参赛承诺书

提交包含此承诺书的 pdf 文件，表明所有此文件的作者共同承诺：

我们完全清楚，在竞赛开始后参赛队员不能以任何方式，包括电话、电子邮件、“贴吧”、QQ 群、微信群等，与队外的任何人（包括指导教师）交流、讨论与赛题有关的问题；无论主动参与讨论还是被动接收讨论信息都是严重违反竞赛纪律的行为。

我们以中国大学生名誉和诚信郑重承诺，严格遵守竞赛章程和参赛规则，以保证竞赛的公正、公平性。如有违反竞赛章程和参赛规则的行为，我们将受到严肃处理。

我们授权北京理工大学数学建模竞赛组织方，可将我们的论文以任何形式进行公开展示（包括进行网上公示，在书籍、期刊和其他媒体进行正式或非正式发表等）。

2024.4

基于二次去噪抑制海杂波下对海面目标的时频分析检测

摘要

海杂波是指雷达电磁波照射到海表面，再接收到的反向散射回波。海杂波产生的物理机理复杂，受到环境因素和雷达设备参数的影响，现阶段是干扰雷达监测海面目标的主要成分。为了研究海杂波问题，并在海杂波影响下得到更好的目标检测效果，本文首先构建了**JONSWAP 谱模型**作为海浪模型，利用**蒙特卡洛法**生成海域内的波浪场和海浪仿真。在上述海浪模型基础上，利用**短时傅里叶变换，提取相对时频脊积累值、相对时频脊多普勒频率变化值**的特征，对回波数据进行分类，实现目标检测。最后，在此基础上，我们采用**高斯滤波、特异性滤波**和 OpenCV 库 **NLM 滤波算法**进行了**二次去噪**并把图像**二值化处理**实现对海杂波的过滤，成功地在海杂波的干扰下分析检测出海面目标。

关键词：海杂波；时频分析检测；海浪谱模型；短时傅里叶变换；二次去噪

基于二次去噪抑制海杂波下对海面目标的时频分析检测

一、问题背景

海杂波是指雷达电磁波照射到海表面后，由于海面起伏、波浪运动等因素而产生的反向散射回波.雷达作为一种重要的海上探测手段，其发射无线电波，然后通过接收反射回来的无线电波来探测目标。其性能的提升对于海洋资源的开发、海上安全监控以及海洋环境研究都具有至关重要的作用。随着雷达技术的不断发展，雷达系统的分辨率和探测距离不断提高[1]，但雷达电磁波在监测海面目标时，不可避免地会受到海杂波的干扰。这种干扰不仅影响雷达对目标的有效探测，还增加了雷达信号处理的复杂性和难度[2]。因此，对海杂波进行深入的建模与分析是提高雷达海面目标检测能力的关键。

海浪模型能够模拟海浪的起伏和运动，从而促进对海杂波的特性和影响的深入研究。通过海浪模型，可以分析不同海况下海浪的特性和规律，为海杂波的建模和分析提供重要依据[3]。

海面目标的检测对于海洋资源的开发、海上交通的监控以及海上安全保卫都具有重要意义。然而，海面目标的多样性和复杂性给雷达探测带来了很大挑战。船只、鲸鱼、岛屿等海面目标具有不同的反射特性和运动规律，需要雷达系统具备较高的分辨率和灵敏度才能进行有效探测[4]。同时，海杂波的干扰也会降低雷达对海面目标的探测性能，增加了探测难度。

海杂波对雷达监测海面目标的影响主要表现在两个方面：一是降低了雷达的探测距离和分辨率，使得雷达难以发现远距离或小型目标；二是增加了雷达的虚警率，使得雷达在探测过程中产生大量无效报警[5]。这些影响都严重制约了雷达在海上探测领域的应用和发展。为了解决海杂波对雷达监测海面目标的影响问题，需要建立一种合理的海杂波过滤数学模型。该模型能够根据雷达接收到的信号特性，有效地分离出海杂波和有用信号，提高雷达对海面目标的探测性能。通过海杂波过滤数学模型的研究和应用，可以为雷达在海上探测领域的应用提供更加可靠和有效的技术支持，以获取更好的海面目标检测效果。

二、问题分析

由于海面的复杂性和动态性，海杂波的特性多变，对雷达目标检测构成严重干扰。为优化雷达性能，需要深入了解海杂波的产生机理和影响因素。构建准确的海浪模型是理解海杂波特性的基础，有助于预测和模拟海浪行为。本文根据所给的雷达对海探测数据集[6]，设计时频分析检测方法，具体探究海杂波建模及分析问题，以更好地理解和应对海杂波带来的挑战。

2.1 问题一的分析

研究海杂波问题，首先需要构建一个海浪模型。现有的海浪模型主要线性模型、非线性模型、谱模型、深水波动方程模型、浅水模型、耦合模型等等。我们分析比较了个海浪模型的优缺点，再加上由于海杂波通常是由多种频率成分的波浪混合而成的，谱模型能够提供波浪能量随频率变化的信息，帮助了解海洋波浪的频谱特性，包括主要频率成分、能量分布等，因此我们选择以谱模型为基础。常见的海浪谱模型有 JONSWAP 谱

模型、Pierson-Moskowitz 谱模型、Bretschneider 谱模型、Ochi-Hubble 谱模型。相较而言，JONSWAP 谱模型考虑了风速廓线效应，其适用范围广泛，考虑了海洋波浪的非线性特性，并可通过实测数据进行验证，因此在研究海杂波问题时具有较好的优点和应用价值。本文主要借助 JONSWAP 谱模型，通过确定仿真区域和时间范围、设置风场和其他外部条件——选择合适的 JONSWAP 参数、建立数值模型、模拟计算、分析与校验，并结合蒙特卡洛(Monte Carlo)方法[7]，将海面视作由无数的正弦波线性叠加而来，因而可以通过对海浪谱进行 Fourier 逆变换获得海面的高度起伏，最后来给出海浪仿真结果。

2.2 问题二的分析

对于问题二，在 JONSWAP 谱模型构建的基础上，我们分析雷达学报网站“雷达对海探测计划数据共享计划”，同时考虑到本题涉及对船、鲸、岛屿等海面目标的特征分析，需要更多的时序信息来捕捉目标的动态变化，且对频谱也有一定要求，因此我们选择时频分析[7]的方法。在众多时频分析方法中，选用数学理论成熟、适用性广泛、计算效率高的短时傅里叶变换模型，得到原始数据的时频谱图。为了区分海面目标与海杂波，我们提取了 2 个特征：相对时频脊积累值和相对时频脊多普勒频率变化值。在特征值的基础上，使用 SVDD 多特征检测法。因为 SVDD 是一种基于支持向量机的无监督异常检测方法，通过将正常数据映射到高维空间中，构建一个超球体包围正常数据，从而实现异常数据的检测。多特征检测器是 SVDD 的一种扩展，它能够利用多个特征来提高异常检测的准确性[9]。多特征检测器将多个特征表示为一个向量，然后利用 SVDD 算法对该向量进行异常检测，从而得到海面目标的检测结果。

2.3 问题三的分析

对于海面目标检测来说，海杂波过滤是至关重要的。海杂波过滤方法有许多：基于门限、基于滤波器[10]、基于时域分析、基于频域分析[8]等等。结合问题二中得到的初始时谱图，我们图中的干扰项多为海杂波的噪点。为了获取更好的海面目标检测效果，我们使用高斯滤波和特异性抑制的方法实现第一次去噪，由于一次去噪的效果并非十分明显，我们又借助 OpenCV 中 NLM 滤波算法使用自然图像中普遍存在的冗余信息来去噪。它利用了整幅图像来去噪，以图像块为单位在图像中寻找相似区域，再对这些区域求平均，能够比较好的去掉图像中存在的高斯噪声。最后进行轮廓提取应用阈值操作将图像转换为二值图，利用图像处理得到 0/1 矩阵，矩阵的值即是过滤海杂波后对海面目标检测的分布“概率”。

本文的研究路径如图 1 所示：

图 1 研究路径

三、模型假设

1. 假设雷达学报网站数据真实可靠
2. 假设雷达每次发射波相同
3. 假设海面不在特殊天气干扰
4. 假设海浪是单向传播的
5. 不考虑远距离海面目标检测时以噪声为主的数据

四、模型构建与求解

4.1 问题 1——构建海浪模型，并给出海浪的仿真结果

4.1.1 分析海浪运动和海浪模型

海浪运动指的是海洋表面上的波浪形成、传播和演变的过程。海浪是由风吹动海面产生的，其能量来源于风对海洋表面的摩擦和转移。海浪运动的特征包括波高、波长、周期、波速等。它的形成可以简单地描述为：风通过对海面的摩擦和转移，将能量传递给海水，引起海面波动，形成波浪。随着风的持续吹动，波浪会逐渐增大，并向远离风向的方向传播。波浪的传播受到水深、海底地形、地球自转等因素的影响，会导致波浪的衍射、折射、反射等现象。在海洋中，波浪会与其他波浪相互作用，形成波浪群、波浪合并等现象，最终演变为复杂的波浪场景。

海浪模型是一种用于描述和预测海洋波浪行为的数学模型。这些模型通常基于物理原理、数值方法或经验规律，用于模拟海洋中波浪的形成、传播和演变过程。现有的海浪模型主要线性模型、非线性模型、谱模型、深水波动方程模型、浅水模型、耦合模型等等。线性模型简单易懂，计算速度快，但只能描述小振幅波浪，不适用于大振幅波浪和非线性效应。非线性模型（如 Korteweg–de Vries 方程、KdV 方程）能够描述更复杂的波浪行为，包括大振幅和非线性效应。但数学上较为复杂，计算量大，通常需要数值模拟来求解。谱模型（如 Pierson-Moskowitz 谱、JONSWAP 谱）通过频谱分析提供了波浪的频谱信息，适用于海洋工程设计和海况预报。通常基于经验参数，对于不同的海域和天气条件，参数需要根据实际情况进行调整。深水波动方程模型考虑了水深对波浪传播的影响，适用于深水区域的波浪描述但不考虑底部摩擦和浅水效应，限制了在浅水区域的应用。浅水模型（如 Boussinesq 方程）考虑了浅水区域的非线性效应和底部摩擦，适用于近岸区域的波浪描述但对于较深水域的波浪描述不够准确。耦合模型结合了多种模型，综合考虑了不同因素对波浪的影响但计算复杂度高，需要大量的数据和计算资源。由于本文主要研究海杂波问题及雷达对海面目标检测的检测效果，并且目前应用于海面电磁辐/散射仿真的海面建模方法主要是基于海浪谱的建模方法，海浪谱模型的能量谱可以作为输入函数直接代入散射模型进行计算，因此我们在谱模型的基础上构建海浪的仿真。海浪谱模型是描述海浪频谱特性的数学模型，用于表示海浪在不同频率范围内的能量分布，通常以频率为自变量，以波浪能量或波高谱密度为因变量。海浪谱模型可以提供关于海浪能量分布在不同频率下的信息，从而帮助了解海浪的频谱特性和频率成分。

常见的海浪谱模型有 JONSWAP 谱模型、Pierson-Moskowitz 谱模型、Bretschneider 谱模型、Ochi-Hubble 谱模型。Pierson-Moskowitz 谱模型是最早被广泛应用的波浪谱模型之一，简单且易于理解，但忽略了风速廓线效应，不能很好地描述风生成波浪的频谱特性。Bretschneider 谱模型是对 Pierson-Moskowitz 谱模型的改进，考虑了风速廓线效应，能更准确地描述风生成波浪的频谱特性，但仍然存在一定的局限性，不能很好地描述极端海况下的波浪特性。Ochi-Hubble 谱模型考虑了非线性特性和随机性，适用于模拟大

范围内的海洋波浪，但对参数敏感，需要根据具体情况进行精细调整和校准。相较而言，JONSWAP 谱模型考虑了风速廓线效应，适用范围广泛，考虑了海洋波浪的非线性特性，并可通过实测数据进行验证，因此在研究海杂波问题时具有较好的优点和应用价值。本文主要借助 JONSWAP 谱模型，通过确定仿真区域和时间范围、设置风场和其他外部条件、选择合适的 JONSWAP 参数、建立数值模型、模拟计算、分析与校验来给出海浪仿真结果。

