基于抽样检测的企业生产过程决策优化模型

摘　要

随着科技的飞速发展，电子产品的普及率急剧上升，众多企业在承担零配件装配任务时，面临着是否进行质量检测以及如何有效处理不合格品的双重挑战。本文基于**抽样检测方法、动态规划模型及单目标规划**等研究方法，对企业生产过程中的各个环节综合分析，建立基于抽样检测的企业生产过程决策优化模型。

针对问题一，首先构建零假设和备择假设，次品数量用**二项分布**拟合，确定检测次数尽可能少的检测方案，然后设定**两阶段抽样检测**，根据 95%和 90%置信度，比较次品率范围，判断是否接收一批零配件。本文在第一阶段采取小样本量的检测，在检测次品率过高或过低的情况下，严格按照**置信度**要求帮助企业快速做出拒收或接收选择。在第二阶段提高检测准确率，做出更准确的选择。

针对问题二，首先假设这一批零件能够生产出的成品数量是 100，然后建立以各零配件、成品的检测成本，不合格成品的拆解成本，以及不合格成品的调换损失为自变量的**目标函数**，目标是使**生产成品的成本最小**，通过对比不同情况下生产出 100 个成品所需要的成本来得出最好的**决策方案**。

针对问题三，首先结合问题二的解题步骤列出影响生产总成本的因素，然后得到计算**总成本数学公式**，此外由于这道题较为复杂因此要**优化解题思路**，我们结合对列出的成本函数采取**贪心算法优化解题策略：**在每个售卖周期内，根据当前检测成本和产品次品率，**局部最优**地选择是否进行检测，以期实现总成本最小化。虽然这种方法不保证全局最优解，但计算效率比较高。

针对问题四，要求在新的条件下重新对问题三和问题四进行求解，因为新增的条件与问题一的某些因素相似，因此对于问题二和问题三的解答过程基于上述模型进行，因为抽样检测出的**样品次品率并不完全准确**，因此在重新解决问题二和问题三时，对于某些决策导致的对于同一零件成品的多次售卖，要对此时的**售卖整体重新评估次品率**，这导致解决问题的方法较为复杂，因此对于问题二和问题三的解题方法再次使用**贪心算法优化策略**，在每个售卖周期内，根据当前检测成本和产品次品率，局部最优地选择是否进行检测，以期实现总成本最小化。

关键词：抽样检测　动态规划　单目标规划　两阶段抽样检测 贪心算法优化策略

一、 问题重述

1.1 问题背景

电子产品的生产过程复杂，有些企业负责将零配件装配成成品，扮演着重要的“集成制造商”的角色，由于电子产品由许多个零配件组装而成，任何一个环节的疏忽或零配件本身的质量问题都是决定成品是否合格的关键因素。不合格品一旦流入市场，不仅会影响企业的品牌形象和企业信誉，还会增加退换、物流等额外成本，这对企业的经济利益造成了损失。

因此，对各个零配件和成品的次品率、检测成本以及不合格成品的调换损失和拆卸费用进行综合分析，从而对从生产到退回不合格成品的各个阶段决策，可以帮助企业实现利润最大化。

1.2 问题要求

在装配过程中，只要其中有一个零配件不合格则整个成品不合格，但是如果两个零配件都合格，装配的成品也不一定合格。对于不合格的成品，企业有两种解决方案，第一种方案是选择将产品报废，第二种方案是将不合格品进行拆解，拆解过程不会对零配件造成损坏，但增加了拆解成本。基于上述背景我们需要建立数学模型解决以下问题：

问题一：该问要求为企业设计抽样检测方案，并且检测次数要尽可能少来检验供应商供应的零配件是否超过某个标称值。当标称值为 10%时，在以下两种情形下得出具体结果：在 95%的信度下认定零配件次品率超过 10%，则拒收零配件；在 90%的信度下认定零配件次品率不超过 10%，则接收零配件。

问题二：该问中展示了企业生产过程中的四个阶段，在已知两种零配件和成品次品率的条件下，为企业生产过程中的各个阶段作决策。表 1 展示了企业在生产过程中遇到的情况，根据做出的决策对表 1 中的情形给出具体的决策方案，并给出决策的依据和相应的指标结果。

问题三：已知零配件、半成品和成品的次品率，在问题二的基础上对 m 道工序、n 个零配件的生产过程进行决策。图 1 展示了 2 道工序、8 个零配件的情形，表 2 给出企业在生产中遇到的情况，针对该情形制定出决策方案，并给出决策的依据及相应指标。

问题四：假设问题二和问题三中各个零配件、半成品及成品的次品率通过抽样检测方法得到，在该假设条件下，重新完成问题二和问题三。

二、 问题分析

2.1 问题一的分析

针对问题一，题目中由于只是给出了这批零件的标称的次品率，没有给定该批零件

的具体的哪些合格，哪些不合格的具体数据，因此我们要自己设计出符合题目要求的这批零件是否合格的数据，通过构建零假设和备择假设，次品数量用二项分布拟合，当样本量足够大时，可选用正态分布，然后用多种方法对这些零件做抽样检测，将不同种抽样方法的结果综合比较，找出检测次数较少且较为可靠的抽样方法，并在两种不同信度下对这种方法进试验判断是否能够在不同信度下认为零配件次品率是否超过标称值，从而接收或拒收这些零件。

2.2 问题二的分析

针对问题二，企业生产过程一共有四个阶段，阶段一和阶段二中涉及到是否检测，阶段三中涉及不合格品是否拆解问题，因为是否检测、是否拆解均只存在两种情况，因此我们首先建立 0-1 模型，因为成本影响因素不单一，因此这是一个单目标规划问题，目标是对各个阶段综合分析进行决策，使最终成本最小。由于各个阶段不独立，每个阶段的决策可能影响后续阶段，因此我们使用动态规划方法来优化决策路径。表 1 提供了企业在生产过程中遇到的六种情况，每种情况中零配件和成品的次品率、检测成本以及不合格成品的调换损失和拆解费用数据明确，我们使用上述构建的模型对表 1 的情形得出决策方案。

2.3 问题三的分析

针对问题三，该问题是在多零配件、多工序情况下建立模型，随着零配件和工序的增加，产品生产过程中要考虑半成品，影响成本的因素包括零配件、半成品和成品的检测费用，半成品、成品的拆卸费用，以及成品的调换损失，因此要对各个因素综合分析进行决策，使最终成本最小。在该问题中我们同样要建立一个单目标规划模型，目标同问题二一样，但问要解决的问题更加麻烦，因此我们不仅需要对第二问的求解模型进一步完善，还要改进优化算法以达到更加快速，使模型更准确贴近实际情况。表 2 提供了企业生产过程中的详细信息，针对表中的信息，我们利用上述模型给出决策方案。

2.4 问题四的分析

针对问题四，该问中没有提供零配件、半成品和成品的次品率，次品率需要通过抽样检验得到，由于问题一中的抽样检测方法具有检测的样本数量较少并且检测准确率较高的优点，因此可以将该抽样检测方案用于问题四，基于此方案将模型进行完善，对问题二和问题三进行决策。

