Problem Chosen	2023 ShuWei Cup Summary Sheet	Team Control Number
D		2023111121187

The Mathematics of Laundry Cleaning

Summary

Optimizing Cleaning Solutions Given Dirt Quantity, Available Water, and a_1 = 0.8: This paper establishes a nonlinear integer programming model with the objective of minimizing residual dirt. The optimal cleaning plan found involves washing twice, each with 50L of water. To explore the influence of the a_k parameter, the Monte Carlo algorithm was used. The minimum solubility required for cleaning in just two washes is 0.6667. It was found that when a_k is greater than 0.6667, two washes are optimal. For a_k between 0.58 and 0.6667, three washes are optimal, and around 0.5, six washes are required. Changing the initial dirt quantity and solving the model showed that different initial dirt amounts do not affect the washing method when a_k is fixed. Regarding the available water quantity, varying this parameter showed that with ample water, dirt can be cleaned effectively when a_k is greater than 0.5, but not when a_k is much less than 0.5. The cleanliness levels for different a_k values with a fixed water quantity of 500L were also calculated.

In addressing the issue of providing the most time-efficient cleaning solution without water limitations, this paper establishes a nonlinear integer programming model. Since the duration of each wash is equal, the objective function is to minimize the number of washes, with the constraint that the final dirt residue does not exceed one-thousandth of the initial dirt quantity. Using the Monte Carlo algorithm, it is found that the most time-efficient cleaning plan, with unlimited water and a_1=0.8, involves washing twice. To discuss the impact of a_k, further exploration based on Problem 1 is conducted. The Monte Carlo algorithm is used to find the a_k value that exactly meets the cleaning standard, which is determined to be 0.4995. As for the impact of the initial dirt quantity, the results are consistent with those of Problem 1.

Cost-Efficient Cleaning Solutions Considering Dirt Quantity and Solubility: A nonlinear integer programming model was developed with the goal of minimizing cleaning costs. The constraints included cleaning clothes or reducing eight types of dirt residues to less than one-thousandth. Different types of detergents and their a_k values for each dirt type were considered, and the Particle Swarm algorithm was used to simulate the lowest approximate cleaning cost. The optimal solution involved using detergents 1, 4, 6, 7, 8, 9, 10 once, and detergents 2, 3, 5 twice, with a total cost of 19.77 yuan.

Cost-Efficient Cleaning Solutions for Non-Mixable Materials: Materials were categorized into four groups based on mixability principles: [13456, 34568, 28, 678]. Clothes were randomly assigned to these groups, ensuring no more than 36 items per group. A nonlinear integer programming model was then established to minimize cleaning costs, with the constraint of reducing each dirt type to less than one-thousandth. The Particle Swarm algorithm was used to find the most cost-effective solution for each group, and the Monte Carlo algorithm iterated 1000 times to identify the optimal solution: group one should use detergents 1, 8, 9, 10 once, and 2, 3, 5, 6, 7 twice, with a total cost of 79.87 yuan.

Keybord: Nonlinear Integer Programming;Monte Carlo;Particle Swarm Algorithm

Content

The Mathematics of Laundry Cleaning....1
Summary....1
1.1 Background....3
1.2 Problem Data....3
1.3 Work....3
2. Problem analysis....4
2.1 Analysis of question One....4
2.2 Analysis of question Two....5
2.3 Analysis of question Three....5
2.4 Analysis of question Four....6
3. Symbol and Assumptions....6
3.1 Symbol Description....6
3.2 Fundamental assumptions....7
4. Model establishment and solution....7
4.1 Problem 1....7
4.1.1 Model Building....7
4.1.2 Model Solution and Results....10
4.2 Problem 2....13
4.2.1 Model Building....13
4.2.2 Model Solution and Results....14
4.3 Problem 3....16
4.3.1 Model Building....16
4.3.2 Model Solution and Results....19
4.4 Problem 4....21
4.4.1 Model Building....21
4.4.2 Model Solution and Results....23
5. Model Evaluation....26
5.1 Advantages of the Model....26
5.2 Disadvantages of the Model....26
References....26
Appendix....27
1 Support material....27
2 Appendix I....27
3 Appendix II....27

1. Introduction

1.1 Background

Laundry cleaning is a routine activity, where people commonly use laundry detergent or other cleaning agents. The surfactants on the surface of laundry detergent possess dirt-removing capabilities. They weaken the intermolecular forces that maintain the cohesion of water molecules, allowing individual molecules to penetrate the surface of the object being cleaned and between dirt particles. This effectively reduces the surface tension of water, enhancing its permeability. The chemical principle involves surfactant molecules having one lipophilic end, which attracts dirt and repels water, and one hydrophilic end, which attracts water molecules.

When washing clothes with a liquid containing laundry detergent, the lipophilic part of the surfactant molecules adheres to the dirt on the clothes, and the hydrophilic part repels these dirt particles. Through the mechanical action of a washing machine or hand rubbing, the dirt particles surrounded by surfactant molecules dislodge and are removed during the rinsing stage.

1.2 Problem Data

Table 1 presents the relationship between 36 pieces of clothing and the quantity of 8 types of contaminants found on each piece of clothing.

Table 2 details the unit prices of 10 types of detergents and their solubility with respect to the 8 types of contaminants.

Table 3 shows the relationship between 132 pieces of clothing and 8 types of materials, along with the quantity of 8 types of contaminants on each piece of clothing.

Table 4 outlines the compatibility of 8 types of clothing materials with 8 types of detergents, indicating whether they can be mixed and washed together.

1.3 Work

Problem 1:Given the quantity of dirt and available water, with the solubility of dirt in the k-th wash as a_k (where $a_1 = 0.80, a_k = 0.5a_{k-1}, k = 2,3, \ldots$), the first requirement is to propose an optimal cleaning plan detailing the quantity of washing and water used per wash, without considering other factors. The second requirement is to analyze and discuss the impact of a_k, initial dirt quantity, and available water on the optimal plan.

Problem 2: Assuming each washing cycle takes the same amount of time and water is unlimited, other conditions similar to Problem 1. The first requirement is to provide the most time-efficient cleaning plan, ensuring that the final dirt residue does not exceed one-thousandth of the initial amount. The second requirement is to consider a_k and initial dirt quantity, analyzing and discussing their impact on the optimal solution.

Problem 3: Assuming the water cost is 3.8 yuan/ton. The requirement is to propose the most cost-effective cleaning plan based on data from Tables 1 and 2, considering water cost, type of clothing, quantity of dirt, types of detergents, unit price of detergents, and the solubility of various detergents on dirt.

Problem 4: Based on data from Tables 3 and 4, considering the impact of non-mixable materials, types and quantities of dirt, and other conditions similar to Problem 2, the requirement is to provide a cost-effective and efficient cleaning plan.

2. Problem analysis

2.1 Analysis of question One

Problem 1: This problem is set with given initial dirt quantities and available water, allowing us to first assume constant values for both and consider each washing cycle to use equal amounts of water. Additionally, knowing the solubility of dirt in the k-th wash and the initial value of a_k, it is feasible to consider the use of only one type of detergent, assuming equal amounts of detergent are used in each wash. To determine the optimal number of washes and the water quantity per wash, we can transform the problem into finding the best number of washing cycles. A nonlinear integer programming mathematical model is established, aiming to minimize the residual dirt quantity. The constraints include the total water used in k washes not exceeding the available quantity, meeting the dirt solubility requirements set in the problem, and the number of washes being more than one. To solve this model, algorithms such as Monte Carlo, Particle Swarm Optimization, or Simulated Annealing could be utilized.

For discussing the impact of a_k, initial dirt quantity, and available water, we can use the control variable method to analyze these factors one by one, all solvable through the Monte Carlo algorithm applied to the aforementioned nonlinear integer programming model. Specifically, to discuss the impact of a_k, considering the efficiency of a_k, if a_k is large, the focus is on finding the minimum number of washes required to clean the clothes; if a_k is small, the goal is to determine the minimum number of washes needed to meet the standard. When discussing the influence of the initial dirt quantity, to simplify the problem, an upper limit of the dirt quantity can be set first, and then the impact of varying dirt quantities within this range on the number of washes can be discussed. As for the influence of available water, scenarios of both abundant water and fixed water quantity can be considered, and solutions can be derived for cleaning or reaching a certain standard of cleanliness given a fixed initial dirt quantity and a_k.

2.2 Analysis of question Two

Problem 2: This problem assumes that each washing cycle takes the same amount of time and requires the most time-efficient cleaning plan. Essentially, this translates to finding the solution with the least number of washes, similar in essence to Problem 1. Additionally, there is no restriction on the available water for this problem, and the termination condition is that the dirt residue does not exceed one-thousandth of the initial dirt quantity. Therefore, a nonlinear programming model can be established with the objective of minimizing the number of washes. The constraints include ensuring that the dirt residue does not exceed one-thousandth of the initial dirt quantity, meeting the dirt solubility requirements, and having more than one wash. The Monte Carlo algorithm can be used to solve this model.

In discussing the impact of $a_k$and the initial dirt quantity, the control variable method can be applied for individual analysis of these factors, and the Monte Carlo algorithm can be employed to solve the above nonlinear integer programming model. Firstly, in discussing the impact of a_k, the aim is to find the minimum solubility a_min required for meeting the cleaning standard. Then, a_k is categorized for further discussion: when $a_k > a_{min}$, the focus is on finding the number of washes needed for cleaning; when $a_k \leqslant a_{min}$, the goal is to determine the number of washes required to reduce the dirt residue to less than one-thousandth of the initial quantity. Secondly, the influence of the initial dirt quantity is analyzed using a similar approach and solution method as in Problem 1.

2.3 Analysis of question Three

Problem 3: This problem requires a comprehensive consideration of the $a_k$values (cleaning efficiency) of 10 detergents for 8 types of dirt, the unit price of each detergent, and the impact of water consumption (total number of washes) on cost to provide the most cost-effective cleaning solution. As every piece of clothing must be cleaned thoroughly, it's not sufficient to just meet a general standard for the total amount of dirt; instead, all 8 types of dirt must individually meet the cleaning standard, with their residual amounts being less than one-thousandth. Additionally, considering the varying a_k values of different detergents for different types of dirt, it's necessary to account for the residual amount of the same dirt type after being treated with different detergents for varying numbers of washes. An important consideration is whether the order of using different detergents affects the cleaning outcome, which can also be a direction for discussion.