4.1.2 构建 JONSWAP 谱模型

把无限个随机的余弦波叠加起来可以描述一个定点的波面：

$$\eta(t)=\sum_{n=1}^{\infty}a_n\cos(\omega t_n+\varepsilon_n) \tag{1}$$

式(2-5)中 a_4 、o, 分别为组成波的振幅和圆频率，ε 为在 0～2 π 范围内随机分布的初相位。

定义：

$$S(\omega)=\frac{1}{\Delta\omega}\sum_{\omega}^{\omega+\Delta\omega}\frac{1}{2}a_n^2 \tag{2}$$

s(o)表示频率间隔△o 内的平均能量，如果△o=1, 则上式表示单位频率间隔内的能量，即能量密度，故 s(o)称为频谱。

JONSWAP（Joint North Sea Wave Project）谱模型是一种用于描述海浪频谱特性的数学模型，用于分析和预测海洋波浪的能量分布和频谱形态。该模型是在 Pierson-Moskowitz 谱模型基础上进行改进的，目的是更准确地描述风生成波浪的频谱特性。

JONSWAP 谱模型表达式为：

$$S(\omega)=\frac{\alpha g^2}{\omega_p^5}\cdot\omega^{-5}\cdot e^{(-\frac{5}{4}\cdot(\frac{\omega_p}{\omega})^4)}\gamma^{a} \tag{3}$$

$$a=e^{-\frac{(\omega-\omega_p)^2}{2\sigma^2\omega_p^2}} \tag{4}$$

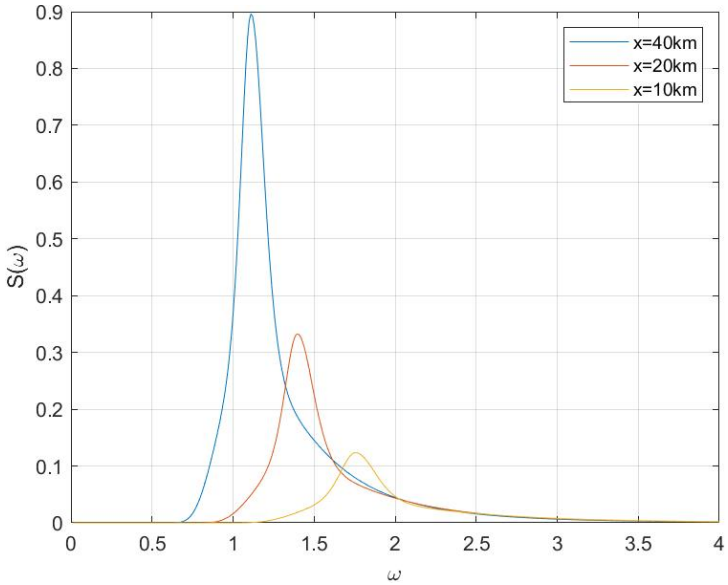


图 2 JONSWAP 谱的示例图

当 $\omega<\omega_p$ 时，σ =0.07；当 $\omega>\omega_p$ 时，σ =0.09；参数越大，JONSWAP 谱的峰值越细长。

ω_p^5 表示波浪频谱，α 是一个修正系数，g 是重力加速度，ω_p 是角频率，是波高峰值频率。

修正系数 α 用于修正 Pierson-Moskowitz 谱模型中对波浪能量分布的过度估计。其值通常根据风速廓线和波浪条件来确定。波高峰值频率是指波浪能量谱中能量最集中的频率，用于描述波浪的主要频率成分。波高峰值频率通常与海域的风速和风向相关。频谱形态因子是 JONSWAP 谱模型中的一个参数，用于描述波浪频谱的形态。较大值表示波浪频谱的形态较陡峭，而较小的值表示波浪频谱的形态较平缓。

根据题目所给数据集来看，本文主要架构于烟台第一海水浴场试验点，在 5 级到 2 级海况变化条件下，试验期间风、浪所对应的的海情是由低到高、再由高到低的过程。开始时浪高为 1.8m 中浪，4 级海况，结束时浪高为 0.4m 轻浪，2 级海况，期间包含风速最高 16.7m/s，浪高最高为 2.8m 大浪，5 级海况。我们选取烟台第一海水浴场的具体情况和海域特性来进行分析。选择合适的 JONSWAP 谱模型参数——波高峰值频率 0.15 Hz、波高峰值频率处的波高 1.5m、波能传播方向东南至正南方向。

修正系数 α 是 JONSWAP 谱模型中的一个重要参数，根据已有的经验法则和相关研究的结论，可以得到该海域 α 的估计值为 0.02。

利用选定的参数，建立 JONSWAP 谱模型的频谱表达式。

$$S(\omega)=\frac{0.02g^2}{0.15^5}\cdot\omega^{-5}\cdot e^{(-\frac{5}{4}\cdot(\frac{0.15}{\omega})^4)}\gamma^{a} \tag{5}$$

由于选取地为烟台第一海水浴场水域，本文取 g=9.81m/s^2

4.1.3 海浪仿真

JONSWAP 谱模型提供了海面建模方法，然而为了通过构建海浪模型研究海杂波问题，需要在空间维对海面进行建模。目前对海面的空间建模最常用的方法是蒙特卡洛(Monte Carlo)方法。其基本假设认为海面可以由无数的正弦波线性叠加而来，因而可以通过对海浪谱进行 Fourier 逆变换获得海面的高度起伏[9]。

确定海域范围为烟台第一海水浴场海域，根据 4.1.2 中建立的 JONSWAP 谱模型，利用蒙特卡洛法生成海域内的波浪场。

一维海浪高度计算公式如下：

$$z(x,y)=k_1\sum_{i=1}^{n}\sqrt{T(i)}\,cos\,(2\pi f_i x+\varphi) \tag{6}$$

其中，k 为常数，φ 为随机数，取值在$[0,2\pi]$之间

$$T(i)=S(f_i) \tag{7}$$

S 为 JONSWAP 模型的能量谱函数

$$f\in[0.01,2] \tag{8}$$

f 元素取值的上下限根据 JONSWAP 模型能量谱值较大部分进行选取，例如以能量谱值中位数附近两数对应的两个频率值作为区间上下限度，以此使得选取有限个点对应的波对整体海浪波有较大贡献。

下图为所得一维海浪平面图与多个正弦波叠加图：

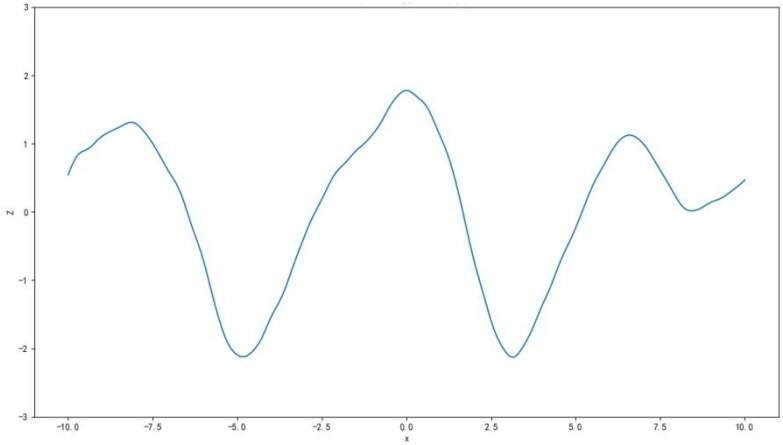


图 3 一维海面高度空间分布图

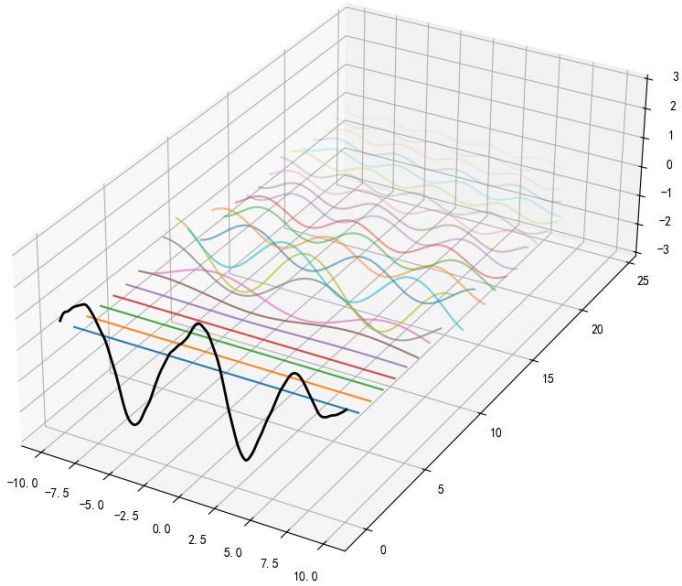


图 4 一维海面高度空间分布图（正弦展开）

二维海浪高度计算公式如下：

$$z(x,y) = k_2 \sum_{i=1}^{n}\sum_{j=1}^{n} \sqrt{T(i,j)}\, cos\,(2\pi(F_{1ij}x + F_{2ij}y) + \varphi) \tag{9}$$

其中，k_2 为常数，φ 为随机数，取值在$[0, 2\pi]$之间

$$T(i,j) = S(\sqrt{{F_{1ij}}^2 x + {F_{2ij}}^2 y}) \tag{10}$$

S 为 JONSWAP 模型的能量谱函数

$$F_1 = \begin{pmatrix} f_{11} & \cdots & f_{1n} \\ \vdots & \ddots & \vdots \\ f_{n1} & \cdots & f_{nn} \end{pmatrix}, F_2 = {F_1}^T, f \in [0.01,2] \tag{11}$$

二维下 f 选取与一维相同。下图为仿真出的二维海浪图：

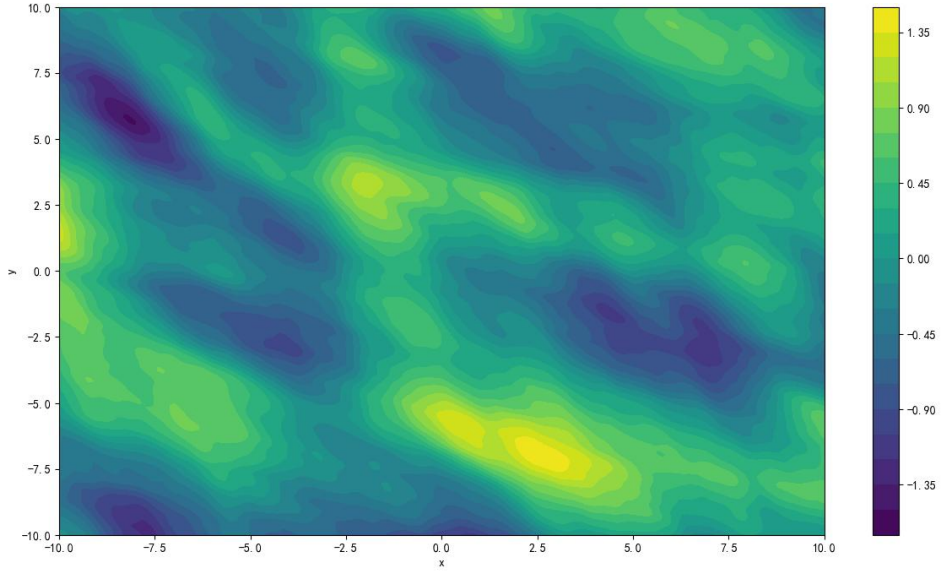


图 5 二维海面高度空间分布图

然后利用生成的波浪场，并根据波浪场的波高、波长等参数，模拟海域内波浪的传播和演变过程。对模拟结果进行分析，实测数据进行验证，并根据实际情况对模型进行调整和优化，得到海浪仿真图：