三、 模型假设

1.企业将零配件装配好后，不存在成品未售出的情况。

2.在抽样检测、拆卸、重新装配之后，半成品或成品的次品率降低，假设次品率的降低按照一定比例。

3.零配件组装生产以及半成品、成品的检测阶段，不出现设备故障、生产事故等情况。

4.抽样检测估计次品率时，假设每次检测样品中的次品数量符合二项分布，抽样检测的样本量根据置信水平来确定，在样本数足够大的情况下，近似于正态分布。

5.假设在每个生产阶段，各个零配件、半成品和成品的检测与装配操作是并行进行的，而且每次循环中，所有产品（包括零配件、半成品和成品）可以同时被检测或装配。

四、 符号说明

符号	说明
n	抽样检测样本量
X	零部件中次品的个数
$\hat{p}$	样品中的次品率
p_0	标称值
$x_{1,2,3}$	布尔变量
$\mathrm{n}_{i,k}$	第 k 次退回的第 i 个零件的数量
$C_{\text{零配件检测成本}}(i,k)$	第 k 次退回的第 i 个零件的检测成本
n_k	第 k 次退回过程中由合格零件能够组装的总的成品数量
p_j	第 j 个工序的次品率
$n_{\text{调换数量}}(k)$	第 k 次退回的成品数量
$p_{\text{装配}}$	总的装配的次品率
$C_{\text{成品以及半成品检测}}(i,k)$	第 k 次退回的第 j 道工序的零件的检测成本
$\alpha_i(\mathrm{k})$	第k次售卖过程中零配件i成品非次品率递减比例
$\mathrm{p}_{\text{合格的零配件}}$	拆解后能回收的合格零配件的概率
$C_{\text{回收的收益}}$	每件回收零配件的价值
$\mathrm{p}_{\text{拆解后的合格零配件}}$	拆解后零配件合格的概率

五、 模型的建立与求解

5.1 问题一模型的建立与求解

问题一模型建立思路图如图 1 所示：

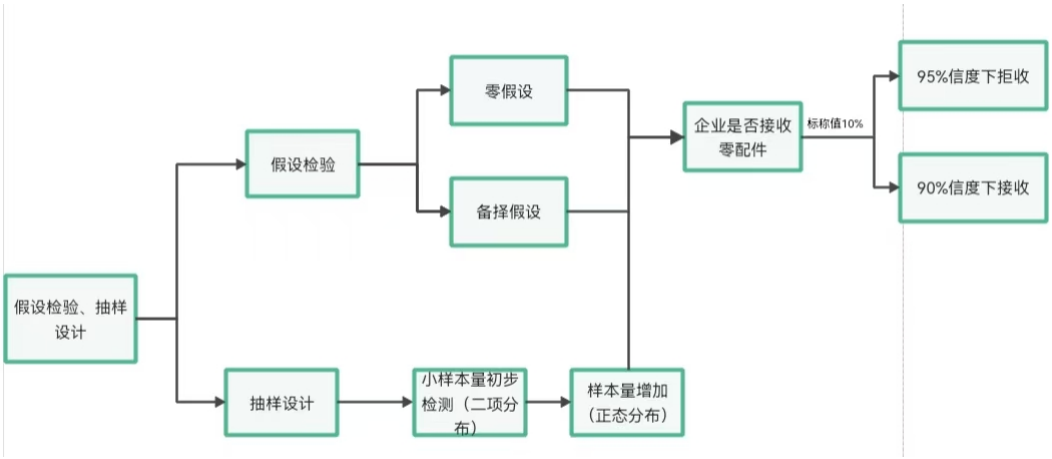


图 1 问题一模型建立思路图

问题一要求我们设计一个抽样检测方案，要求我们用尽可能少的次数来判断零配件的次品率是否超过标称值。由于题目中并没有给我们任何这批零件的是否合格的数据，因此我们要自己设计出这些数据来检验方法，由于题目还要求我们给出有抽样检测的次数，因此这就要求我们在确定方法之前事先确定不同信度水平下最小的样本数。

5.1.1.1 假设检验

假设企业需要检测的一批零部件次品率超过标称值 p_0,我们可以通过假设检验的方式来实现，假设次品率 p 不超过 p_0，检验结果也应当符合正态分布，可以构建如下假设：

零假设 H_0：次品率 $p \le p_0$ ；

备择假设 H：次品率 $p > p_0$ 。

使用二项分布检测不合格品的数学公式：

$$X \sim B(n, p)$$

其中 n 为样本量，X 为零部件中次品的个数。

根据原理在假设 H_0 下，不合格品数量的概率：

$$P(X = k) = C_n^k p_0^k (1 - p_k)^{n-k}$$

其中 k 表示不合格的数量。

当样品量取得数目足够多时可以将二项分布近似等于正态分布来计算，这时根据中心极限定理，次品数的期望值和方差为：

$$\mu = n \cdot p$$

$$\sigma = \sqrt{n \cdot p \cdot (1-p)}$$

可得近似的正态分布函数:

$$Z = \frac{\hat{p} - p_0}{\sqrt{p_0 \cdot (1-p_0)}}$$

其中 $\hat{p}$是样品中的次品率， p_0 是标称值， Z 是正态分布检验统计值。

5.1.1.2 两次抽样设计

企业可以选择进行两次抽样检测，在第一次检测时抽取较少量零配件样本进行检测，如果检测零配件得出的次品率明显高于标称值，则可以直接拒收；相反，如果检测得到的次品率明显低于标称值，可以直接接收。如果第一次检测不能得到明确的结论，则进行第二次抽样检测。

根据假设检验的公式，在 95%置信水平下需要检测的样本数量计算公式为：

$$n = \frac{Z^2 \cdot p(1-p)}{E^2}$$

其中：

$Z = 1.96$ 是 95%置信水平对应的标准正态分布的临界值；

$p = 0.10$ 是次品率的标称值；

$E = 0.01$ 是误差范围。

计算得出$n \approx 346$，说明在 95%置信水平下，需要检测 346 个样本才能做出可靠判断。

第一次抽样设定样本数量a_1为 100，检测出的次品数量记为b_1，从而推出零配件次品率 $p_1 = {}^{b_1}\!/_{a_1}$ 设定拒收和接收的临界次品率：

当 $p_1 > 15\%$ 时，企业拒收；

当 $p_1 < 15\%$ 时，企业接收。

如果次品率在 5%-15%之间，则不能妄下定论，需要进行第二次抽样检测。

在第二次抽样检测中，样本数量a_2设置为 250，总的样本数量n为 350，在此次检测中，检测出的次品数量计为b_2，综合第一二次抽样检测得到的总次品率为 $p = {}^{b_1+b_2}\!/_{a_1+a_2}$，最终企业根据次品率判断接收还是拒收这一批零件。

5.1.2 模型的建立

根据 5.1.1 使用到的检验方法以及得出的最小样本数的值，我们还要求解置信区间来确定样品次品率是否落在可以接受的范围内：

$\hat{p} \pm Z_{\alpha/2}\sqrt{\frac{\hat{p}\,(1-\hat{p})}{n}}$，其中 $\hat{p}$ 是样品的次品率， $Z_{\alpha/2}$ 是对应置信水平的标准正太分布