Based on the above considerations, this problem can still be formulated as a nonlinear integer programming model. The objective is to minimize the washing cost, with the constraint that the residual amounts of all 8 types of dirt are less than one-thousandth. The Particle Swarm Optimization algorithm is used to solve this model, iterating multiple times to simulate the lowest approximate solution for cleaning costs.

2.4 Analysis of question Four

Problem 4: This problem requires considering the issue of non-mixable materials and providing the most cost-effective cleaning solution. Due to the large number of clothes, the approach involves grouping the clothes, similar to Problem 3, with the stipulation that each group contains no more than 36 items. Since the primary basis for grouping is material, grouping can be guided by information about special materials from Table 4 (such as Material 2 only being mixable with Material 8). On this basis, clothes are randomly assigned to groups, ensuring that each group does not exceed the limit of 36 items.

A nonlinear integer programming model is then established with the objective of minimizing cleaning costs. The constraints include ensuring that the residual amount of each type of dirt is less than one-thousandth. The Particle Swarm Optimization algorithm is utilized to compute the most cost-effective solution for each group. Then, the Monte Carlo algorithm is employed for multiple iterations to find the optimal solution.

3. Symbol and Assumptions

3.1 Symbol Description

symbol	meaning
D	Initial amount of dirt
D_k	The amount of dirt remaining in the kth time
W	Water availability
i	Type i detergent
j	Dirt of the jth type
n_i	The number of times the ith detergent was used
m_i	The unit price of the ith detergent
D_{jN}	The amount of residue of the j-th type of dirt after washing N times
a_{ij}	Solubility of type j-type dirt in type i detergent
b_{1c_i}	Contains the c_i th piece of clothing in Material 1
d_{1c_i}	The amount of dirt corresponding to the c_i piece of clothing containing material 1

3.2 Fundamental assumptions

(1) It is assumed that the detergent used in each cleaning has fully reacted;

(2) It is assumed that at least 1g of detergent is required to completely dissolve 1g of dirt;

(3) Assuming that the upper limit of the initial amount of dirt is 10g;

(4) It is assumed that the water consumption adopts the rule of rounding down;

(5) Suppose 50 liters of water are used for each wash;

(6) Suppose that for the same amount of detergent, the solubility of the dirt in the k-th wash is 0.5 of the k-1 wash, that is, $a_k = 0.5a_{k-1}$is constant;

4. Model establishment and solution

4.1 Problem 1

4.1.1 Model Building

(One) Optimal Cleaning Solution

(1) Analysis of Conditions

The problem provides initial dirt quantity and available water volume. For the initial dirt quantity, let's assume a fixed value, say 10g. As for the available water volume, let's denote it as W and assume that the water volume used in each washing cycle is equal. According to the researched data, the traditional water washing method uses 40-60L per cycle, so we take the average value of 50L as the water volume for each washing cycle.

Furthermore, the problem specifies the solubility of dirt in the k-th washing cycle and the initial value of a_k, which is a_1=0.8. Therefore, in this paper, we only consider using the same type of detergent and assume equal amounts of detergent in each wash. Thus, for a unit of detergent, the solubility of dirt is 0.8 in the first wash, 0.4 in the second wash, 0.2 in the third wash, and so on.

(2) Establishment of the Model

This question requires determining the optimal number of washes and the water volume per wash. Based on the above analysis and the assumption that the water volume per wash is equal, the problem is transformed into finding the best number of washing cycles. We now establish a nonlinear integer programming model:

The objective function aims to minimize the residual dirt quantity, which is

$$\min D_k \tag{1}$$

Given the specified amount of available water and the assumption that the water volume used in each wash is equal, the maximum number of washing cycles is fixed.

Therefore, the constraint condition is that the number of washes must be greater than or equal to 1 and not exceed the maximum number of washes, which is

$$1 \le k \le \frac{W}{50} \tag{2}$$

Given the provided solubility of dirt in the k-th wash and the initial value of a_k, the residual quantity of dirt follows the relationship

$$D_k = D - \sum_{k=1}^{k} a_1 \cdot D \times 0.5^{k-1} \tag{3}$$

$$a_1 = 0.8 \tag{4}$$

$$a_k = 0.5a_{k-1} \tag{5}$$

well organized

$$\min D_k \tag{6}$$

$$s.t. \begin{cases} 1 \le k \le \frac{W}{50} \\ D_k = D - \sum_{k=1}^{k} a_1 \cdot D \times 0.5^{k-1} \\ a_1 = 0.8 \\ a_k = 0.5a_{k-1} \end{cases} \tag{7}$$

(Two) Discuss the impact of a_k

In discussing the impact of a_k, we consider the efficiency of a_k. To define this efficiency, we use the number of washes obtained earlier as the standard number p. Simultaneously, we solve for the smallest solubility that allows all dirt to be removed in no more than p washes, denoted as $a^p{}_{min}$. Then, a_k is categorized for discussion:

When $a_k > a^p{}_{min}$, the number of washes needed to clean the dirt is consistently p.

When $a_k \le a^p{}_{min}$, the Monte Carlo algorithm is used to solve formulas (6) - (7) with p=2, determining the minimum number of washes to completely remove the dirt.

Another consideration is the scenario where the solubility of dirt under a certain detergent is too low, implying that infinite washing cycles might not remove all dirt. Hence, this paper introduces a cleaning standard: the final dirt residue should not exceed a certain proportion of the initial dirt quantity, denoted as r.

Given that the available water quantity in this problem is fixed, another scenario to consider is the possibility of having sufficient washes to remove all dirt or meet the cleaning standard, but the number of washes is too high, resulting in insufficient water. Thus, we assume W as the upper limit of water volume.

Next, we consider the reasonable value of available water volume:

(1) Determining the value of W

Since we have already assumed that each wash uses an equal amount of water, 50L, to determine the reasonable value of W, we need to establish a reasonable number of washes. Here, the Monte Carlo algorithm is applied to solve formulas (6) - (7), with the results shown as follows,

Table 1 The number of washing times corresponding to different a_k (part)

a_k	$k-th$
1	1
0.67, 0.99	2
0.58−0.67	3
0.54−0.58	4
0.52−0.54	5
0.50−0.52	6
0−0.50	∞

From the table, it is evident that when $a_k \geq 0.50$, the number of washes required to completely remove dirt is finite. However, when $a_k < 0.50$, the number of washes required becomes infinite, indicating that using a single detergent is ineffective for completely removing dirt when a_k is below a certain threshold.

To further determine a reasonable value for W, this paper re-applies the Monte Carlo algorithm with a_k in the range of 0.50 to 0.51. The results obtained are as follows,

Table 2 The corresponding washing times when a_k is between 0.50-0.51

a_k	$k-th$
0.5	20
0.51	6

From the table, it is clear that when $a_k = 0.50$, the number of washes required to clean the dirt is 20, and when $a_k = 0.51$, the number of washes required is 6. Therefore, choosing 10 as the maximum number of washes, which falls within the range of 6 to 20, the maximum available water volume is determined to be 500L.

(2) Discussing the Impact When $a_k \leq a^p{}_{min}$ under the Condition of W=500

Based on the above analysis, we need to discuss the following two scenarios: inability to completely remove dirt and the ability to completely remove dirt but constrained by water volume. Here, we introduce the solubility required to just completely remove dirt, denoted as a_{min}.

(i) Inability to Completely Remove Dirt

The cleaning standard mentioned above is such that the final dirt residue should not exceed a certain proportion of the initial dirt quantity, denoted as r. This satisfies the following relationship:

$$D_k < r \cdot D \tag{8}$$

In this context, D_k represents the final dirt residue, and D denotes the initial dirt quantity.

Therefore, it is necessary to modify the existing nonlinear integer programming model by incorporating formula (8) as a constraint. The specific modification is as follows:

$$\min D_k \tag{9}$$

$$s.t. \begin{cases} D_k < r \cdot D \\ 1 \leq k \leq 10 \\ D_k = D - \sum_{k=1}^{k} a_1 \cdot D \times 0.5^{k-1} \\ a_k = 0.5 a_{k-1} \end{cases} \tag{10}$$

Furthermore, considering the scenario where the cleanliness standard is not met even after 10 washes, the residual dirt quantity after 10 washes under different values of a_k can be calculated using the following formula:

$$D_k = D - \sum_{k=1}^{k} a_1 \cdot D \times 0.5^{k-1}, k = 10, a_k < a_{min} \tag{11}$$

(ii) Ability to Completely Remove Dirt but Constrained by Water Volume

For this scenario, the solution involves calculating the residual dirt quantity after 10 washes under different $a_k$values. The formula for this calculation is:

$$D_k = D - \sum_{k=1}^{k} a_1 \cdot D \times 0.5^{k-1}, k = 10, a_{min} \le a_k \le a^p{}_{min}. \tag{12}$$

(Three) Discussing the Impact of Initial Dirt Quantity

In determining the optimal solution, the upper limit of the dirt quantity has been set at 10g. Hence, the effect of the dirt quantity varying between 0-10g on the number of washes is discussed. To simplify the problem, the upper limit of water available for 10 washes is used, and the Monte Carlo algorithm is applied to solve formulas (9) - (10).

(Four) Discussing the Impact of Available Water Volume

The discussion of available water volume in this paper is divided into two scenarios: abundant water and fixed water volume.

(1) When Water is Abundant

Given the initial dirt quantity, witha_k fixed, the Monte Carlo algorithm is used to solve formula (8) to determine the number of washes required to completely remove the dirt or meet the cleanliness standard for different values ofa_k.

(2) When Water Volume is Fixed

Following the analysis above, the upper limit of available water is first set at 500L, which is sufficient for 10 washes. Then, given the initial dirt quantity and with a_k fixed, the Monte Carlo algorithm is used to solve formulas (9) - (10). In this case, a_k's efficiency needs to be categorized for discussion, with the solution method similar to the "Discussion of the Impact of a_k".

4.1.2 Model Solution and Results

（One）Optimal Cleaning Solution

Based on $a_1 = 0.8$and $a_k = 0.5a_{k-1}$, it can be determined that for a unit of dirt, the quantity of dirt removed in the first wash is 0.8. Theoretically, the quantity of dirt removed in the second wash would be

$$a_2 = 0.5a_1 = 0.4 \tag{13}$$

From the above results, it can be theoretically determined that the amount of dirtremoved in two washes is

$$0.8 + 0.4 = 1.2 > 1 \tag{14}$$

This indicates that for a detergent with $a_1 = 0.8$, only two washes are needed to completely remove the dirt, which means

$$k = 2 \tag{15}$$

Therefore, the optimal cleaning solution for this question is to wash twice, using 50L of water each time.