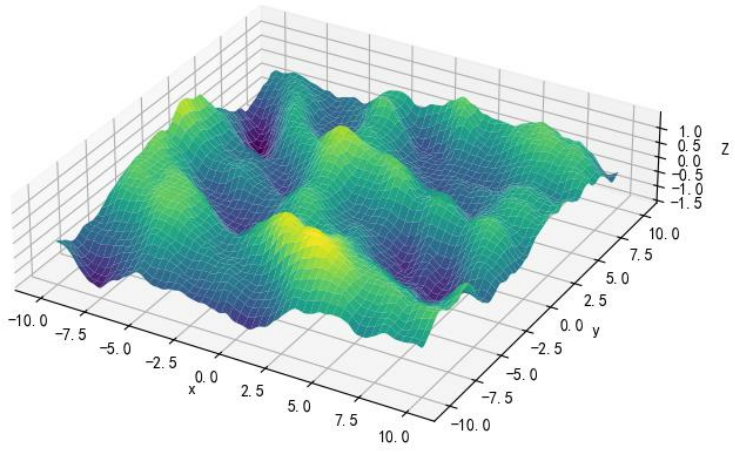


图 6 三维海浪仿真图

4.2 问题 2——探讨各海面目标特征分析模型及方法，利用所构建的检测方法给出仿真结果

4.2.1 目标回波采集数据介绍

本文所采用的数据来源于雷达学报网站“雷达对海探测计划数据共享计划”中《高/低海况下杂波与漂浮目标回波连续采集试验数据报告》[7]。采用 SPPR50P-HH 和 SPPR50P-VV 两部独立的固态功放监视/导航雷达，除波导裂缝天线分别为水平极化（1.8 米长）和垂直极化（2.4 米长）外，其余配置完全相同。两部雷达均工作在 X 波段，采用固态功放组合脉冲发射体制，利用脉冲压缩技术，发射时间为 40ns~100us，发射功率为 100W，最高距离分辨率为 6 米。利用接收信号和发射信号的时差计算目标距离，可以在水平面内顺时针 360o 全方位扫描，转速为 2~48 转/分，也可以在设定的角度上凝视工作。

首先我们对雷达学报网站所给的 20221112150043_stare_HH.mat 的 amplitude_complex_T1 部分复数数据变量进行取模处理，得到原始数据的总体概况：

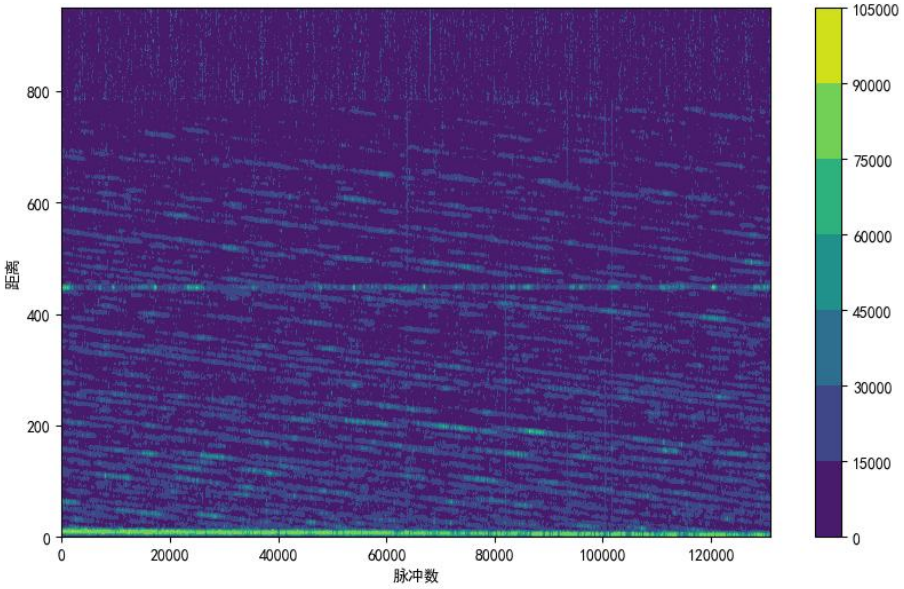


图 7 amplitude_complex_T1 模值

接着，由官网数据表可知，距离 445-450 间存在浮标这一海面目标，为了更详细观察幅角图的细节，我们放大比例尺，对 435-460 距离间的 0-50 脉冲进行作图：

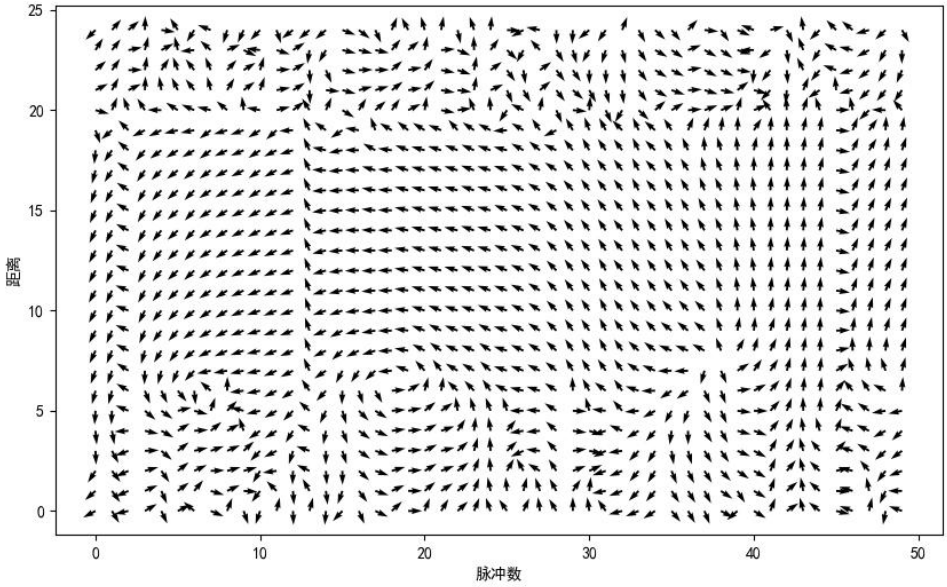


图 8 amplitude_complex_T1 相位（435:460，0:50）

由两图初步得到对数据的整理直观了解。

4.2.2 海面目标特征分析

海面目标特征分析是一种通过对海面目标的形态、颜色、纹理、运动等特征进行分析，对其进行识别和分类的方法。在海洋领域，海面目标的种类繁多，包括船只、潜艇、浮标、浮筒等等。对这些目标特征的分析，可以提高对海洋环境的感知能力，对于海洋资源开发、安全监测等具有重要意义。

题目所涉及的是雷达对海面目标的监测，而雷达的基本工作原理是：发射无线电波，然后通过接收反射回来的无线电波来探测目标。当无线电波遇到目标时，会反射回来，被雷达接收并处理。通过对这些反射回来的无线电波进行分析，可以获得目标的距离、方位、速度等信息。由于雷达是利用物体反射电磁波的特性来发现并明确目标的距离、速度、高度和方位等信息的，因此我们选用信号处理模型。

但是，在海面目标检测中，海浪中海杂波的存在会对检测结果产生影响的。海杂波

是雷达电磁波照射到海表面时接收到的海表面后向散射回波。海杂波产生的物理机理复杂，依赖于很多因素，其中包括了复杂海面本身的状况以及雷达的工作状态。由于受到环境因素和雷达设备参数的影响，海杂波的特性也随之不断地改变，相较于地面杂波，海杂波的空时变化要更为复杂。同时，海尖峰（Sea spikes）是微波段雷达照射动态演化海表面出现的一种特有散射现象，高分辨情况下表现为孤立的、高功率、亚秒级-秒级持续时间的海杂波尖峰，是高分辨海杂波中海面小目标检测的主要虚警来源。因此，海面目标的检测不可避免地面临着海杂波的干扰。海杂波背景下目标检测的难点主要在于：慢速小目标回波微弱；空时变海杂波异常复杂，海杂波特性认知难度大；目标模型难以建立等。而这些问题的重要问题之一就是海杂波特性的精细化感知问题。

下图为海杂波示例图：

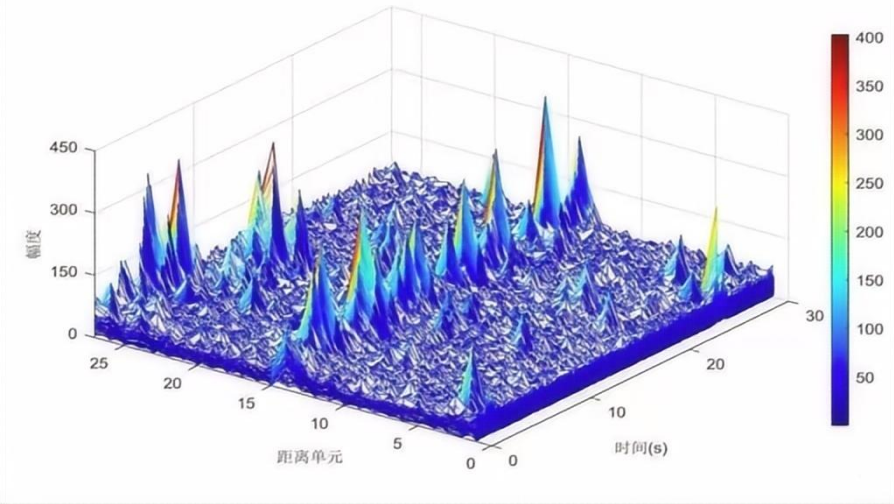


图 9 海杂波示例图

在雷达进行目标检测中，一般会用到频谱分析和时域分析两种常用的信号分析方法。频谱分析可以清晰地展示信号在频率域上的特征，包括频率成分和能量分布，有助于识别目标的频率特征。对于具有周期性的信号（如雷达和声纳返回的信号），频谱分析可以准确地提取信号的频率信息，有助于检测目标。但是频谱分析通常无法提供信号在时间上的变化情况，因此可能无法捕捉到目标的时序特征，如目标的运动轨迹。时域分析可以清晰地展示信号在时间上的波形特征，包括脉冲宽度、持续时间等，有助于捕捉目标的时序特征。它通常比频谱分析更直观和易于理解，对于一些简单的信号分析任务可能更适用。对于非周期性的信号，时域分析可以提供更准确的描述和分析。但时域分析无法提供信号在频率域上的详细特征，可能难以准确识别目标的频率特征。由于本题涉及到对船、鲸、岛屿等海面目标的特征分析，需要更多的时序信息来捕捉目标的动态变化，且对频谱也有一定要求，因此我们选择时频分析的方法。

4.2.3 构建检测方法

时频分析是一种用于分析信号在时域和频域上的特性的方法，常用于处理海面目标的信号。时频分析分为线性时频分析和二次型时频分析两种，线性时频分析有短时傅里叶变换、小波变换等，二次型时频分析有 Wigner-Ville 分布、Cohen 类时频变换等[2]。在本题的分析中，我们选用数学理论成熟、适用性广泛、计算效率高的短时傅里叶变换（Short-Time Fourier Transform，STFT）模型。

普通傅里叶变换只反映出信号在频域的特性，无法在时域内对信号进行分析。为了将时域和频域相联系，Gabor 于 1946 年提出了短时傅里叶变换，其实质是加窗的傅里叶变换。STFT 的过程是：在信号做傅里叶变换之前乘一个时间有限的窗函数 h(t)，并假定非平稳信号在分析窗的短时间隔内是平稳的，通过窗函数 h(t)在时间轴上的移动，对

信号进行逐段分析得到信号的一组局部“频谱”。

短时傅里叶变换的表达式为：

$$STFT(t,f)=\int_{-\infty}^{+\infty}x(\tau)g^{*}(\tau-t)e^{-j2\pi f\tau}d\tau \qquad (12)$$

式中：$h(\tau-t)$为分析窗函数

由上式知，信号 x(t)在时间 t 处的短时傅里叶变换就是信号乘上一个以 t 为中心的“分析窗”h(τ −t)后所作的傅里叶变换。x(t)乘以分析窗函数 h(τ −t)等价于取出信号在分析时间点 t 附近的一个切片。对于给定时间 t，STFT(t,f)可以看作是该时刻的频谱。特别是，当窗函数取 h(t)≡1 时，则短时傅里叶变换就退化为传统的傅里叶变换。要得到最优的局部化性能，时频分析中窗函数的宽度应根据信号特点进行调整，即正弦类信号用大窗宽，脉冲型信号用小窗宽。下图为短时傅里叶变化示例图与时频谱图：

