文件
！！此时将文件的表头去掉
xlsx = pd.read_excel('data.xlsx')
print(xlsx)
arr = xlsx.values
x = np.arange(0,4.02,0.02)
y = np.arange(0,5.02,0.02)
X,Y = np.meshgrid(x,y)
z = arr
Z = z[:,1:202]
！！Z代表实际的深度数据矩阵

#绘制曲面图
import plotly.graph_objects as go

fig = go.Figure(data=[go.Surface(x=X, y=Y, z=-Z/1852)])
fig.update_layout(title='Surface', autosize=False,
                  width=500, height=500,
                  margin=dict(l=65, r=50, b=65, t=90))
fig.show()

#绘制等高线图
import plotly.graph_objects as go

fig = go.Figure(data =
    go.Contour(
        z=-z/1852,
        x=x, # horizontal axis
        y=y # vertical axis
    ))
fig.update_layout(title='Contour', autosize=False,
                  width=400, height=500,
                   xaxis=dict(
        title='x',
        range=[0, 4] # X轴范围
                  ),
                  yaxis=dict(
        title='y',
        range=[0, 5] # X轴范围
                  ))
fig.show()
```

```
#绘制截面
def drawCut(x):
    """
    构建指定x坐标下，沿y方向的z-y截面
    """
    fig = plt.figure(figsize=(5,3))
    ax1 = fig.add_subplot(1,1,1)
    n = int(x/0.02)
    ax1.plot(y,-Z[:,n]/1852)
    ax1.grid('on')
    ax1.set_xlabel('y')
    ax1.set_ylabel('z')
    ax1.set_title(f'x={x},z-y')

drawCut(3.0)

#多项式拟合

z1 = np.polyfit(y,-Z[:,0]/1852 ,3) #用5次多项式拟合，输出系数从高到0
p1 = np.poly1d(z1) #使用次数合成多项式
y_pre = p1(y)
plt.plot(y,-Z[:,0]/1852,'.')
plt.plot(y,y_pre)
plt.title('x=0 ,z-y')
plt.show()

#绘图


import plotly.graph_objs as go
import numpy as np
import plotly.offline as of


trace0 = go.Scatter(
    x = y_try,
    y = z_try,
    mode = 'markers',
    name = '实际数据'
)
trace1 = go.Scatter(
    x = y_try,
    y = y_pre1,
    mode = 'lines',
    name = '二项式拟合'
)
```

```
trace2 = go.Scatter(
    x = y_try,
    y = y_pre2,
    mode = 'lines',
    name = '四项式拟合'
)
data = [trace0, trace1, trace2]

fig = go.Figure(data = data)
fig.show()

#通过扫描曲面得到近似角度

def Alpha_2(dx):
    """
    针对不同dx,返回沿x轴切片得到的等效倾角
    考虑到提供的数据dx为0.02的倍数
    res_a矩阵代表我们得到的倾角序列
    flag矩阵代表倾角和我们初始点的朝向
    1: 从深到浅
    0: 中间区域
    -1: 从浅到深
    返回弧度
    """
    res_a = []
    flag = []
    x_cut = np.arange(0,4,dx)
    for i in x_cut:
        n = int(i/0.02)
        if i < 2.0:
            a = np.arctan((Z[-1,n]-Z[0,n])/5.0/1852)##记得化为海里
            res_a.append(a)
            flag.append(1)
        if i >=2.0 and i <=3.0:
            ##暂时还没决定怎么划分等效角
            a = np.arctan(np.abs(Z[-1,n]-Z[0,n])/5.0/1852)
            res_a.append(a)
            flag.append(0)
        if i >3.0:
            a = np.arctan((Z[0,n]-Z[-1,n])/5.0/1852)
            res_a.append(a)
            flag.append(-1)
    return res_a,flag