（Two）Discussing the Impact of a_k

Referring to the solution results from the previous problem, this paper takes 2 washes as the standard number of washes. By substituting $k = 2$ into formulas (9) - (10), the results obtained are as shown in the following figure,

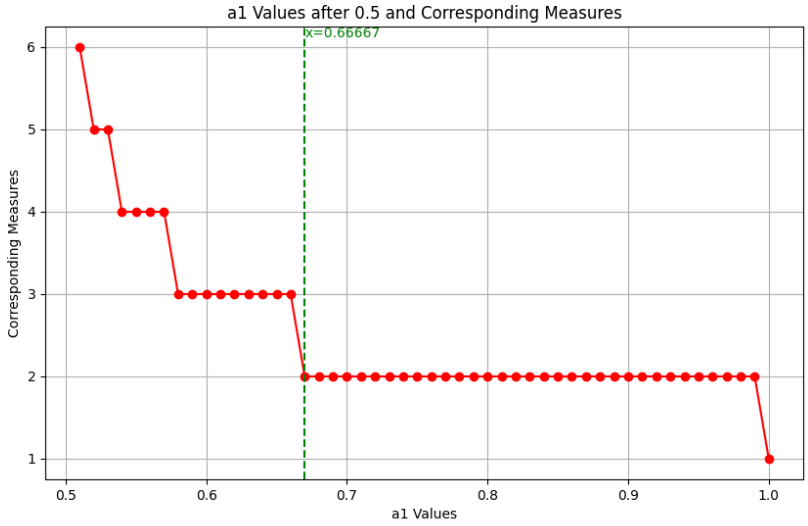


Figure 1 The number of washing times corresponding to different a_k (part)

The graph indicates that when $a^p{}_{min} = 0.6667$, which corresponds to exactly 2 washes, the minimum solubility is 0.6667. It is also evident that when a_k is between 0.58 and 0.6667, the number of washes required to completely remove the dirt is 3; when a_k is between 0.54 and 0.58, it takes 4 washes; when a_k is between 0.52 and 0.54, it takes 5 washes; and when a_k is around 0.5, it takes 6 washes.

Next, under the premise that W = 500L, we discuss the impact when$a_k \leq 0.6667$, specifically solving for the following two scenarios,

(i) Can't completely remove dirt

Using the Monte Carlo algorithm, formulas (9) - (11) are solved, where for $0.50 < a_k < 0.6667$, the results are as shown in Figure 1. The following table shows the dirt residue after 10 washes when $a_k < 0.50$

Table 3: Dirt Residue After 10 Washes When $a_k < 0.50$ (Partial)

a_k	0. 1	0. 2	0. 3	0. 4	0. 49
Dirt residue	8. 00195	6. 00390	4. 00586	2. 00781	0. 20957

Note: For complete data, refer to the supplementary materials.

The table shows that the smaller the value of $a_k = 0.50$, the greater the amount of dirt residue. With an initial dirt quantity of 10g, when $a_k = 0.1$, the dirt residue is 8.00195g; when $a_k = 0.2$, the residue is 6.00390g; when$a_k = 0.3$ it is 4.00586g; when $a_k = 0.4$ it is 2.00781g; and when $a_k = 0.49$, the residue is 0.20957g.

(ii) Ability to Completely Remove Dirt but Constrained by Water Volume

For this scenario, the residual dirt quantity after 10 washes is calculated using formula (12). The results are as shown in the following table,

Table 4: **Dirt Residue After 10 Washes When $a_k > 0.50$ (Partial)**

a_k	0.50	0.51
Dirt residue	0.00977	0

Note: See supporting materials for complete data

The table indicates that with an initial dirt quantity of 10g, when $a_k = 0.50$ the dirt residue is 0.00977g, and when $a_k = 0.51$, the dirt residue is 0g.

(Three)Discussing the Impact of Initial Dirt Quantity

Based on the above analysis, this paper sets an upper limit of 10g for the dirt quantity and 500L for the available water quantity. The Monte Carlo algorithm is applied to solve formulas (9) - (10), and the results obtained are as shown in the following figure,

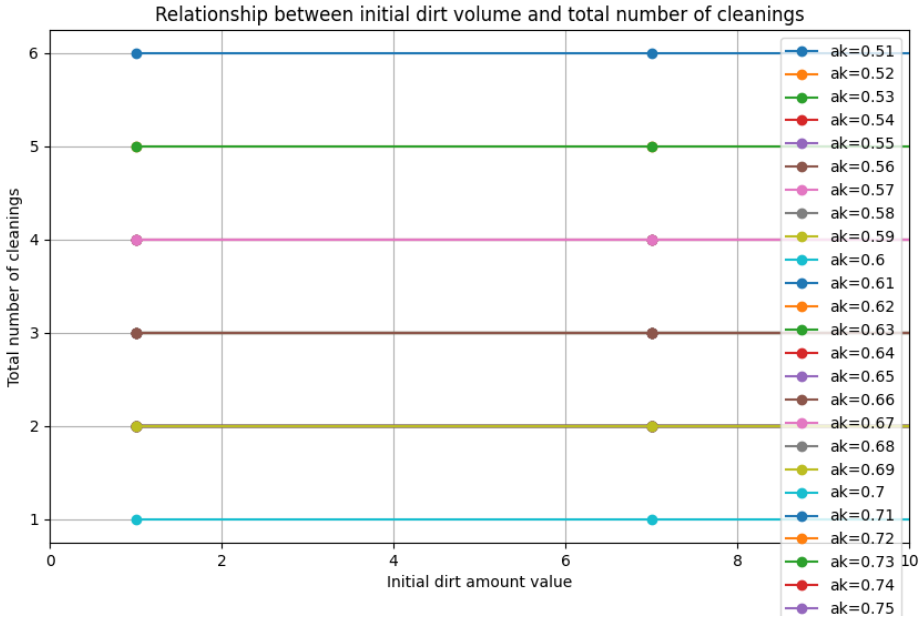


Figure 2 Relationship between initial dirt vilume and total number of cleanings

The graph shows that when a_k is fixed, the number of washes required to completely clean different masses of initial dirt is the same. Therefore, varying initial dirt quantities do not influence the washing method.

(Four)Discussion on the Impact of Available Water

(1) When Water is Abundant

Given an initial amount of dirt, under the condition of a fixed a_k, the Monte Carlo algorithm is used to solve equation (8). This process determines the number of washes needed to completely remove the dirt or to meet the cleaning standard for different values of a_k. The results are shown in Figure 1 above.

(2) When Water Quantity is Fixed

With a limit of 500L of available water and an initial dirt amount of 10g, we analyze the impact of different a_k values on the cleaning effectiveness. The results are shown in the figure below,

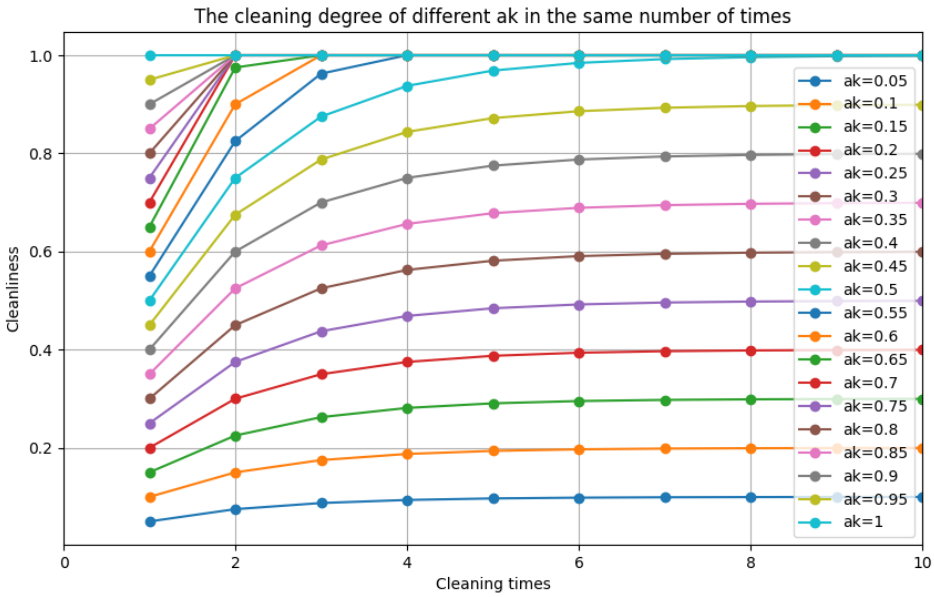


Figure 3 The cleaning degree of different ak in the same number of times

The graph shows that under the condition of a fixed water quantity, when a_k is greater than 0.5, multiple cleaning cycles can achieve the required cleaning standard. However, when a_k is less than 0.5, as a_k decreases, the final cleaning effectiveness also diminishes, and the amount of dirt removed in each cleaning cycle is reduced.

4.2 Problem 2

4.2.1 Model Building

(One) The most time-saving cleaning solution

The problem assumes that each washing takes the same amount of time and requires the identification of the most time-efficient cleaning strategy. Therefore, this is essentially a problem of finding the solution with the minimum number of washes, similar to problem one. Additionally, there are no limitations on the availability of water for this problem, and the termination condition is defined as the dirt residue not exceeding one-thousandth of the initial amount of dirt. Hence, a nonlinear integer programming model can be established. The objective function is to minimize the number of washes, which is expressed as

$$\min k \tag{16}$$

With the constraint that the dirt residue does not exceed one-thousandth of the initial amount of dirt, which is expressed as

$$D_k < \frac{D}{1000} \tag{17}$$

Other conditions are the same as in problem one, leading to the following formulation:

$$s.t.\begin{cases} D_k < \dfrac{D}{1000} \\ D_k = D - \sum_{k=1}^{k} a_1 \cdot D \times 0.5^{k-1} \\ a_1 = 0.8 \\ a_k = 0.5a_{k-1} \\ k \geq 1 \end{cases} \tag{19}$$

(Two) Discuss the impact of a_k

Removing the previously mentioned constraint of $a_1 = 0.8$, results in the following:

$$\begin{cases} D_k < \dfrac{D}{1000} \\ D_k = D - \sum_{k=1}^{k} a_1 \cdot D \times 0.5^{k-1} \\ a_k = 0.5a_{k-1} \\ k \geq 1 \end{cases} \tag{20}$$

Solve for the minimum solubility a_{min} that just meets the cleaning standard. Additionally, determine the number of washes required to meet the cleaning standard when $a_k > a_{min}$.