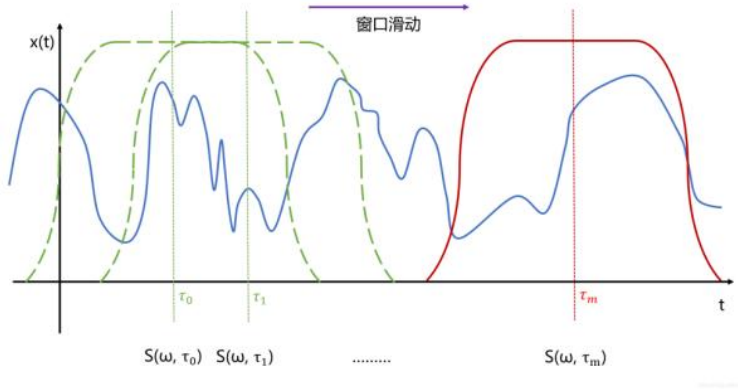


图 10 短时傅里叶变化示例图

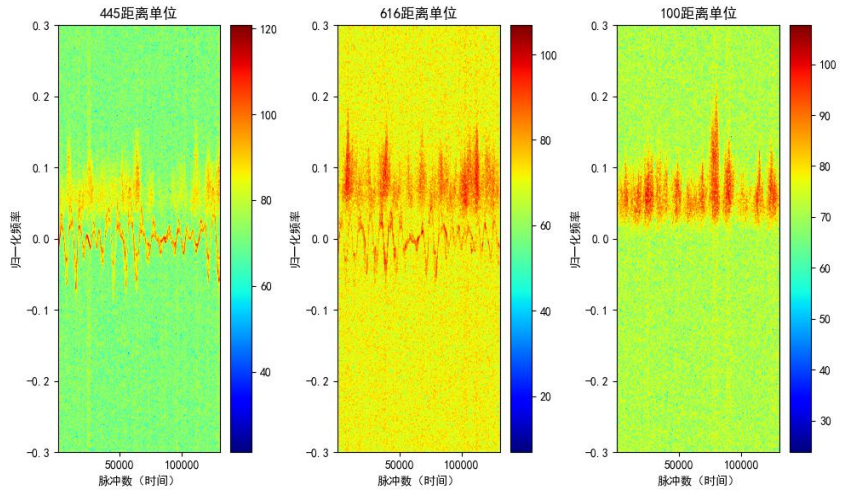


图 11 时频谱图

多普勒频带是一种由多普勒效应引起的频率偏移现象。多普勒效应是指当一个物体向着一个接收者运动时，接收者所接收到的物体反射回来的信号会发生频率偏移。这是因为当物体朝向接收者运动时，信号的波长会被压缩，导致频率升高；而当物体背离接收者运动时，信号的波长会被拉伸，导致频率降低。多普勒频带常用于雷达系统中，用于确定目标的速度和运动方向。通过测量反射回来的信号的频率变化，可以计算出目标相对于雷达系统的速度和运动方向。在雷达波范围内，海面包含大量不同速度的散射点，因此海杂波的多普勒频带较宽。而目标在观测时间内，速度变化范围有限，因此运动目标的多普勒频带相对较窄。我们提取以下特征对海面目标与海杂波进行检测：相对时频脊积累值（Relative Ridge Integration，RRI）、相对时频脊多普勒频率变化值（Relative Ridge Doppler Frequency Variation，RFV）。

时频脊线是一种在时频分析中经常使用的工具，用于表示信号在时频域中的局部特征。它可以用来描述信号的瞬时频率和瞬时振幅等信息。时频脊线通常通过在时频域内寻找信号局部峰值来确定。这些峰值通常会被连接成一条曲线，称为时频脊线。

$$Ridge_{TF}(t\,|\,z)=\max\{STFT(f,t\,|\,z)\}$$
$$f_{TF}(t\,|\,z)=\arg\max\{STFT(f,t\,|\,z)\} \tag{13}$$

其中，$STFT(f,t\,|\,z)$ 为对回波序列 z 进行 STFT 处理得到的时频图，$Ridge_{TF}(t\,|\,z)$ 表示回波序列 z 对应的时频脊线，$f_{TF}(t\,|\,z)$ 是时频脊线对应的多普勒频率值。

下图为所得时频图与对应的时频脊线图：

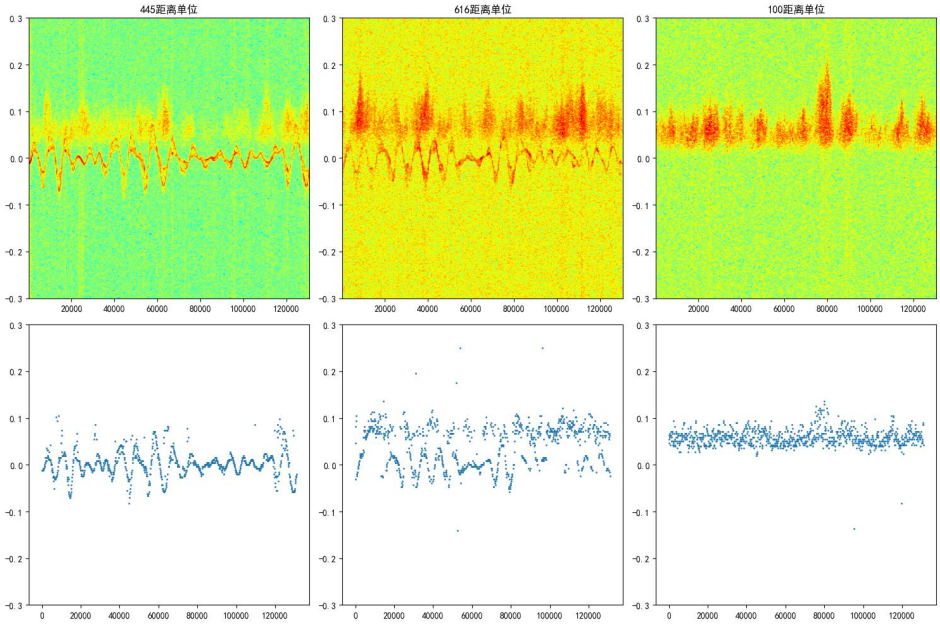


图 12 时频图与对应的时频脊线图

图 12 上部分为目标回波序列经过 STFT 后得到的时频图。由于信号受海杂波影响较大，在时频图上海杂波能量聚集到一个带状区域中，为主杂波区。在主杂波区中，海杂波能量分布不均匀，带状区域的亮度和宽度会随时间变化，反映海杂波回波序列在时间上的非平稳性。目标归一化频率曲线为窄带蛇形，其变化范围为-0.4Hz~0.4Hz，围绕零频波动，反映目标在某一位置处随海面起伏。对于回波序列，时频脊线可以看作是连续曲线，近似表示为海面目标的瞬时频率曲线，反映了对象的运动信息。

沿时频脊线进行积累就得到了脊积累值（Ridge Integration，RI），通常目标单元 RI 值较大，纯海杂波 RI 值较小，因此利用 RI 可在一定程度上区分海杂波和目标，RI 值计算表达式为：

$$RI(z)=\sum_{t}Ridge_{TF}(t\,|\,z) \tag{14}$$

求待测单元回波序列 RI 的值与参考单元回波序列 RI 值之比，可得满足恒虚警特征的 RRI：

$$RRI(z)=\frac{RI(z)}{\Theta(RI(z_k),k=1,2,\ldots,K)} \tag{15}$$

因为目标时频脊线可以视作连续，相邻点多普勒频率差值小，但海杂波时频脊为主杂波区的不连续点，时频脊相邻点多普勒频率差值大，因此我们借助 RFV 区别目标与海杂波：

$$FV(z)=\sum_{t}|f_{TF}(t\,|\,z)-f_{TF}(t-1\,|\,z)| \tag{16}$$

其中，FV 为时频脊多普勒频率变化值，将待测回波序列与参考回波序列相比，即可得到满足恒虚警的特征 RFV：

$$RFV(z)=\frac{FV(z)}{\Theta(FV(z_k),k=1,2,\cdots K)} \tag{17}$$

根据这 2 个特征，我们使用 SVDD（Support Vector Data Description）的多特征检测器进行数据的分类检测。SVDD 是一种基于支持向量机的无监督异常检测方法，它通过将正常数据映射到高维空间中，构建一个超球体包围正常数据，从而实现异常数据的检测。多特征检测器是 SVDD 的一种扩展，它能够利用多个特征来提高异常检测的准确性。多特征检测器将多个特征表示为一个向量，然后利用 SVDD 算法对该向量进行异常检测。具体来说，多特征检测器通过对每个特征提取有用的信息，然后将这些信息融合为一个向量来描述样本。然后将这个向量作为输入传入 SVDD 模型，进行异常检测。相比于单特征检测器，多特征检测器能够更全面地考虑数据的不同方面，提高异常检测的准确性。

最终得到混有海杂波与目标物时频图的检测分析图，途中橙色点表示为海面目标物，蓝色小点为海杂波的噪声干扰。下图为所得分类结果：

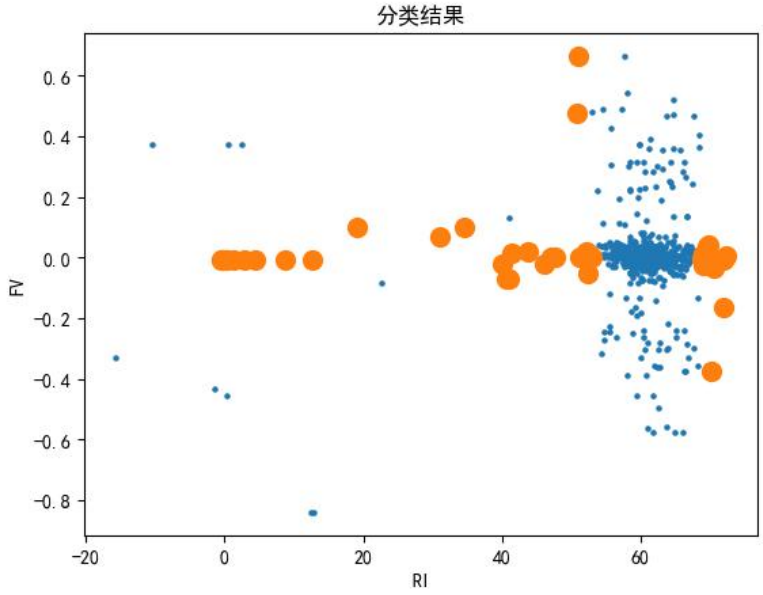


图 13 分类结果

4.3 问题 3——给出一种海杂波过滤数学模型，并测试其有效性

4.3.1 海杂波过滤模型和仿真

由 4.2.1 可知，在海面目标检测中，海杂波会对检测产生影响。海杂波会产生噪声，使得图像的信噪比降低，从而使得目标的特征提取和检测变得困难。不仅如此，海杂波还会模糊目标的边缘，使得目标的形状和大小难以准确地确定，进而影响目标的分类和识别。因此，对于海面目标检测来说，海杂波过滤是至关重要的。通过海杂波过滤技术，可以有效地去除海洋图像中的噪声，提高海面目标的对比度和边缘特征，从而使目标更加明显和容易被识别。同时，海杂波过滤还可以提高图像的分辨率和清晰度，进一步提高目标检测的准确性和可靠性。在本文中，为了获取更好的海面目标检测效果，我们利用高斯滤波和特异性抑制实现第一次去噪，由于一次去噪的效果并非十分明显，我们利用 OpenCV 库进行了二次去噪并将处理后的图形转换为 RGB 格式显示，最后进行轮廓提取应用阈值操作将图像转换为二值图，实现对海杂波的过滤。