的临界值。

1.在 95%的信度下认定零配件次品率超过标称值，则拒收：

（1）对于第一次抽样检测，零配件次品率标称值p_0=0.10,次品的理论期望值和标准差为：

$$\mu_0 = a_1 \cdot p_0 = 10$$

$$\sigma_1 = \sqrt{a_1 \cdot p_0 \cdot (1 - p0)} = 3$$

若检测得到的零件次品率$p_1 = 0.15$时，次品数量$b_1 = a_1 \cdot p_1 = 15$，标准化 Z 值为:

$$Z = \frac{b_1 - \mu_0}{\sigma\sqrt{n}} \approx 1.67$$

Z 值对应的置信水平为 95.25%，可以直接拒收。

（2）对于第二次抽样检测，总样本量 n 为 350，95%置信水平下的临界次品数量为：

$$\mu_2 = n \cdot p_0 = 35$$

$$\sigma_2 = \sqrt{n \cdot p_0 \cdot (1 - p_0)} \approx 5.61$$

$$b_2 = Z\sigma_2 + \mu_0, 其中Z = 1.96$$

若总次品量大于b_2，则拒收这一批零配件。

2.在 90%的信度下认定零配件次品率低于标称值，则接收：

（1）对于第一次抽样检测，当样本次品率 p_1 实际检测为 0.05 时，次品数量$c_1 = a_1 \cdot p_1 = 5$，标准化 Z 值为：

$$Z = \frac{c_1 - \mu_0}{\sigma\sqrt{n}} \approx -1.67$$

Z 值对应置信水平为 4.75%，可以直接接收。

（2）对于第二次抽样检测，90%置信水平下的临界次品数量为：

$$c_2 = \mu_0 - Z\sigma_2, 其中Z = 1.645$$

若总次品量小于c_2，则接收这一批零配件。

5.1.3 模型的求解

程序运行结果：

（1）第一次检测：

如果检测到次品数量大于 15.0，则拒收；

如果检测到次品数量小于 6.0，则接收。

（2）第二次检测：

如果检测到次品数量大于 44.0，则拒收；

如果检测到次品数量小于 28.0，则接收。

由此结果我们可以看到，在第一次检测时次品数量大于 15 个，即次品率大于 15%时，可以直接选择拒收零配件。次品数量小于 6 个，即次品率小于 6%时，可以直接接收零件。次品数大于 6 小于 15 时，不太能选择是拒收还是接收，进行第二次抽样检测。

第一次检测阶段能够让企业快速做出选择，当次品率明显过高或过低时，企业只需要检测少量样品就能得出结论，很有效的减少了成本。

第二次检测时，次品数量大于 44 个，即次品率大于 12.6%时，可以直接选择拒收零件。次品数量小于 28 个，即次品率小于 8%时，可以选择接收。

第二次检测阶段使得企业能够更准确的判断零配件的次品率，降低了企业做出错误选择的可能性。

总的来说，一般情况下，企业很可能只需要进行第一次检测就能做出选择，有效地降低检测成本。同时检测的过程均满足 95%置信度的拒收标准和 90%置信度的接受标准，并且能够保证有很小的检测误差。相比一次直接检测 350 个样本，两次抽样检测能够节省大量经济和时间成本。

5.2 问题二模型的建立与求解

5.2.1 模型的建立

在生产过程中，每个环节的决策都影响着企业最后的收入，因此我们要分析各个情况下企业的最终成本，通过最终成本对比来确定各个环节的优化方案。根据题目中零配件与成品的次品率不同的情况，我们有如下两种思路：

思路 1 是购买相同数量的零配件 1 和零配件 2，选择是否进行检测后装配；

思路 2 是保证企业有相同的成品产出，在购买零配件之前就预先考虑是否检测。

如果要对某一零配件进行检测，就根据次品率标称值计算零配件损失期望值，适当增加购买量。如果对某一零配件不进行检测，就按照企业产出要求，购买等量的零配件即可。

经过联系实际对比，我们认为思路 2 具有更好的经济效益，能够减少两种零配件使用量不平衡的问题，降低不合理的浪费。

企业生产各个阶段流程如图 2 所示：

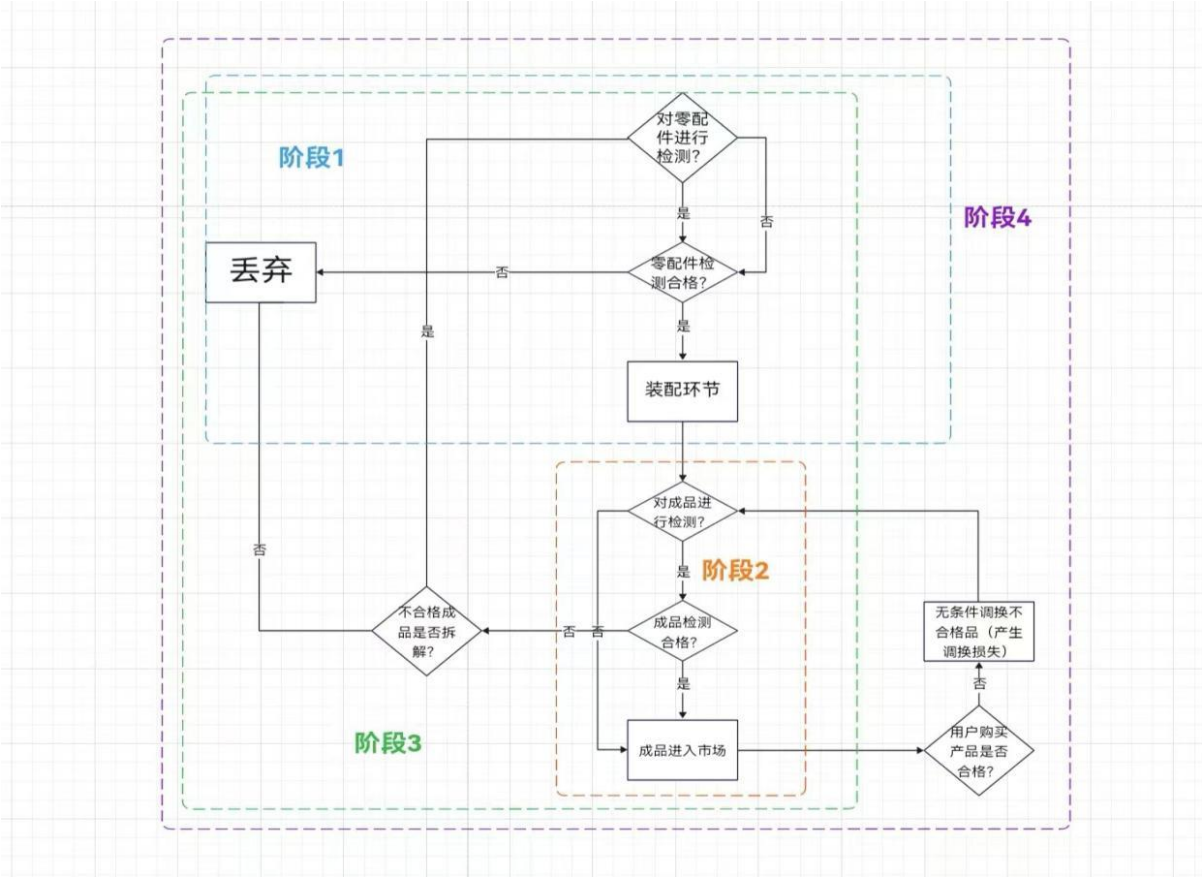


图 2 企业生产各个阶段流程图

5.2.1.1 变量说明

N： 生产的成品数。

$p_{01,02,03}$：零配件 1、2，成品的次品率。

$a_{1,2}$： 零配件 1、2 的采购价。

$b_{1,2,3}$： 零配件 1、2，成品的检测成本。

$c_{1,2,3,4}$： 成品的装配成本，拆解费用，调换损失，市场售价。

$x_{1,2,3,4}$:布尔变量，表示是否对零配件 1 进行检测，是否对零配件 2 进行检测，是否对零配件 3 进行检测，是否对不合格的成品进行拆解。

5.2.1.2 成本效益公式

1. 采购成本

零配件 1、2 的采购数量:

