基于混沌自适应鲸鱼算法的波浪能转换运动模型优化研究

摘要

本文分别考虑波浪能装置的垂荡与纵摇两种运动形式，建立浮子与振子的运动模型，并进行数值求解；最后优化相关参数以获得最大输出功率.

针对问题一，建立描述波浪能装置垂荡运动形式的微分方程组，通过**四阶龙格库塔方法**数值求解. 首先，分别对浮子和振子进行受力分析，运用牛顿第二定理得到情况一与情况二的微分方程组，并将该方程组转化为矩阵形式；其次，根据**四阶龙格-库塔方法**，采用 **ode45 求解器**数值求解，得到 40 个波浪周期内时间间隔为 0.2 s 的垂荡位移和速度，结果存放在支撑文件 **result1-1.xlsx** 和 **result1-2.xlsx** 中，给定时刻数据见 (情况一) 表1和 (情况二) 表2. 最后，利用 Bode 图对海浪能装置系统进行**性能分析**，讨论其稳定性和抗干扰能力.

针对问题二，首先确定系统瞬时功率为阻尼器的阻尼力与浮子和振子的相对速度的乘积，平均输出功率为系统**稳定状态**一段时间内功率的均值；其次，以最大平均输出功率为目标函数，情况一以阻尼系数为决策变量，情况二以比例系数和幂指数为决策变量，以参数的上下界为约束条件，建立优化模型. 采用**混沌自适应鲸鱼算法**求解，得到情况一的最大功率为 **232.922W**，最优阻尼系数为 **36384**；情况二的最大功率为 **234.169W**，最优比例系数为 **97546**，幂指数为 **0.4110**. 最后，对情况一、二参数变化进行仿真模拟，验证结果正确性.

针对问题三，分别分析波浪能装置的垂荡和纵摇两种运动形式，取浮子竖直坐标、振子竖直坐标、浮子角位移、振子角位移为**广义坐标**，通过隔离法受力分析列出运动学方程，通过隔离力矩分析列出角动量方程，联合建立耦合的**微分方程组**，并转化为矩阵形式，仍然通过四阶龙格库塔方法求解. 求解结果见支撑文件 **result3.xlsx** 中，其中给定时刻结果见表4.

针对问题四，首先确定阻尼器输出功率为阻尼力与相对速度之积、阻尼力矩与相对角速度之积这两者之和，再按照问题二方法，以阻尼器输出功率为目标函数，以直线阻尼器系数和旋转阻尼器系数为决策变量，建立优化模型. 仍然采用**混沌自适应鲸鱼算法**进行求解，得到最大输出功率为 **331.61W**，最优直线阻尼系数为 **58254**，最优旋转阻尼系数为 **81689**.

本文优点：(1) 将非线性的微分方程组转换成线性形式，方便计算;(2) 采用四阶龙格库塔方法迭代求解，计算精度高; (3) 混沌反馈自适应鲸鱼算法能很好地避免陷入局部最优，并且收敛速度明显优于等步长搜索.

关键词：运动模型　常微分方程模型　四阶龙格-库塔方法　混沌自适应鲸鱼算法

1 问题重述

1.1 问题背景

能源是人类赖以生存的重要保障. 海洋中的潮汐涨落，波浪翻涌蕴含巨大的潜在可利用能源. 其分布广泛、安全清洁、储量丰富，有十分光明的应用前景. 近年来，已有许多关于波浪能发电的相关研究. 波浪能量转换方式有振荡水柱式、振荡浮子式、摆式和越浪式，其中振荡浮子式具有体积小、效率高、成本低等优点，有利于广泛灵活布局远海海域进行波浪能收集工作[1] .

振荡浮子波浪能转换装置主要由浮子、振子和 PTO(能量输出系统) 组成. 浮子外形为圆柱形，底部连接圆锥形壳体. 内部振子通过弹簧和阻尼器与隔层相连，工作时振子沿中轴振动驱动阻尼器做功，产生能量输出. 图1为振荡浮子结构示意图.

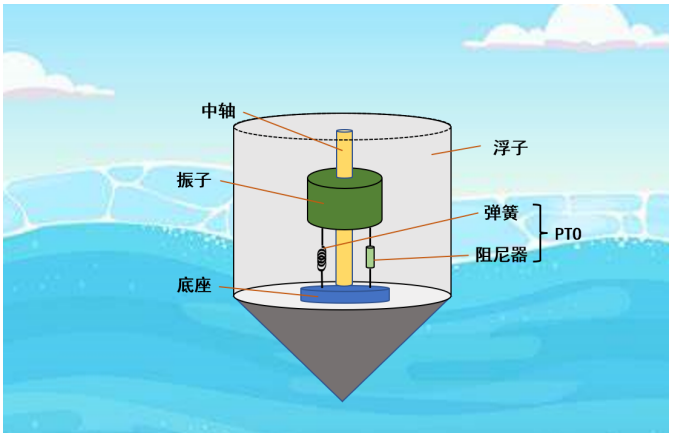


图 **1**　振荡浮子结构示意图

1.2 问题提出

问题一：初始时浮子和振子在海水中平衡静止，当海浪激励力为 $f\cos\omega t$ 时，仅考虑垂荡运动，试分析浮子和振子的运动状态. 分别讨论：(1) 阻尼器阻尼系数为常数;(2) 阻尼系数与相对速度的绝对值的根值成正比.

问题二：试建立数学模型，分别在问题一的两种讨论下确定阻尼器最优阻尼系数，使波浪转换装置的平均能量输出功率最大.

问题三：PTO 系统连接转轴架，使得中轴能绕转轴摆动. 在转动架上增加旋转阻尼器和扭转弹簧，在海浪作用下，浮子不仅上下垂荡，而且左右纵摇. 建立运动学模型，计算浮子和振子在波浪激励力和激励力矩作用下的运动状态.

问题四：当直线阻尼器和旋转阻尼器阻尼系数为常数时，试建立优化模型，计算最大平均输出功率.

2 模型假设

1. 不考虑海水的粘性和海水的涡流，将海水看做理想流体;
2. 忽略中轴、隔层底座和 PTO 的质量和运动中的摩擦;
3. 假设能量输出系统是线性的阻尼系统;
4. 忽略转轴架高度，振子到转轴的距离即为直线弹簧的长度;
5. 假设波浪能装置在垂荡和纵摇过程中，圆锥形区域始终不露出水面;

3 符号说明

符号	说明	单位
F_e, M_e	波浪激励力 (矩)	N
F_w, M_w	兴波阻尼力 (矩)	N
F_s, M_s	静水恢复力 (矩)	N
F_k, M_k	弹簧弹力 (矩)	N
F_d, M_r	阻尼器阻尼力 (矩)	N
m_1, m_2, m_a	浮子、振子、附加质量	kg
I_1, I_2, I_a	浮子、振子、附加转动惯量	$kg \cdot m^2$
k	弹簧刚度	N/m
K_m	扭转弹簧刚度	$N \cdot m$
K_d	直线阻尼器阻尼系数	$N \cdot s/m$
K_r	旋转阻尼器阻尼系数	$N \cdot s/m$
K_{wv}	垂荡兴波阻尼系数	$N \cdot s/m$
K_{wh}	纵摇兴波阻尼系数	$N \cdot s/m$
K_{ms}	静水恢复力矩系数	$N \cdot m$

注：表中未列出的符号以第一次出现时说明为准.

4 问题一模型的建立及求解

4.1 问题分析

问题一要去对波浪能转换装置进行运动学分析. 装置整体受到波浪激励力、兴波阻尼力、静水恢复力、附加惯性力. 波浪激励力是推动装置运动的主动力；行波阻尼力与运动速度成正比；静水恢复力是使浮体回到平衡状态的力，为重力和浮力的合力；附加惯性力可以等效成附加虚拟质量. 振子和浮子的相互作用为弹簧弹力和直线阻尼器阻力，其中弹簧弹力满足胡克定律，直线阻尼器阻力与浮子和振子相对速度成正比. 通过隔离法，可以列出浮子和振子相互关联的运动组. 给定初始体，求解该方程组得到浮子和振子位置和速度关系. 图2为问题一思维导图.

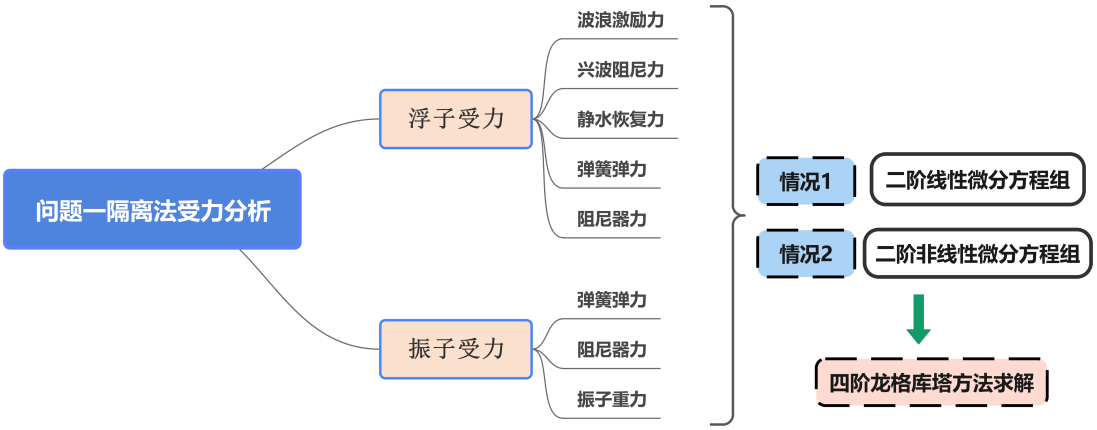


图 2　问题一思维导图

4.2 运动学模型建立

以平衡位置为参考点，垂直向上为坐标正方向，记浮子坐标位置为 x_1，振子坐标位置为 x_2，初始时刻浮子 $x_1 = 0$，振子 $x_2 = l_0$，l_0 是弹簧被振子自然压缩长度.

4.2.1 浮子受力分析

浮子在海水中受到主动力波浪激励力 F_e，兴波阻尼力 F_w, 静水恢复力 F_s, 图3为浮子受力分析图.

根据牛顿第二定律列出运动学方程

$$(m_1 + m_a)\ddot{x_1} = F_e - F_w - F_s - F_k - F_d. \tag{1}$$

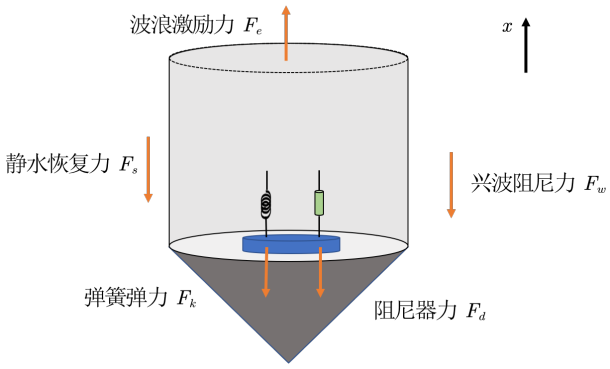


图 **3**　浮子受力分析图

4.2.2 振子受力分析

振子在浮子内部，其只受到弹簧弹力和阻尼器的阻力. 图4为振子受力分析图.

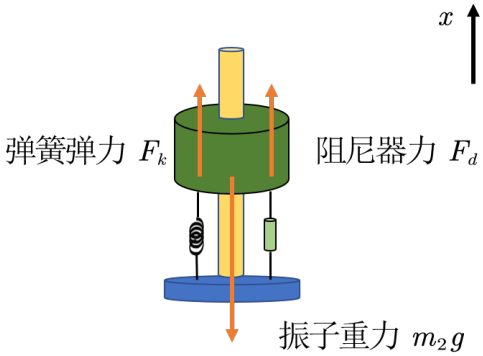


图 **4**　振子受力分析图

根据牛顿第二定律列出运动学方程

$$m_2\ddot{x}_2 = -m_2g + F_k + F_d. \tag{2}$$

4.2.3 几种作用力

- **波浪激励力**

问题一给出的波浪激励力为

$$F_e = f\cos\omega t \tag{3}$$

式中 f 为波浪激励力振幅，ω 为入射波浪的频率.

- **兴波阻尼力**

振荡浮子在海水中做垂直振荡运动，会带动周围海水运动，由此对浮子本身的振动起到阻碍作用的力. 其与浮子振荡速度成正比，背离速度方向 (受力分析时取负号，这里给出正值).

$$F_w = K_{wv}\dot{x}_1. \tag{4}$$

式中 K_{wv} 为垂荡兴波阻尼系数.

• 静水恢复力

振荡浮子在水中受到浮力和重力的作用，两者共同组成浮子平衡恢复力. 其中浮力等于平衡位置的浮力加上位移变化引起的浮力改变量. 波浪激励不足以使得振荡浮子下端圆锥形区域露出水面，浮子吃水深度始终在圆柱区域，浮力表达式为

$$F_f = (m_1 + m_2)g - \rho_s g\pi r_1^2 \cdot x_1,$$

带入得到静水恢复力 (方向向下) 表达式

$$\begin{aligned} F_s &= m_2 g - F_f \\ &= \rho_s g\pi r_1^2 \cdot x_1 - m_1 g. \end{aligned} \tag{5}$$

式中 $\rho_s g$ 为海水密度，r_1 为圆柱底半径.