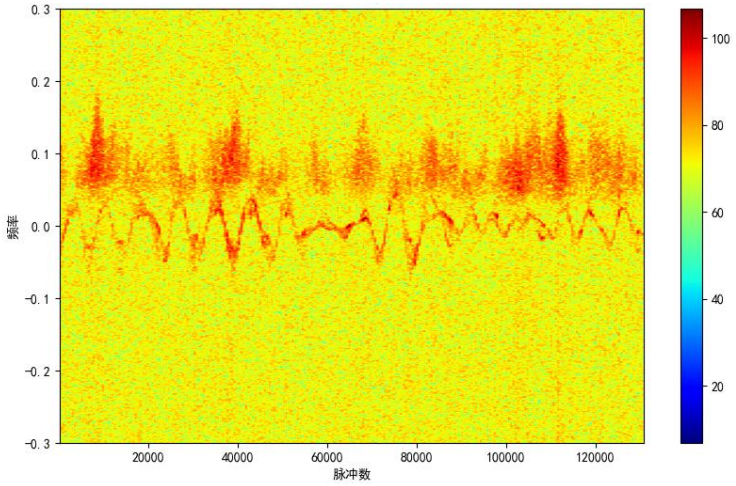


图 14 原始频谱图

高斯滤波是一种线性平滑滤波器，它可以用于去除图像中的噪声和平滑图像。这种滤波器的基本思想是用一个高斯函数来平滑图像。高斯函数是一种钟形曲线，它可以很好地模拟自然界中的很多现象。高斯滤波的过程是将图像中每个像素周围的像素值按照一定的权重进行加权平均，得到该像素的新值。这些权重由高斯函数决定，离中心点越远的像素其权重越小，因此它们对最终结果的影响也越小。这样就可以达到平滑图像、去除噪声的效果。高斯滤波的具体操作是：用一个用户指定的模板（或称卷积、掩膜）去扫描图像中的每一个像素，用模板确定的邻域内像素的加权平均灰度值去替代模板中心像素点的值。

一维高斯分布：

$$G(x)=\frac{1}{\sqrt{2\pi\sigma}}e^{-\frac{x^2}{2\sigma^2}} \tag{18}$$

二维高斯分布：

$$G(x,y)=\frac{1}{2\pi\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{19}$$

再结合 sigmoid 抑制函数实现特异性抑制。sigmoid 函数是一种常用的数学函数，它的输出值在 0 到 1 之间。它常被用作抑制函数，使得数据输出更加平滑，避免出现极大或极小的输出值。

Sigmoid 函数由下列公式定义：

$$Sig(x)=(1+e^{-x})^{-1} \tag{20}$$

下图为所得 Sigmoid 函数曲线图：

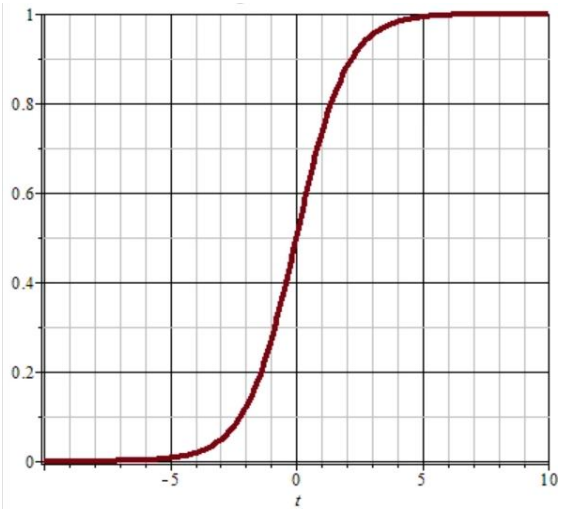


图 15 Sigmoid 函数曲线图

用高斯滤波和特异性抑制进行第一次去噪，利用 sigmoid 实现对海杂波明显的区域深度过滤，对弱明显的区域进行浅度过滤，得到一次去噪的频谱图：

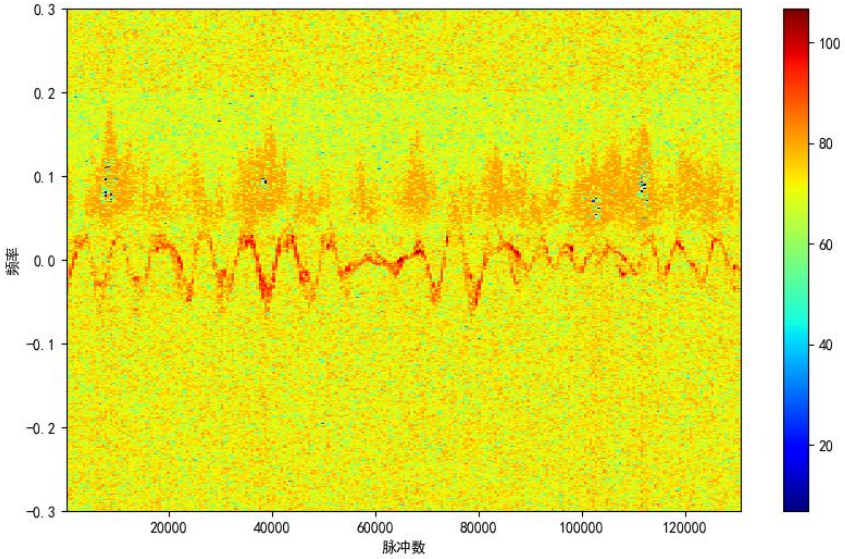


图 16 一次去噪的频谱图

由于一次降噪的频谱图效果并没有达到预想，我们又进行了二次去噪：利用 OpenCV 中 NLM 算法使用自然图像中普遍存在的冗余信息来去噪。对于图像中的每个像素，NLM 算法定义一个邻域窗口（通常为矩形区域）。在图像中寻找与该邻域窗口相似的其他像素块。这些像素块可以来自图像的其他位置，也可以来自邻域窗口周围。计算找到的每个相似像素块与当前像素块之间的相似度，通常使用均方差或 PSNR 等度量。根据相似度计算每个像素块的权重，并将这些像素块的像素值加权平均，得到当前像素点的去噪后值。对所有像素点重复上述步骤，直到所有像素点均被处理。NLM 算法的优点在于能够有效地去除图像中的高斯噪声和其他噪声，同时保留图像的边缘和细节特征。所得图 17：

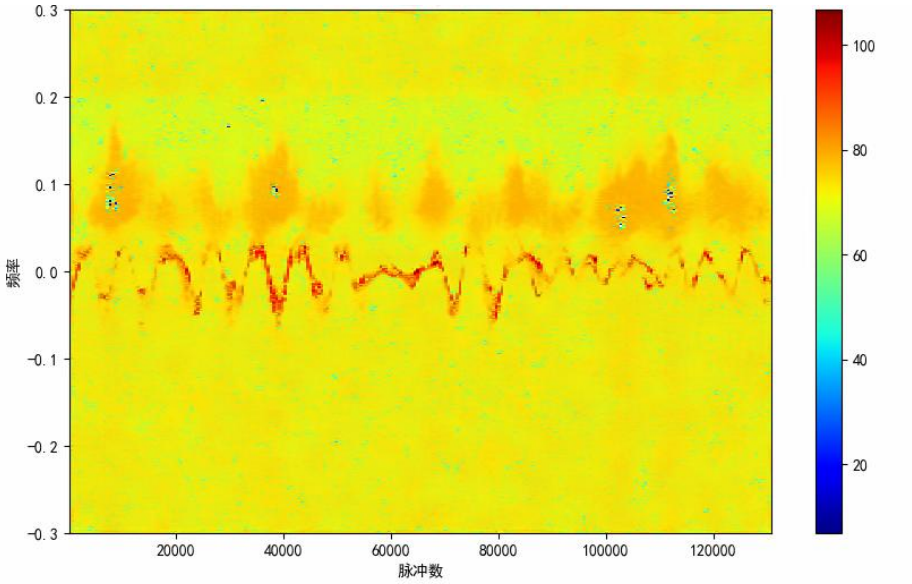


图 17 二次去噪频谱图

我们发现，经历了两次去噪后，海杂波被较好地过滤，为了更直观地被机器读取识别，我们将图形从 BGR 转换为 HSV，定义红色的 HSV 范围后创建掩码并提取红色区域，最后应用阈值操作将图像转换生成二值图：

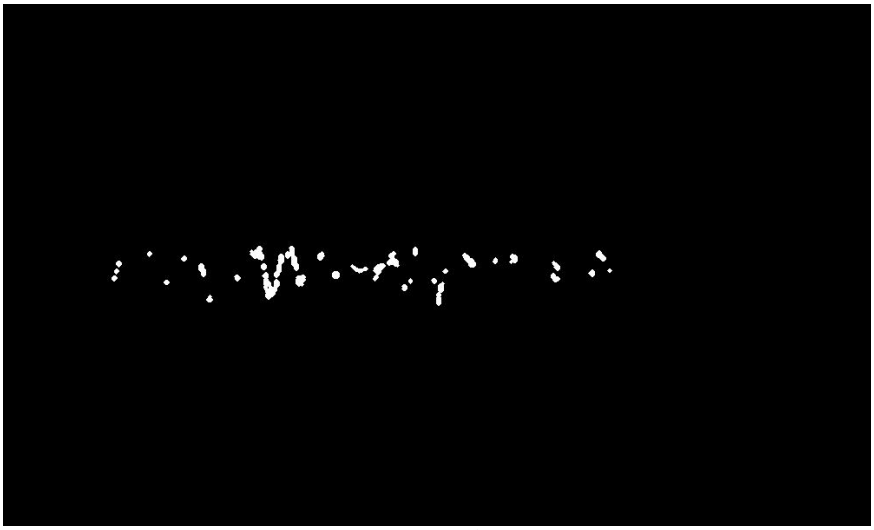


图 18 二值图

最后利用图像处理将二值图转换成最原始的 0/1 矩阵，计算出矩阵的值可以表征过滤海杂波后对海面目标检测的分布“概率”，转换成位置变换图如下：

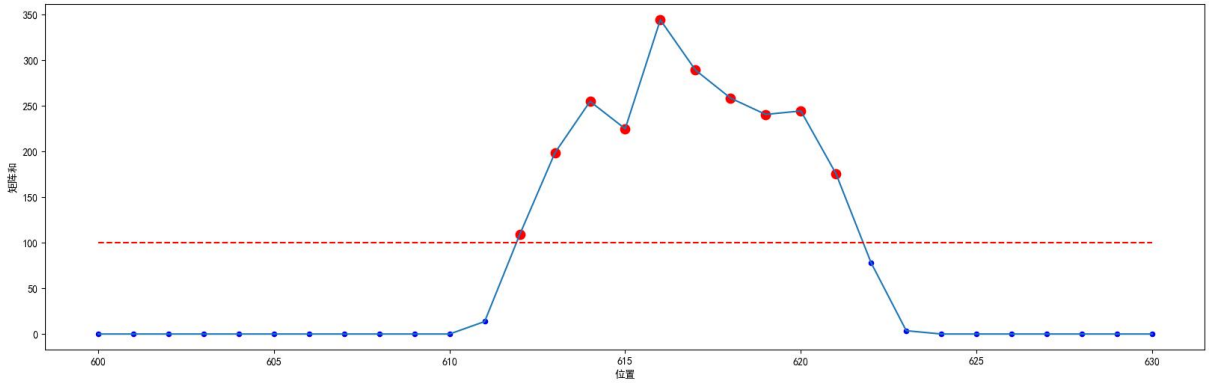


图 19 浮标 2 附近矩阵和随位置变化图

为了排除过滤的偶然性，我们又选用了多组目标数据进行过滤，均能得到较好的过

滤图像：

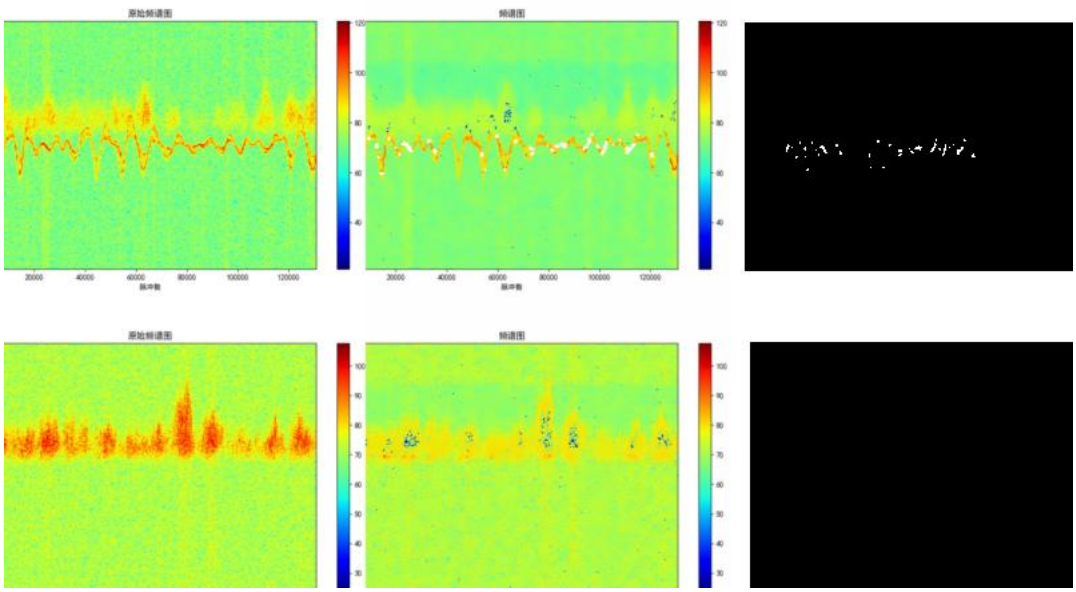


图 20 图像处理

左边为初始时频图，右边为过滤海杂波后时频图

进一步的，我们使用了另一数据集（20221112180016_stare_HH.mat）进行测试，检测效果良好，目标点全部检出的同时保持了很低的虚警率，如下图所示：