$$N_1 = \frac{N}{1 - p_{01} \cdot (1 - x_1)}$$

$$N_2 = \frac{N}{1 - p_{02} \cdot (1 - x_2)}$$

零配件 1、2 的采购成本:

$$C_{采购配件一} = N_1 \cdot a_1$$

$$C_{采购配件二} = N2 \cdot a2$$

2. 检测成本

对于零配件一二是否检测，若检测则全部都检测，不检测则全部都不检测，可以直观的得到检测成本。

$$C_{检测配件一} = N_1 \cdot b_1 \cdot x_1$$

$$C_{检测配件二} = N_2 \cdot b_2 \cdot x_2$$

$$C_{检测成品} = N \cdot b_3 \cdot x_3$$

3. 装配成本

$$C_{装配} = N \cdot c_1$$

4. 拆解和调换成本

装配后的成品可能由合格的成品和不合格的成品组成，因此企业需要决定否对成品进行检测以避免流入市场造成损失。

不合格成品拆解后，可能得到合格零配件

$$C_{拆解} = x_4 \cdot N \cdot \left(1 - P_{合格成品}\right) \cdot c_2 - x_4 \cdot N \cdot \left(1 - P_{合格成品}\right) \cdot k \cdot c\_2$$

其中，k 为比例系数，反应得到合格零配件的可能性。

$$P_{合格成品} = \left(1 - p_{01}(1 - x_1)\right) \cdot \left(1 - p_{02}(1 - x_2)\right) \cdot (1 - p_{03}(1 - x_3))$$

不检测成品造成次品流入市场，带来的调换损失为

$$C_{调换} = (1 - x3) \cdot N \cdot (1 - p_{合格成品}) \cdot c_3$$

其中优化的目标函数为：

$$Min \ \ C_{总成本} = C_{采购配件一} + C_{采购配件二} + C_{检测配件一} + C_{检测配件二} + C_{检测成品} + C_{调换} + C_{拆解} + C_{装配}$$

5. 2. 2 解决措施

在解决问题的过程中每一步都决定着接下来每一阶段的收益与损失，每种情况的 4

种决策构成 16 种方案，采用枚举的方法来逐个对比决策路径。

5.2.3 模型的实际求解

（1)**初始设定：**设定企业额定生产量，由生产量要求设定零配件一和零配件二的购买量。这些参数将作为后续计算的基础。

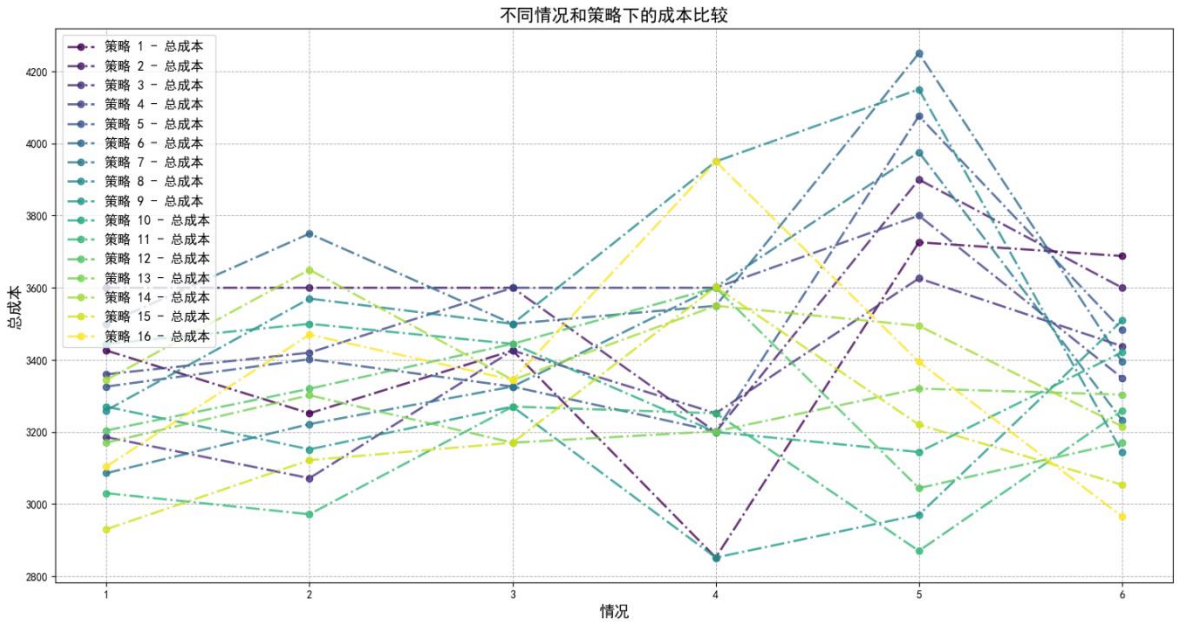
（2)**成本求解函数：**列出影响成本的各个部分的成本函数。这些函数将用于计算不同决策情况下的生产总成本。成本函数可能包括生产成本、拆解成本、检测成本等。

（3)**输出和记录：**输出每种决策及其对应的总成本，并记录每种决策所对应的生产最低总成本。总结每种决策的成本效果，找出最优的决策方案。

利用 python 代码可以得到如下的结果：

	检测零配件一	检测零配件二	检测成品	拆解	总成本	调换成本
1	否	否	否	是	2930.44	60
2	否	是	否	是	2972.00	120
3	否	否	否	是	3170.44	300
4	否	是	是	是	2852.50	0
5	否	是	是	是	2870.44	100
6	否	否	否	否	2965.79	50

结果进行可视化处理：



图三 不同情况和策略下的成本比较

5.2.4 结果分析

针对零配件一的检测：

经过分析结果发现，在所有情况下都未对零件一进行检测，题目中零件一购买单价较低，若要进行检测，零配件一的成本花费直接翻倍，直接进行装配可能是更好的方式。

针对零件二的检测：

经过分析结果发现，在部分情况下选择对零件二进行检测，因为零件 2 的购买单价较高，次品流入市场会有较大的损失，相比不进行检测导致次品进入装配，检测要更好一点。

针对成品的检测：

经过分析结果发现，在多有些情况下会对成品进行检测，成品检测能有效防止次品流入市场，但其检测带来的成本较高，如果后续流程有更高的调换成本和拆解成本的话，最好是进行成品检测。

拆解的策略选择：

经过对结果的分析，我们发现，在测试的六种情况中，有五种情况通过拆解不合格成品再利用可以获得更高的效益。而在第六种情况中，选择不拆解零件进行丢弃，说明在这种情况下，拆解成本过高，导致不拆解零件的效益反而更高。

总成本分析：

经过分析结果可以发现，情况 6 既不检测，也不拆解就能达到很低的成本。这可能是因为在这种情况下，成品次品率较低，从而减少了拆解和检测的成本。其他情况下的成本也相对接近，说明这些情况下的决策选择较为均衡，即使初始值较高，通过调整策略，最终使成本趋于一致。

对于企业来说，当零配件的采购价较低时，没有太多必要进行检测，只有当零配件的成本在生产线上占比较大时，才优先考虑进行检测。对于企业信誉来说，退换货会影响企业信誉，降低长期运营收益，为了保证企业公信度和持久力，建议在适当的成本损失内对装配的成品进行检验，避免产品以次充好的情况。