• 弹簧弹力

根据胡克定律弹簧弹力与弹簧伸长量成正比.

$$F_k = k(L - x_2 + x_1). \tag{6}$$

式中 k 为弹簧刚度.

• 阻尼器阻力

直线阻尼器的阻尼力与浮子和振子的相对速度成正比.

$$F_d = \begin{cases} \text{情况 1：} K_d(\dot{x}_1 - \dot{x}_2), \\ \text{情况 2：} \alpha|\dot{x}_1 - \dot{x}_2|^{\beta}(\dot{x}_2 - \dot{x}_1). \end{cases} \tag{7}$$

式中 K_d 为直线阻尼器的阻尼系数，α 为比例系数，β 为幂指系数.

4.2.4 模型总述

情况 1

将上述讨论的几种力带入式 (1) 和 (2) 中得到浮子和振子满足的二阶微分方程组

$$\begin{cases} (m_1 + m_a)\ddot{x}_1 + (K_{wv} + K_d)\dot{x}_1 - K_d\dot{x}_2 + (k + \rho_s g\pi r_1^2)x_1 - kx_2 + kL - m_2 g = f\cos\omega t \\ m_2\ddot{x}_2 - K_d(\dot{x}_1 - \dot{x}_2) - k(L + x_1 - x_2) + m_2 g = 0 \end{cases} \tag{8}$$

情况 2

将情况 2 的阻尼力 $F_d = \alpha|\dot{x}_1 - \dot{x}_2|^{\beta}(\dot{x}_1 - \dot{x}_2)$ 带入方程 (1) 和方程 (2) 中，得到微分方程组.

$$\begin{cases} (m_1 + m_a)\ddot{x}_1 + K_{wv}\dot{x}_1 + \alpha|\dot{x}_1 - \dot{x}_2|^{\beta}(\dot{x}_1 - \dot{x}_2) + (k + \rho_s g\pi r_1^2)x_1 - kx_2 + kL - m_2 g = f\cos\omega t \\ m_2\ddot{x}_2 - \alpha|\dot{x}_1 - \dot{x}_2|^{\beta}(\dot{x}_1 - \dot{x}_2) - k(L + x_1 - x_2) + m_2 g = 0 \end{cases} \tag{9}$$

为了便于求解，记 $\boldsymbol{X}=\begin{bmatrix}x_1\\x_2\end{bmatrix}$，将上述方程写成矩阵形式

$$\boldsymbol{A}\ddot{\boldsymbol{X}}+\boldsymbol{B}\dot{\boldsymbol{X}}+\boldsymbol{C}\boldsymbol{X}=\boldsymbol{b}. \tag{10}$$

相关系数矩阵写在附录 A 中.

初始条件

以平衡位置为参考点，初始时刻浮子 $x_1=0$，振子 $x_2=l_0$，$l_0=L-\dfrac{m_2g}{k}$ 是弹簧被振子自然压缩长度. 初始时刻浮子和振子没有速度，则初始条件表示为

$$x_1(0)=0, \qquad \dot{x}_1(0)=0, \tag{11}$$

$$x_2(0)=l_0, \qquad \dot{x}_2(0)=0. \tag{12}$$

4.3 模型求解

对二阶微分方程组式 (8) 和式 (9) 采用数值求解的方法，在 Matlab 中采用 ode45 求解器求解，ode45 基本原理为四阶龙格-库塔方法.

• 四阶龙格-库塔方法

龙格库塔方法是进行微分方程数值解常用方法之一. 其主要优点是计算精度较高且程序简单. 常规的欧拉公式和梯形公式分别只具有一阶精度和二阶精度. 本文采用具有四阶精度的龙格-库塔方法. 龙格-库塔方法的基本思想是计算 $[x_i,x_{i+1}]$ 内多点处曲线的斜率，再用它们的某种线性组合作为曲线在 $[x_i,x_{i+1}]$ 内平均斜率的近似值，得到更高的计算精度.

下面给出标准四阶龙格-库塔公式：

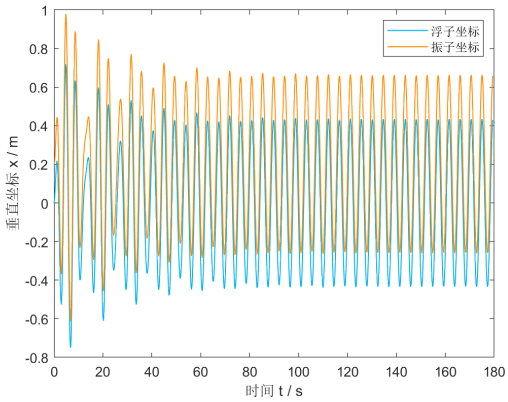
$$\begin{cases} y_{i+1}=y_i+\frac{h}{6}(k_1+2k_2+2k_3+k_4)\\ k_1=f(x_i,y_i)\\ k_2=f(x_i+\frac{h}{2},y_i+\frac{h}{2}k_1)\\ k_3=f(x_i+\frac{h}{2},y_i+\frac{h}{2}k_2)\\ k_4=f(x_i+h,y_i+hk_3) \end{cases} \tag{13}$$

二阶导数项可以用一阶导数的龙格-库塔公式给出.

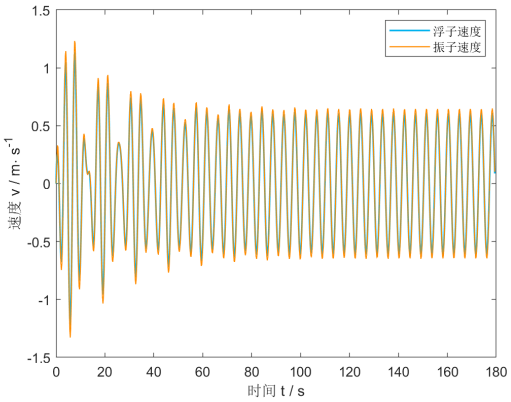
4.3.1 情况一求解结果

通过计算求解得到图5a的浮子和振子的位移运动图像，图5b为浮子的速度变化图像,

振子的初始坐标位置为 $x_2=l_0=0.2m$，因此振子坐标比浮子坐标位置高. 从图5a可以看出，第一次垂荡时，海浪激励浮子向上运动，带动振子同向上运动. 接着浪的波谷



(a) 情况一 **40** 个周期的位移变化图像



(b) 情况一 **40** 个周期的速度变化图像

图 **5**　情况一求解结果

带动浮子和振子向下运动. 再次向上运动时，弹簧拉扯浮子、带动振子向上运动到达峰值. 接下来循环往复垂荡. 经历两次较大的振幅周期后会有一次较小的振幅周期. 随着时间变化，振幅的差异越来越小，最终趋向稳定，且浮子和振子趋向同周期振荡，振荡频率为海水激励频率. 一段时间后，浮子垂荡振幅和振子垂荡振幅近似相等.

前 40 个波浪周期内时间间隔为 0.2 s 的垂荡位移和速度相关数据存入支撑文件 result1-1 中，表1列出 10s,20s,40s,60s,100s 的相关数据.

表 **1**　情况一结果

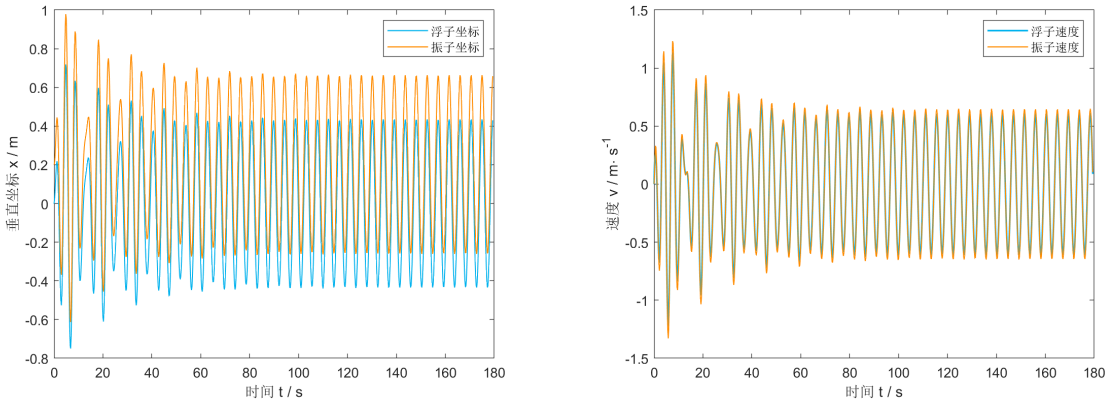
时间	浮子位移	浮子速度	振子位移	振子速度
10s	-0.1906	-0.6405	-0.2115	-0.6937
20s	-0.5905	-0.2406	-0.6341	-0.2721
40s	0.2854	0.3134	0.2966	0.3332
60s	-0.3144	-0.4792	-0.3314	-0.5154
100s	-0.0836	-0.6041	-0.0840	-0.6429

4.3.2 情况二求解结果

情况二是一个非线性方程组，采用龙格库塔方法递推得到浮子位置速度与振子位置速度的数值解.

图6a为情况二下位置变化图像，可以看出当直线阻尼器的阻尼系数与浮子和振子的相对速度的绝对值的幂成正比时，相比情况一阻尼系数为常数，位置结果变化并不大.

前 40 个波浪周期内时间间隔为 0.2 s 的垂荡位移和速度存入支撑文件 result1-2 中，表2列



(a) 情况一 **40** 个周期的位移变化图像　　**(b)** 情况一 **40** 个周期的速度变化图像

图 **6**　情况二求解结果

出 10 s、20 s、40 s、60 s、100 s 时，浮子与振子的垂荡位移和速度. 对比表1和表2，其

表 **2**　情况二结果

时间	浮子位移	浮子速度	振子位移	振子速度
10s	-0.2058	-0.6527	-0.2347	-0.6978
20s	-0.6111	-0.2539	-0.6608	-0.2764
40s	0.2687	0.2959	0.2802	0.3139
60s	-0.3273	-0.4904	-0.3492	-0.5259
100s	-0.0883	-0.6099	-0.0937	-0.6489

中特定时刻浮子的位置速度与振子的位置速度变化较小，情况一与情况二相比，浮子和振子位移变化在厘米量级，速度变化的数量级则为厘米每秒.

4.4 波浪能装置系统性能分析

对于线性定常系统，由谐波输入产生的输出稳态分量仍然是与输入同频率的谐波函数，而幅值和相位的变化是频率 ω 的函数，且与系统数学模型相关. 为此，定义谐波输入下，输出响应中与输入同频率的谐波分量与谐波输入的幅值之比 $A(\omega)$ 为幅频特性，相位之差 $\varphi(\omega)$ 为相频特性，并称其指数表达形式

$$G(j\omega) = A(\omega)e^{j\varphi\omega}, \tag{14}$$

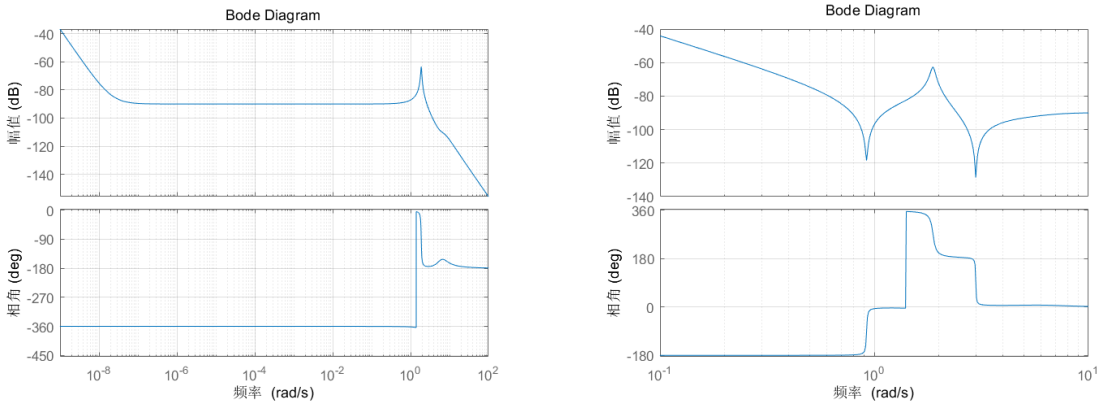
为系统的频率特性.

对数频率特性曲线又称为伯德图，由对数幅频曲线和对数相频曲线组成. 对数频率特性曲线的横坐标按 $\lg\omega$ 分度，单位为弧度/秒（rad/s），对数幅频曲线的纵坐标按

$$L(\omega) = 20 \cdot \lg|G(j\omega)| = 20\lg(A(\omega)) \tag{15}$$

线性分度，单位是分贝 (dB). 对数相频曲线的纵坐标按 $\varphi(\omega)$ 线性分度，单位为度.

在该波浪能装置系统中输入为波浪激励力，输出为浮子与振子振子的位置，分别作出两个输出关于输入的伯德图，如图7所示，上方为对数幅频曲线，下方为对数相频曲线.