(Three) Discuss the influence of initial dirt amount

The model is the same as in problem one. Furthermore, it is known from the results of problem one that, when a_k is fixed, the number of washes required to clean different initial dirt masses is the same. Therefore, varying initial dirt masses do not influence the washing method.

4.2.2 Model Solution and Results

(One) The most time-saving cleaning solution

From Problem 1, it is known that when a_k equals 0.8, only two washes are required to completely clean. Since the time required for each wash is the same, the most time-efficient cleaning plan remains two washes.

(Two) Discuss the impact of a_k

The Monte Carlo algorithm is used to solve equation (20) to find the minimum solubility a_{min}, and the results are presented in the following figure,

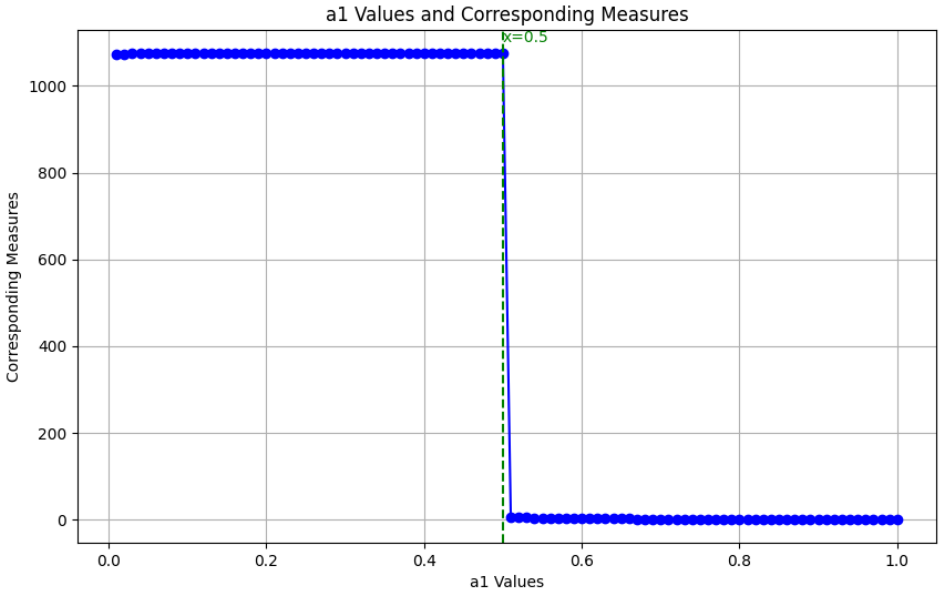


Figure 4 a1 Values and Corresponding Measures

As shown in the graph, under the condition of sufficient water, when a_k is approximately equal to 0.5, the cleaning process is just sufficient. To determine a more precise value, the range of 0.44-0.50 is selected for further calculation. The results are presented in the following figure,

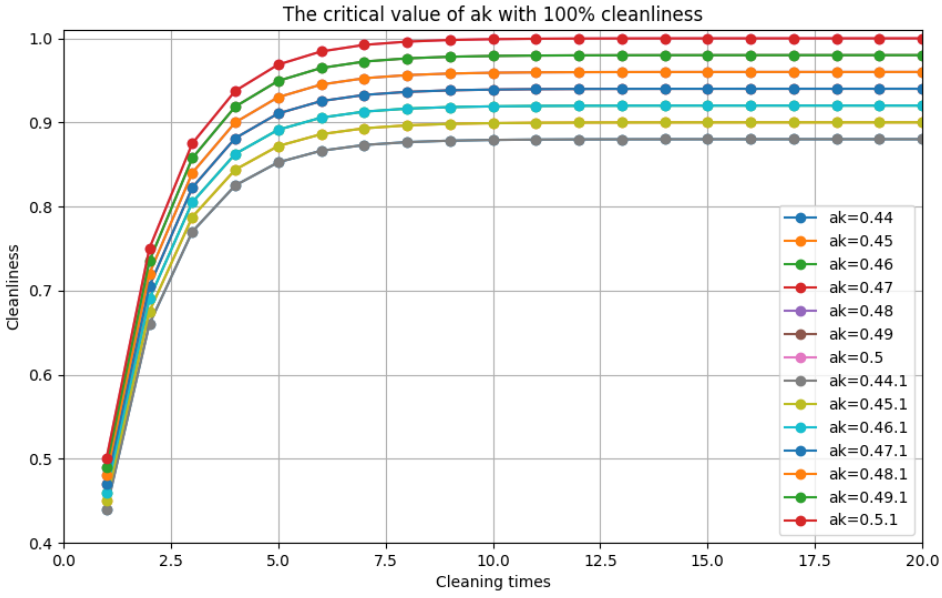


Figure 5 The critical value of ak with 100%cleanliness

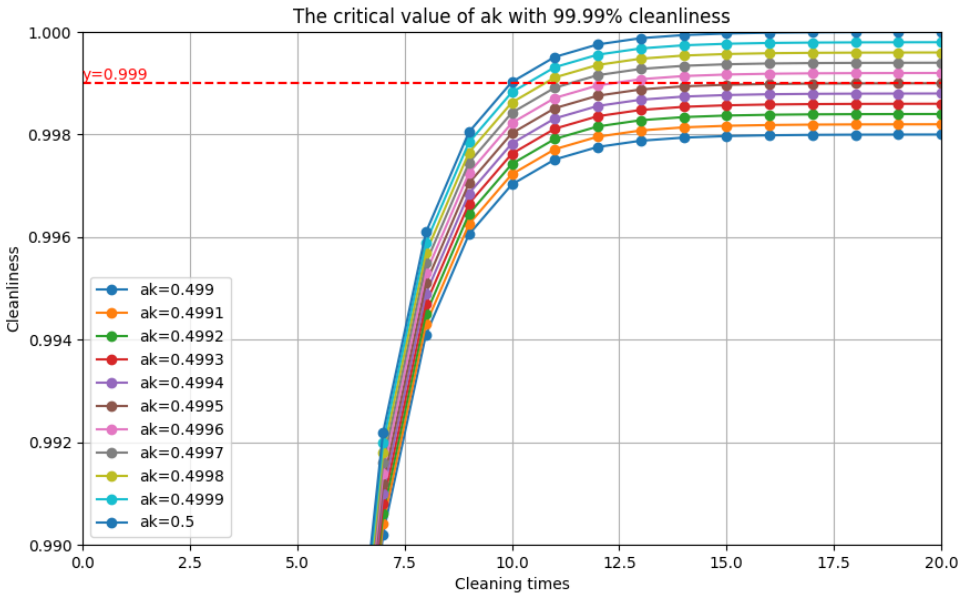


Figure 6 The critical value of ak with 99.99%cleanliness

According to the graph, when using only a single detergent, the solubility $a_k$that just achieves a one-thousandth concentration is 0.4995. It is also observed that when a_k is between 4.4 and 4.9995, the cleaning effectiveness increases with an increase in a_k , but it does not reach 99.9%, which means it cannot meet the one-thousandth standard. This indicates that using a single detergent alone is insufficient for cleaning.

The following discussion further addresses a_k in two scenarios:

(i)when $a_k \geq 0.4995$

Solving equation (11) yields the number of washes required to meet the cleaning standards for different values of a_k. The results are shown in the table below:

Table 5: Number of Washes Required to Meet Cleaning Standards for $a_k \geq 0.4995$

a_k	0.4995	0.4996	0.4997	0.4998	0.4999
k-th	21	13	12	11	11

According to the table, when $a_k = 0.4995$, the number of washes required to meet the cleaning standard is 21. When $a_k = 0.4996$, it takes 13 washes to meet the standard; for $a_k = 0.4997$, it takes 12 washes; when $a_k = 0.4998$, the required number of washes is 11; and similarly, for $a_k = 0.4999$, it also takes 11 washes to reach the cleaning standard.

4.3 Problem 3

4.3.1 Model Building

To simplify the problem, this paper assumes that the 36 pieces of clothing given in the problem can all be washed together, treating the 36 items as a single entity.

Suppose the total number of washes is N, and an equal amount of detergent is used in each wash, denoted as A. We further assume A=10g.

The problem requires providing the most cost-effective cleaning solution. Therefore, this paper establishes a nonlinear integer programming model with the objective of minimizing the costs of detergent and water, which is

$$\min \sum_{i=1}^{8} A \cdot m_i n_i + 3.8M \tag{21}$$

In this model, i represents the i-th type of detergent, j represents the j-th type of dirt, and n_i represents the number of times the i-th type of detergent is used, which must satisfy

$$N = \sum_{i=1}^{10} n_i \tag{22}$$

$$n_i \geq 0, n_i \in Z^+ \tag{23}$$

In the model, M represents the unit price of the i-th type of detergent, and M represents the total water usage in tons. Given the assumption that each wash uses 50L of water, it follows that

$$M = \frac{50N}{1000} \tag{24}$$

In order to define a standard for cleanliness, this paper refers to Problem 2, adopting the criterion that the dirt residue should be less than one-thousandth of the initial dirt quantity. Consequently, the constraint is that after N washes, the residue of the j-th type of dirt should be less than the initial residue of that type of dirt, which is

$$D_{jN} \leq \frac{D_j}{1000} \tag{25}$$

Since different types of dirt have different solubilities in various detergents, this paper denotes the solubility of the j-th type of dirt in the i-th type of detergent as a_{ij}Thus, after N washes, the relationship satisfied by the residue of the j-th type of dirt is

$$D_{jN} = D_j \left(1 - \sum_{n=1}^{n_1} a_{1j} \cdot 0.5^{n-1}\right)\left(1 - \sum_{n=1}^{n_2} a_{2j} \cdot 0.5^{n-1}\right) \ldots \left(1 - \sum_{n=1}^{n_{10}} a_{10j} \cdot 0.5^{n-1}\right) \tag{26}$$

Regarding the above equation, it is apparent that the order of adding multiple detergents does not affect the final outcome of the washing. To prove this conclusion, the paper briefly provides the following proof:

Taking two types of detergents as an example, suppose the solubility of detergent 1 is a_1, and it is used for n_1 times, while the solubility of detergent 2 is a_2, and it is used for n_2 times. Since the types of detergents used are different, their effects can be likened to cleaning directions in two different dimensions, as shown in the following figure,

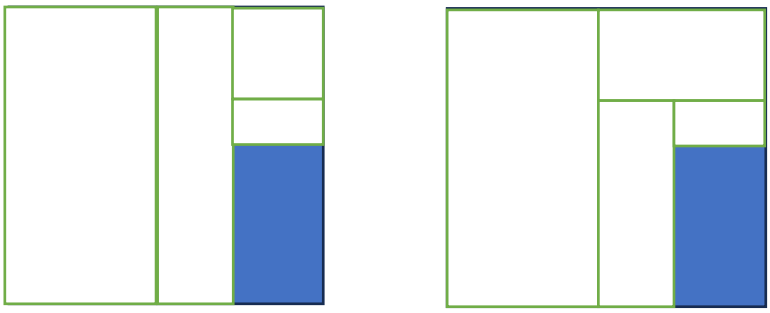


Figure 7 Explores the impact of different addition orders of detergents on the final results

In one direction, the remaining quantity is given by

$$1-\sum_{n=1}^{n_1} a_1 \cdot 0.5^{n-1} \tag{27}$$

In the other direction, the remaining quantity is

$$1-\sum_{n=1}^{n_2} a_2 \cdot 0.5^{n-1} \tag{28}$$

So then the remaining dirt amount is

$$D\left(1-\sum_{n=1}^{n_1} m \cdot 0.5^{n-1}\right)\left(1-\sum_{n=1}^{n_2} n \cdot 0.5^{n-1}\right) \tag{29}$$

Here's the English translation of the section from your mathematical modeling paper for the international competition:

The results are the same as those achieved by washing in sequence, conforming to the original equation (26).