5.3 问题三模型的建立与求解

5.3.1 模型建立

在问题二的基础上，问题三要求我们完善模型，以建立一个针对多工序和多零配件生产流程决策的数学模型。由于每一道工序不仅会影响后续工序，还会对最终的利润产生影响，因此必须考虑多个因素：包括不同装配件的次品率对半成品和成品次品率的影响，以及在不同工序中做出的不同决策对后续工序的潜在影响等。由上述可以建立的一个总的成本的函数为：

$$\text{Min}\ \sum_{i=0}^{8}\sum_{k=0}^{30}C_{\text{检测配件的费用}}(i,k)+\sum_{k=0}^{30}C_{\text{调换费用}}(k)+\sum_{k=0}^{30}C_{\text{拆解费用k}}(k)+\sum_{k=0}^{30}\sum_{j=0}^{2}C_{\text{零件装配费用}}(j,k)$$

其中：

$C_{\text{检测配件的费用}}(i,k)$表示第k次零件组装售卖过程中在检测时对所拆解出的第 i 个部件进行检测所花费的费用；

$C_{\text{调换费用}}(k)$表示第k次零件组装售卖过程中不合格成品退回的调换费用；

$C_{\text{拆解费用}}(k)$表示第k次零件组装售卖过程中不合格成品的拆解费用；

$C_{\text{零件装配费用}}(j,k)$第k次零件组装售卖过程中，不合格成品拆解后，第 j 次重新装配的费用。

5.3.1.1 检测与装配决策的期望公式

本文中，每个工序或者零件是否检测都会影响着最终的成本和收益，因此我们要通过分析问题得出每一个 生产过程对于总结果的影响从而得出决策的期望公式。经过分析得到的决策的期望公式为：

$$S_5 = n_{i,k}\cdot C_{\text{零配件检测成本}}(i,k)+n_k\cdot p_j\cdot C_{\text{调换成本}}+n_{j,k}\cdot C_{\text{装配成本}}(j,k)$$
$$+n_{\text{调换数量}}(k)\cdot p_{\text{装配}}\cdot C_{\text{调换成本}}$$

其中$n_{i,k}$表示第k次退回的第i个零件的数量，$C_{\text{零配件检测成本}}(i,k)$表示第k次退回的第i个零件的检测成本，n_k表示第k次退回过程中由合格零件能够组装的总的成品数量，p_j表示第j个工序的次品率，$n_{\text{调换数量}}(k)$表示第k次退回的成品数量，$p_{\text{装配}}$表示总的装配的次品率，$C_{\text{成品以及半成品检测}}(i,k)$表示第$k$次退回的第$j$道工序的零件的检测成本。

5.3.1.2 次品率的递减

由于企业的某些决策可能导致零配件被多次组装和销售，而之前销售的零配件会影响后续半成品和成品的合格率，从而影响到流入市场的产品退换货成本，因此需要考虑每次销售后次品率的变化。经过分析可以得到次品率的变化函数为：