(a) 浮子 **Bode** 图　　**(b)** 振子 **Bode** 图

图 **7**　**Bode** 图

● 系统的稳定性

通过分析伯德图，依据奈奎斯特稳定判据[2]，该波浪能装置系统处于临界稳定状态. 临界稳定状态按李雅普洛夫的定义属于稳定的状态，但是如果系统参数发生变化，系统就会进入非稳态，在系统参数不变情况下经历一段时间后，系统则又会进入临界稳定态[3] . 这与实际情况相吻合，在波浪能装置的参数发生变化后，振子与浮子将会打破原来的规律性运动，而在一段时间后，他们又会在新的系统下达到稳定.

● 系统的抗干扰能力

系统开环对数幅频在高频段的幅值，直接反映了系统对输入高频干扰信号的抑制能力. 高频特性的分贝值愈低，衰减越强，系统抗干扰能力愈强. 由两张系统的幅频曲线图可以看出，在高频段幅值的分贝数很低，可见系统抗干扰能力较强.

5 问题二模型的建立与求解

5.1 问题分析

本问要在问题一讨论的两种情况下建立直线阻尼器的最优阻尼优化模型，使得能量输出系统 (PTO) 的平均输出功率最大. 目标函数表示为最大平均输出功率，功率可以表示为阻尼器的阻尼力乘以相对速度. 情况一，决策变量即为阻尼系数；情况二，决策变量为比例系数和幂指数. 通过优化算法带入寻找最优参数. 图8为问题二思维导图.

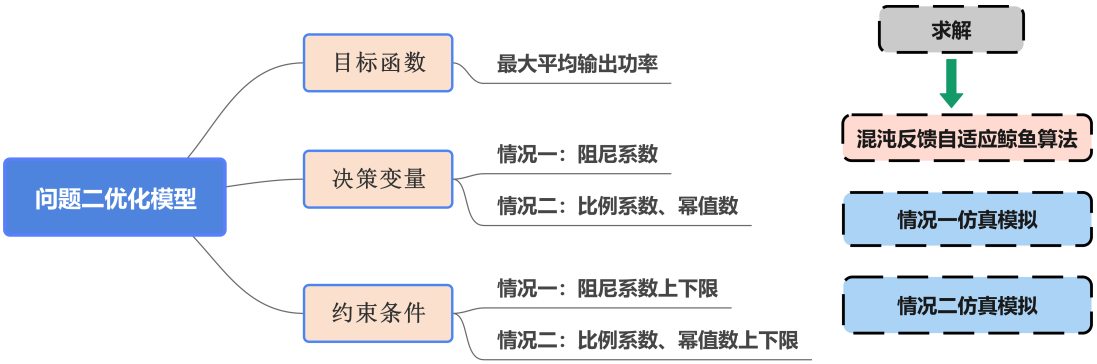


图 **8**　问题二思维导图

5.2 最大输出功率优化模型

- **决策变量**

情况 1：阻尼系数为常量，决策变量为阻尼系数 K_d；

情况 2：阻尼系数与浮子和振子的相对速度的绝对值的幂指数成正比 $K_d = \alpha|\dot{x}_1 - \dot{x}_2|^{\beta}$，决策变量为比例系数 α 和幂指数 β.

- **目标函数**

目标函数为最大平均输出功率，功率可以表示为力与速度的乘积. 刚开始时，波浪能装置受到海浪激励力开始垂荡一段时间后趋向稳定. 考虑到波浪能装置一般处于稳定工作状态，PTO 系统的平均输出功率应为稳定状态的输出功率. 本文取一段时间后垂荡趋向稳定过程的平均功率作为 PTO 输出功率.

记数值计算过程某一时刻浮子速度 $\dot{x}_1^k$ 振子速度 $\dot{x}_2^k$，则平均功率即为等时间步长下瞬时功率的均值.

$$P = \begin{cases} \text{情况一：} K_d \sum\limits_{k=n_1}^{n_2} \dfrac{(\dot{x}_1^k - \dot{x}_2^k)^2}{N} \\ \text{情况二：} \alpha \sum\limits_{k=n_1}^{n_2} \dfrac{|\dot{x}_1^k - \dot{x}_2^k|^{\beta}(\dot{x}_1^k - \dot{x}_2^k)^2}{N} \end{cases} \tag{16}$$

其中 K_d 为常量，α 为比例系数，β 为幂值数.N 为计算区间 $[n_1, n_2]$ 内数值解的个数. 本

文采取计算精度 0.001s，针对情况一计算区间采取 $20s \sim 113.502s$，情况二计算区间采取 $40s \sim 113.502s$.

• **约束条件**

题目给出优化参量的上下限：

$$\begin{cases} 0 \leq K_d \leq 100000, \\ 0 \leq \alpha \leq 100000, \\ 0 \leq \beta \leq 1. \end{cases} \tag{17}$$

5.2.1 模型总述

情况 1：

$$\begin{aligned} \max_{K_d} P = K_d \sum_{k=n_1}^{n_2} \frac{(\dot{x}_1^k - \dot{x}_2^k)^2}{N}, \\ 0 \leq K_d \leq 100000. \end{aligned} . \tag{18}$$

情况 2：

$$\max_{\alpha,\beta} P = \alpha \sum_{k=n_1}^{n_2} \frac{|\dot{x}_1^k - \dot{x}_2^k|^\beta (\dot{x}_1^k - \dot{x}_2^k)^2}{N}$$

$$s.t. \quad \begin{cases} 0 \leq \alpha \leq 100000, \\ 0 \leq \beta \leq 1. \end{cases} \tag{19}$$

5.3 鲸鱼算法求解优化模型

式 (18) 和式 (19) 均为简单约束的优化模型，这一类模型常采用等步长搜索等暴力算法或常规 PSO 算法，但针对本文模型上述算法有计算复杂度高，或容易陷入局部最优的局限性. 本文采用混沌反馈自适应鲸鱼算法克服这些局限性.

5.3.1 鲸鱼算法基本原理

鲸鱼优化算法（whale optimization algorithm, WOA）是 Mirjalili 和 Lewis 于 2016 年受鲸鱼独特的泡泡网觅食行为启发而提出的一种群体智能优化算法[4]，其源于对座头鲸独有的泡泡网络觅食行为. 混沌反馈自适应鲸鱼算法相较于鲸鱼优化算法做出了三点优化：利用 Sine 混沌初始化种群，增加反馈阶段，自适应调节惯性权值. 实验表明，混沌反馈自适应鲸鱼算法 (CFAWOA) 相较于鲸鱼优化算法提高了收敛速度与精度，同时使程序更容易脱离局部最优从而提升全局搜索能力[5] .

5.3.2 混沌反馈自适应鲸鱼算法

(1) **混沌初始化：**鲸鱼种群 N，搜索空间 d，第 i 只鲸鱼在空间中的位置为

$$X_i = (X_i^1, X_i^2, X_i^3, \cdots, X_i^d)$$

其代表一个可行解. 鲸鱼算法的初始值会影响算法的收敛速度和精度. 采用混沌映射, 增加算法随机性和遍历性，可以得到更快的收敛速度和更强的全局搜索能力. 本文**采用 Sine 混沌**初始化鲸鱼种群[3].Sine 混沌自映射为

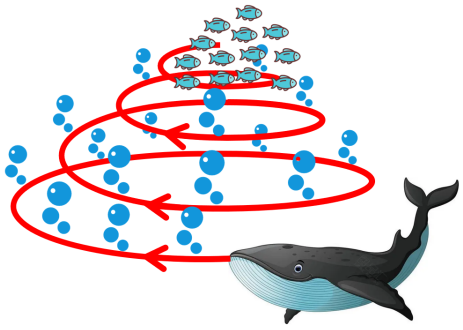


图 **9**　鲸鱼优化算法原理示意图

$$x_{n+1} = \frac{4}{a}\sin \pi x_n . n = 1, 2, \cdots, d.$$

其中 $a \in (0, 4]$.

(2) 计算鲸鱼种群的适应能力 $f(X_i)$;

(3) 识别猎物位置并包围猎物. 假定当前鲸鱼个体位置为最理想位置，其他鲸鱼向最优位置靠拢，更新方式为

$$X(t+1) = X_p(t) - A \cdot |C \cdot X_p(t) - X(t)| \tag{20}$$

式中 t 代表迭代次数，$X_p = (X_p^1, X_p^2, \cdots, X_p^d)$ 为目标猎物位置.$A = 2a \cdot r_1 - a, C = 2 \cdot r_2$,$r_1, r_2 = rand(0, 1)$;a 为收敛因子，其随迭代次数线性减小.

(4) 捕食行为分为收缩包围和螺旋运动两种. 两者以概率 p 进行选择. 收缩包围迭代公式同式 (20)；螺旋运动是个体以螺旋行进方式向最佳鲸鱼靠近，其迭代形式为

$$X(t+1) = X_p(t) + |X_p(t) - X(t)| \cdot e^{bx} \cdot \cos 2\pi l \tag{21}$$

式中常数 b 控制对数螺旋形状，$l = rand(-1, 1)$.

(5) **反馈调节:** 鲸鱼觅食过程中距离较远的鲸鱼与距离较近的鲸鱼进行反馈信息交流，使距离较远的鲸鱼快速移动到猎物附近，从而提高算法的收敛速度. 反馈调节为

$$X_{worst_new} = X_{worst} - r \cdot (X_p - X_{worst})$$

式中 r 为 $[0, 1]$ 的随机数，当前最差解 X_{worst} 更新为 X_{worst_new}.

(6) **自适应惯性权重:** 迭代初期，惯性权重应较大，保证算法的全局搜索能力；迭代后期，惯性权重应较小，保证较强的局部搜索能力. 本文因为自适应惯性权重 ω 为

$$\omega = 0.2 + \frac{1}{0.4 + \exp -f(x)/f_0^{\,t}}$$

其中 t 表示当前的迭代次数，$f(X_i)$ 为鲸鱼 X_i 的适应度值，f_0 表示第一次迭代计算中最佳适应度值. 鲸鱼捕食行为包括收缩包围和螺旋运动，加入自适应权重后，迭代公式

写为

$$\begin{aligned} &\text{收缩包围:} X(t+1) = \omega \cdot X_p(t) - A \cdot |C \cdot X_p(t) - X(t)|, \\ &\text{螺旋运动：} X(t+1) = \omega X_p(t) + |X_p(t) - X(t)| \cdot e^{bx} \cdot \cos 2\pi l. \end{aligned} \tag{22}$$

下面给出**混沌反馈自适应鲸鱼算法伪代码**

Algorithm 1 CFAWOA

Input: 最大迭代次数 T; 种群规模 N

Output: $X_p, f(X_p)$

Initialize X_i ▷ 采用 Sine 混沌映射初始鲸鱼种群

$f(X_i)$ ▷ 计算当前每个鲸鱼个体适应度值，并记录当前最优适应度

while $t \leq T$ **do**

  for $i = 1 \rightarrow N$ **do**

    计算自适应惯性权重 ω，并更新参数 A, C, l, p, a ▷ 更新参数

    if $p < 0.5$ **then**

      if $|A| < 1$ **then**

        收缩包围更新鲸鱼位置 ▷ 鲸鱼个体向猎物位置包围

      else if $|A| \geq 1$ **then**

        X_{worst_new} 选择最差鲸鱼个体并跟新位置 ▷ 反馈调节

      end if

    else if $p \geq 0.5$ **then**

      螺旋收缩更新鲸鱼位置 ▷ 螺旋运动向猎物包围

    end if

    重新计算适应度值 $f(X_i)$，并更新最优个体

    $t \leftarrow t + 1$

  end for

end while

return $X_p, f(X_p)$

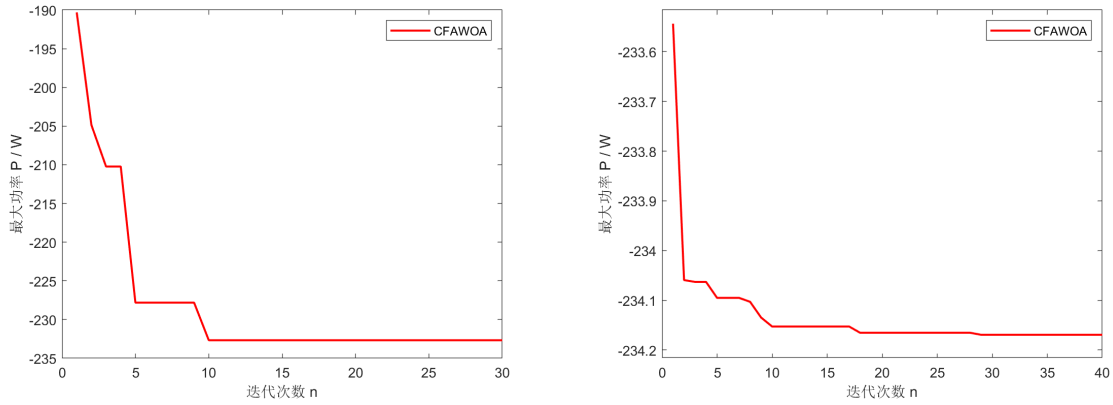
5.4 问题二结果呈现

通过求解得到情况一、二结果如表3所示. 对情况一，直线阻尼器阻尼系数 $K_d = 36258$；对情况二，比例系数 $\alpha = 96535$，幂指数 $\beta = 0.4108$.