theta = np.pi/3*2
beta = np.pi/2
def two_line_parallel(h, D):
```

```
    global alpha,theta,beta
    y1h=math.tan(theta/2)/(1-math.tan(alpha) * math.tan(theta/2))*h
    y2h=math.tan(theta/2)/(1+math.tan(alpha) * math.tan(theta/2))*h
    return [y1h,y2h]
def down_to_h(y_down):
    global alpha,theta,beta
    h=y_down*(1-math.tan(alpha) * math.tan(theta/2))/math.tan(theta/2)
    return h
def up_to_h(y_up):
    global alpha,theta,beta
    h=y_up*(1+math.tan(alpha) * math.tan(theta/2))/math.tan(theta/2)
    return h
def H_iteration_down(h):
    global alpha,theta,beta
    n=0
    H=(2*h*math.tan(theta/2)*n/(1-math.tan(theta/2)**2*math.tan(alpha)**2)+h/math.tan(alpha)-h
        *math.tan(theta/2)/(1 +
        math.tan(theta/2)*math.tan(alpha)))/(math.tan(theta/2)/(1-math.tan(theta/2)*math.tan(alpha))+1/math.tan
    return H
def H_iteration_up(h):
    global alpha,theta,beta
    n=0
    H=(2*h*math.tan(theta/2)*n/(1-math.tan(theta/2)**2*math.tan(alpha)**2)-h/math.tan(alpha)-h
        *math.tan(theta/2)/(1 -
        math.tan(theta/2)*math.tan(alpha)))/(math.tan(theta/2)/(1+math.tan(theta/2)*math.tan(alpha))-1/math.tan
    return H
# 自深到浅
def line_From_deep_to_shallow(h_max,h_min,length):
    H_list=[h_max]
    D_list=[length]
    D_list.append(length-h_max*math.tan(theta/2))
    H_list.append(down_to_h(D_list[0]-D_list[1]))
    i=1
    while ((length-(H_list[0]-H_list[i])/math.tan(alpha))>(two_line_parallel(H_list[i],0)[1])):
        h=H_iteration_down(H_list[i])
        if h<=h_min:
            break
        H_list.append(h)
        i+=1
#     print(H_list)
    D_list=length-(h_max-np.array(H_list))/math.tan(alpha)
#     print(D_list)
    print("总条数",i)
    return i,np.array(H_list),D_list

# 自浅到深
def line_From_shallow_to_deep(h_max,h_min,length):
```

```
    H_list=[h_min]
    D_list=[0]
    D_list.append(h_min*math.tan(theta/2))
    H_list.append(up_to_h(D_list[1]-D_list[0]))
   # print(1)
    i=1
    #while (length-(H_list[i]-H_list[0])/math.tan(alpha))>two_line_parallel(H_list[i],0)[0]:
     #   H_list.append(H_iteration_up(H_list[i]))
      # print(2)
       # i+=1
    while (length-(H_list[i]-H_list[0])/math.tan(alpha))>two_line_parallel(H_list[i],0)[0]:
        h=H_iteration_up(H_list[i])
        if h>=h_max:
            break
        H_list.append(h)
        i+=1
#     print(H_list,i)
    D_list=(-h_min+np.array(H_list))/math.tan(alpha)
    print("总条数",i)
    return i,np.array(H_list),D_list
#     print(D_list)


#针对不同dx扫描得到的点列
##设置常数
dx = 0.02
length = 5
res_D = []
res_H = []
alpha_list,flag_list = Alpha_2(dx)
#print(alpha_list)
#print(flag)

##由dx可知一共将x分为
xj_range =np.arange(0,4.0,dx)

##对应矩阵中的下标为
j_range = xj_range/0.02

for j in range(int(4.0/dx)):
    alpha = alpha_list[j]
    flag = flag_list[j]
    if flag ==1:
        ##深到浅
        h_max = Z[-1,int(j_range[j])]/1852
        h_min = Z[0,int(j_range[j])]/1852
        i,res_H_list,res_D_list = line_From_deep_to_shallow(h_max,h_min,length)
```

```
        res_D.append(res_D_list)
        res_H.append(res_H_list)
    if flag == 0:
        ##二次函数拟合(考虑到倾角较小，我们近似为三角情况)
        if Z[-1,int(j_range[j])]/1852 > Z[0,int(j_range[j])]/1852 :
            h_max = Z[-1,int(j_range[j])]/1852
            h_min = Z[0,int(j_range[j])]/1852
            i,res_H_list,res_D_list = line_From_deep_to_shallow(h_max,h_min,length)
            res_D.append(res_D_list)
            res_H.append(res_H_list)
        if Z[-1,int(j_range[j])]/1852 == Z[0,int(j_range[j])]/1852 :
            print("出现异常")
        if Z[-1,int(j_range[j])]/1852< Z[0,int(j_range[j])]/1852 :
            h_max = Z[0,int(j_range[j])]/1852
            h_min = Z[-1,int(j_range[j])]/1852
            i,res_H_list,res_D_list = line_From_deep_to_shallow(h_max,h_min,length)
            res_D.append(res_D_list)
            res_H.append(res_H_list)
    if flag ==-1:
        ##浅到深
        h_max = Z[0,int(j_range[j])]/1852
        h_min = Z[-1,int(j_range[j])]/1852
        i,res_H_list,res_D_list = line_From_shallow_to_deep(h_max,h_min,length)
        res_D.append(res_D_list)
        res_H.append(res_H_list)