When there are three types of detergents, this can be analogized to cleaning directions in three dimensions, and the proportion of dirt removed in each direction is the same for the final result. When there are n types of detergents, it can be viewed as having n dimensions of cleaning directions, with each cleaning direction removing a proportion of dirt as when these n detergents are used a fixed number of times, the final result still holds. Therefore, in this model, it can be concluded that the order of adding detergents does not affect the final outcome of the washing.

In summary, for solving the problem of finding the most cost-effective cleaning solution, the formula is as follows,

$$\min \sum_{i=1}^{8} A \cdot m_i n_i + 3.8M \tag{30}$$

$$s.t.\begin{cases} N=\sum_{i=1}^{10} n_i \\ n_i \geq 0, n_i \in Z^+ \\ M=\dfrac{50N}{1000} \\ D_{jN} \leq \dfrac{D_j}{1000} \\ D_{jN}=D_j\left(1-\sum_{n=1}^{n_1} a_{1j}\cdot 0.5^{n-1}\right)\left(1-\sum_{n=1}^{n_2} a_{2j}\cdot 0.5^{n-1}\right)\ldots\left(1-\sum_{n=1}^{n_{10}} a_{10j}\cdot 0.5^{n-1}\right) \end{cases} \tag{31}$$

4.3.2 Model Solution and Results

To solve the above nonlinear integer programming model, this paper employs the Particle Swarm Optimization algorithm [3]. The core idea of this algorithm is to use the sharing of information among individuals in a group, allowing the group's motion in the problem-solving space to evolve from disorder to order, thereby obtaining feasible solutions for the problem.

To prevent the Particle Swarm Optimization algorithm from getting trapped in local optima in this paper, we have employed an adaptive neighborhood model of Particle Swarm Optimization. This model allows the swarm to be divided into several different subgroups, facilitating search in multiple areas and avoiding local optima, thus aiming to find a globally optimal solution [4].

（One）Solution Steps and Flowchart

$$v_i^d = wv_i^{d-1} + c_1 r_1\left(p_i^d - x_i^d\right) + c_2 r_2\left(g_i^d - x_i^d\right) \tag{32}$$

Where r_1, r_2 are random numbers between [0,1].

Step1: Selection of default parameters

（1）Number of particles：min{100,10*nvars}, where nvars is the number of variables.

（2）Inertia weight：[0.1,1.1]，Note that the inertia weight will adaptively adjust during the iteration process.

（3）Cognitive learning factor：1.49

（4）Social learning factor：1.49

（5）Proportion of particles within the neighborhood：0.25

(6) Minimum number of particles in the neighborhood： minNeighborhoodSize =max{2, The integer part of (number of particles * proportion of particles in the neighborhood) }，At the start of the iteration, each particle has a neighborhood, initially set to the "minimum neighborhood particle number," which adaptively adjusts during subsequent iterations.

Step2: Variable Initialization and Fitness Calculation

（1）Velocity initialization: vmax =ub - lb； v= - vmax +2*vmax .*rand(n,narvs);

（2）Position initialization: Each particle's position is uniformly distributed within the upper and lower bound constraints.

（3）Calculate each particle's fitness: Fitness is still set as the target function we want to optimize.

（4）Initialize the personal best position: The position after initialization.

（5）Initialize the global best position for all particles: The particle with the lowest fitness becomes the best position for all particles.

Step3: Update Particle Velocity and Position

Update each particle's information in each iteration.

（1）Randomly generate a neighborhood for particle i, containing Q particles (including particle i), and find the best-positioned particle among them; this particle has the minimum objective function value, denoted as lbest.

（2）Update the velocity for particle i using the formula: v = w*v + c1*u1*(pbest - x) + c2*u2*(lbest - x)

（3）Update the velocity for particle i using the formula: x=x+v，similar to the basic Particle Swarm Optimization algorithm.

（4）Correct the position and velocity: If particle i's position exceeds the constraints, adjust it to the boundary; additionally, if the particle's position is at the boundary, ensure its velocity does not exceed the maximum allowed, otherwise set the velocity to zero.

（5）Calculate the fitness for particle i; if it is less than its historical best fitness, update its historical best position to its current position. Also, determine if particle i's fitness is less than the smallest fitness found by all particles so far, and if so, update the best position for all particles to particle i's position.

Step4: Adaptive Parameter Adjustment

Assuming that after the d-th iteration, all particles' information has been updated, it's necessary to update the model parameters before starting the next iteration, which reflects the adaptive process.

The rules are as follows: Let the current minimum fitness of all particles be a, and the minimum fitness of all particles after the previous iteration be b. Compare the relative size of a and b; if a < b, then set flag = 1; otherwise, set flag = 0. If flag = 0, perform the following operations:

（1）Update c=c+1 ； here, c represents the "stagnation counter," initially set to 0 at the beginning of the iteration.

（2）Update Q = min{Q+ minNeighborhoodSize, SwarmSize} ； Q: number of particles in the neighborhood; minNeighborhoodSize: minimum number of particles in the neighborhood; SwarmSize: total number of particles.

If flag = 1, perform the following operations：

（1）Update Q = minNeighborhoodSize

（2）Update c = max{c - 1, 0}

（3）Consider the value of c，ifc<2，then update w = 2*w； ifc>5，thenupdate w = w/2；Here, w is the inertia weight, and if the calculated result exceeds the upper or lower bounds of the inertia weight, it should be adjusted to the boundary.

Step5: Automatic Termination of the Iteration Loop

4.4 Problem 4

4.4.1 Model Building

（One）Grouping of Clothing

Referring to problem three, assuming a maximum of 36 garments can be washed at once, it is considered to divide the 132 garments given in this problem into four groups. Due to the different materials of the clothing, some materials cannot be mixed in the wash, hence the first step is to group the garments by material.

（1）Grouping by Material

Table 7 Restrictions on washing mixed laundry of different materials

material	1	2	3	4	5	6	7	8
1		×	√	√	√	√	×	×
2			×	×	×	×	×	√
3				×	×	√	√	√
4					×	√	√	√
5						×	√	√
6							√	√
7								√
8								

The table indicates that Material 2 can only be mixed with Material 8 for washing, and Material 7 can be mixed with Materials 6 and 8. The remaining Materials 1, 3, 4, 5, and 6 can all be washed together. Therefore, in grouping, priority is given to Materials 2 and 7 to cover as many grouping possibilities as possible. The proposed grouping method is as follows, with the results shown in the table below:

Table 8 Grouping of materials

Constituencies	material
First	1、3、4、5、6
Second	3、4、5、6、8
Third	2、8
fourth	6、7、8

From the above table, it is evident that the same material can be distributed into different groups. However, there are also cases where a material is allocated to only one group. For instance, Material 1 is only assigned to the first group, Material 2 to the third group, and Material 7 to the fourth group.

(2) Grouping Clothing by Material

Based on the grouping of materials, priority is given to grouping the garments made of Materials 1, 2, and 7. Then, the rest of the clothes are grouped randomly.

Random Allocation: Initialize four empty arrays, named group1, group2, group3, and group4. The original data, where each row represents a garment, is randomly reordered. After completely assigning the fixed materials 1, 2, and 7, each row is

checked in sequence. If a garment meets the conditions for a group, it is assigned to that group; if not, it is checked against the next group's conditions until all 83 flexibly allocatable garments out of the 132 are assigned.

(Two) For a Given Scenario, Calculate the Amount of Dirt for Each Group

Assuming a specific scenario as shown in the table below,

Table 9 A certain situation of clothing grouping

Constituencies	The material and the corresponding amount of clothing				
First	material 1	material 3	material 4	material 5	material 6
	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}
Second	material 3	material 4	material 5	material 6	material 8
	C_{21}	C_{22}	C_{23}	C_{24}	C_{25}
Third	material 2	material 8	\	\	\
	C_{31}	C_{32}	\	\	\
fourth	material 6	material 7	material 8	\	\
	C_{41}	C_{42}	C_{43}	\	\

The table shows that the amount of dirt for the first group is the sum of the dirt amounts from Materials 1, 3, 4, 5, and 6. Specifically, the dirt amount for Material 1 in the first group corresponds to the dirt amount for C_{11} pieces of clothing, which is expressed as

$$\sum_{c_i=1}^{C_{11}} b_{1c_i} \cdot d_{1c_i} \tag{33}$$

Similarly, the dirt amount for Material 2 in the first group satisfies the following formula:

$$\sum_{c_i=1}^{C_{12}} b_{1c_i} \cdot d_{1c_i} \tag{34}$$

It is evident that the total amount of dirt for the first group satisfies the formula:

$$D_{1j} = \sum_{c_i=1}^{C_{11}} b_{1c_i} \cdot d_{1c_i} + \sum_{c_i=1}^{C_{12}} b_{1c_i} \cdot d_{1c_i} + \sum_{c_i=1}^{C_{13}} b_{1c_i} \cdot d_{1c_i} + \sum_{c_i=1}^{C_{14}} b_{1c_i} \cdot d_{1c_i} + \sum_{c_i=1}^{C_{15}} b_{1c_i} \cdot d_{1c_i} = \sum_{s=1}^{5} \sum_{c_i=1}^{C_{1s}} b_{1c_i} \cdot d_{1c_i} \tag{35}$$