$$\hat{p}_i^{(k+1)}=\hat{p}_i^{(k)}\cdot(1-\alpha)$$

其中，$\hat{p}_i^{(k)}$是第 k 次循环的次品率估计，α是递减比例。次品率的递减是一个动态更新过程。

5.3.2 模型的实际求解

（1）初始设定各个零配件等的产品次品率以及每次循环过程中零配件的次品的递减比例。

（2）然后列出影响成本的各个部分的成本求解函数，通过这些函数能够求解出各种决策情况下的生产总成本。

（3）由于某些决策会导致零件多次被组装并售卖因此需要确定某些决策下的最多售卖次数和企业要求的最终次品率标准作为计算成本的终止条件。

（4）输出决策和总成本 记录每种决策所对应的生产最低总成本以及所对应的决策。

通过 python 得到的结果如下：

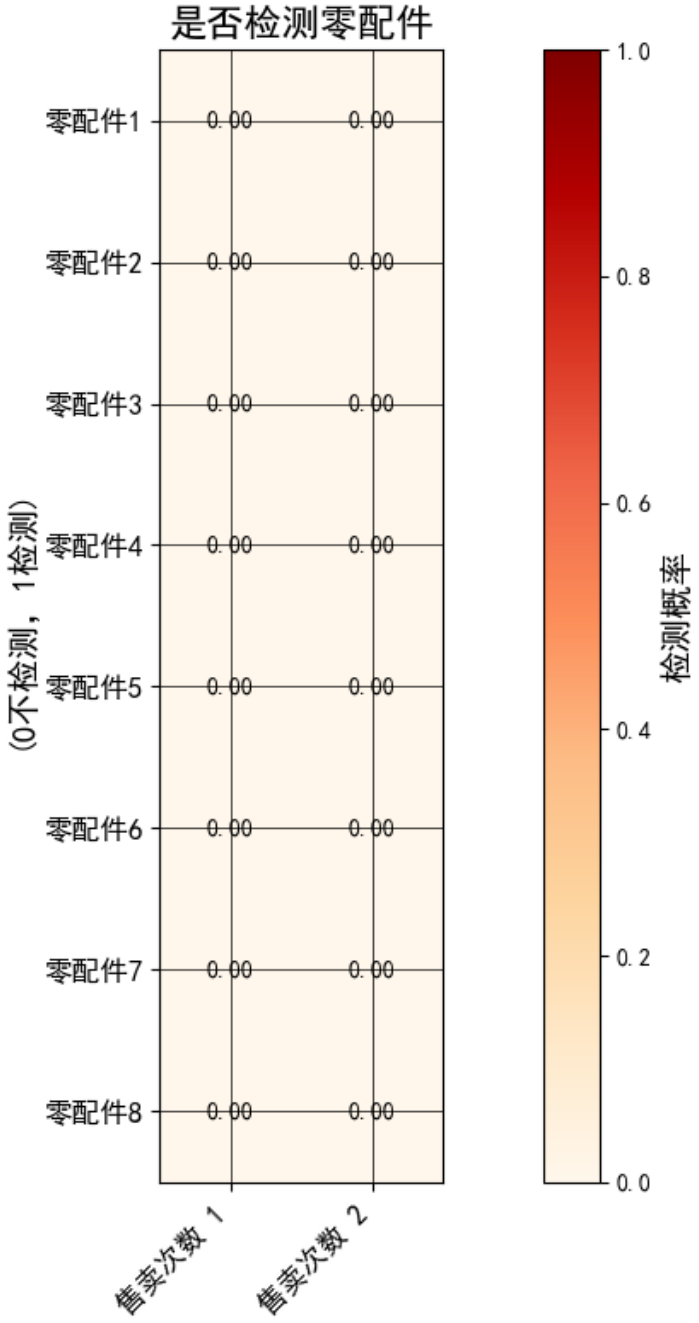


图 4 是否检测零配件可视图

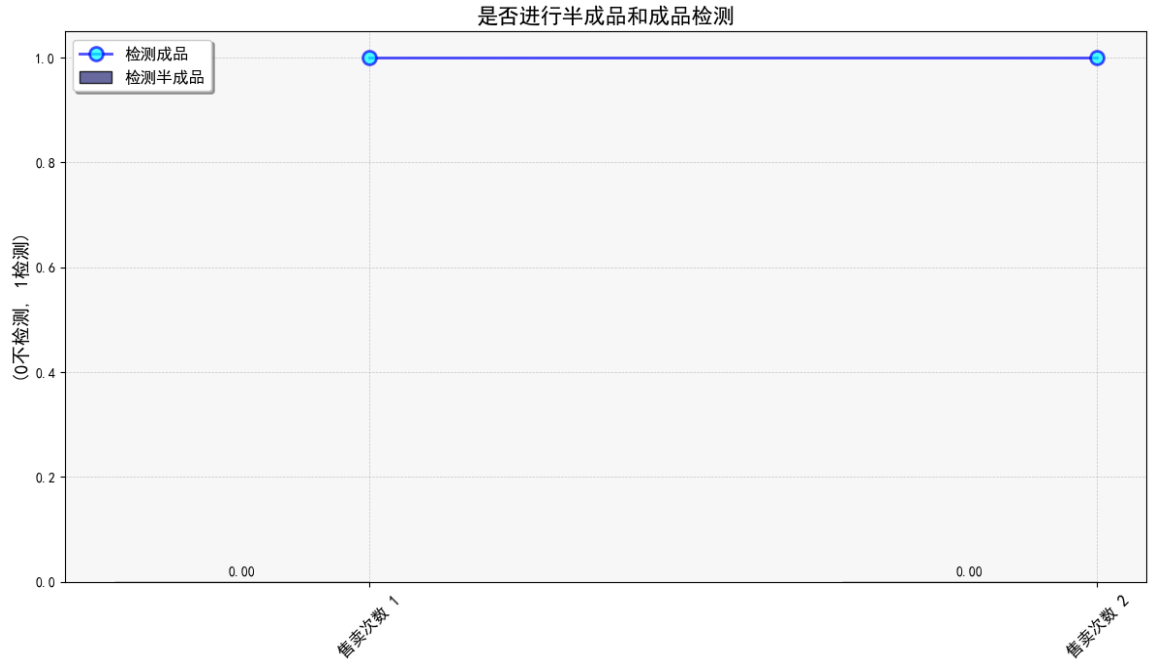


图 5 是否进行半成品和成品检测展示图

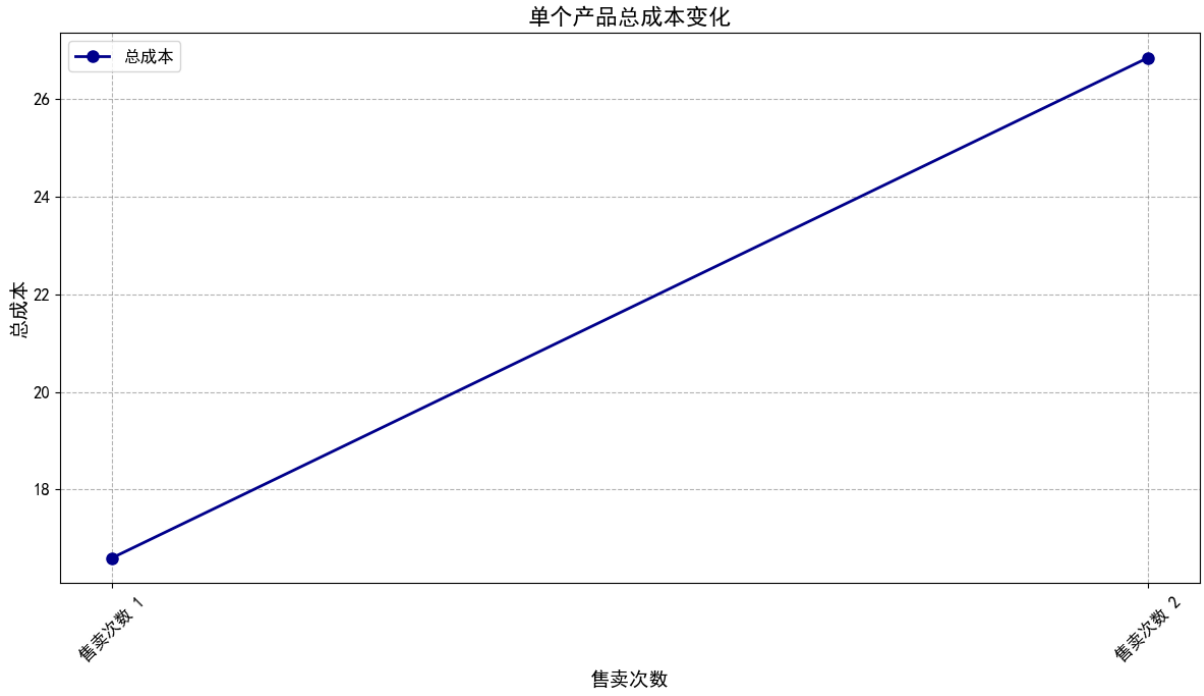


图 6 总成本变化图

5.3.3 结果分析

针对各个零配件的检测：

根据上图所示，在设定的两次成品退回并拆解重装的售卖过程中，若各个零件未经过检测，成品生产成本可能会降低，从而实现更高的利润。这表明可能是由于零件故障

的发生频率较低等原因导致的。

针对半成品以及成品的检测：

根据上图以及求解的数据经分析可得：在设定的两次成品退回并拆解重装的售卖过程中，半成品都被检测所产生得成本较低，这意味着在这种情况下，检测半成品所造成的损失大于不检测所造成的损失，而在成品的检测与否的选择恰恰与半成品的检测与否的选择相反则表明不对成品进行检测所造成的损失是大于检测得损失的，这可能是由于未检测成品所导致的产生的调换损失太大导致的。

针对是否对不合格成品进行拆解的分析：

由第二问知，拆解决策再大多数情况下被启用，进行拆解带来的收益高于拆解成本，因此默认拆解。

5.4 问题四模型的建立与求解

5.4.1 模型建立

在该问中，假设所有的次品率都是通过抽样检测得到的，要求重新考虑前面的问题二和问题三，计算每种决策所对应的成本以及能获得的利润。

5.4.1.1 抽样检测的样品数量确定以及次品率的计算公式

为了保证抽样检测所得到得次品率具有较高的准确率和较小的误差，因此需要先确定抽样的样本量，根据数学知识可以知道样本大小的计算公式为：

$$n = \frac{Z_{\alpha/2} \cdot \hat{p} \cdot (1-\hat{p})}{E^2}$$

其中$Z_{\alpha/2}$是标准正态分布在置信水平在下的临界值，是允许的误差，是初步估计的次品率。这样假设通过抽样检测，我们在选择的样本中检测到的次品数量为 x，则整体的样本次品率可表示为：

$$\hat{p} = \frac{x}{n}$$

在问题三中，次品率的递减是基于固定值的，在本问中，次品率通过抽样检测估计，递减过程变复杂，每次循环次品率减小的值由前一次抽样的次品率估计。

$$\hat{p_i}^{(k+1)} = \hat{p_i}^{(k)} \cdot (1-\alpha)$$

其中，$\hat{p_i}^{(k)}$是第 k 次循环的次品率估计，α是递减比例。次品率的递减是一个动态更新过程。

5.4.1.2 基于问题四的问题对与问题二和问题三建立模型的改进原因

由于在第四问中次品率是通过抽样检测到的，这个值的大小与可信度受检测的样品数量有密切关系，随着检测样品的数量的增多，估计出的样品次品率越能够接近真实值但也会造成检测成本的上升从而影响各个环节的决策，因此要对这一过程产生的成本进