#绘制点列散点图
#切片该x时的散点
res_x = []
res_y = []
for i in range(len(res_D)):
    length_x = dx*i
    ##扫描每一个点作图
    for y in res_D[i]:
        res_x.append(length_x)
        res_y.append(y)
        import plotly.offline as of
import plotly.graph_objs as go
import plotly
trace = go.Scatter(
    x = res_x,
    y = res_y,
    mode ='markers', #纯散点图
    marker = dict(
        size = 1,
        color = 'rgba(255,0,100,9)'
    )
```

```
)
data = trace
fig = go.Figure(data = data)
fig.show()

##测量截面最大角
max_Z = []
min_Z = []
for t in range(len(x)):
    max_Z.append(max(Z[:,t]))
    min_Z.append(min(Z[:,t]))
theta_max = max((np.array(max_Z)-np.array(min_Z))/1852/5)
theta = np.arctan(theta_max)
##转化为角度
np.degrees(theta)

##测量漏测面积

def ABC(i):
    k1 = np.tan(np.pi/6)
    k2 = -np.tan(np.pi/6)
    D_list_try = res_D[i]
    z_try = -Z_mile[:,i]
    y_try = Y[:,i]
    #对于给定的粒子列表d我们有
    def varD1(var):
        """
        输入指定的Dlist,返回不同斜率的直线群
        直线均为
        z = k1(y-D_list)
        """
        return k1*(y-d_i)
    def varD2(var):
        """
        输入指定的Dlist,返回不同斜率的直线群
        直线均为
        z = k1(y-D_list)
        """
        return k2*(y-d_i)

    pz = np.polyfit(y_try,z_try ,2) #用2次多项式拟合，输出系数从高到0
    p = np.poly1d(pz) #使用次数合成多项式
    res_1 = []
    res_2 = []
    ##拟合后得到函数p1
    for d_i in D_list_try:
        result1 = fsolve(lambda var: p(var) - varD1(var), 0) # 初始猜测值为0
```

```
        result2 = fsolve(lambda var: p(var) - varD2(var), 0) # 初始猜测值为0
        if result1 <= 5 and result1>= 0 :
            res_1.append(result1)
            #print(result1)
        ##联立p1与直线系求解交点的y
        if result2 <= 5 and result2 >= 0 :
            res_2.append(result2)
            #print(result2)
    res_1 = np.array(res_1)
    res_2 = np.array(res_2)
    #print(res_2)
    ##计算最短距离
    distance = 0
    if res_2.size > 0:
        if res_1.size > 0:
            for res_i in res_2:
                    dis = (res_1 - res_i)
                    #print(dis)
                    if max(dis)>0:
                        distance += min(dis[dis > 0])
    return distance

dis_all = []
sqre = 0
for i in range(199):
    dis_res = ABC(i)
    dis_all.append(dis_res)
    print(dis_res)
    sqre += dis_res * 0.02

sqre_res = sqre/20*100

#计算覆盖宽度
from numba import jit
from numba import cuda
import numpy as np
import math
import matplotlib.pyplot as plt
def coverWidth(l,D,alpha,beta,gama):
    """
    本函数旨在通过对于指定的深度和坡度返回一个覆盖宽度
    l:据海域距离
    beta:侧线方向夹角
    D:海域中心深度
    alpha:坡度
    gama:开角
    """
```

```
    theta = np.arctan(np.sin(beta)*np.tan(alpha))
    z =np.cos(beta)*np.tan(alpha)*l + D
    s = np.sqrt(3)*0.5*(1/np.cos(gama/2-theta) + 1/np.cos(gama/2+theta))*z
    return s
## 投影到x-y平面的覆盖宽度
from numba import jit
from numba import cuda
import numpy as np
import math
import matplotlib.pyplot as plt

def NcoverWidth(l,D,alpha,beta,gama,n,d):
    """
    本函数旨在通过对于指定的深度和坡度返回一个覆盖宽度
    计算n条时的间距
    l:据海域距离
    beta:侧线方向夹角
    D:海域中心深度
    alpha:坡度
    gama:开角
    n:代表第n条线(设初始直线为0)
    d:代表不同线的间距
    """
    z =np.cos(beta)*np.tan(alpha)*l + D - n*d*np.tan(alpha)*np.sin(beta)
    s1 = np.sqrt(3)/(1-np.tan(gama/2)*np.sin(beta)*np.tan(alpha))*z
    s2 = np.sqrt(3)/(1+np.tan(gama/2)*np.sin(beta)*np.tan(alpha))*z
    return s1,s2