Similarly, the amount of dirt for the second group is calculated as follows:

$$D_{2j} = \sum_{s=1}^{5} \sum_{c_i=1}^{C_{2s}} b_{1c_i} \cdot d_{1c_i} \tag{36}$$

The third group is

$$D_{3j} = \sum_{c_i=1}^{C_{31}} b_{1c_i} \cdot d_{1c_i} + \sum_{c_i=1}^{C_{32}} b_{1c_i} \cdot d_{1c_i} = \sum_{s=1}^{2} \sum_{c_i=1}^{C_{3s}} b_{1c_i} \cdot d_{1c_i} \tag{37}$$

The thourth group is

$$D_{4j} = \sum_{s=1}^{3} \sum_{c_i=1}^{C_{4s}} b_{1c_i} \cdot d_{1c_i} \tag{38}$$

(Three) Determining the Washing Cost for Different Scenarios and Selecting the Optimal Solution

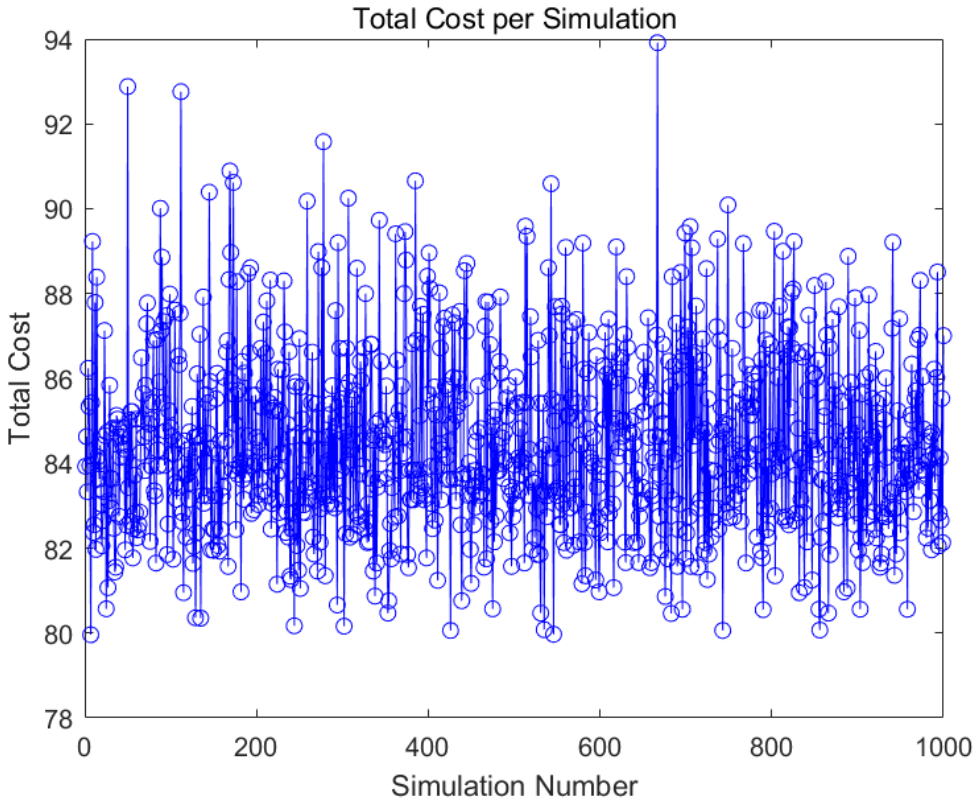
Calculate for each group using the method described in problem three.

$$\min \sum_{i=1}^{8} A \cdot m_i n_i + 3.8M \tag{39}$$

$$s.t. \begin{cases} N = \sum_{i=1}^{10} n_i \\ n_i \geq 0, n_i \in Z^+ \\ M = \dfrac{50N}{1000} \\ D_{jN} \leq \dfrac{D_j}{1000} \\ D_{qjN} = D_{qj}\left(1 - \sum_{n=1}^{n_1} a_{1j} \cdot 0.5^{n-1}\right)\left(1 - \sum_{n=1}^{n_2} a_{2j} \cdot 0.5^{n-1}\right) \ldots \left(1 - \sum_{n=1}^{n_{10}} a_{10j} \cdot 0.5^{n-1}\right) \end{cases} \tag{40}$$

4.4.2 Model Solution and Results

Utilize the Particle Swarm Optimization algorithm to solve equation (40) and calculate the cost-effective suboptimal solution for each group. Then, employ the Monte Carlo [5] algorithm for 1000 iterations to determine the optimal solution. The results of the iterations are shown in the following figure:



The graph shows that over 1000 iterations, the average cost is primarily around 84 Yuan, with the lowest cost reaching below 80 Yuan. Due to the large number of iterations and the multitude of scenarios, this paper will not enumerate each one. Below, only the results of the optimal scenario are presented.

(One) Grouping Clothing by Material

Table 10 Clothing grouping results (optimal situation)

Group1	Group2	Group3	Group4
19、20、11、12、36、124、123、103、49、104、99、70、80、100、53、56、50、125、57、27、32、25、7、67、106、54、1、9、23、29、4、96、72、47、118、102	111、115、22、119、31、64、116、65、112、24、89、84、108、120、55、94、130、113、83、15、2、39、37、126、105、48、28、87、93、42、107、60、63、73、3、58	44、110、43、129、40、17、52、10、61、90、109、92、45、16、21、88、122、8、77、82	75、34、26 33、81、114、59、78、71、68、62、95、46、18、97、35、98、79、69、5、132

(二) Amount of dirt in each group

According to equations (35) to (38), the amount of dirt for each material in each group is calculated, as shown in the table below:

Table 11 Amount of dirt in each group (optimal situation)

Group1	Group2	Group3	Group4
120	144	62	86
125	144	51	77
116	104	48	57
104	111	69	60
138	109	60	83
123	130	71	66
123	109	51	67
116	109	68	64

The table indicates that the amount of dirt for each material in each group is approximately equal, suggesting that the grouping in this scenario is reasonably balanced.

(Three)Washing costs

Substituting the above results into equations (39) and (40), and using the Particle Swarm Optimization algorithm as in problem three, the results of the solution are as follows：

Table 12 Washing scheme (optimal situation)

Group1	Group2	Group3	Group4
1	1	1	1
2	1	2	2
2	3	1	2
0	1	1	1
2	3	2	1
2	2	2	2
2	2	1	2
1	1	1	0
1	0	1	1
1	1	1	1

The table reveals that in the first group, the first to tenth types of detergents were used for washing 1, 2, 2, 0, 2, 2, 2, 1, 1, and 1 times, respectively. In the second group, they were used 1, 1, 3, 1, 3, 2, 2, 1, 0, and 1 times, respectively. In the third group, the usage was 1, 2, 1, 1, 2, 2, 1, 1, 1, and 1 times, respectively. For the fourth group, the detergents were used 1, 2, 2, 1, 1, 2, 2, 0, 1, and 1 times, respectively.

Table 13 Laundry cost results

Group1	Group2	Group3	Group4	Total Cost
19.77	20.07	20.36	16.67	79.87

The table shows that the cost for the first group is 19.77 Yuan, the second group is 20.07 Yuan, the third group is 20.36 Yuan, and the fourth group is 16.67 Yuan, with a total cost of 79.87 Yuan.

5. Model Evaluation

5.1 Advantages of the Model

(1) This paper applies the Monte Carlo algorithm and Particle Swarm Optimization algorithm, among others, dealing with a large volume of data, which makes the results more realistic and reliable
(2) The model established in this paper can be applied to a wide range of specific laundry scenarios, demonstrating its broad applicability.

5.2 Disadvantages of the Model

(1) The response to the initial amount of dirt is not significant, and there may be substantial deviations in extreme cases.
(2) The model may produce deviations when a large variety of detergents are used.

References

[1] nqi Li, Feng Yang, Reha Uzsoy, Jie Xu,A metamodel-based Monte Carlo simulation approach for responsive production planning of manufacturing systems,Journal of Manufacturing Systems,Volume 38,2016,Pages 114-133,ISSN 0278-6125,
[2] Kabiri, N.N., Emami, S. & Safaei, A.S. Simulation–optimization approach for the multi-objective production and distribution planning problem in the supply chain: using NSGA-II and Monte Carlo simulation. Soft Comput 26, 8661–8687 (2022).
[3]KennedyJ,Eberhart R .Particle swarm optimization[C]//Proceedings of ICNN'95 - International Conference on Neural Networks. IEEE, 1995
[4] 钱锋《粒子群算法及其工业应用 》科学出版社，2013，P34
[5] Vaidya, O.B., DSouza, R.T., Dhavala, S., Saha, S., Das, S. (2023). HMC-PSO: A Hamiltonian Monte Carlo and Particle Swarm Optimization-Based Optimizer. In: Tanveer, M., Agarwal, S., Ozawa, S., Ekbal, A., Jatowt, A. (eds) Neural Information Processing. ICONIP 2022. Lecture Notes in Computer Science, vol 13623. Springer, Cham.

Appendix

1 Support material

Code
Picture
Data
The result of the fourth question
The result of the third question

2 Appendix I

	The first	The second	The third	The fourth	The fifth	The sixth	The seventh	The eighth	The ninth	The tenth	Total cost
1	1	2	2	1	1	2	2	0	1	1	20.07
2	0	3	1	1	2	2	2	1	1	2	21.85
3	2	2	1	1	4	2	1	1	0	1	22.35
4	1	2	2	1	1	2	1	1	1	1	19.97
5	0	2	3	2	2	2	1	1	1	1	21.65
6	1	2	2	1	2	2	2	1	0	1	20.36
7	1	2	2	1	2	2	1	1	0	2	21.36
8	1	2	2	1	2	1	1	1	1	1	19.77
9	1	2	2	1	1	1	2	0	1	2	21.27
10	1	1	3	1	2	3	1	0	1	1	21.06
11	0	2	3	1	2	1	2	1	2	1	21.75
12	1	2	2	1	2	2	2	1	0	1	20.36
13	1	1	3	0	2	3	2	1	1	1	21.65
14	1	1	2	2	1	2	2	0	1	1	21.07
15	1	2	3	1	2	2	1	0	1	1	20.96
16	1	2	2	2	3	1	1	2	0	1	22.15
17	1	3	1	1	1	1	2	1	1	1	19.97
18	1	1	2	1	1	1	2	1	2	1	21.07
19	1	2	2	0	1	2	2	1	1	2	21.66
20	0	2	1	2	2	2	2	2	1	1	21.75