对比情况一和情况二，非线性阻尼系数最大平均功率略优于线性阻尼系数，但两者最大平均输出功率差距并不大.

表 **3**　问题二结果

情况	最大功率	最优参数值
情况一	232.922	$K_d = 36384$
情况二	234.169	$\alpha = 97546.\ \beta = 0.4110$



(a) 情况一收敛曲线　　**(b)** 情况二收敛曲线

图 **10**　收敛曲线

5.5 问题二结果分析

5.5.1 仿真模拟验证

- **情况一仿真模拟**

利用 CFAWOA 求得情况一阻尼系数为基础，±1000 范围，采用等步长搜索法，仿真模拟阻尼系数与输出功率的关系. 从图11中可以看出，输出功率随阻尼系数曲线振荡剧烈，存在多个局部最优位置. 采用 CFAWOA 与等步长搜索法仿真模拟得到的最优阻尼系数结果一致，但有更低的时间复杂度，体现了算法的优越性.

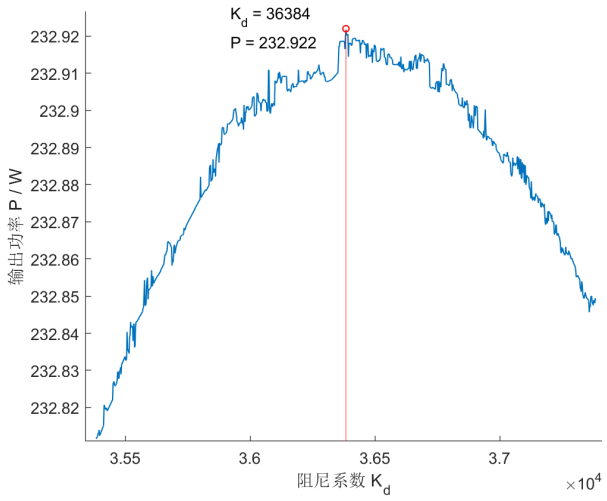


图 **11**　情况一仿真结果

- **情况二仿真模拟**

利用 CFAWOA 求得情况二结果为基础，采用等步长搜索，对全局进行搜索仿真模拟. 图12为仿真模拟热力图，从中可以看出输出功率随幂值数变化比较敏感，而比例系数对功率影响较小. 通过 CFAWOA 算法得到的最优参数为 97546 和 0.4110.

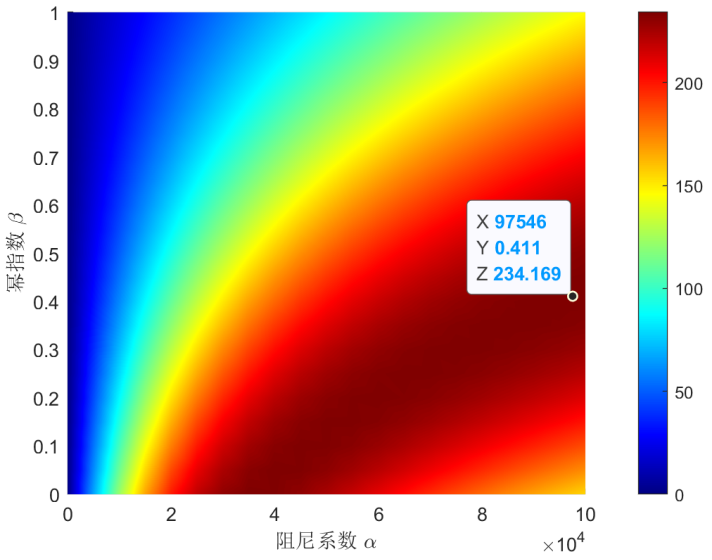


图 **12**　情况二仿真结果

5.5.2 弹簧刚度调节对结果影响

振子振动的主动力主要来自弹簧弹力，弹簧刚度对输出功率有决定性影响，对. 如图13，输出功率与弹簧刚度呈负相关，弹簧刚度越大，输出功率越低.

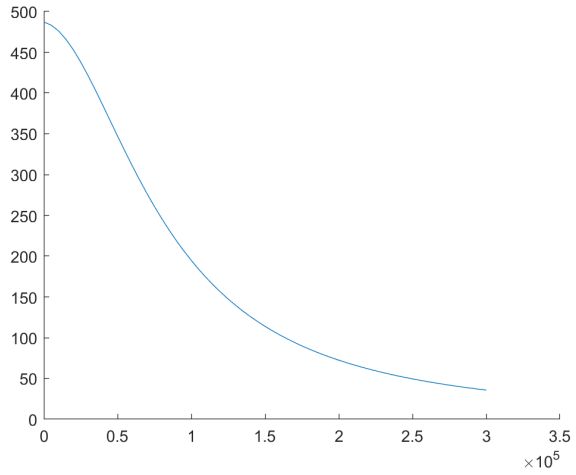


图 **13**　弹簧刚度调节对结果影响

6 问题三模型的建立与求解

6.1 问题分析

问题三通过添加中轴底座增加波浪能装置工作的自由度，从原本的只在垂直方向上振荡，加上可以左右纵摇. 沿振子沿中轴方向振动，通过直线阻尼器对外输出能量. 在左右纵摇过程中，扭转弹簧提供力矩，浮子和振子产生角度差，旋转阻尼器输出能量. 海浪激励力带动浮子垂荡，海浪激励力矩推动浮子纵摇. 同问题一建系，设置浮子竖直坐标、振子竖直坐标、浮子角位移、振子角位移为四个广义坐标，分别表示系统的四个自由度. 通过隔离法分析，列出运动学方程，通过微分方程数值方法求解. 图14为问题三思维导图.

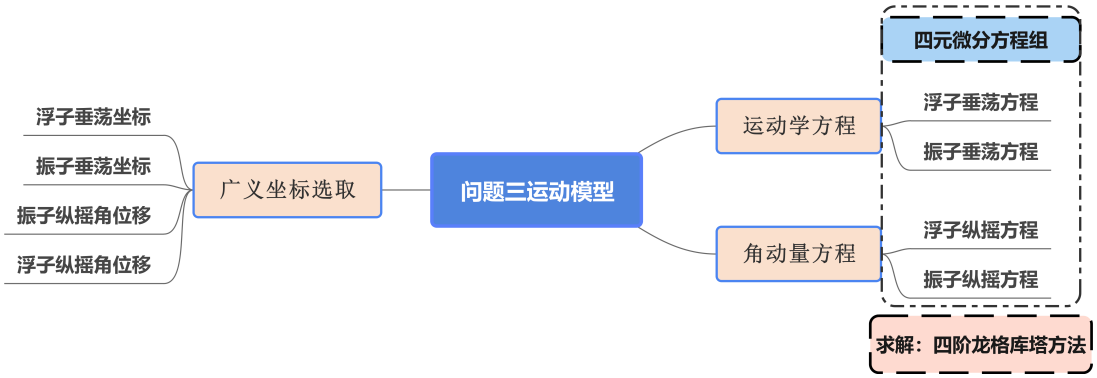


图 **14**　问题三思维导图

6.2 运动学模型建立

图15展示了两自由度波浪能装置的工作原理，同问题一建系，垂直向上为 x 轴正方向，水平向右为 y 轴正方向. 系有四个自由度，取浮子竖直坐标 x_1、振子竖直坐标 x_2、浮子角位移 θ_1、振子角位移 θ_2 为系统四个广义坐标.

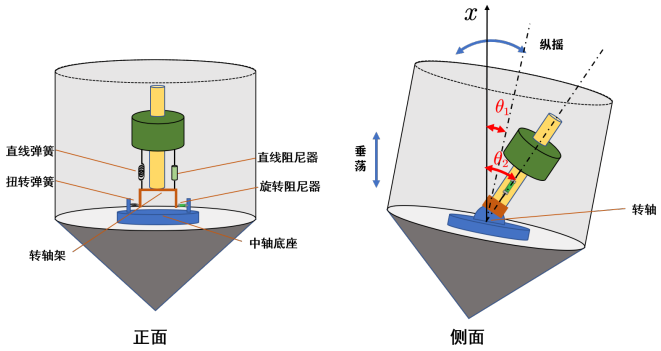


图 **15**　两自由度装置结构图

6.2.1 垂荡方向

根据牛顿第二定律，竖直方向上浮子和振子动力学方程为

$$\begin{cases} \text{浮子：} (m_1+m_a)\ddot{x}_1 = F_e - F_w - F_s - F_k\cos\theta - F_d\cos\theta, \\ \text{振子：} m_2\ddot{x} = -m_2 g + F_k\cos\theta + F_d\cos\theta. \end{cases} \tag{23}$$

6.2.2 纵摇方向

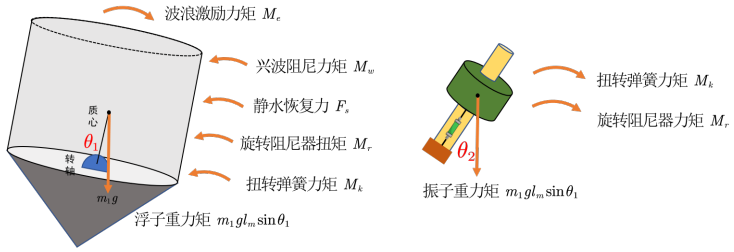


图 **16**　转动惯量分析

根据图16分析，根据角动量定理，浮子和振子的纵摇方向角动量方程为

$$\begin{cases} (I_1+I_a)\ddot{\theta}_1 = M_e - M_w - M_s - M_k - M_r + m_1 g\sin\theta_1 l_m, \\ I_2\ddot{\theta}_2 = M_k + M_r + m_2 g\tan\theta_2(x_2 - x_1). \end{cases} \tag{24}$$

式中 I_1, I_2, I_a 分别为浮子转动惯量、振子转动惯量和附加转动惯量.

6.2.3 几种作用力和力矩

• **波浪激励力 (矩)** 问题三给出波浪激励力和激励力矩为

$$\begin{aligned} F_e &= f\cos\omega t, \\ M_e &= L\cos\omega t. \end{aligned} \tag{25}$$

式中 f 为垂荡激励力振幅，L 为纵摇激励力矩振幅.

• **兴波阻尼力 (矩)**

兴波阻尼力 (矩) 与浮子振荡 (角) 速度成正比，表示为

$$\begin{aligned} F_w &= K_{wv}\dot{x}_1, \\ M_w &= K_{wh}\dot{\theta}_1. \end{aligned} \tag{26}$$

式中 K_{wv} 为垂荡兴波阻尼系数，K_{wh} 为纵摇兴波阻尼系数.

• **静水恢复力 (矩)**

图17表明，与问题一相比，加入纵摇自由度的浮子静水恢复力表达式不变.

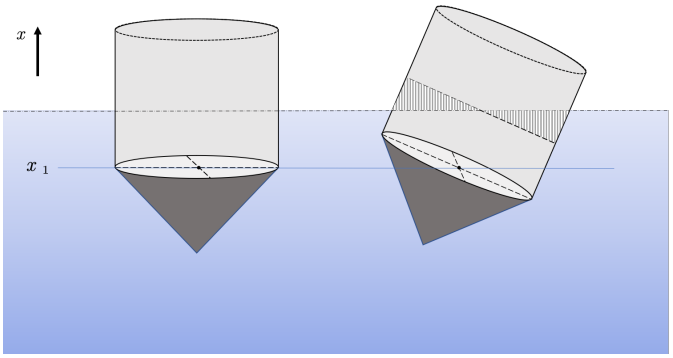


图 **17**　静水恢复力

$$F_s = \rho_s g\pi r_1^2 \cdot x_1 - m_1 g. \tag{27}$$

在海水中倾斜的浮体会受到海水扶正的力矩，其大小与浮体相对静水面的转角成正比. 静水恢复力矩表示为

$$M_s = K_{ms}\theta_1 \tag{28}$$

式中 K_{ms} 为静水恢复力矩系数.

• **弹簧弹力 (矩)**

根据几何关系，容易得到直线弹簧伸长量变为 $\Delta l = l - \dfrac{x_2 - x_1}{\cos\theta_2}$，则弹簧弹力表示为

$$F_k = k\left(l - \frac{x_2 - x_1}{\cos\theta_2}\right). \tag{29}$$

扭转弹簧的扭矩与浮子和振子的相对角位移成正比，则扭转弹簧力矩表示为

$$M_k = K_m(\theta_1 - \theta_2) \tag{30}$$

式中 K_m 为扭转弹簧的刚度.

• 阻尼器阻尼力 (矩)

沿着直线阻尼器方向振子和浮子的相对位移为 $\Delta = (x_2 - x_1)\tan\theta_2$，其关于时间一阶导数即为振子和浮子相对速度. 为了便于计算，取 $\Delta v \approx \dfrac{\dot{x}_1 - \dot{x}_2}{\cos\theta_2}$，则阻尼器阻尼力为

$$F_d = K_d \frac{\dot{x}_1 - \dot{x}_2}{\cos\theta_2} \tag{31}$$

式中 K_d 为直线阻尼器系数.