def AllcoverWidth(l,D,alpha,beta,gama,n1,n2,d):
    """
    本函数旨在通过对于指定的深度和坡度返回一个覆盖宽度
    计算n条时的间距
    l:据海域距离
    beta:侧线方向夹角
    D:海域中心深度
    alpha:坡度
    gama:开角
    n:代表第n条线(设初始直线为0)
    d:代表不同线的间距
    """
    z1 = np.cos(beta)*np.tan(alpha)*l + D + n1*d*np.tan(alpha)*np.sin(beta)
    z2 = np.cos(beta)*np.tan(alpha)*l + D + n2*d*np.tan(alpha)*np.sin(beta)
    s1a,s1b = NcoverWidth(l,D,alpha,beta,gama,n1,d)
    s2a,s2b = NcoverWidth(l,D,alpha,beta,gama,n2,d)
    width = s1a+d+s2b
    return width
```

```
##重叠率计算
def ita(l,D,alpha,beta,gamma,d1,d2):
    """
    定义重叠率，考虑相邻两条纹
    l:据海域距离
    beta:侧线方向夹角
    D:海域中心深度
    alpha:坡度
    gama:开角
    n:代表第n条线(设初始直线为0，以海域最深点为参考点)
    d:代表不同线的间距
    返回s1,s2分别为投影长度
    """
    za = np.cos(beta)*np.tan(alpha)*l + D - d1*np.tan(alpha)*np.sin(beta)
    zb = np.cos(beta)*np.tan(alpha)*l + D - d2*np.tan(alpha)*np.sin(beta)
    sa1 = np.tan(gamma/2)/(1-np.tan(gamma/2)*np.sin(beta)*np.tan(alpha))*za
    sa2 = np.tan(gamma/2)/(1+np.tan(gamma/2)*np.sin(beta)*np.tan(alpha))*za
    ita = 1 - (np.tan(gamma/2)*(d2-d1)*np.tan(alpha)*np.sin(beta))/((1 -
        np.tan(gamma/2)*np.sin(beta)*np.tan(alpha))*(sa1+sa2))- (d2-d1)/(sa1+sa2)
    return ita
gamma = np.pi*2/3
alpha = math.radians(1.5)
beta = np.arange(-np.pi,np.pi,0.01)
res = ita(0,70,alpha,beta,gamma,0,200)

from numba import jit
from numba import cuda
import numpy as np
import math
import matplotlib.pyplot as plt
def HcoverWidth(l,D,alpha,beta,gama,h1,h2):
    """
    本函数旨在通过对于指定的深度和坡度返回一个覆盖宽度
    计算n条时的间距
    l:据海域距离
    beta:侧线方向夹角
    D:海域中心深度
    alpha:坡度
    gama:开角
    n:代表第n条线(设初始直线为0)
    d:代表不同线的间距
    """
    z =np.cos(beta)*np.tan(alpha)*l + D - h1*np.tan(alpha)*np.sin(beta)
    s1 = np.sqrt(3)/(1-np.tan(gama/2)*np.sin(beta)*np.tan(alpha))*z
    s2 = np.sqrt(3)/(1+np.tan(gama/2)*np.sin(beta)*np.tan(alpha))*z
    s = (s1+s2)/np.cos(math.radians(1.5))
    return s
```

附录 B　斜坡上截线方程——Python

```
def Zip_y1_1(beta,alpha,theta):
return
    math.cos(beta)*math.tan(theta/2)*math.tan(alpha)/(1-math.tan(theta/2)*math.sin(beta)*math.tan(alpha))
def Zip_y2_1(beta,alpha,theta):
return
    -math.cos(beta)*math.tan(theta/2)*math.tan(alpha)/(1+math.tan(theta/2)*math.sin(beta)*math.tan(alpha))
def Zip_y1_2(beta,alpha,theta,h):
return math.tan(theta/2)/(1-math.tan(theta/2)*math.sin(beta)*math.tan(alpha))*h
def Zip_y2_2(beta,alpha,theta,h):
return -math.tan(theta/2)/(1+math.tan(theta/2)*math.sin(beta)*math.tan(alpha))*h

