Problem Chosen	**2026**	**Team Control Number**
b	**MCM/ICM** **Summary Sheet**	**1234**

XXXXXXX

Summary

Keywords: k222eyword1; keyword2

Contents

1 Introduction 3
1.1 Background . . . 3
1.2 The Description of the Problem . . . 3
1.3 Our work . . . 3
2 Assumptions 4
3 Notations 5
4 Models 6
4.1 Modeling and Solving of Problem 1 . . . 6
4.1.1 Problem Analysis . . . 6
4.1.2 Model Preparation . . . 6
4.1.3 Model Construction . . . 6
4.1.4 Model Solution . . . 6
4.2 Modeling and Solving of Problem 2 . . . 6
4.2.1 Problem Analysis . . . 6
4.2.2 Model Preparation . . . 6
4.2.3 Model Construction . . . 6
4.2.4 Model Solution . . . 6
5 Sensitivity Analysis 6
6 Model Evaluation and Promotion 6
6.1 Model Evaluation . . . 6
6.1.1 Advantages . . . 6
6.1.2 Limitations . . . 6
6.2 Future Work . . . 6
6.2.1 Model extension . . . 6
6.2.2 Model application . . . 6

7 Conclusions 6

1 Introduction

1.1 Background

1.2 The Description of the Problem

本问题聚焦于《与星共舞》(Dancing with the Stars, DWTS) 节目的投票机制分析。在该竞技真人秀中，名人选手 (celebrity contestants) 与专业舞者 (professional dancers) 配对表演，通过评委打分 (judge scores) 和观众投票 (fan votes) 的组合决定每周淘汰结果。然而，观众投票数据作为商业机密从未公开，且节目在 34 季中采用了不同的投票组合规则 (voting combination methods)。核心挑战在于：在观众投票未知的情况下，如何评估投票系统的公平性并提出优化方案。

具体任务包括：

- 观众投票估算：构建数学模型，基于评委得分、选手特征和淘汰结果反向推断每位选手的观众投票数。评估估算结果与实际淘汰情况的一致性 (consistency)，并量化估算的不确定性 (uncertainty measures)。
- 投票方法对比：对比分析节目使用的两种组合方法——基于排名法 (rank-based method) 和基于百分比法 (percentage-based method) 在各赛季产生的结果差异。针对争议案例 (如 Jerry Rice、Bobby Bones 等评委低分但观众高票的选手)，评估方法选择和评委复议机制 (judges choosing from bottom two) 对最终排名的影响，并推荐未来赛季的最优方法。
- 影响因素分析：构建模型量化专业舞者经验、名人年龄 (age)、行业背景 (industry) 等因素对比赛结果的影响。分析这些因素对评委得分和观众投票的作用机制是否存在差异。
- 新系统设计：提出一种更”公平”(fair) 或更具观赏性 (exciting) 的投票组合系统。通过历史数据模拟验证新系统的有效性，并为制作方提供可操作的实施建议。
- 报告撰写：提交不超过 25 页的分析报告及 1-2 页备忘录，为 DWTS 制作方提供关于投票组合方式影响的总结和未来赛季的决策建议。

1.3 Our work

2 Assumptions

Assumption 1: The light attenuation and scattering in the underwater environment adhere to the Jaffe-McGlamery imaging model, where the propagation of light follows an exponential decay law.

3 Notations

The core symbols and their definitions used in this study are summarized in Table 1, providing an overview of the key parameters and their related meanings.

Table 1: Notations used in this literature

Symbol	Description
PSNR	Peak signal-to-noise ratio for image quality evaluation
UCIQE	Underwater Color Image Quality Evaluation metric
UIQM	Underwater Image Quality Measure

4 Models

4.1 Modeling and Solving of Problem 1

4.1.1 Problem Analysis

4.1.2 Model Preparation

4.1.3 Model Construction

4.1.4 Model Solution

4.2 Modeling and Solving of Problem 2

4.2.1 Problem Analysis

4.2.2 Model Preparation

4.2.3 Model Construction

4.2.4 Model Solution

5 Sensitivity Analysis

6 Model Evaluation and Promotion

6.1 Model Evaluation

6.1.1 Advantages

6.1.2 Limitations

6.2 Future Work

6.2.1 Model extension

6.2.2 Model application

7 Conclusions

References

[1] L. Prandtl, Fluid motions with very small friction, Proceedings of the 3rd International Mathematical Congress, Heidelberg: H. Schlichting, 1904, 484-491.

Appendices

Report on Use of AI

1. Bing AI
   Query1: write a sort algorithm
   Output: Sure, I can help you with that. Here is a simple implementation of the selection sort algorithm in Python:

   ```
   def selection_sort(arr):
       for i in range(len(arr)):
           min_idx = i
           for j in range(i + 1, len(arr)):
               if arr[min_idx] > arr[j]:
                   min_idx = j
           arr[i], arr[min_idx] = arr[min_idx], arr[i]
       return arr
   ```

   This algorithm sorts an array by repeatedly finding the minimum element from the unsorted part of the array and putting it at the beginning. The time complexity of this algorithm is O(n^2).

   There are many other sorting algorithms available such as bubble sort, insertion sort, merge sort, quick sort, heap sort, and more. Each of these algorithms has its own advantages and disadvantages. You can learn more about them on GeeksforGeeks and Programiz.

   I hope this helps!

2. OpenAl Enie (Nov 5, 2023 version, Ernie 4.0)
   Query1: <insert the exact wording of any subsequent input into the Al tool>
   Output: <insert the complete output from the second query>

3. Github CoPilot (Feb 3, 2024 version)
   Query1: <insert the exact wording you input into the Al tool>
   Output: <insert the complete output fron the Al tool>

4. Google Bard (Feb 2,2024 version)
   Query: <insert the exact wording of your query>
   Output: <irsert the complete output from the Al tool>

5. Bing AI
   Query: write a FFT algorithm
   Output: Sure, here is a Python implementation of the Fast Fourier Transform (FFT) algorithm:

   ```
   import numpy as np

   def fft(x):
       n = len(x)
       if n == 1:
           return x
       even = fft(x[0::2])
   ```

```
    odd = fft(x[1::2])
    T = [np.exp(-2j * np.pi * k / n) * odd[k] for k in range(n // 2)]
    return [even[k] + T[k] for k in range(n // 2)] + [even[k] - T[k] for k
        ↪ in range(n // 2)]
```

The FFT is a widely used algorithm for computing the **Discrete Fourier Transform (DFT)** of a sequence. It reduces the number of computations needed from O(n^2) to O(n log n) 12. The basic idea of the FFT is to apply divide and conquer. We divide the coefficient vector of the polynomial into two vectors, recursively compute the DFT for each of them, and combine the results to compute the DFT of the complete polynomial 3.

I hope this helps!