旋转阻尼器的扭矩与浮子和振子的相对角速度成正比，则旋转阻尼器扭矩为

$$M_r = K_r(\dot{\theta}_1 - \dot{\theta}_2) \tag{32}$$

式中 K_r 为旋转阻尼器系数.

6.2.4 模型总述

将上述力 (矩) 表达式带入动力学方程式 (23) 和角动量方程式 (24)，得到微分方程组

$$\begin{cases} (m_1 + m_a)\ddot{x}_1 + \cos\theta_2 \cdot k\Delta l + \cos\theta_2 \cdot K_d\Delta v + K_{wv}\dot{x}_1 + \rho_s g\pi r_1^2 \cdot x_1 - m_2 g = f\cos\omega t, \\ m_2\ddot{x} - \cos\theta_2 \cdot k\Delta l - \cos\theta_2 \cdot K_d\Delta v + m_2 g = 0, \\ (I_1 + I_a)\ddot{\theta}_1 + K_{wh}\dot{\theta}_1 + K_r(\dot{\theta}_1 - \dot{\theta}_2) + K_{ms}\theta_1 + K_m(\theta_1 - \theta_2), \\ I_2\ddot{\theta}_2 - K_r(\dot{\theta}_1 - \dot{\theta}_2) - K_m(\theta_1 - \theta_2) - mg\tan\theta_2 \cdot (x_2 - x_1) = 0. \end{cases} \tag{33}$$

这是一个四元非线性微分方程组，为了便于数值求解，记 $\boldsymbol{X} = \begin{bmatrix} x_1 & x_2 & \theta_1 & \theta_2 \end{bmatrix}^{\mathbf{T}}$，将上述方程写成矩阵形式为

$$\boldsymbol{A}\ddot{\boldsymbol{X}} + \boldsymbol{B}\dot{\boldsymbol{X}} + \boldsymbol{C}\boldsymbol{X} = \boldsymbol{b}. \tag{34}$$

限于篇幅系数矩阵在附录 A 中给出. 此为一个非线性方程组，系数矩阵中含有未知变量，可以通过数值方法迭代求解.

转动惯量计算

假设忽略中轴、底座、隔层及 PTO 的质量，均匀浮子外壳质量面密度为 $\rho_{suf} = 187kg/m^2$. 利用在线机械设计工具 (https://www.mechtool.cn/)，结合平行轴定理和垂直轴定理，浮子绕轴转动惯量等于圆柱上表面转动惯量、圆柱壳侧面转动惯量和圆锥壳转动惯量之和，计算得到浮子转动惯量 $I_1 = 18044kg \cdot m^2$.

振子转动惯量与绕轴转动半径有关，忽略转轴架高度，振子绕轴转动半径即为弹簧长度. 振子转动惯量 $I_2 = m_2\left(\dfrac{x_2 - x_1}{\cos\theta_2}\right)^2$.

初始条件: 以静水平衡位置为参考点，初始时刻浮子坐标为 0，振子坐标为 l_0，浮子和振子速度，角位移和角速度均为 0，则初始条件写为

$$\boldsymbol{X}(0) = [0, l_0, 0, 0]^T, \qquad \dot{\boldsymbol{X}}(0) = [0, 0, 0, 0]^T. \tag{35}$$

6.3 问题三求解结果

带入附件中相关参数，同样用 Matlab 中 ode45 求解器求解微分方程式 (34)，得到 40 个波浪周期内时间间隔为 0.2s 的数据存放在支撑文件 result3.xlsx 中.

图18画出前四十个周期内，在波浪激励力和激励力矩作用下，浮子垂荡坐标位置、振子坐标位置与相对位置的变化曲线. 振子初始坐标位置高于浮子位置 $l_0 = 0.202m$，因此在图 18上，振子坐标位置整体高于浮子坐标 0.202m.

刚开始时，系统垂荡振幅较小，几个海浪周期后，垂荡振幅大幅增大. 接着振幅急剧减小，如此循环几次后系统趋向平衡状态，振幅在 1.1m. 振子和浮子相对位移也随浮子和振子垂荡同周期变化，最大相对位移为 0.1m.

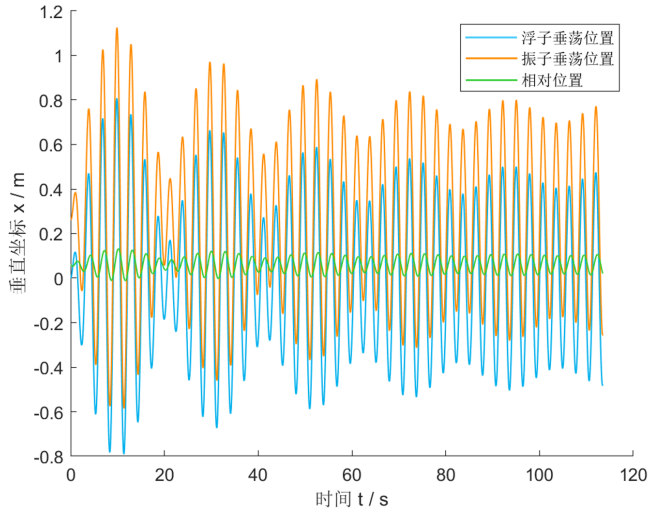
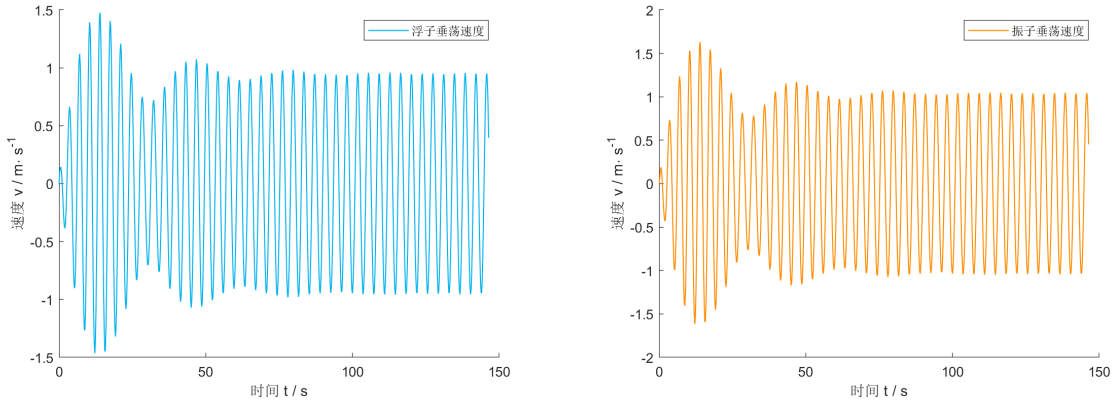


图 **18**　问题三垂荡位移

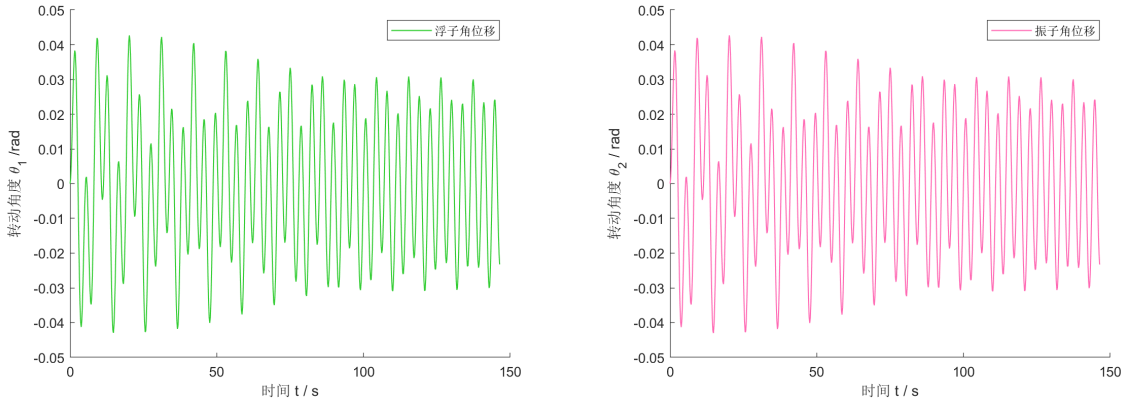
图19为浮子和振子垂荡速度图像，浮子和振子同步变化，角速度变化区间范围在 $-1m/s \sim 1m/s$ 之间.



(a) 浮子垂荡速度图像　　**(b)** 振子垂荡速度图像

图 **19**　浮子和振子垂荡速度曲线

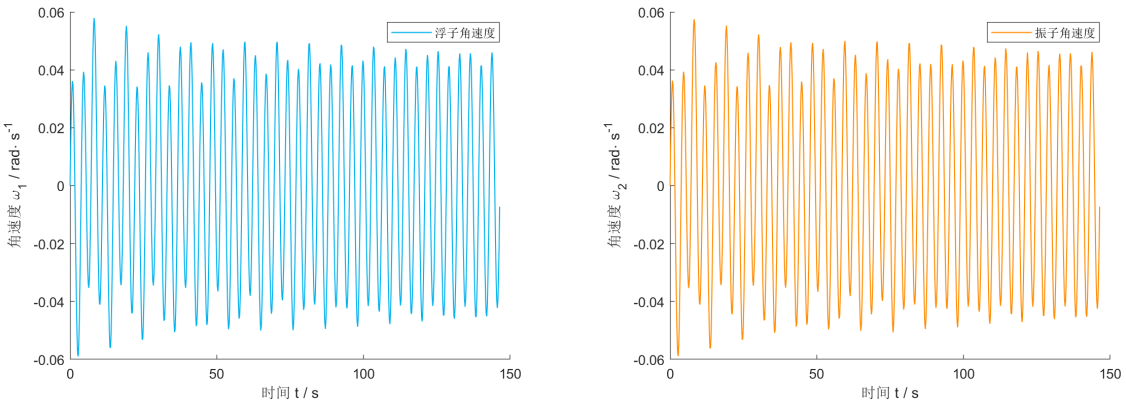
图20表示纵摇过程浮子角位移和振子角位移的变化过程，浮子角位移和振子角位移同步变化，趋于稳定后最大角位移为 0.03rad，约 1.72°.



(a) 浮子角位移图像　　**(b)** 振子角位移图像

图 **20**　浮子和振子角位移

图21为浮子和振子角速度变化图像，浮子和振子同步变化，角速度变化区间为 $-0.04rad/s \sim +0.04rad/s$.



(a) 浮子角速度图像　　**(b)** 振子角速度图像

图 **21**　浮子和振子角速度曲线

限于篇幅，表4仅给出 10 s、20 s、40 s、60 s、100 s 时，浮子与振子的垂荡位移与速度和纵摇角位移与角速度.

表 **4**　问题三结果

时间	浮子				振子			
	垂荡位移	垂荡速度	纵摇角位移	纵摇角速度	垂荡位移	垂荡速度	纵摇角位移	纵摇角速度
10s	-0.528	0.969	0.024	-0.040	-0.599	1.038	0.024	-0.041
20s	-0.705	-0.269	0.041	0.017	-0.773	-0.319	0.041	0.017
40s	0.369	0.757	-0.019	-0.012	0.393	0.845	-0.019	-0.012
60s	-0.321	-0.722	0.014	0.037	-0.342	-0.799	0.014	0.036
100s	-0.050	-0.947	0.001	0.041	-0.043	-1.036	0.001	0.042

7 问题四模型的建立与求解

7.1 问题分析

问题四考虑在问题三的基础上对线性直线阻尼器和旋转阻尼器的阻尼系数进行优化，使得直线阻尼器和旋转阻尼器共同输出功率最大. 同问题二，以最大平均输出功率为目标函数，两种阻尼系数为决策变量，给定系数取值区间为约束条件建立优化模型，依附件更改相应参数，采用鲸鱼算法求解.

7.2 最大输出功率优化模型

• 输出功率计算

两自由度波浪能装置的输出功率由两部分组成，即直线阻尼器输出和旋转阻尼器输出. 直线阻尼器输出瞬时功率可以表示为阻尼力与相对速度的乘积；旋转阻尼器的输出瞬时功率可以表示为扭转阻尼扭矩与相对角速度的乘积；一段时间后系统进入稳定振荡状态，取稳定工作状态下一段时间长度，等时间步长输出瞬时功率的算数平均值即为系统平均输出功率. 记数值计算过程某一时刻浮子速度 $\dot{x}_1^k$、振子速度 $\dot{x}_2^k$、浮子角速度 $\dot{\theta}_1^k$、振子角速度 $\dot{\theta}_2^k$，则输出功率表示为

$$P = K_d \sum_{k=n_1}^{n_2} \frac{(\dot{x}_1^k - \dot{x}_2^k)^2}{N} + K_r \sum_{k=n_1}^{n_2} \frac{(\dot{\theta}_1^k - \dot{\theta}_2^k)^2}{N}. \tag{36}$$

式中 K_d, K_r 分别表示直线阻尼器阻尼系数和旋转阻尼器的阻尼系数.N 为计算区间 $[n_1, n_2]$ 内数值解的个数. 因为考虑纵摇过程系统的不稳态时间较长，本文采取计算 $100s \sim 126.9s$.