def Get_k_d(h,beta,theta=np.pi*2/3, alpha=1.5/180*np.pi):
k1=(Zip_y1_1(beta,alpha,theta)*math.cos(beta)+math.sin(beta))/(math.cos(beta)-Zip_y1_1(beta,alpha,theta)*mat
b1=Zip_y1_2(beta,alpha,theta,h)/(math.cos(beta)-Zip_y1_1(beta,alpha,theta)*math.sin(beta))
k2=(Zip_y2_1(beta,alpha,theta)*math.cos(beta)+math.sin(beta))/(math.cos(beta)-Zip_y2_1(beta,alpha,theta)*mat
b2=Zip_y2_2(beta,alpha,theta,h)/(math.cos(beta)-Zip_y2_1(beta,alpha,theta)*math.sin(beta))
return k1, b1, k2, b2

def Get_( beta,theta=np.pi*2/3, alpha=1.5/180*np.pi):
#    h=110/1852-2*math.tan(1.5/180*np.pi)
global H
h=H
k1=(Zip_y1_1(beta,alpha,theta)*math.cos(beta)+math.sin(beta))/(math.cos(beta)-Zip_y1_1(beta,alpha,theta)*mat
b1=Zip_y1_2(beta,alpha,theta,h)/(math.cos(beta)-Zip_y1_1(beta,alpha,theta)*math.sin(beta))
k2=(Zip_y2_1(beta,alpha,theta)*math.cos(beta)+math.sin(beta))/(math.cos(beta)-Zip_y2_1(beta,alpha,theta)*mat
b2=Zip_y2_2(beta,alpha,theta,h)/(math.cos(beta)-Zip_y2_1(beta,alpha,theta)*math.sin(beta))
return k1, b1, k2, b2
```

附录 C　方程式化简——Mathematica

```
 In[431]:= (*定义变量*)h=Symbol["h"];
beta=Symbol["beta"];
alpha=Symbol["alpha"];
theta=Symbol["theta"];

(*定义斜率和截距表达式*)
y1prime=(h*Cos[beta]^2*Tan[alpha]*Tan[theta/2]-(Sin[beta]*Tan[alpha]*Tan[theta/2]-1)*Sin[beta])/((-h*Sin[bet
y2prime=(-h*Cos[beta]^2*Tan[alpha]*Tan[theta/2]+(Sin[beta]*Tan[alpha]*Tan[theta/2]+1)*Sin[beta])/((h*Sin[bet
y10=h*Tan[theta/2]/((-h*Sin[beta]*Tan[alpha]*Tan[theta/2]-Sin[beta]*Tan[alpha]*Tan[theta/2]+1)*Cos[beta]);
```

```
y20=-h*Tan[theta/2]/((h*Sin[beta]*Tan[alpha]*Tan[theta/2]+Sin[beta]*Tan[alpha]*Tan[theta/2]+1)*Cos[beta]);

(*化简直线斜率和截距*)
y1prime_simplified=FullSimplify[y1prime];
y2prime_simplified=FullSimplify[y2prime];
y1_ 0_simplified=FullSimplify[y10];
y2_ 0_simplified=FullSimplify[y20];

(*打印化简结果*)
Print["斜率 y1': ",y1prime_simplified];
Print["斜率 y2': ",y2prime_simplified];
Print["截距 y1(0): ",y1_ 0_simplified];
Print["截距 y2(0): ",y2_ 0_simplified];
\:6B63\:5728\:8BA1\:7B97In[431]:= Set::write: 0 _simplified y1_ 中的标签 Times 被保护.
\:6B63\:5728\:8BA1\:7B97In[431]:= Set::write: 0 _simplified y2_ 中的标签 Times 被保护.
\:6B63\:5728\:8BA1\:7B97In[431]:= 斜率 y1': y1prime_simplified
\:6B63\:5728\:8BA1\:7B97In[431]:= 斜率 y2': y2prime_simplified
\:6B63\:5728\:8BA1\:7B97In[431]:= 截距 y1(0): 0
\:6B63\:5728\:8BA1\:7B97In[431]:= 截距 y2(0): 0
In[447]:= FullSimplify[y1prime]
Out[447]= (-Tan[beta]+(-((1+h) Cos[beta])+Sec[beta]) Tan[alpha] Tan[theta/2])/(-1+(1+h)
    Sin[beta] Tan[alpha] Tan[theta/2])
In[448]:= FullSimplify[y2prime]
Out[448]= (Tan[beta]+(-((1+h) Cos[beta])+Sec[beta]) Tan[alpha] Tan[theta/2])/(1+(1+h)
    Sin[beta] Tan[alpha] Tan[theta/2])
In[449]:= FullSimplify[y10]
Out[449]= -((h Sec[beta] Tan[theta/2])/(-1+(1+h) Sin[beta] Tan[alpha] Tan[theta/2]))
In[450]:= FullSimplify[y20]
Out[450]= -((h Sec[beta] Tan[theta/2])/(1+(1+h) Sin[beta] Tan[alpha] Tan[theta/2]))
In[451]:=
    FullSimplify[(2*h*Tan[theta/2]*n/(1-Tan[theta/2]^2*Tan[alpha]^2)+h/Tan[alpha]-h*Tan[theta/2]/(1+Tan[thet
Out[451]= (h (Cot[alpha]+(-1+2 n) Tan[theta/2]))/(Cot[alpha]+Tan[theta/2])
In[452]:=
    FullSimplify[(2*h*Tan[theta/2]*n/(1-Tan[theta/2]^2*Tan[alpha]^2)-h/Tan[alpha]-h*Tan[theta/2]/(1-Tan[thet
Out[452]= (h (Cot[alpha]+(1-2 n) Tan[theta/2]))/(Cot[alpha]-Tan[theta/2])
```

附录 D　前两问题求解——MATLAB