行量化以找到一个较为合适的检测的样本数量的值,既能够很好的反映整体次品率也能够较好的节约成本。

5.4.2 模型求解

5.4.2.1 模型的实际求解

（1）设定每次循环过程中零配件等的次品率的递减比例等参数。

（2）然后列出影响成本的各个部分的成本求解函数，通过这些函数能够求解出各种决策情况下的生产总成本。

（3）由于某些决策会导致零件多次被组装并售卖因此需要确定某些决策下的最多售卖次数和企业要求的最终次品率标准作为计算成本的终止条件。

（4）输出决策和总成本，记录每种决策所对应的生产最低总成本以及所对应的决策。

5.4.2.2 对于问题二的改进

由于是用抽样检测所得到的次品率，因此在多此售卖过程中要对每轮的产品的次品率进行更加准确的更新，每一轮的次品率与之前相比可能有着较大的变化，也要考虑误差的影响。根据数学知识可以知道每次组装售卖零配件和成品的次品率的计算公式为：

$$\hat{p}_i^{(k+1)} = \hat{p}_i^{(k)} \cdot (1-\alpha)$$

其中，$\hat{p}_i^{(k)}$是第 k 次循环的次品率估计，α是递减比例。次品率的递减是一个动态更新过程。

5.4.2.3 求解结果

在这里我们采用第一问所提到的抽样检测的次品率的两种置信水平进行计算：

通过 python 得到的结果如下：

在 90%置信度下的零配件，6 种情况下成品，配件次品率的变化。

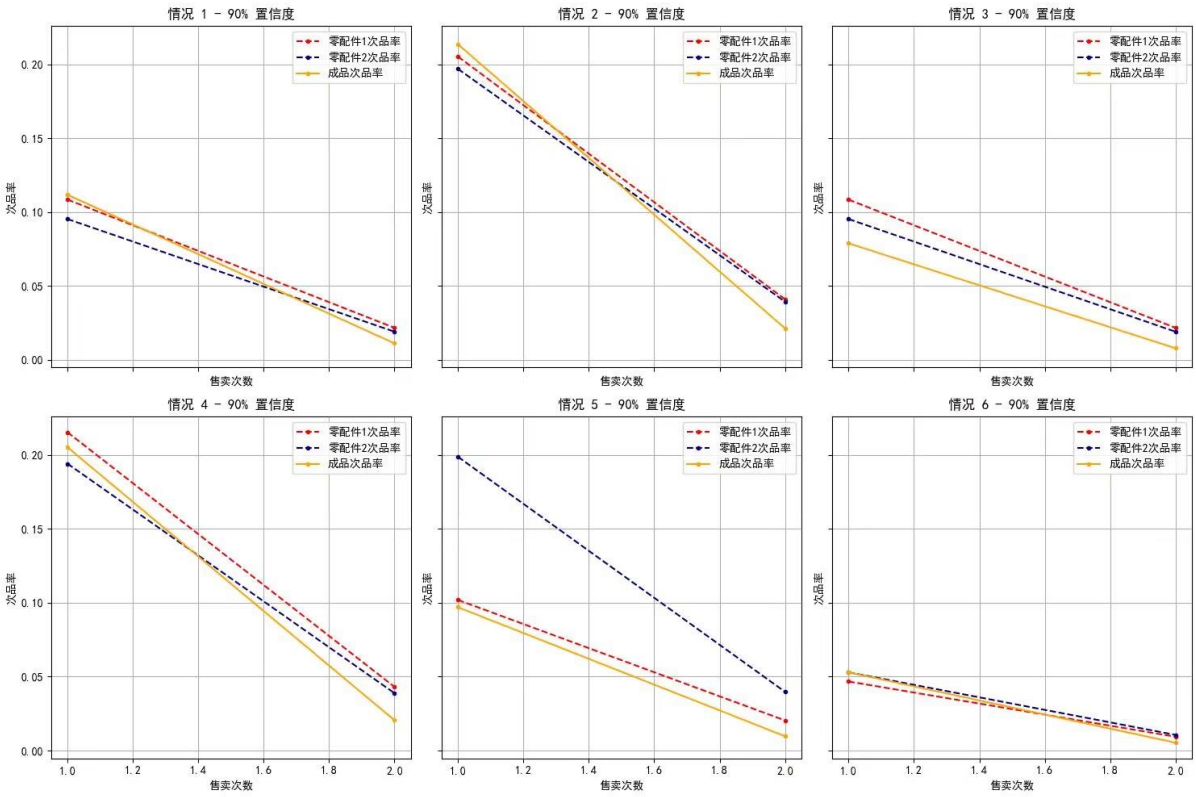


图 7 90%置信度水平下的决策分析结果

表 1 检测及拆解决策表：

	检测零配件一	检测零配件二	检测成品	拆解	平均成本	售卖次数
1	否	否	否	是	30.85	2
2	否	否	否	是	36.25	2
3	否	否	否	是	30.65	2
4	否	否	否	是	33.23	2
5	否	否	否	是	31.59	2
6	否	否	否	否	4.70	2

在 95%置信度下的零配件，6 种情况下成品，配件次品率的变化：

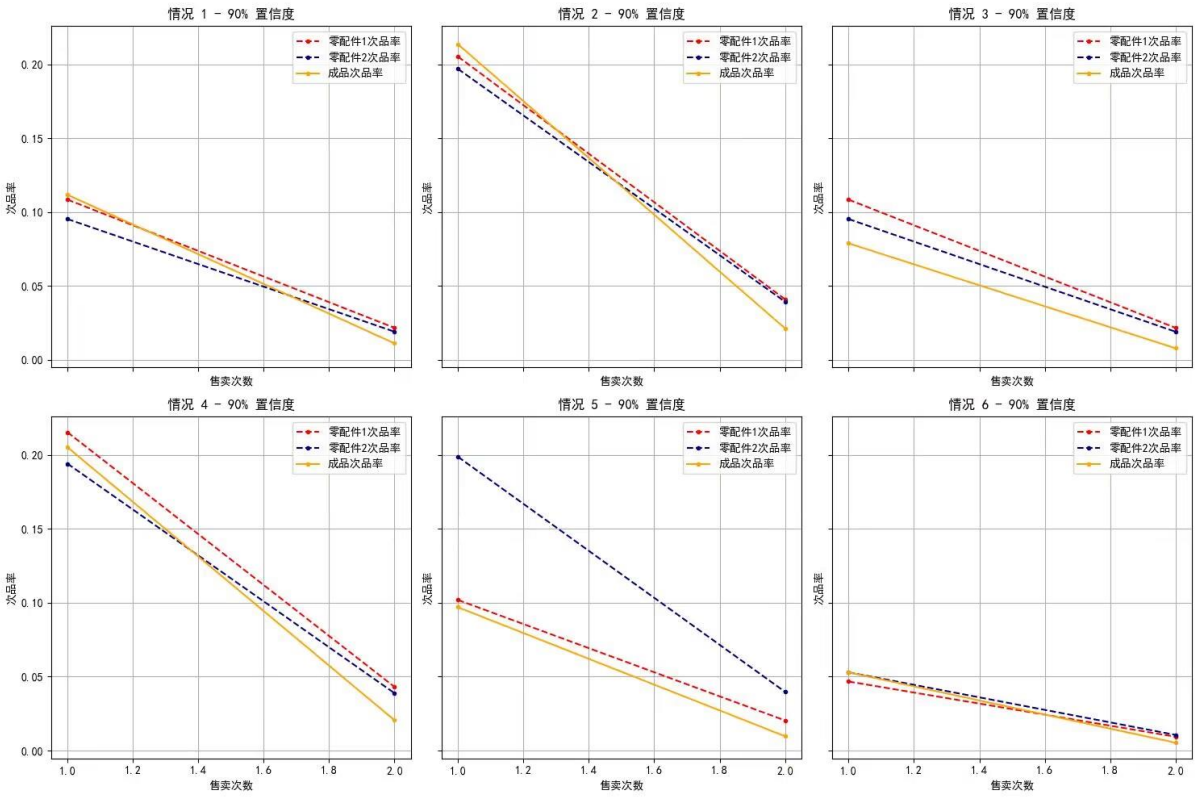


图 8 95%置信度水平下的决策分析结果

表 2 检测及拆解决策表：

	检测零配件一	检测零配件二	检测成品	拆解	平均成本	售卖次数
1	否	否	否	是	31.03	2
2	否	否	否	是	36.38	2
3	否	否	否	是	30.89	2
4	否	否	否	是	32.94	2
5	否	否	否	是	31.55	2
6	否	否	否	否	4.74	2