7.2.1 模型总述

决策变量为两种阻尼系数，约束条件为提给取值区间，得到模型总述为

$$\max_{K_d,K_r} P = K_d \sum_{k=n_1}^{n_2} \frac{(\dot{x}_1^k - \dot{x}_2^k)^2}{N} + K_r \sum_{k=n_1}^{n_2} \frac{(\dot{\theta}_1^k - \dot{\theta}_2^k)^2}{N}$$

$$s.t. \quad \begin{cases} 0 \le K_d \le 100000, \\ 0 \le K_r \le 100000. \end{cases} \tag{37}$$

7.3 问题四结果呈现

同样采用混沌反馈自适应鲸鱼算法求解式 (37) 优化模型. 得到结果如表5.

表 **5**　问题四结果

最大功率	直线阻尼系数	旋转阻尼系数
331.61W	58254	81689

图22为计算迭代收敛曲线.

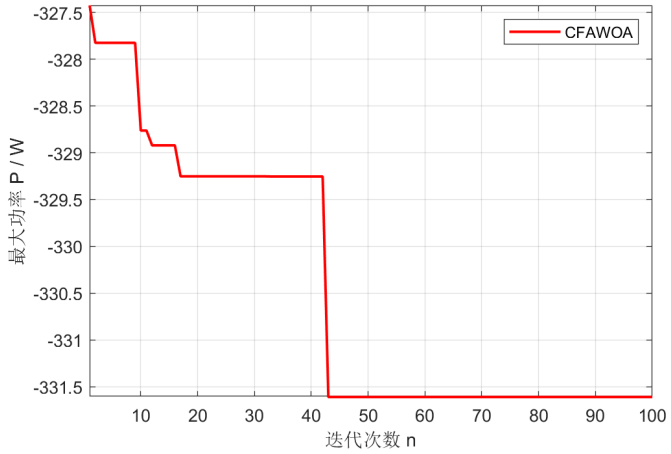


图 **22**　迭代收敛曲线

7.4 问题四结果分析

采用等步长搜索算法，对直线阻尼系数 K_d 和旋转阻尼系数 K_r 与输出功率的影响结果仿真. 得到结果如图23. 从图中看出，直线阻尼系数 K_d 对功率影响较大，再直线阻尼系数不变时，旋转阻尼系数对功率影响很小。因此，输出功率中扭矩项输出功率远小于直线阻尼器阻尼力输出功率，直线阻尼器在波浪能装置功率输出中占主要作用.

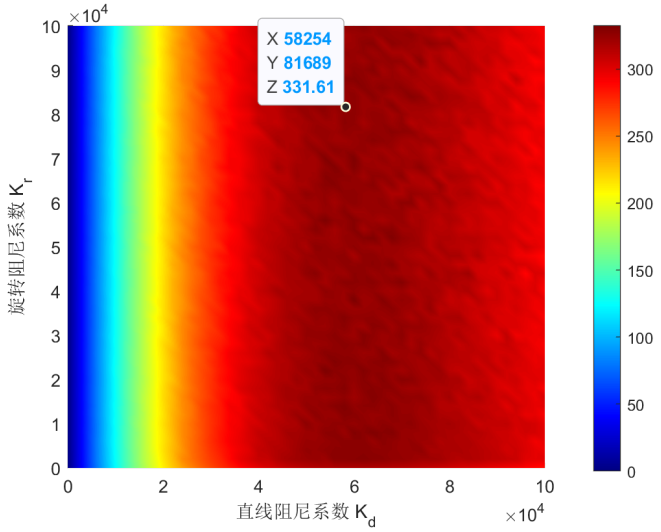


图 **23**　问题四仿真图

8 模型总结与评价

8.1 模型优点

(1) 将非线性的微分方程组转换成线性形式，方便计算;

(2) 采用四阶龙格库塔方法迭代求解，计算精度高;

(3) 混沌反馈自适应鲸鱼算法能很好地避免陷入局部最优，并且收敛速度明显优于等步长搜索.

8.2 模型缺点

(1) 问题三振子和浮子相对速度采用了近似值，结果可能存在误差;

8.3 模型改进

本文因简化模型，我们对系统的运动仅研究了垂荡运动以及横摇运动，实际生活中，波浪能装置实际上是三个方向运动的. 并且本文为了检查模型，忽略了环境因素. 如果将因素考虑齐全，就可以得到一个三维摆动模型，能加贴近实际模型.

参考文献

[1] 陈佳, 兰飞, 郭昊霖, 黎静华. 波浪能发电控制技术研究综述 [J/OL]. 电力自动化设备:1-20[2022-09-18].

[2] 陈佳, 兰飞, 郭昊霖, 黎静华. 波浪能发电控制技术研究综述 [J/OL]. 电力自动化设备:1-20[2022-09-16].

[3] 李彧野. 基于阻抗模型的柔性直流输电系统稳定性分析与控制方法研究 [D]. 武汉大学,2020.

[4] Khalil H K. Lyapunov stability[J]. Control Systems, Robotics and AutomatioN, 2009, 12: 115.

[5] 涂春梅, 陈国彬, 刘超. 混沌反馈自适应鲸鱼优化算法研究 [J]. 统计与决策,2019,35(07):17-20.

[6] 谭铭, 杨宇轩, 岑雨昊, 司玉林, 钱鹏, 张大海. 基于遗传算法的多自由度波浪能装置浮体形状优化 [J]. 中国舰船研究,2022,17(03):228-236.

[7] 高琅, 朱良生, 王磊, 周斌珍. 垂荡式波浪能装置与风机平台集成系统水动力性能研究 [J]. 广东造船,2022,41(03):17-21.

附录 A　系数矩阵

问题一系数矩阵为

$$\boldsymbol{A}=\begin{bmatrix} m_1+m_2 & 0 \\ 0 & m_2 \end{bmatrix}.\boldsymbol{B}=\begin{bmatrix} K_{wv}+K_d & -K_d \\ -K_d & K_d \end{bmatrix},$$

$$\boldsymbol{C}=\begin{bmatrix} k+\rho_s g\pi r_1^2 & -k \\ -k & k \end{bmatrix}.\boldsymbol{b}=\begin{bmatrix} f\cos\omega t-kl+m_2 g \\ -m_2 g+kl \end{bmatrix}.$$

其中阻尼系数 K_d 依情况一和情况二不同：

$$K_d=\begin{cases} \text{情况一：}K_d \\ \text{情况二：}\alpha|\dot{x}_1-\dot{x}_2|^{\beta} \end{cases}$$

问题三整理得系数为

$$\boldsymbol{A}=\begin{bmatrix} m_1+m_a & 0 & 0 & 0 \\ 0 & m_2 & 0 & 0 \\ 0 & 0 & I_1+I_a & 0 \\ 0 & 0 & 0 & I_2 \end{bmatrix}.\boldsymbol{B}=\begin{bmatrix} K_{wv}+K_d & -K_d & 0 & 0 \\ -K_d & K_d & 0 & 0 \\ 0 & 0 & K_{wh}+K_r & -K_r \\ 0 & 0 & -K_r & K_r \end{bmatrix},$$

$$\boldsymbol{C}=\begin{bmatrix} \rho_s g\pi r_1^2+k & -k & 0 & 0 \\ -k & k & 0 & 0 \\ 0 & 0 & K_{ms}+K_m & -K_m \\ m_2 g\tan\theta_2 & -m_2 g\tan\theta_2 & -K_m & K_m \end{bmatrix}.\boldsymbol{b}=\begin{bmatrix} f\cos\omega t+m_2 g-kl\cos\theta_2 \\ kl\cos\theta_2-m_2 g \\ L\cos\omega t+m_1 g\sin\theta_1 l_m \\ 0 \end{bmatrix}$$

附录 B　问题一源代码

2.1　问题一主程序程序

```
global l g f w r1 kwv kd k m1 m2 ma p beta %定义各种变量
l=0.5;
g=9.8;
f=6250;
w=1.4005;
r1=1;
kwv=656.3616; kd=10000; k=80000;
m1=4866; m2=2433; ma=1335.535;
p=1025;
beta=0.5;
%%
%隔离法（1.1）
X0 = [0;l-(m2*g)/k;0;0];          %初始值，包括位置和速度
```

```
jz.A = [m1+ma 0;0 m2];          %距离矩阵
jz.B = [kwv+kd -kd;-kd kd];     %速度矩阵
jz.C = [p*g*pi*r1^2+k -k;-k k]; %加速度矩阵
[t,x] = ode45(@Ode_45,[0:0.01:80*pi/w],X0,[],jz); %求常微分方程组的数值解

figure(2); %画图
plot(t,x(:,1),'Color',[0 0.69804 0.93333],'lineWidth',0.8);
hold on
plot(t,x(:,2),'Color',[1 0.54902 0],'lineWidth',0.8);
xlabel('时间 t / s');
ylabel('垂直坐标 x / m');
legend('浮子坐标','振子坐标');

figure(3);
plot(t,x(:,3),'Color',[0 0.69804 0.93333],'lineWidth',1.2);
hold on
plot(t,x(:,4),'Color',[1 0.54902 0],'lineWidth',0.8);
xlabel('时间 t / s');
legend('浮子速度','振子速度');


%%
%隔离法（1.2）

X0 = [0;l-(m2*g)/k;0;0];          %初始值，包括位置和速度
jz.A = [m1+ma 0;0 m2];          %距离矩阵
jz.B=[kwv+kd*(abs(X0(1)-X0(2)))^beta -kd*(abs(X0(1)-X0(2)))^beta;-kd*(abs(X0(1)-X0(2)))^beta
    kd*(abs(X0(1)-X0(2)))^beta];
jz.C = [p*g*pi*r1^2+k -k;-k k]; %加速度矩阵
[t2,x2] = ode45(@Order3,[0:0.01:80*pi/w],X0,[],jz); %求常微分方程组的数值解

figure(4);
plot(t2,x2(:,1),'Color',[0 0.69804 0.93333],'lineWidth',0.8);
hold on
plot(t2,x2(:,2),'Color',[1 0.54902 0],'lineWidth',0.8);
xlabel('时间 t / s');
ylabel('垂直坐标 x / m');
legend('浮子坐标','振子坐标');

figure(5);
plot(t2,x2(:,3),'Color',[0 0.69804 0.93333],'lineWidth',1.2);
hold on
plot(t2,x2(:,4),'Color',[1 0.54902 0],'lineWidth',0.8);
xlabel('时间 t / s');
legend('浮子速度','振子速度');

%%
```

```
%微分方程组的表达式1.1
function ans1=Ode_45(t,X0,jz) %迭代的函数
global g f w k m2 l
i = length(X0);
X = X0(1:i/2);       %前一半是0阶导项
DX = X0(i/2+1:i); %后一半是1阶导项
b = [f*cos(w*t)-k*l+m2*g;-m2*g+k*l];
DDX = inv(jz.A)*(-jz.B*DX-jz.C*X+b); %移项求解二阶导
ans1 = [DX; DDX];
end

%微分方程组的表达式1.2
function ans2=Order3(t,X0,jz) %迭代的函数
global g f w kwv kd k m2 l beta
i = length(X0);
X = X0(1:i/2);       %前一半是0阶导项
DX = X0(i/2+1:i); %后一半是1阶导项
b = [f*cos(w*t)-k*l+m2*g;-m2*g+k*l];
jz.B=[kwv+kd*(abs((DX(1)-DX(2))))^beta
    -kd*(abs((DX(1)-DX(2))))^beta;-kd*(abs((DX(1)-DX(2))))^beta kd*(abs((DX(1)-DX(2))))^beta];
DDX = inv(jz.A)*(-jz.B*DX-jz.C*X+b); %移项求解二阶导
ans2 = [DX; DDX];
end
```

2.2 伯德图程序

```
clc
syms x1 x2 s t
m1=4866;
m2=2433;
ma=1335.535;
l=0.5;
g=9.8;
f=6250;
w=1.4005;
r1=1;
kw=656.3616;
ka=10000;
k=80000;
p=1025;
[x1,x2]=solve((m1+ma)*s^2*x1==f*s/(s^2+w^2)-(kw+ka)*s*x1+ka*(s*x2-0.202)- ...
(p*g*pi*r1^2+k)*x1+k*x2-(k*l-m2*g)/s,m2*(s^2*x2-0.202*s)...
==-m2*g/s+k*(x1-x2)+ka*(s*x1-s*x2+0.202)+k*l/s,x1,x2);
R=f*s/(s^2+w^2);
q=x1/R;
```

```
qq=x2/R;
aa=8833361435024399/4503599627370496;
bb=29429165953087953637335669166571 52;
cc=120892581961462889323119928934400 00;
dd=96717203959002340529253807279833088;
ee=-5520850896890249375;
ff=-44166807175121995000;
A=4503599627370496;
B=8833361435024399;
C=4147449849237020247;
D=24173389246075117695;
E=212784450812223451228;
F=101177585019344360000;
G=693952326358649280000;
%num00=32*[bb cc dd ee ff];
%den00=5*[A*C A*D A*E+B*C A*F+B*D A*G+B*E B*F B*G 0];
%sys00=tf(num00,den00);
%step(sys00);
xlim([0 300]);
num0=16*[bb cc dd+aa*bb ee+aa*cc ff+aa*dd aa*ee aa*ff];
den0=15625*[A*C A*D A*E+B*C A*F+B*D A*G+B*E B*F B*G 0 0];
sys=tf(num0,den0);
bode(sys);
grid on;
y1=8833361435024399/4503599627370496;
y2=64820464644895094454372092589995326308159062016;
y3=37780573121595613084015004836694586874472488960;
y4=34522971339425901500800147562234818701747426 42688;
y5=36515125575513730883567344458467114301451036 26240;
y6=27999284345436203931689158063807998333795749190671;
y7=31014765141144022279954733528423530617787141 54375;
y8=21268432116824247047674974925242921005426084007500;
z1=107374182400000000;
z2=4503599627370496;
z3=8833361435024399;
z4=4147449849237020247;
z5=24173389246075117695;
z6=212784450812223451228;
z7=101177585019344360000;
z8=693952326358649280000;
%num11=[y2 y3 y4 y5 y6 y7 y8];
%den11=z1/f*[z2*z4 z2*z5 z2*z6+z3*z4 z2*z7+z3*z5 z2*z8+z3*z6 z3*z7 z3*z8 0];
%sys11=tf(num11,den11);
%step(sys11);
num1=[y2 y3 y1*y2+y4 y5+y1*y3 y6+y1*y4 y7+y1*y5 y8+y1*y6 y1*y7 y1*y8];
den1=z1*[z2*z4 z2*z5 z2*z6+z3*z4 z2*z7+z3*z5 z2*z8+z3*z6 z3*z7 z3*z8 0 0];
sys1=tf(num1,den1);
```

```
bode(sys1);
grid on;
```