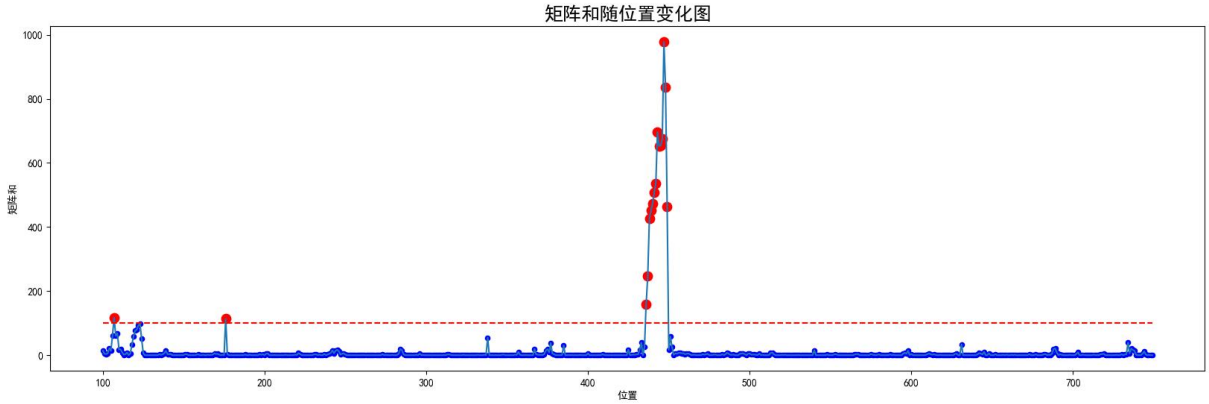


图 21 另一数据集矩阵和随位置变化图

五、模型结果分析

为了研究海杂波问题，我们构建了 JONSWAP 谱模型作为最基础的海浪模型，通过确定仿真区域和时间范围、设置风场和其他外部条件，并结合蒙特卡洛(Monte Carlo)方法，将海面视作由无数的正弦波线性叠加而来，然后对海浪谱进行 Fourier 逆变换获得海面的高度起伏，最后来给出海浪仿真结果。模型所得出的三维海浪仿真图直观地反映出了海浪的模型，为后续研究海面目标探测和海杂波问题奠定了基础。

接着，在检测船、鲸、岛屿等海面目标中，我们借助短时傅里叶变换得到时频图，为了区分海面目标与海杂波，我们提取了相对时频脊积累值和相对时频脊多普勒频率变化值特征，使用 SVDD 多特征检测法对该向量进行异常检测，从而得到海面目标的检测结果。从最终得到的分类结果看，能够检测并区分海杂波与海面目标物，但仍需要更进一步的分析。

因此为了获取较好的海面目标检测效果，也就是在过滤海杂波的情况下检测目标，我们采用高斯滤波和特异性抑制的方法实现第一次去噪，这个模型可以初步地对海杂波

进行过滤，但是一次去噪的效果并非十分明显，我们又借助 OpenCV 中 NLM 算法二次去噪，最后进行轮廓提取应用阈值操作将图像转换为二值图，实现对海杂波的过滤。对比原始频谱图与最终二次去噪后的频谱图，我们可以直观地观察出模型对海杂波较好的过滤效果：海杂波所带来的噪点干扰大部分被很好的过滤，这证明了我们模型选取的有效性。

六、总结

6.1 模型的优、缺点评价分析

6.1.1 模型的优点

JONSWAP 谱模型：

JONSWAP 谱模型能够比较准确地预测海浪的频谱特征，尤其适用于预测海上风电场等海洋工程应用中的海浪条件。能够较好地描述风浪相互作用的特征，因此对于需要考虑风浪相互作用的问题；具有较高的精度，一定程度上考虑了风速、海水深度、波浪周期等因素对海浪谱特征的影响。

蒙特卡洛法生成波浪场：

蒙特卡洛法适用范围广，不依赖于具体的物理方程模型，适用于各种不同类型的海面波浪场的模拟。其通过多次采样可以得到更加精确的结果，尤其是在处理非线性系统时，蒙特卡洛法可以得到比传统方法更准确的结果。蒙特卡洛法中每次采样都是独立的，可以进行并行计算，提高计算效率。

时频分析：

时频分析可以提供目标的高精度位置信息，对于精确定位目标非常有用；可以探测到小型目标，对于海上救援和搜救等方面具有重要意义；适应不同水深、不同海况的探测需求，具有一定的适应性；可以通过多目标跟踪技术，同时探测多个目标，提高探测效率。

短时傅里叶变换：

短时傅里叶变换可以在时间上对信号进行局部分析，对非平稳信号分析更为准确；能够提供一定的时间和频率分辨率，可以更好地观察信号特征；在信号存在瞬态现象时，可以提供更好的分析结果。

SVDD 多特征检测法：

SVDD 多特征检测法可以处理包含多个特征的数据集，因此可以更准确地识别异常值。该方法对于噪声和异常值的鲁棒性较强，能够有效地减少这些因素对结果的影响。对非线性数据有良好的处理能力。使用核函数可以将非线性问题转化为线性问题，从而增强了算法的泛化能力。

6.1.2 模型的缺点

JONSWAP 谱模型：

其对于极端海况的预测精度有限，因此在实际应用中需要进行适当修正。该模型假设海浪是单向传播的，因此对于复杂海况的描述能力有限。模型涉及到多个参数的确定，这些参数的取值会影响到预测精度，因此需要进行合理的参数选取。

蒙特卡洛法生成波浪场：

蒙特卡洛法计算成本高，需要进行多次独立采样，因此计算成本相对较高。其在收敛速度上相对比较慢，需要进行大量的采样才能达到比较准确的结果。由于采样是随机的，因此蒙特卡洛法在某些情况下可能出现算法不稳定的情况。

时频分析：

时频分析法在探测海面目标时受到天气、海况等自然因素的影响较大，可能会影响探测效果；目标的探测距离较短，需要在近距离范围内进行探测；对于非静止的目标，需要实时追踪目标的移动轨迹，算法较为复杂。

短时傅里叶变换：

短时傅里叶变换中时间窗口大小的选取对分析结果影响较大，需要根据实际情况进行优化；其需要进行大量的乘加运算，计算复杂度较高；在频域上的采样点数较多时，该方法会产生频率泄漏（Frequency Leakage）现象，影响分析精度。

SVDD 多特征检测法：

SVDD 多特征检测法对参数的敏感性较强，需要通过选择合适的核函数和参数来进行模型训练，而这些参数的选择对结果的影响较大。其计算复杂度较高。该方法的计算复杂度随着数据集的增大而增加，因此在处理大规模数据集时需要耗费大量时间和计算资源。

6.2 模型的改进与推广

6.2.1 模型的改进

在探讨船、鲸、岛屿等海面目标的特征分析模型与方法中，由于数据的有限性，我们仅区别了海杂波与大类海面目标间的时频信息，对于船、鲸、岛等不同类型海面目标的区别分析的还不够深入。具体来说，海面目标特征分析可以从以下几个方面进行解读：形态特征——船只通常呈长条状，鲸等鱼类呈流线型条状，岛屿一般呈较大的半球状；纹理特征——船只通常有明显的甲板、舷窗等特征，鲸等鱼类表面光滑且带有黏液，岛屿表面多覆盖植物、泥沙等；运动特征——船只通常沿着直线航行，鲸等鱼类在海面上多呈流线型深浅游动，岛屿则不会在海面上留下明显的运动轨迹。后续可以在这基础上展开，着重分析不同特征海面目标的时频图。

在利用 SVDD 多特征检测法分析时，我们仅取用了相对时频脊积累值和相对时频脊多普勒频率变化值这两个特征值，在分类中可以会存在一些误差，遗漏少量海面目标的时频点。后续我们可以多取用一些特征值，从更高维度上利用 SVDD 检测分析。

由于实际情况下高海况海杂波下近距离单元以海杂波为主，远距离单元以噪声为主，我们在分析时仅仅考虑了近距离单位以海杂波为主的噪点，过滤了海杂波来检测海面目标。后续我们可以考虑远距离单元以噪声为主时的噪点，过滤噪声带来的噪点对海面目标进行检测。

6.2.2 模型的推广

基于二次去噪抑制海杂波下对海面目标的时频分析模型可以用于对海洋环境下的声纳信号进行处理和分析，提高信号的可靠性和准确性，应用于海洋资源勘探、海底地形测绘等领域。同时也可以用于对水下目标进行识别和定位，提高水下目标探测和监测的效率和精度，应用于海洋安全保卫、海底管道巡检等领域；对水声通信信号进行处理和分析，提高通信信号的传输质量和稳定性，应用于水下通信、水声导航等领域。后续随着科技的进步，该模型能够与其他海洋观测技术结合使用，如水声测深、潮汐观测等，提高对海洋环境的认知和理解。

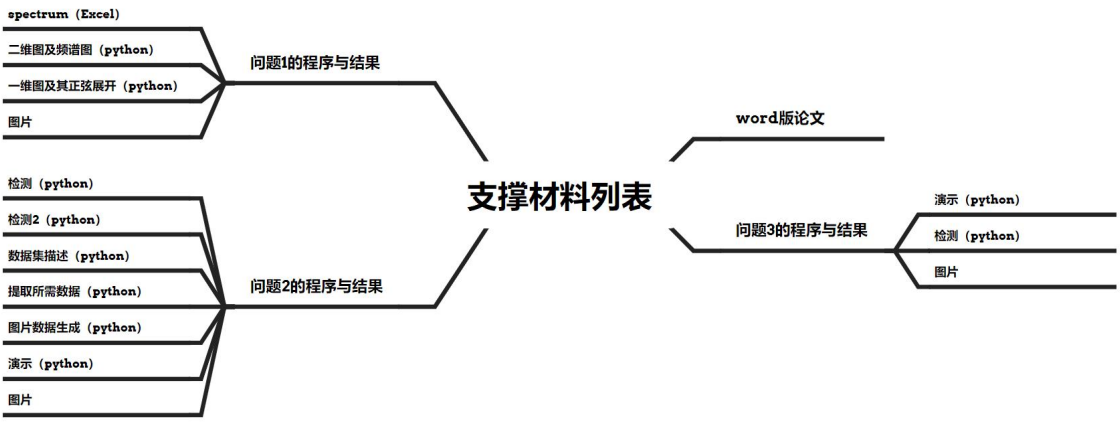
七、参考文献

[1]姜杰.基于雷达极化信息的海面目标检测与识别[D].哈尔滨工业大学,2022.DOI:10.27061/d.cnki.ghgdu.2022.003550.