```
%第一问求解
x1=-800:200:800; %一条测线距中心位置处的距离
d=200;
x2=x1-d; %与其相邻的测线
%参数
h0=70;
alpha=1.5/180*pi;
```

```
theta=2*pi/3;
%测线1
s1=(h0-x1.*tan(alpha)).*tan(theta/2)./(1-tan(alpha).*tan(theta/2));
s2=(h0-x1.*tan(alpha)).*tan(theta/2)/(1+tan(alpha).*tan(theta/2));
%与其相邻测线
s3=(h0-x2.*tan(alpha)).*tan(theta/2)./(1-tan(alpha).*tan(theta/2));
s4=(h0-x2.*tan(alpha)).*tan(theta/2)./(1+tan(alpha).*tan(theta/2));
%海水深度
h=h0-x1.*tan(alpha);
%覆盖宽度
W=(s1+s2)/cos(alpha);
%重叠率
eta=(s1+s4-d)./(s3+s4);

%第二问求解
R=0:0.3:2.1;
R=R*1852;
phi_list=0:pi/4:7*pi/4;
result=zeros([8,8]);
for i=1:8
    phi=phi_list(i);
    s1=(120+R.*(cos(phi).*tan(1.5/180*pi))).*tan(pi/3)./(1-sin(phi).*tan(1.5/180*pi).*tan(pi/3));
    s2=(120+R.*(cos(phi).*tan(1.5/180*pi))).*tan(pi/3)./(1+sin(phi).*tan(1.5/180*pi).*tan(pi/3));
    s=(s1+s2)./cos(atan(sin(phi)*tan(1.5/180*pi)));
    result(i,:)=s;
end

%第二问L与 关系图
R=yy;
phi=xx;
s1=(120+1852*R.*(cos(phi).*tan(alpha))).*tan(pi/3)./(1-sin(phi).*tan(alpha).*tan(pi/3));
s2=(120+1852*R.*(cos(phi).*tan(alpha))).*tan(pi/3)./(1+sin(phi).*tan(alpha).*tan(pi/3));
s=(s1+s2)./cos(atan(sin(phi)*tan(1.5/180*pi)));
% s1=s/1852; %单位换成海里
surf(xx,yy,s);
xlabel(' (rad)')
ylabel('L(海里)')
zlabel('W(m)')

%第三问xOy坐标系下测线与截线的方程
beta=pi/6;
alpha=1.5/180*pi;
y=tan(beta)*x;
h0=1;
x=0:0.01:5;
M1=(cos(beta).*tan(alpha).*tan(pi/3))./(1-tan(pi/3).*sin(beta)*tan(alpha))*h0;
M2=(cos(beta).*tan(alpha).*tan(pi/3))./(1+tan(pi/3).*sin(beta)*tan(alpha))*h0;
```

```
y=tan(beta)*x;
y1=((sin(beta)+cos(beta).*M1)*x+M1./cos(beta)./tan(alpha))./(cos(beta)-sin(beta)*M1);
y2=((sin(beta)-cos(beta).*M2)*x-M2./cos(beta)./tan(alpha))./(cos(beta)+sin(beta)*M2);
plot(x,y,x,y1,x,y2)
```