附录 C　问题二源代码

3.1　主程序

```
clear all
clc
x_num=10; % 种群数量
fun_name='F1'; % 适应度函数名称
max_num=100; % 最大迭代次数

[lb,ub,narvs,fobj]=Get_Functions_details(fun_name); %获取边界以及优化函数
%原始鲸鱼算法
[Best_score,Best_pos,WOA_cg_curve]=WOA(x_num,max_num,lb,ub,narvs,fobj);
%改进的混沌反馈自适应鲸鱼算法
[Best_score1,Best_pos1,WOA_cg_curve1]=CFAWOA(x_num,max_num,lb,ub,narvs,fobj);

figure(1)
plot(WOA_cg_curve,'Color','r','linewidth',1.5)
hold on
plot(WOA_cg_curve1,'Color','r','linewidth',1.5);
xlabel('迭代次数 n');
ylabel('最大功率 P / W');
legend('CFAWOA');
axis tight
grid on
box on


%WOA算法
function [lead_score,lead_pos,Con_curve]=WOA(x_num,max_num,lb,ub,narvs,fobj)

lead_pos=zeros(1,narvs);
lead_score=inf;

Pos=initialization(x_num,narvs,ub,lb); %常规随机初始化种群
Con_curve=zeros(1,max_num);

cnt=0;

while cnt<max_num %循环寻找最优解
disp(cnt)
for i=1:size(Pos,1)
```

```
Flag4ub=Pos(i,:)>ub;
Flag4lb=Pos(i,:)<lb;
Pos(i,:)=(Pos(i,:).*(~(Flag4ub+Flag4lb)))+ub.*Flag4ub+lb.*Flag4lb;
% 计算对应的适应度
fit=fobj(Pos(i,:));
% 更新新的最优解
if fit<lead_score
lead_score=fit; % 更新最优解适应度
lead_pos=Pos(i,:);
end
end
a=2-cnt*((2)/max_num);
a2=-1+cnt*((-1)/max_num);
% 更新位置
for i=1:size(Pos,1)
r1=rand();
r2=rand();
A=2*a*r1-a;
C=2*r2;
b1=1;
len=(a2-1)*rand+1;
p1 = rand();
for j=1:size(Pos,2)
%鲸鱼算法座头鲸泡泡网觅食行为的具体算法
if p1<0.5
if abs(A)>=1
rand_leader_index = floor(x_num*rand()+1);
X_rand = Pos(rand_leader_index, :);
D1_X_rand=abs(C*X_rand(j)-Pos(i,j));
Pos(i,j)=X_rand(j)-A*D1_X_rand;

elseif abs(A)<1
D1_Leader=abs(C*lead_pos(j)-Pos(i,j));
Pos(i,j)=lead_pos(j)-A*D1_Leader;
end
elseif p1>=0.5
dToLeader=abs(lead_pos(j)-Pos(i,j));
Pos(i,j)=dToLeader*exp(b1.*len).*cos(len.*2*pi)+lead_pos(j);
end
end
end
cnt=cnt+1;
Con_curve(cnt)=lead_score;
end

%CFAWOA
```

```
function [lead_score,lead_pos,Con_curve]=CFAWOA(x_num,max_num,lb,ub,narvs,fobj)
lead_pos=zeros(1,narvs);
lead_score=inf;
% 改进点1: sine混沌映射
Pos=initializationNew(x_num,narvs,ub,lb); %改进的初始化种群
Con_curve=zeros(1,max_num);
cnt=0;
while cnt<max_num %循环寻找最优解
disp(cnt);
for i=1:size(Pos,1)
FlagForub=Pos(i,:)>ub;
FlagForlb=Pos(i,:)<lb;
Pos(i,:)=(Pos(i,:).*(~(FlagForub+FlagForlb)))+ub.*FlagForub+lb.*FlagForlb;
% 计算对应的适应度
fit(i)=fobj(Pos(i,:));
% 更新新的最优解
if fit(i)<lead_score
lead_score=fit(i);
lead_pos=Pos(i,:);
end
end
u = max(fit);
a=2-cnt*((2)/max_num);
a2=-1+cnt*((-1)/max_num);
for i=1:size(Pos,1)
%%改进点2: 自适应权重
w = 0.2 + (0.4 + exp(-fit(i)/u)^cnt);
r1=rand();
r2=rand();
A=2*a*r1-a;
C=2*r2;
b=1;
l=(a2-1)*rand+1;
p = rand();
for j=1:size(Pos,2)
%鲸鱼算法座头鲸泡泡网觅食行为算法
if p<0.5
if abs(A)>=1
rand_leader_index = floor(x_num*rand()+1);
X_rand = Pos(rand_leader_index, :);
D1_X_rand=abs(C*X_rand(j)-Pos(i,j));
Pos(i,j)=X_rand(j)-A*D1_X_rand;
elseif abs(A)<1
D1_Leader=abs(C*lead_pos(j)-Pos(i,j));
Pos(i,j)=w*lead_pos(j)-A*D1_Leader;
end
```

```
elseif p>=0.5

dToLeader=abs(lead_pos(j)-Pos(i,j));
% Eq. (2.5)
Pos(i,j)=dToLeader*exp(b.*l).*cos(l.*2*pi)+w*lead_pos(j);

end

end
FlagForub=Pos(i,:)>ub;
FlagForlb=Pos(i,:)<lb;
Pos(i,:)=(Pos(i,:).*(~(FlagForub+FlagForlb)))+ub.*FlagForub+lb.*FlagForlb;
fit(i) = fobj(Pos(i,:));
% 最差鲸鱼位置
[~,ind1] = max(fit(i));
Pos_Worst = Pos(ind1,:);
% 改进点3: 最差位置反馈交流
if p<0.5 && abs(A)<1
temp = Pos_Worst + rand(1,narvs).*(lead_pos - Pos_Worst);
FlagForub=temp>ub;
FlagForlb=temp<lb;
temp=(temp.*(~(FlagForub+FlagForlb)))+ub.*FlagForub+lb.*FlagForlb;
if fobj(temp)<fit(ind1)
Pos(i,:) = temp;
fit(ind1) = fobj(temp);
end
end
end
cnt=cnt+1;
%记录每次迭代最优值
Con_curve(cnt)=lead_score;
end

%适应度函数
function [lb,ub,narvs,fobj] = Get_Functions_details(F)
switch F
case 'F1'
fobj = @fun1;
lb=0;              %下界
ub=100000;  %上界
narvs=1;       %维度=1

case 'F2'
fobj = @fun2;
lb=[0 0.3];         %下界
ub=[100000 1]; %上界
narvs=2;           %维度=2
```

```
end
end

function ans1 = fun1(kdx)
global l g f w r1 kwv k m1 m2 ma p
l=0.5;
g=9.8;
f=4890;
w=2.2143;
r1=1;
kwv=167.8395;
kd=kdx;
k=80000;
m1=4866;
m2=2433;
ma=1165.992;
p=1025;

%隔离法（1）
X0 = [0;l-(m2*g)/k;0;0];          %初始值，包括位置和速度
jz.A = [m1+ma 0;0 m2];          %加速度矩阵
jz.B = [kwv+kd -kd;-kd kd];    %速度矩阵
jz.C = [p*g*pi*r1^2+k -k;-k k]; %距离矩阵
[t,x] = ode45(@funq1,[0:0.01:2*40*pi/w],X0,[],jz); %微分方程组的数值解
length1=size(x,1);
ans1=0;
for i=4261:length1
ans1=ans1+(x(i,3)-x(i,4))^2;
end
ans1=ans1*kdx/(length1-4261+1);
ans1=-ans1;
end

function ans1=funq1(t,X0,jz)
global g f w k m2 l
i = length(X0);
X = X0(1:i/2);       %前一半是0阶导项
DX = X0(i/2+1:i); %后一半是1阶导项
b = [f*cos(w*t)-k*l+m2*g;-m2*g+k*l];
DDX = inv(jz.A)*(-jz.B*DX-jz.C*X+b); %移项求解二阶导
ans1 = [DX; DDX];
end

function ans2 = fun2(x)
global l g f w r1 kwv k m1 m2 ma p kd beta
l=0.5;
```

```
g=9.8;
f=4890;
w=2.2143;
r1=1;
kwv=167.8395;
kd=x(1);
beta=x(2);
k=80000;
m1=4866;
m2=2433;
ma=1165.992;
p=1025;
%隔离法（2）
x0 = [0;l-(m2*g)/k;0;0];          %初始值位置和速度
jz.A = [m1+ma 0;0 m2];          %加速度矩阵
jz.B=[kwv+kd*(abs(x0(1)-x0(2)))^beta -kd*(abs(x0(1)-x0(2)))^beta;-kd*(abs(x0(1)-x0(2)))^beta
    kd*(abs(x0(1)-x0(2)))^beta];
jz.C = [p*g*pi*r1^2+k -k;-k k]; %距离矩阵
[t2,x2] = ode45(@Order3,[0:0.01:80*pi/w],x0,[],jz);
ans2=0;
length1=size(x2,1);
for i=2001:length1
ans2=ans2+kd*(abs(x2(i,3)-x2(i,4)))^beta*(x2(i,3)-x2(i,4))^2;
end
ans2=-ans2/(length1-2001+1);
end


function ans2=Order3(t,x0,jz)
global g f w kwv kd k m2 l beta
i = length(X0);
X = X0(1:i/2);        %前一半是0阶导项
DX = X0(i/2+1:i); %后一半是1阶导项
b = [f*cos(w*t)-k*l+m2*g;-m2*g+k*l];
jz.B=[kwv+kd*(abs((DX(1)-DX(2))))^beta
    -kd*(abs((DX(1)-DX(2))))^beta;-kd*(abs((DX(1)-DX(2))))^beta kd*(abs((DX(1)-DX(2))))^beta];
%disp(param.B);
DDX = inv(jz.A)*(-jz.B*DX-jz.C*X+b);   %移项求解二阶导
ans2 = [DX; DDX];
end

% Sine映射的种群初始化
function Pos=initializationNew(x_num,narvs,ub,lb)

Bd_no= size(ub,2);

if Bd_no==1
```

```
for i = 1:x_num
temp = SineMap(narvs).*(ub-lb)+lb;
temp(temp>ub) = ub;
temp(temp<lb) = lb;
Pos(i,:)=temp;
end
end

if Bd_no>1
for j = 1:x_num
sine_Val = SineMap(narvs);
for i=1:narvs
ub_x=ub(i);
lb_x=lb(i);
Pos(j,i)=sine_Val(i).*(ub_x-lb_x)+lb_x;
if(Pos(j,i)>ub_x)
Pos(j,i) = ub_x;
end
if(Pos(j,i)<lb_x)
Pos(j,i) = lb_x;
end
end
end
end
end

%Sine映射
function [x] = SineMap(max_num)
x(1)=rand; %随机生成第一个初始点
for i=1:max_num-1
x(i+1)=sin(pi*x(i)); %通过Sine映射迭代后面的初始值
end
end
```

3.2 仿真程序

```
  plot3(resx,resy,res);
  figure(3);

  z=zeros(21,21);
  for i=1:21
  z(:,i)=res((i-1)*21+1:i*21,1);
  end
  surf(0:5000:100000,0:0.05:1,z);
  shading interp;
```

```
hold on
plot3(97546,0.411,234.169,'.','Color','y','MarkerSize',10);

function DStateVarDt=Order3(t,X0,jz)
global g f w kwv kd k m2 l beta
i = length(X0);
X = X0(1:i/2);       %前一半是0阶导项
DX = X0(i/2+1:i); %后一半是1阶导项
b = [f*cos(w*t)-k*l+m2*g;-m2*g+k*l];
%disp(XDot);
%param.B=[kwv+kd*(abs(XDot(1)-XDot(2)))^beta
    -(abs(kd^2*(XDot(1)-XDot(2))))^beta;-(abs(kd^2*(XDot(1)-XDot(2))))^beta
    (abs(kd^2*(XDot(1)-XDot(2))))^beta];
jz.B=[kwv+kd*(abs((DX(1)-DX(2))))^beta
    -kd*(abs((DX(1)-DX(2))))^beta;-kd*(abs((DX(1)-DX(2))))^beta
    kd*(abs((DX(1)-DX(2))))^beta];
%disp(param.B);
DDX = inv(jz.A)*(-jz.B*DX-jz.C*X+b);%降阶后的表达式
DStateVarDt = [DX; DDX];
end
```