3 Appendix II

code1
matlab：Applying Monte Carlo simulation to solve integer nonlinear programming

models

```
function [bestStrategy, cleanedAmounts] = findOptimalCleaningStrategy(D_parts, X, W, a1)
    % Parameter initialization
    maxCleans = W / 50;


    numParts = length(D_parts);
    cleansUsed = 0;
    Dk = D_parts;
    bestStrategy = zeros(1, maxCleans);
    cleanedAmounts = zeros(1, maxCleans); % Stores the amount of dirt removed by each cleaning
    cleansCount = zeros(1, numParts); % Store the number of washes for each part

    % Simulating different cleaning strategies under water constraints
    while cleansUsed < maxCleans
        maxClean = 0;
        selectedPart = 0;

        % Select the part to clean that removes the most dirt
        for i = 1:numParts
            if Dk(i) > 0
                cleanPower = X;
                if i == numParts  %For the last part, use the actual amount of dirt
                    cleanPower = D_parts(i);
                end
                potentialClean = cleanPower * a1 * 0.5^cleansCount(i); % Calculate the amount of dirt that may be removed
                currentClean = min(Dk(i), potentialClean); % Make sure the amount of remaining dirt is not exceeded
                if currentClean > maxClean
                    maxClean = currentClean;

                    selectedPart = i;
                end
            end
        end

        % If there are no parts to clean, terminate the cycle
        if selectedPart == 0
            break;
        end
```

```
39.
40.         % Update cleaning strategy, dirt volume and dirt removal volume
41.         bestStrategy(cleansUsed + 1) = selectedPart;
42.         cleanedAmounts(cleansUsed + 1) = maxClean;
43.         Dk(selectedPart) = Dk(selectedPart) - maxClean;
44.         cleansCount(selectedPart) = cleansCount(selectedPart) + 1;
45.         cleansUsed = cleansUsed + 1;
46.     end
47. end
48.
49. %%%%%%%%%dividing line%%%%%%%%%
50. % parameter settings
51. D_parts = 10; % initial dirt amount
52. X = 1000; % Detergent amount
53. W = 500; % Amount of water available
54. W= fix(W/50) * 50;
55. a1 = 0.8; % cleaning efficiency factor
56.
57. % Find the optimal cleaning strategy
58. [optimalStrategy, cleanedAmounts] = findOptimalCleaningStrategy(D_parts, X,
      W, a1);
59.
60. % Output results
61. %disp('Optimal Cleaning Strategy (Part Chosen per Step):');
62. %disp(optimalStrategy);
63. disp('Cleaned Amounts per Step:');
64. disp(cleanedAmounts);
65. nonZeroCount = nnz(cleanedAmounts);
66. disp('Number of non-zero cleaned amounts:');
67. disp(nonZeroCount);
```

code2

matlab：Change the value of ak based on Monte Carlo

```
1.  % parameter settings
2.  D_parts = 10; % initial dirt amount
3.  X = 1000; % Detergent amount
4.  W = 1100*50; % Amount of water available
5.  W = fix(W/50) * 50; % Ensure that W is a multiple of 50
6.  a1_values = 0:0.01:1; % Assume that a1 ranges from 0 to 1 with a step size of 0
      .1
7.
8.  % Initialize variables
```

```
nonZeroCounts = zeros(size(a1_values)); % Used to store the nonZeroCount corresponding to each a1 value
cleanedAmountsMatrix = []; % Used to store Cleaned Amounts per Step under each a1 value

% Loop through different a1 values
for i = 1:length(a1_values)
    a1 = a1_values(i);

    % Find the optimal cleaning strategy
    [optimalStrategy, cleanedAmounts] = findOptimalCleaningStrategy(D_parts, X, W, a1);

    % Calculate the number of non-zero cleaning quantities and store the results
    nonZeroCounts(i) = nnz(cleanedAmounts);
    cleanedAmountsMatrix = [cleanedAmountsMatrix; cleanedAmounts]; % Add the results to the matrix
end

% Output table with non-zero degree
nonZeroCountsTable = table(a1_values', nonZeroCounts', 'VariableNames', {'a1', 'NonZeroCounts'});
disp('Table of Non-Zero Counts for Different a1 Values:');
disp(nonZeroCountsTable);

% Output the results of Cleaned Amounts per Step
disp('Cleaned Amounts per Step for Different a1 Values:');
for i = 1:length(a1_values)
    disp(['a1 = ', num2str(a1_values(i)), ':']);
    disp(cleanedAmountsMatrix(i, :));
```

code3

matlab：Change the value of w based on Monte Carlo

```
% parameter settings
D_parts = 10; % initial dirt amount
X = 1000; % Detergent amount
a1 = 0.8; % cleaning efficiency factor

% Initialize variables
W_values = 0:10:1000;
nonZeroCounts = zeros(size(W_values)); % Used to store the nonZeroCount corresponding to each W value

```

```
% Loop through different W values
for i = 1:length(W_values)
    W = W_values(i);
    W = fix(W/50) * 50;

    % Find the optimal cleaning strategy
    [optimalStrategy, cleanedAmounts] = findOptimalCleaningStrategy(D_parts, X, W, a1);

    % Calculate the number of non-zero cleaning quantities
    nonZeroCounts(i) = nnz(cleanedAmounts);
end

% Visualization
plot(W_values, nonZeroCounts, 'b-o'); % Draw a line chart
hold on; % Keep the current diagram so you can add more layers

% Mark the locations W = 50 and W = 100
w50_index = find(W_values == 50);
w100_index = find(W_values == 100);
plot(W_values(w50_index), nonZeroCounts(w50_index), 'ro', 'MarkerSize', 10, 'MarkerFaceColor', 'r');
plot(W_values(w100_index), nonZeroCounts(w100_index), 'go', 'MarkerSize', 10, 'MarkerFaceColor', 'g');

xlabel('Water Volume (W)');
ylabel('Number of  Cleaned Amounts');
title(' Cleaned Amounts vs Water Volume');
grid on;
legend('Counts', 'W = 50', 'W = 100');
hold off; % End layer addition
```

code4

python：(ak-times)Import excel with different ak values obtained in Program 2, and obtain visual images through python

```
###python
###(ak-times)Import excel with different ak values
obtained in Program 2, and obtain visual images through python

import pandas as pd
import matplotlib.pyplot as plt

# Load data
```

```
file_path = 'C:/Users/35274/Desktop/数维杯/第一题/Data.xlsx'
df = pd.read_excel(file_path)

# Extract data
a1_values = df.iloc[:, 0]  # first column is the value of a1
corresponding_values = df.iloc[:, 1]  # the second column is the corresponding quantity

# Plot complete data
plt.figure(figsize=(10, 6))
plt.plot(a1_values, corresponding_values, marker='o', color='blue')
plt.axvline(x=0.5, color='green', linestyle='--')  # Add a vertical line with a1=0.66667
# Add text label
plt.text(0.5, plt.ylim()[1], 'x=0.5', color='green', verticalalignment='top')

plt.title('a1 Values and Corresponding Measures')
plt.xlabel('a1 Values')
plt.ylabel('Corresponding Measures')
plt.grid(True)
plt.show()

# Make sure the data is sorted by the value of a1
df.sort_values(by=df.columns[0], inplace=True)

# Extract data
a1_values = df.iloc[:, 0]  # Assume the first column is the value of a1
corresponding_values = df.iloc[:, 1]  #Assume that the second column is the corresponding quantity

# Split data
#a1=0 to 0.5 data
mask1 = a1_values <= 0.5
a1_values_0_to_05 = a1_values[mask1]
corresponding_values_0_to_05 = corresponding_values[mask1]

# Data after a1=0.5
mask2 = a1_values > 0.5
a1_values_after_05 = a1_values[mask2]
corresponding_values_after_05 = corresponding_values[mask2]

# Plot the data for a1=0 to 0.5
plt.figure(figsize=(10, 6))
```

```
plt.plot(a1_values_0_to_05, corresponding_values_0_to_05, marker='o', color='blue')
plt.title('a1 Values from 0 to 0.5 and Corresponding Measures')
plt.xlabel('a1 Values')
plt.ylabel('Corresponding Measures')
plt.grid(True)
plt.show()

# Plot the data after a1=0.5
plt.figure(figsize=(10, 6))
plt.plot(a1_values_after_05, corresponding_values_after_05, marker='o', color='red')
plt.axvline(x=0.67, color='green', linestyle='--')  # add a1=0.66667
# Add text label
plt.text(0.67, plt.ylim()[1], 'x=0.66667', color='green', verticalalignment='top')


plt.title('a1 Values after 0.5 and Corresponding Measures')
plt.xlabel('a1 Values')
plt.ylabel('Corresponding Measures')
plt.grid(True)
plt.show()
```

code5

python：(cleanliness,Dirt amount)Import excel with different ak values obtained in Program 2 and Program 3, and obtain visual images through python

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

file_path = 'C:/Users/35274/Desktop/数维杯/第一题/Data.xlsx'

data = pd.read_excel(file_path, sheet_name=4)  # Change the requirements worksheet according to the desired requirements

# Extract data
cleaning_steps = data.iloc[1:, 0].tolist()
ak_values = data.columns[1:]  # Skip the first column and extract all ak values
selected_ak_values = ak_values[::1]  # Every few ak values can be selected upon request

# Create an empty DataFrame
df = pd.DataFrame({'Cleaning Step': cleaning_steps})

# Traverse the selected ak value and extract the corresponding data
```

```
for ak in selected_ak_values:
    df[f'Dirt Removed (ak={ak})'] = data[ak].iloc[1:].tolist()

# Draw a graph
plt.figure(figsize=(10, 6))
for ak in selected_ak_values:
    plt.plot(
        'Cleaning Step',
        f'Dirt Removed (ak={ak})',
        data=df,
        marker='o',
        label=f'ak={ak}'
    )

plt.title('Amount of Dirt Removed for Different D Values') #Change the title according to the desired requirements
plt.xlabel('D Values')  #Change the name of the x-axis according to your desired requirements
plt.ylabel('Counts')  #Change the name of the y-axis according to your desired requirements
plt.xlim(0, 7)  # Set x-axis range
plt.legend()
plt.grid(True)
plt.show()
```

code6

matlab：The objective function of particle swarm algorithm