[2]吴梦姣.中近程海防雷达的海杂波特性分析及目标检测设计[D].西安电子科技大学,2022.DOI:10.27389/d.cnki.gxadu.2022.000587.
[3]殷安云.基于雷达遥感的海浪信息研究与仿真模型验证[D].广西师范大学,2018.
[4]谭文清,宋杰,庄敬敏,等.被动雷达海上目标探测实验研究[J].雷达科学与技术,2024,22(01):87-92+103.
[5]宫玉坤,柏宇,马意彭.浅析海杂波对雷达检测的影响[J].中国新通信,2018,20(11):239.
[6]关键, 刘宁波, 王国庆, 等. 雷达对海探测试验与目标特性数据获取——海上目标双极化多海况散射特性数据集[J]. 雷达学报, 2023, 12(2): 456–469. doi: 10.12000/JR23029
[7]杜延磊.随机粗糙海面微波散射/辐射的仿真与分析：解析近似模型和数值方法[D].中国科学院大学(中国科学院遥感与数字地球研究所),2019.DOI:10.27612/d.cnki.gyyys.2019.000014.
[8]张俊玲.基于时频分析的海面目标检测方法研究[D].西安电子科技大学,2022.DOI:10.27389/d.cnki.gxadu.2022.000216.
[9]梁壮.海面弱目标检测方法研究[D].西安电子科技大学,2022.DOI:10.27389/d.cnki.gxadu.2022.000205.
[10]张劲东,张弓,潘汇,等.基于滤波器结构的压缩感知雷达感知矩阵优化[J].航空学报,2013, 34(4).DOI:10.7527/S1000-6893.2013.0147.

八、附录

8.1 支撑材料列表



8.2 主要程序代码

8.2.1 第一题

一维图及其正弦展开.py

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['axes.unicode_minus'] = False

# 定义 JONSWAP 函数
def JONSWAP(f, f_p, alpha, gamma):
    sigma = np.zeros_like(f)
```

```
    sigma[f <= f_p] = 0.07
    sigma[f > f_p] = 0.09
    Sw = alpha / f**5 * np.exp(-1.25 * (f_p / f)**4) * gamma**np.exp(- (f - f_p)**2
/ (2 * sigma**2 * f_p**2))
    return Sw

# 定义蒙特卡洛模拟
def monte_carlo_simulation_1d(n, f_p, alpha, gamma):
    f = np.linspace(0.01, 2, n)  # 频率范围
    spectrum = JONSWAP(f, f_p, alpha, gamma)  # 计算频谱
    phase = np.random.uniform(0, 2*np.pi, n)  # 随机相位
    return f, spectrum, phase

# 参数设置
n = 200  # 模拟的频率点数
f_p = 0.1
alpha = 0.0081
gamma = 3.3
hmax = 2.8
kfix = 0.6 # 修正系数
# 进行蒙特卡洛模拟
f, spectrum, phase = monte_carlo_simulation_1d(n, f_p, alpha, gamma)
k = kfix * hmax/np.sqrt(sum(spectrum)) # 波高系数

# 生成一维海浪的空间分布
x = np.linspace(-10, 10, n)
Z = np.zeros_like(x)

fig = plt.figure(figsize=(20, 16))
ax = fig.add_subplot(111, projection='3d')

# 在三维空间中的不同位置绘制 1D 海浪的空间分布的正弦展开

nn = 25 # 展示的展开项数

for i in range(n):
    Z += k * np.sqrt(spectrum[i]) * np.cos(2 * np.pi * f[i] * x + phase[i])
for i in range(1, nn):
    y = k * np.sqrt(spectrum[i]) * np.cos(2 * np.pi * f[i] * x + phase[i])
    ax.plot(x, y, zs=i, zdir='y', alpha=1-(1/nn)*i)
ax.plot(x, Z, zs=0, zdir='y', linewidth=2, color='black')
# ax.grid(False)
# 改变坐标轴的比例
scale_x = 1
scale_y = 1.8
scale_z = 0.8

def short_proj():
    return np.dot(Axes3D.get_proj(ax), np.diag([scale_x, scale_y, scale_z, 1]))
```

```
ax.get_proj = short_proj
ax.set_zlim(-3, 3)
plt.title('一维海面高度空间分布图（正弦展开）', y=1.2)
plt.savefig('1.png')
plt.show()


# 绘制 1D 海浪的平面图
plt.figure(figsize=(20, 16))
plt.plot(x, Z)
plt.xlabel('x')
plt.ylabel('Z')
plt.ylim(-3, 3)
plt.title('一维海面高度空间分布图')
plt.savefig('2.png')
plt.show()
```

二维图及频谱图.py

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from mpl_toolkits.mplot3d import Axes3D
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['axes.unicode_minus'] = False

# 参数设置
n = 200
f_p = 0.1
alpha = 0.0081
gamma = 3.3
hmax = 2.8
kfix = 0.3 # 修正系数
# 定义 JONSWAP 模型
def JONSWAP(f, f_p, alpha, gamma):
    sigma = np.where(f <= f_p, 0.07, 0.09)
    r = np.exp(- (f - f_p)**2 / (2 * sigma**2 * f_p**2))
    return alpha * f**-5 * np.exp(-5/4 * (f_p / f)**4) * gamma**r

f1 = np.linspace(0.01, 2, n)
f2 = np.linspace(0.01, 2, n)
# 定义蒙特卡洛模拟
def monte_carlo_simulation(n, f_p, alpha, gamma):
    F1, F2 = np.meshgrid(f1, f2)
    spectrum = JONSWAP(np.sqrt(F1**2 + F2**2), f_p, alpha, gamma)
    phase = np.random.uniform(0, 2*np.pi, (n, n))
    return F1, F2, spectrum, phase

# 进行蒙特卡洛模拟
```

```
F1, F2, spectrum, phase = monte_carlo_simulation(n, f_p, alpha, gamma)

# 将 spectrum 转换为 DataFrame
spectrum_df = pd.DataFrame(spectrum, index=F1[0, :], columns=F2[:, 0])
# print(spectrum)
k = kfix * hmax/np.sqrt(np.sum(spectrum)) # 波高系数

# 生成海浪的二维空间分布
x = np.linspace(-10, 10, n)
y = np.linspace(-10, 10, n)
X, Y = np.meshgrid(x, y)
Z = np.zeros_like(X)

for i in range(n):
    for j in range(n):
        Z += k * np.sqrt(spectrum_df.iloc[i, j]) * np.cos(2 * np.pi * (f1[i] * X
+ f2[j] * Y) + phase[i, j])

# 绘制等高线图

plt.figure(figsize=(10, 8))
plt.contourf(X, Y, Z, levels=20, cmap='viridis')
plt.colorbar()
plt.title('二维海面高度空间分布图（等高线图）')
plt.xlabel('x')
plt.ylabel('y')
plt.savefig('3.png')
plt.show()

# 绘制三维图

fig = plt.figure(figsize=(20, 16))
ax = fig.add_subplot(111, projection='3d')
ax.plot_surface(X, Y, Z, cmap='viridis')

# 缩小 z 轴的比例
scale_x = 1
scale_y = 1
scale_z = 0.3

def short_proj():
    return np.dot(Axes3D.get_proj(ax), np.diag([scale_x, scale_y, scale_z, 1]))

ax.get_proj = short_proj

ax.set_title('三维海面高度空间分布图', y=0.9)
ax.set_xlabel('x')
ax.set_ylabel('y')
ax.set_zlabel('Z')
```

```
plt.savefig('4.png')
plt.show()


# 输出显示频谱并保存频谱 DataFrame 到 Excel 文件

print(spectrum_df)
spectrum_df.to_excel('spectrum.xlsx', index=False)

# 绘制频谱图
spectrum_df_diag = np.diagonal(spectrum_df.values)

plt.figure(figsize=(10, 8))
plt.title('频谱图')
plt.xlabel('频率')
plt.ylabel('能量')
plt.plot(spectrum_df_diag)
plt.savefig('5.png')
plt.show()
```

8.2.2 第二题

提取所需数据.py

```
import numpy as np
import h5py
import pandas as pd
from scipy.signal import stft
with h5py.File(r"C:\Users\Timothy\Desktop\数学建模相关\二轮\数据集
\20221112150043_stare_HH.mat", 'r') as data:
    acT1 = np.array(data['amplitude_complex_T1']) # 得到的数据类型是 numpy.void
acT1_complex = np.array([complex(x[0], x[1]) for x in acT1.flat],
                        dtype=complex).reshape(acT1.shape) # 选择所需要的部分，并
将 numpy.void 转换为 complex
np.save(r"C:\Users\Timothy\Desktop\数学建模相关\二轮\数据集
\data_amplitude_T1_complex.npy",
        acT1_complex) # 保存为.npy 文件，以后读取时直接读取.npy 文件即可，读取到的数
据类型是 complex


# 保存 STFT 结果
df = pd.DataFrame(acT1_complex)
# 短时傅里叶变换（STFT）
f, t, Zxx = stft(df, return_onesided=False)
np.save(r"C:\Users\Timothy\Desktop\数学建模相关\二轮\数据集
\acT1_cplx_stft_f.npy", f)
```

```
np.save(r"C:\Users\Timothy\Desktop\数学建模相关\二轮\数据集
\acT1_cplx_stft_t.npy", t)
    np.save(r"C:\Users\Timothy\Desktop\数学建模相关\二轮\数据集
\acT1_cplx_stft_Zxx.npy", Zxx)
```

图片数据生成.py

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

f = np.load(r"C:\Users\Timothy\Desktop\数学建模相关\二轮\数据集
\acT1_cplx_stft_f.npy")
t = np.load(r"C:\Users\Timothy\Desktop\数学建模相关\二轮\数据集
\acT1_cplx_stft_t.npy")
Zxx = np.load(r"C:\Users\Timothy\Desktop\数学建模相关\二轮\数据集
\acT1_cplx_stft_Zxx.npy")
acT1_complex = np.load(r"C:\Users\Timothy\Desktop\数学建模相关\二轮\数据集
\data_amplitude_T1_complex.npy")
df = pd.DataFrame(acT1_complex)
Zxx100p_max_f = np.argmax(np.abs(Zxx[100:, :, :]), axis=1)


Bone = np.array(f[Zxx100p_max_f])

# 创建 images
# 遍历数据
for i in range(850):
    # 创建一个新的图像
    fig = plt.figure(figsize=(10, 6))
    plt.scatter(t, Bone[i], s=1)
    plt.ylim(-0.3, 0.3)
    # 保存图像到特定位置，名称为图像序号
    plt.savefig(f'C:/Users/Timothy/Desktop/数学建模相关/二轮/数据集
/images/{i}.png')

    plt.close(fig)
# 创建 spectrums
# 遍历数据
for i in range(850):
    # 创建一个新的图像
    fig = plt.figure(figsize=(10, 6))
    plt.specgram(df.iloc[i+100], NFFT=1024, Fs=2, noverlap=500, cmap='jet')
    plt.ylim(-0.3, 0.3)
    # 保存图像到特定位置，名称为图像序号
    plt.savefig(f'C:/Users/Timothy/Desktop/数学建模相关/二轮/数据集
/spectrums/{i}.png')
    plt.close(fig)
```

数据集描述.py

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

plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['axes.unicode_minus'] = False

# 模值图
acT1_complex = np.load(r"C:\Users\Timothy\Desktop\数学建模相关\二轮\数据集
\data_amplitude_T1_complex.npy")
modulus = np.abs(acT1_complex)
phase = np.angle(acT1_complex)

plt.figure(figsize=(10, 6))
plt.contourf(modulus)
plt.title("data_amplitude_T1 模值")
plt.xlabel("脉冲数")
plt.ylabel("距离")
plt.colorbar()
plt.show()

# 幅角图
phase1 = phase[435:460, 0:50]

# 创建 x 和 y 的值
x = np.arange(phase1.shape[1])
y = np.arange(phase1.shape[0])

# 创建一个二维网格
X, Y = np.meshgrid(x, y)

# 根据角度创建向量
U = np.cos(phase1)
V = np.sin(phase1)

# 绘制向量场图，所有的箭头都有相同的长度
plt.figure(figsize=(10, 6))
plt.quiver(X, Y, U, V)
plt.title("data_amplitude_T1 相位(435:460, 0:50)")
plt.xlabel("脉冲数")
plt.ylabel("距离")
# 显示图像
plt.show()
```

演示.ipynb

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy.signal import stft
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['axes.unicode_minus'] = False
    acT1_complex = np.load(r"C:\Users\Timothy\Desktop\数学建模相关\二轮\数据集
\data_amplitude_T1_complex.npy")

df = pd.DataFrame(acT1_complex)
df445 = df.iloc[445]
df616 = df.iloc[616]
df100 = df.iloc[100]
dflist = [df445,df616,df100]
titlelist = ['445','616','100']