附录 D　问题三源代码

```
% h1=3; %圆柱高度
% h3=0.8; %圆锥高度
% r1=1; %半径
% ll=sqrt(1+0.8^2); %圆锥母线
% s=pi*r1^2+2*pi*r1*h1+pi*r1*ll; %表面积
% p=4866/s; %面密度
% I1=0.25*p*pi*r1^4;
% I2=I1+p*pi*h1^2; %圆柱顶转动惯量
% I3=1/3*pi*r1*h1*p*(3*r1^2+2*h1^2); %圆柱侧边壳转动惯量
% I4=1/12*pi*r1*ll*p*(3+2*h3^2); %圆锥转动惯量
% I=I2+I3+I4
clear;clc
global l g f w r1 kwv kd k m1 m2 ma p L kwh kms km kr I1 Ia
l=0.5;
g=9.8;
f=3640;
w=1.7152;
r1=1;
kwv=683.4558;
kd=10000;
k=80000;
```

```
m1=4866;
m2=2433;
ma=1028.876;
p=1025;
L=1690;
kwh=654.3383;
kms=8890.7;
km=250000;
kr=1000;
I1=1.8044e+04;
Ia=7001.914;

X0=[0;l-m2*g/k;0;0;0;0;0;0];
jz.A=[m1+ma 0 0 0;0 m2 0 0;0 0 I1+Ia 0;0 0 0 m2*((X0(2)-X0(1))/cos(X0(4)))^2];

jz.B=[kwv+kd -kd 0      0;
-kd     kd  0        0;
0       0   kwh+kr   -kr;
0       0   -kr      kr];

jz.C=[p*g*pi*r1^2+k -k 0 0;
-k k 0 0;
0 0 kms+km -km;
m2*g*tan(X0(4)) -m2*g*tan(X0(4)) -km km];

[t,x] = ode45(@funq3,[0:0.2:2*40*pi/w],X0,[],jz);

% 0:1:2*40*pi/w
% figure(2);
% %plot(t,x(:,1),'Color',[0 0.69804 0.93333],'lineWidth',0.8);
% for i=1:size(x,1)-1
%    line([t(i) t(i+1)],[x(i,1) x(i+1,1)],'Color',[0 0.69804 0.93333],'lineWidth',0.8);
%    hold on
% end
% hold on
% %plot(t,x(:,2),'Color',[1 0.54902 0],'lineWidth',0.8);
% for i=1:size(x,1)-1
%    line([t(i) t(i+1)],[x(i,2) x(i+1,2)],'Color',[1 0.54902 0],'lineWidth',0.8);
%    hold on
% end
% hold on
% for i=1:size(x,1)-1
%    line([t(i) t(i+1)],[x(i,2)-x(i,1)-0.202 x(i+1,2)-x(i+1,1)-0.202],'Color',[0.19608 0.80392
    0.19608],'lineWidth',0.8);
%    hold on
% end
%
```

```
% xlabel('时间 t / s');
% ylabel('垂直坐标 x / m');
% legend('浮子垂荡位置','振子垂荡位置','相对位置');
%
% figure(3);
% %plot(t,x(:,3),'Color',[0 0.69804 0.93333],'lineWidth',1.2);
% for i=1:size(x,1)-1
%     line([t(i) t(i+1)],[x(i,3) x(i+1,3)],'Color',[0.19608 0.80392 0.19608],'lineWidth',0.8);
%    hold on
% end
% xlabel('时间 t / s');
% ylabel('转动角度 ');
% legend('浮子角位移');
% hold on
% %plot(t,x(:,4),'Color',[1 0.54902 0],'lineWidth',0.8);
% %figure(3);
% for i=1:size(x,1)-1
%    line([t(i) t(i+1)],[x(i,4) x(i+1,4)],'Color',[1 0.41176 0.70588],'lineWidth',0.8);
%    hold on
% end
% xlabel('时间 t / s');
% ylabel('转动角度 ');
% legend('振子角位移');
%
% figure(4);
% %plot(t,x(:,3),'Color',[0 0.69804 0.93333],'lineWidth',1.2);
% for i=1:size(x,1)-1
%    line([t(i) t(i+1)],[x(i,5) x(i+1,5)],'Color',[0 0.69804 0.93333],'lineWidth',0.8);
%    hold on
% end
% xlabel('时间 t / s');
% ylabel('速度 v / ');
% legend('浮子垂荡速度');
% hold on
%plot(t,x(:,4),'Color',[1 0.54902 0],'lineWidth',0.8);
% figure(4);
% for i=1:size(x,1)-1
%    line([t(i) t(i+1)],[x(i,6) x(i+1,6)],'Color',[1 0.54902 0],'lineWidth',0.8);
%    hold on
% end
% xlabel('时间 t / s');
% ylabel('速度 v / ');
% legend('振子垂荡速度');
%
% figure(5);
% %plot(t,x(:,3),'Color',[0 0.69804 0.93333],'lineWidth',1.2);
% for i=1:size(x,1)-1
```

```
%    line([t(i) t(i+1)],[x(i,7) x(i+1,7)],'Color',[0 0.69804 0.93333],'lineWidth',0.8);
%    hold on
% end
% xlabel('时间 t / s');
% ylabel('角速度 ');
% legend('浮子角速度');
% hold on
% %plot(t,x(:,4),'Color',[1 0.54902 0],'lineWidth',0.8);
% figure(5);
% for i=1:size(x,1)-1
%    line([t(i) t(i+1)],[x(i,8) x(i+1,8)],'Color',[1 0.54902 0],'lineWidth',0.8);
%    hold on
% end
% xlabel('时间 t / s');
% ylabel('角速度 ');
% legend('振子角速度');

function ans1=funq3(t,x0,jz)
global l g f w kwv kd k m1 m2 ma L kwh kr I1 Ia
%disp(t);
i = length(x0);
X = x0(1:i/2);       %前一半是0阶导项
DX = x0(i/2+1:i); %后一半是1阶导项
b = [f*cos(w*t)-k*l*cos(X(4))+m2*g;-m2*g+k*l*cos(X(4));L*cos(w*t);0];
jz.A=[m1+ma 0 0 0;0 m2 0 0;0 0 I1+Ia 0;0 0 0 m2*((X(2)-X(1))/cos(X(4)))^2];
DDX = inv(jz.A)*(-jz.B*DX-jz.C*X+b);%移项求解二阶导
ans1 = [DX; DDX];
end
```

附录 E　问题四源代码

5.1　主程序

```
clear all
clc
x_num=10; % 种群数量
fun_name='F3'; % 适应度函数名称
max_num=100; % 最大迭代次数

[lb,ub,dim,fobj]=Get_Functions_details(fun_name); %获取边界以及优化函数
%原始鲸鱼算法
[Best_score,Best_pos,WOA_cg_curve]=WOA(x_num,max_num,lb,ub,dim,fobj);
%改进鲸鱼算法
[Best_score1,Best_pos1,WOA_cg_curve1]=CFAWOA(SearchAgents_no,Max_iteration,lb,ub,dim,fobj);
```

```
figure(1)
plot(WOA_cg_curve,'Color','r','linewidth',1.5)
hold on
plot(WOA_cg_curve1,'Color','r','linewidth',1.5);
xlabel('迭代次数 n');
ylabel('最大功率 P / W');
legend('CFAWOA');

axis tight
grid on
box on

function [lb,ub,narvs,fobj] = Get_Functions_details(F)
switch F

case 'F3'
fobj = @fun3;
lb=[0 0];
ub=[100000 100000];
narvs=2;

end
end

function ans3 = fun3(x)
global l g f w r1 kwv kd k m1 m2 ma p L kwh kms km kr I1 Ia
l=0.5;
g=9.8;
f=1760;
w=1.9806;
r1=1;
kwv=528.5018;
kd=x(1);
k=80000;
m1=4866;
m2=2433;
ma=1091.099;
p=1025;
L=2140;
kwh=1655.909;
kms=8890.7;
km=250000;
kr=x(2);
I1=1.8044e+04;
Ia=7142.493;

X0=[0;l-m2*g/k;0;0;0;0;0;0];
```

```
jz.A=[m1+ma 0 0 0;0 m2 0 0;0 0 I1+Ia 0;0 0 0 m2*((X0(2)-X0(1))/cos(X0(4)))^2];
jz.B=[kwv+kd -kd 0      0;
-kd   kd  0        0;
0     0   kwh+kr   -kr;
0     0   -kr      kr];

jz.C=[p*g*pi*r1^2+k -k 0 0;
-k k 0 0;
0 0 kms+km -km;
m2*g*tan(X0(4)) -m2*g*tan(X0(4)) -km km];

[t,x3] = ode15s(@funq3,[0:0.01:2*40*pi/w],X0,[],jz);
ans3=0;
length1=size(x3,1);
for i=10000:length1
ans3=ans3+(kd*((x3(i,5)-x3(i,6))/cos(x3(4)))^2+kr*(x3(i,7)-x3(i,8))^2)/(length1-10000+1);
end
ans3=-ans3;
end

function ans1=funq3(t,X0,jz)
global l g f w kwv kd k m1 m2 ma L kwh kr I1 Ia
i = length(X0);
X = X0(1:i/2);    %前一半是0阶导项
DX = X0(i/2+1:i); %后一半是1阶导项
b = [f*cos(w*t)-k*l*cos(X(4))+m2*g;-m2*g+k*l*cos(X(4));L*cos(w*t);0];
jz.A=[m1+ma 0 0 0;0 m2 0 0;0 0 I1+Ia 0;0 0 0 m2*((X(2)-X(1))/cos(X(4)))^2];
DDX = inv(jz.A)*(-jz.B*DX-jz.C*X+b); %移项求解二阶导
ans1 = [DX; DDX];
end
```

5.2 仿真程序

```
  res=zeros(51*51,1);
  resx=zeros(51*51,1);
  resy=zeros(51*51,1);
  num=0;
  for i=0:2000:100000
  for z=0:2000:100000
  num=num+1;
  resx(num,1)=i;
  resy(num,1)=z;
  global l g f w r1 kwv kd k m1 m2 ma p L kwh kms km kr I1 Ia
  l=0.5;
```

```
g=9.8;
f=1760;
w=1.9806;
r1=1;
kwv=528.5018;
kd=i;
k=80000;
m1=4866;
m2=2433;
ma=1091.099;
p=1025;
L=2140;
kwh=1655.909;
kms=8890.7;
km=250000;
kr=z;
I1=1.8044e+04;
Ia=7142.493;
%%
X0=[0;l-m2*g/k;0;0;0;0;0;0];
jz.A=[m1+ma 0 0 0;0 m2 0 0;0 0 I1+Ia 0;0 0 0 m2*((X0(2)-X0(1))/cos(X0(4)))^2];
jz.B=[kwv+kd -kd 0     0;
-kd   kd  0        0;
0               0            kwh+kr   -kr;
0               0            -kr      kr];

jz.C=[p*g*pi*r1^2+k -k 0 0;
-k k 0 0;
0 0 kms+km -km;
m2*g*tan(X0(4)) -m2*g*tan(X0(4)) -km km];

[t,x3] = ode15s(@funq3,[0:0.01:2*40*pi/w],X0,[],jz);
ans3=0;
length1=size(x3,1);
for j=10000:length1
ans3=ans3+(kd*((x3(j,5)-x3(j,6))/cos(x3(4)))^2+kr*(x3(j,7)-x3(j,8))^2)/(length1-10000+1);
end

res(num,1)=ans3;

end
end
% figure(2);
% plot3(resx,resy,res);

figure(3);
z=zeros(51,51);
```

```
for i=1:51
z(:,i)=res((i-1)*51+1:i*51,1);
end
surf(0:2000:100000,0:2000:100000,z);
shading interp;
hold on;
plot3(58254,81689,331.61,'.','Color','y','MarkerSize',10);

function ans1=funq3(t,X0,jz)
global l g f w kwv kd k m1 m2 ma L kwh kr I1 Ia
%disp(t);
i = length(X0);
X = X0(1:i/2);       %前一半是X
DX = X0(i/2+1:i); %后一半是X关于时间的导数
b = [f*cos(w*t)-k*l*cos(X(4))+m2*g;-m2*g+k*l*cos(X(4));L*cos(w*t);0];
jz.A=[m1+ma 0 0 0;0 m2 0 0;0 0 I1+Ia 0;0 0 0 m2*((X(2)-X(1))/cos(X(4)))^2];
DDX = inv(jz.A)*(-jz.B*DX-jz.C*X+b);%降阶后的表达式
ans1 = [DX; DDX];
end
```