```
function cost = objectiveFunction(n)
    % Parameter definition
    m = [0.25, 0.09, 0.11, 0.19, 0.08, 0.1, 0.12, 0.11, 0.18, 0.22]; % Detergent unit price array
    m=m*10;
    A = 1;   % Detergent dosage factor
    D_j = [116, 91, 96, 106, 99, 71, 86, 76]; % Initial dirt amount array
    a_ij = [0.54, 0.75, 0.78, 0.69, 0.8, 0.66, 0.55, 0.73;
            0.77, 0.67, 0.59, 0.58, 0.71, 0.48, 0.45, 0.66;
            0.7, 0.64, 0.62, 0.71, 0.63, 0.78, 0.5, 0.8;
            0.51, 0.54, 0.66, 0.82, 0.7, 0.6, 0.46, 0.55;
            0.39, 0.72, 0.43, 0.57, 0.46, 0.53, 0.4, 0.6;
            0.45, 0.6, 0.53, 0.48, 0.55, 0.45, 0.49, 0.77;
            0.69, 0.55, 0.62, 0.56, 0.47, 0.38, 0.44, 0.64;
            0.52, 0.44, 0.63, 0.71, 0.56, 0.68, 0.36, 0.66;
            0.8, 0.65, 0.56, 0.49, 0.73, 0.55, 0.47, 0.61;
```

```
0.47, 0.81, 0.77, 0.53, 0.64, 0.59, 0.53, 0.42]; % Solubility matrix of soil in detergents

    % Calculate the total number of washes N
    N = sum(n);

    % Calculate the total water volume M
    M = 50 * N / 1000;

    % Computing costs
    cost = A * sum(m .* n) + 3.8 * M;
    large_penalty = 10000; % or other sufficiently large values

    % Calculate the remaining amount for each dirt
    for j = 1:length(D_j)
        D_jN = D_j(j);
        for i = 1:10
            for k = 1:n(i)
                D_jN = D_jN * (1 - a_ij(i, j) * 0.5^(k-1));
            end
        end

        % Add penalties to satisfy constraints
        if D_jN > D_j(j)/1000
            cost = cost + large_penalty;
        end
    end
end
```

code7

matlab：Run the particle swarm algorithm and parameters

```
nVars = 10;  % Number of variables, corresponding to 10 types of detergents
lb = zeros(1, nVars);  % Nether
ub = [15,15,15,15,15,15,15,15,15,15];  % Upper bound

% Particle swarm parameters
%options = optimoptions('particleswarm', 'SwarmSize', 50, 'HybridFcn', @fmincon);
%options = optimoptions('particleswarm','Display','iter');
% Plot the optimal function value as a function of the number of iterations
options = optimoptions('particleswarm','PlotFcn','pswplotbestf');
[optimal_n, optimal_cost] = particleswarm(@objectiveFunction, nVars, lb, ub, options);
```

```
11. optimal_n
12. optimal_cost
```

code8

matlab

```
1.  % Import Data
2.  [~,~,rawData] = xlsread('garments.xlsx');
3.  garmentsData = rawData(2:end, :); % Assume the second line is the title
4.
5.  % Set the number of Monte Carlo simulations
6.  numSimulations = 1000;
7.
8.  % Set a maximum number of items per group
9.  maxPerGroup = 36;
10.
11. % Parameter settings of particle swarm algorithm
12. nVars = 10; % Number of variables, corresponding to 10 types of detergents
13. lb = zeros(1, nVars); % lower bound of variable
14. ub = ones(1, nVars) * 15; % upper bound of variable
15. options = optimoptions('particleswarm','MaxIterations',10000);
16.
17. % Initialize storage variables for best simulation results
18. bestSimulation = struct();
19. bestTotalCost = inf; % Initialize optimal total cost to infinity
20. % Initialize an array storing total costs for visualization
21. totalCosts = zeros(numSimulations, 1);
22. % Perform a Monte Carlo simulation
23. for sim = 1:numSimulations
24.     % Initialize four groups
25.     group1 = []; group2 = []; group3 = []; group4 = [];
26.
27.     % Shuffle the order of garmentsData to achieve random distribution
28.     shuffledIndices = randperm(size(garmentsData, 1));
29.     shuffledData = garmentsData(shuffledIndices, :);
30.
31.     % Assign fixed material to clothing
32.     for i = 1:size(shuffledData, 1)
33.         garment = shuffledData(i, :);
34.         material = garment{2}; % Use curly braces to extract data from cells
35.
36.         % allocation logic
37.         if material == 1 && size(group1, 1) < maxPerGroup
38.             group1 = [group1; garment];
```

```
39.         elseif material == 7 && size(group3, 1) < maxPerGroup
40.             group3 = [group3; garment];
41.         elseif material == 2 && size(group4, 1) < maxPerGroup
42.             group4 = [group4; garment];
43.         end
44.     end
45.
46.     % Assign clothing with optional materials
47.     for i = 1:size(shuffledData, 1)
48.         garment = shuffledData(i, :);
49.         material = garment{2};
50.
51.         %Check if this item of clothing has been assigned
52.         if any(cellfun(@(x) isequal(x, garment), [group1; group2; group3; group4]
    ))
53.             continue;
54.         end
55.
56.         % Try randomly assigning items of clothing into a qualifying group
57.         if ismember(material, [3,4,5,6]) && size(group1, 1) < maxPerGroup
58.             group1 = [group1; garment];
59.         elseif ismember(material, [3,4,5,6,8]) && size(group2, 1) < maxPerGroup
60.             group2 = [group2; garment];
61.         elseif ismember(material, [6,8]) && size(group3, 1) < maxPerGroup
62.             group3 = [group3; garment];
63.         elseif material == 8 && size(group4, 1) < maxPerGroup
64.             group4 = [group4; garment];
65.         end
66.     end
67.
68.     % Assume contaminantsData starts at column 3 and ends at column 10
69.     contaminantColumns = 3:10;
70.
71.     % Calculate the total amount of dirt in each group
72.     totalContaminantsGroup1 = sum(cell2mat(group1(:, contaminantColumns)),
    1);
73.     totalContaminantsGroup2 = sum(cell2mat(group2(:, contaminantColumns)),
    1);
74.     totalContaminantsGroup3 = sum(cell2mat(group3(:, contaminantColumns)),
    1);
75.     totalContaminantsGroup4 = sum(cell2mat(group4(:, contaminantColumns)),
    1);
76.
77.     % Perform particle swarm optimization on each group
```

```
78.    [optimal_n1, optimal_cost1] = particleswarm(@(n) objectiveFunction(n, total ContaminantsGroup1), nVars, lb, ub, options);
79.    [optimal_n2, optimal_cost2] = particleswarm(@(n) objectiveFunction(n, total ContaminantsGroup2), nVars, lb, ub, options);
80.    [optimal_n3, optimal_cost3] = particleswarm(@(n) objectiveFunction(n, total ContaminantsGroup3), nVars, lb, ub, options);
81.    [optimal_n4, optimal_cost4] = particleswarm(@(n) objectiveFunction(n, total ContaminantsGroup4), nVars, lb, ub, options);
82.
83.     % Calculate the total cost of the current simulation and store it
84.    currentTotalCost = optimal_cost1 + optimal_cost2 + optimal_cost3 + optimal _cost4;
85.    totalCosts(sim) = currentTotalCost;
86.
87.
88.    % Check if it is the best simulation
89.    if currentTotalCost < bestTotalCost
90.        bestTotalCost = currentTotalCost;
91.        bestSimulation.Group1 = group1;
92.        bestSimulation.Group2 = group2;
93.        bestSimulation.Group3 = group3;
94.        bestSimulation.Group4 = group4;
95.        bestSimulation.OptimalCosts = [optimal_cost1, optimal_cost2, optimal_co st3, optimal_cost4];
96.        bestSimulation.OptimalSolutions = [optimal_n1; optimal_n2; optimal_n3; optimal_n4];
97.        bestSimulation.SimulationNumber = sim;
98.    end
99.     % Visualize the total cost of the current simulation
100.        figure(1); % Make sure you draw in the same window every time
101.        plot(1:sim, totalCosts(1:sim), 'b-o');
102.        title('Total Cost per Simulation');
103.        xlabel('Simulation Number');
104.        ylabel('Total Cost');
105.        drawnow; % Update charts in real time
106.    end
107.
108.    % Show best simulation results
109.    disp('Best Simulation Result:');
110.    disp(['Simulation Number: ', num2str(bestSimulation.SimulationNumber)]);
111.    disp('Group 1:');
112.    disp(bestSimulation.Group1);
113.    disp('Group 2:');
114.    disp(bestSimulation.Group2);
```

```
disp('Group 3:');
disp(bestSimulation.Group3);
disp('Group 4:');
disp(bestSimulation.Group4);
disp('Optimal Costs:');
disp(bestSimulation.OptimalCosts);
disp('Optimal Solutions:');
disp(bestSimulation.OptimalSolutions);
```

Code9

matlab

```
function cost = objectiveFunction(n, D_j)
    % Parameter definition
    m = 10*[0.25, 0.09, 0.11, 0.19, 0.08, 0.1, 0.12, 0.11, 0.18, 0.22]; % Detergent unit price array
    A = 1;   % Detergent dosage coefficient
    a_ij = [0.54, 0.75, 0.78, 0.69, 0.8, 0.66, 0.55, 0.73;
            0.77, 0.67, 0.59, 0.58, 0.71, 0.48, 0.45, 0.66;
            0.7, 0.64, 0.62, 0.71, 0.63, 0.78, 0.5, 0.8;
            0.51, 0.54, 0.66, 0.82, 0.7, 0.6, 0.46, 0.55;
            0.39, 0.72, 0.43, 0.57, 0.46, 0.53, 0.4, 0.6;
            0.45, 0.6, 0.53, 0.48, 0.55, 0.45, 0.49, 0.77;
            0.69, 0.55, 0.62, 0.56, 0.47, 0.38, 0.44, 0.64;
            0.52, 0.44, 0.63, 0.71, 0.56, 0.68, 0.36, 0.66;
            0.8, 0.65, 0.56, 0.49, 0.73, 0.55, 0.47, 0.61;
            0.47, 0.81, 0.77, 0.53, 0.64, 0.59, 0.53, 0.42]; % foul solubility matrix

    % Calculate the total number of washes N
    N = sum(n);

    % Calculate the total water volume M
    M = 50 * N / 1000;

    % Computing costs
    cost = A * sum(m .* n) + 3.8 * M;
    large_penalty = 10000; % Penalty value

    % Calculate the remaining amount for each dirt
    for j = 1:length(D_j)
        D_jN = D_j(j);
        for i = 1:10
            for k = 1:n(i)
                D_jN = D_jN * (1 - a_ij(i, j) * 0.5^(k-1));
```

```
            end
        end

        % Add penalties to satisfy constraints
        if D_jN > D_j(j)/1000
            cost = cost + large_penalty;
        end
    end
end
```