# 绘制频谱图（注意：specgram 函数内置了 STFT 的过程，所以输入数据只包含了 dflist 中的元
素，
# 而不需要如 Zxx445 这样的 STFT 结果，因此将 STFT 部分下移）
# 创建一个 1x3 的子图布局
fig, axs = plt.subplots(1, 3, figsize=(10, 6))
for i in range(3):
    cax = axs[i].specgram(dflist[i], NFFT=1024, Fs=1, noverlap=500, cmap='jet')
    fig.colorbar(cax[3], ax=axs[i])
    axs[i].set_ylim(-0.3, 0.3)
    axs[i].set_title(f'{titlelist[i]}距离单位')
    axs[i].set_xlabel('脉冲数（时间）')
    axs[i].set_ylabel('归一化频率')
plt.tight_layout()
plt.savefig('图/1.png')
plt.show()

# 短时傅里叶变换（STFT）
f, t, Zxx445 = stft(df445, return_onesided=False)
f, t, Zxx616 = stft(df616, return_onesided=False)
f, t, Zxx100 = stft(df100, return_onesided=False)
Zxx445_max_f = np.argmax(np.abs(Zxx445), axis=0)
Zxx616_max_f = np.argmax(np.abs(Zxx616), axis=0)
Zxx100_max_f = np.argmax(np.abs(Zxx100), axis=0)
# 创建一个 2x3 的子图布局
fig, axs = plt.subplots(2, 3, figsize=(15, 10))

# 绘制时频谱图
for i, Zxx in enumerate([Zxx445, Zxx616, Zxx100]):
    axs[0, i].specgram(dflist[i], NFFT=1024, Fs=1, noverlap=500, cmap='jet')
    axs[0, i].set_ylim(-0.3, 0.3)
    axs[0, i].set_title(f'{titlelist[i]}距离单位')

# 绘制散点图
```

```
for i, Zxx_max_f in enumerate([Zxx445_max_f, Zxx616_max_f, Zxx100_max_f]):
    axs[1, i].scatter(t, f[Zxx_max_f], s=1)
    axs[1, i].set_ylim(-0.3, 0.3)
    print(f"{titlelist[i]}距离单位 RI:", sum(f[Zxx_max_f]))
    print(f"{titlelist[i]}距离单位 FV:", sum(np.diff(f[Zxx_max_f])))
plt.tight_layout()
plt.savefig('图/2.png')
plt.show()
```

检测.ipynb

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy.signal import stft
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['axes.unicode_minus'] = False
f = np.load(r"C:\Users\Timothy\Desktop\数学建模相关\二轮\数据集
\acT1_cplx_stft_f.npy")
t = np.load(r"C:\Users\Timothy\Desktop\数学建模相关\二轮\数据集
\acT1_cplx_stft_t.npy")
Zxx = np.load(r"C:\Users\Timothy\Desktop\数学建模相关\二轮\数据集
\acT1_cplx_stft_Zxx.npy")
print(f.shape, t.shape, Zxx.shape)
Zxx100p_max_f = np.argmax(np.abs(Zxx[100:, :, :]), axis=1)
print(Zxx100p_max_f.shape)

RI = np.array([sum(f[Zxx100p_max_f[i]]) for i in range(Zxx100p_max_f.shape[0])])
FV = np.array([sum(np.diff(f[Zxx100p_max_f[i]])) for i in
range(Zxx100p_max_f.shape[0])])
# FV_std = np.array([np.std(np.diff(f[Zxx100p_max_f[i]])) for i in
range(Zxx100p_max_f.shape[0])])
print(min(abs(RI)))
print(min(abs(FV)))
# print(min(abs(FV_std)))

from sklearn.svm import OneClassSVM
X0 = np.column_stack((RI[0:650], FV[0:650]))
X = np.column_stack((np.concatenate((RI[:320], RI[360:500], RI[576:650])),
                     np.concatenate((FV[:320], FV[360:500], FV[576:650]))
                     ))# 选择合适的训练集
# X = np.column_stack(Zxx100p_max_f[:250, :])# 选取合适的训练集
# 创建 OneClassSVM 模型
model = OneClassSVM(kernel='rbf', nu=0.038, gamma='scale')
# 训练模型
model.fit(X)

# 使用训练好的模型来预测新的样本
y_pred = model.predict(X0)
print(y_pred[345], y_pred[516])
```

```
print(sum(y_pred))
ys = []
for i, y in enumerate(y_pred):
    if y == -1:
        ys.append(i)
print(ys)
targets = 20

print(targets/((650-sum(y_pred))/2))
mask = np.ones(RI.shape, dtype=bool)
mask[ys] = False

# 使用遮罩选择要绘制的点
masked_RI = RI[mask]
masked_FV = FV[mask]

# 在散点图中绘制这些点
plt.scatter(masked_RI[:650], masked_FV[:650], s=5)

# 在散点图中绘制选出的点，使用不同的颜色和大小
plt.scatter(RI[ys], FV[ys], s=100)
plt.title("分类结果")
plt.xlabel('RI')
plt.ylabel('FV')
plt.show()
```

8.2.3 第三题

第三题演示.ipynb

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy.ndimage import gaussian_filter
from matplotlib.colors import Normalize
import cv2

plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['axes.unicode_minus'] = False

acT1_complex = np.load(r"C:\Users\Timothy\Desktop\数学建模相关\二轮\数据集
\data_amplitude_T1_complex.npy")
df = pd.DataFrame(acT1_complex)

plt.figure(figsize=(10, 6))
Pxx, freqs, bins, im = plt.specgram(df.iloc[445], NFFT=1024, Fs=1, noverlap=500,
cmap='jet')
plt.ylim(-0.3, 0.3)
plt.colorbar()
```

```
plt.xlabel('脉冲数')
plt.ylabel('频率')
plt.title('原始频谱图')
plt.savefig('original.png')
plt.close()

# 对 Pxx 应用高斯滤波
Pxx_filtered = gaussian_filter(Pxx, sigma=0.1)
# 保存原来的颜色映射
vmin = np.min(10 * np.log10(Pxx_filtered))
vmax = np.max(10 * np.log10(Pxx_filtered))
norm = Normalize(vmin=vmin, vmax=vmax)
# 特异性抑制
index = np.where((freqs >= 0.03) & (freqs <= 0.2))
def sigmoid(x):
    return 1 / (1 + np.exp(-x))
suppress_func = 1 - sigmoid(Pxx_filtered[index]/500000000) # 计算抑制函数
Pxx_filtered[index] = Pxx_filtered[index] * suppress_func

# 保存原图像
plt.figure(figsize=(10, 6))
plt.pcolormesh(bins, freqs, 10 * np.log10(Pxx_filtered), cmap='jet', norm=norm)
plt.ylim(-0.3, 0.3)
plt.colorbar()
plt.xlabel('脉冲数')
plt.ylabel('频率')
plt.title('频谱图')
plt.savefig('p1.png')
plt.close()

img = cv2.imread('p1.png')
dst = cv2.fastNlMeansDenoisingColored(img, None, 10, 10, 7, 21)
cv2.imwrite('p2.png', dst)

# 读取图像，并确定处理区域
img = dst
roi = img[:, :700]

# 提取红色区域
hsv = cv2.cvtColor(roi, cv2.COLOR_BGR2HSV)
lower_red = np.array([0, 70, 50])
upper_red = np.array([10, 255, 255])

# 查找轮廓
mask = cv2.inRange(hsv, lower_red, upper_red)
res = cv2.bitwise_and(roi, roi, mask=mask)
gray = cv2.cvtColor(res, cv2.COLOR_BGR2GRAY) # 将图像转换为灰度图
_, thresh = cv2.threshold(gray, 1, 255, cv2.THRESH_BINARY) # 应用阈值操作，将图像
转换为二值
```

```
contours, _ = cv2.findContours(thresh, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)

# 在图像上绘制轮廓并保存
cv2.drawContours(img, contours, -1, (255, 255, 255), 3)
cv2.imwrite('p31.png', img)
# 绘制二值图像并保存
new_img = np.zeros_like(img) # 创建一个与 img 相同大小的全黑图像
cv2.drawContours(new_img, contours, -1, (255, 255, 255), 3)
cv2.imwrite('p32.png', new_img)

# 将图像的像素值除以 255，得到 0 和 1 的矩阵
binary_matrix = gray / 255
# 打印矩阵
print(img.shape)
print(np.sum(binary_matrix))
```

第三题检测.ipynb

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy.ndimage import gaussian_filter
from matplotlib.colors import Normalize
import cv2

plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['axes.unicode_minus'] = False

acT1_complex = np.load(r"C:\Users\Timothy\Desktop\数学建模相关\二轮\数据集
\data_amplitude_T1_complex.npy")
df = pd.DataFrame(acT1_complex)

classified = []
classified_p = []
# for i in range(100, df.shape[0]):
# for i in range(430, 461):
# for i in range(600, 631):
for i in range (720, 751):
    # 一次去噪+特异性滤波

    # 生成 Pxx, freqs, bins
    plt.figure(figsize=(10, 6))
    Pxx, freqs, bins, im = plt.specgram(df.iloc[i], NFFT=1024, Fs=1, noverlap=500,
cmap='jet')
    plt.close()
    # 对 Pxx 应用高斯滤波
    Pxx_filtered = gaussian_filter(Pxx, sigma=0.1)
    # 保存原来的颜色映射
    vmin = np.min(10 * np.log10(Pxx_filtered))
```

```
    vmax = np.max(10 * np.log10(Pxx_filtered))
    norm = Normalize(vmin=vmin, vmax=vmax)
    # 特异性抑制
    index = np.where((freqs >= 0.03) & (freqs <= 0.2))
    def sigmoid(x):
        return 1 / (1 + np.exp(-x))
    suppress_func = 1 - sigmoid(Pxx_filtered[index]/500000000) # 计算抑制函数
    Pxx_filtered[index] = Pxx_filtered[index] * suppress_func
    # 使用 matplotlib 绘制图像
    fig, ax = plt.subplots(figsize=(10, 6))
    cax = ax.pcolormesh(bins, freqs, 10 * np.log10(Pxx_filtered), cmap='jet',
norm=norm)
    ax.set_title('频谱图')
    ax.set_xlabel('脉冲数')
    ax.set_ylabel('频率')
    ax.set_ylim(-0.3, 0.3)
    fig.colorbar(cax)
    plt.close()
    # 将 matplotlib 图像转换为 OpenCV 图像
    fig.canvas.draw()
    img = np.array(fig.canvas.renderer.buffer_rgba())
    img = cv2.cvtColor(img, cv2.COLOR_RGBA2BGR)


    # 轮廓提取（二值图）

    # 读取图像，并确定处理区域
    img = cv2.fastNlMeansDenoisingColored(img, None, 10, 10, 7, 21) # 二次去噪
（OpenCV）
    roi = img[:, :700]
    # 提取红色区域
    hsv = cv2.cvtColor(roi, cv2.COLOR_BGR2HSV)
    lower_red = np.array([0, 70, 50])
    upper_red = np.array([10, 255, 255])
    # 查找轮廓
    mask = cv2.inRange(hsv, lower_red, upper_red)
    res = cv2.bitwise_and(roi, roi, mask=mask)
    gray = cv2.cvtColor(res, cv2.COLOR_BGR2GRAY) # 将图像转换为灰度图
    _, thresh = cv2.threshold(gray, 1, 255, cv2.THRESH_BINARY) # 应用阈值操作，将
图像转换为二值
    contours, _ = cv2.findContours(thresh, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)


    # 分类

    # 将图像的像素值除以 255，得到 0 和 1 的矩阵
    binary_matrix = gray / 255
    threshcls = 100
```

```
    print(f"{i}:", np.sum(binary_matrix))
    if (np.sum(binary_matrix) > threshcls):
        classified.append(1)
        # classified_p.append([i, np.sum(binary_matrix)])
    else:
        classified.append(0)
        # classified_p.append([i, np.sum(binary_matrix)])
    # 打印分类结果
    print(classified)

plt.plot([i[0] for i in classified_p], [i[1] for i in classified_p] , 'o-')

plt.plot([i[0] for i in classified_p], threshcls * np.ones(len(classified_p)),
'r--')
plt.xlabel('位置')
plt.ylabel('矩阵和')
plt.title('浮标 1 附近矩阵和随位置变化图')
plt.show()
print(len(classified_p))
```