5.4.2.4 结果分析

95%的置信水平下的决策：

针对各个零配件和成品的检测：由计算得到的数据和图可得在大多数情况下并没有对零配件进行检测，这说明在 95%的置信水平下，对于零件的检测成本较高导致不检测反而都能够节约成本。而对于成品，有多种情况都选择检测则说明成品的次品率对于成本的影响比较大因此为了节约成本，要对成品进行检测。

针对是否拆解的决策：由计算得到的数据和图可得在大多数情况下，企业选择对于不合

格成品进行拆解能够获得更小的生产成本。

在 90%的置信水平下的决策：

针对各个零配件和成品的检测：在对零配件的检测的决策中，由计算得到的数据和图可得在同在 95%的置信水平下的决策类似，在 90%的置信水平下依然是在大多数情况下未对零配件进行检测，所得出的是否对零配件进行检测的结论与在 90%的置信度水平的结论相似，通过观察数据也可以看到，两种情况下，对成品的检验两种情况也相似。

针对是否拆解的决策：

由计算得到的数据和图可得在同在 95%的置信水平下的决策类似，在 90%的置信水平下依然是选择对不合格成品进行拆解能够节约更多的成本。

两种信度对结果的影响分析：

由计算得到的数据和图经分析可以发现在两种信度水平下系统做出的决策类似，差异较小。而在 90%置信水平下最后的总成本偏低则说明在这种情况下，当企业在放宽信度要求时，可能能够节约更多的成本。

5.4.2.5 对于问题三的改进

同问题二一样，由于是通过抽样检测得到的次品率，因此在问题三中每次组装售卖零配件、半成品、成品的次品率的计算公式也为：

$$\hat{p}_i^{(k+1)} = \hat{p}_i^{(k)} \cdot (1-\alpha)$$

其中，$\hat{p}_i^{(k)}$是第 k 次循环的次品率估计，α是递减比例。次品率的递减是一个动态更新过程。

5.4.2.6 求解结果

在这里我们依然采用第一问所提到的抽样检测的次品率的两种置信水平进行计算：

通过 python 得到的结果如下：

在 95%的置信水平下，在设定的情况下生产样品的平均成本是 29.08，在设定的总的售卖次数受最大循环次数和设定的阈值的限制下最多售卖次数是 4 次。

表 3：在 95%置信水平下配件，半成品，成品检测决策情况：

售卖次数	零配件 1	零配件 2	零配件 3	零配件 4	零配件 5	零配件 6	零配件 7	零配件 8	半成品 1	半成品 2	半成品 3	成品
1	否	否	否	否	否	否	否	否	否	否	否	是
2	否	否	否	否	否	否	否	否	否	否	否	否
3	否	否	否	否	否	否	否	否	否	否	否	否
4	否	否	否	否	否	否	否	否	否	否	否	否

在 90%的置信水平下，在设定的情况下生产样品的平均成本是 29.14 ，在设定的总的售卖次数受最大循环次数和设定的阈值的限制下最多售卖次数是 4 次。

售卖次数	零配件 1	零配件 2	零配件 3	零配件 4	零配件 5	零配件 6	零配件 7	零配件 8	半成品 1	半成品 2	半 成品 3	成品
1	否	否	否	否	否	否	否	否	否	否	否	是
2	否	否	否	否	否	否	否	否	否	否	否	是
3	否	否	否	否	否	否	否	否	否	否	否	否
4	否	否	否	否	否	否	否	否	否	否	否	否

5.4.2.7 结果分析

在 95%的置信水平下的决策：

针对各个零配件，半成品以及成品的检测：在对零配件的检测的决策中，由计算得到的数据和图可得所选取的情况下都没有对零配件进行检测，这说明在这种情况中，成品的次品率不高导致检测的成本反而高于不检测造成的损失。在对半成品是否进行检测的决策中可以发现系统未打算对半成品进行检测，得到的结论与对零配件进行检测的结论类似，都是检测的成本高于不检测造成的损失。而对成品的检测的选择中可以发现系统在初始几次售卖过程中选择了对成品进行检测，说明在这个置信度水平下，在最初的几次售卖过程种系统倾向于对成品进行检测以节约成本。

在 90%的置信水平下的决策：

针对各个零配件，半成品以及成品的检测：在对零配件、半成品和成品的检测决策中，计算得到的数据和图表显示，在 90%的置信水平下，系统的决策在零配件的检测上类似，均未选择对其进行检测。这可能表明，在较高置信水平下，不检测零配件有助于更好地控制成本。对于半成品的检测，决策与前一个置信水平下类似，说明也存在相同的问题。而在成品的检测中，90%的置信水平下仅在售卖过程的少数几次中中选择检测成品，表明随着售卖次数的增加，成品的次品率下降，因此不进行检测可以节约更多成本。

两种信度水平下的对结果的对比分析：

通过上述结果和分析可以看出，在两种信度条件下做出的决策类似，主要的区别在于对成品检测的选择。这表明，在较高信度条件下，系统对零配件、半成品和成品的检验要求是相似的。

六、 模型的评价、改进与推广

6.1 模型的优点

（1）该模型采用动态检验方法，具有动态性、准确性的优点，实现对各阶段次品率

的动态估计，因此该模型更符合实际情况。

（2）该模型综合考虑各个零配件、半成品、成品的次品率，以及检测成本等因素，对各个阶段给出详细的策略，实现企业的利润最大化。

（3）该模型中使用的算法具有考虑方面细致、优化能力强等优点，可推广至其他产品的决策优化场景。

6.2 模型的缺点

（1）该模型中假设抽样检测过程中的次品数量符合二项分布，而实际情况中次品数量并不是严格符合，在一定程度上影响模型的准确性。

（2）该模型没有考虑生产零件、检测零件以及零件调换的各个生产阶段中可能会发生设备故障、生产事故等状况。

6.3 模型的推广与改进

（1）对设备发生故障情况进行统计，将设备发生故障概率综合考虑进模型中，可以完善该模型，进一步优化生产过程决策。

（2）该模型可以推广到其他类型产品加工过程，不少其他企业担当“集成制造商”的角色，该模型实现了对各种零部件、半成品及成品是否进行抽样检验的决策优化，可以为企业带来最大利润，增加收益。

（3）该模型对其他类型抽样方法（如分层抽样）同样具有适应性，因此在满足其他类型的分布特征的检验中可以使用该模型。

七、 参考文献

[1]孙小素,尚书钰.一种新型可靠性抽样设计方法——基于经济优化视角[J].统计学报,2024,5(03):76-94.DOI:10.19820/j.cnki.ISSN2096-7411.2024.03.007.

[2]企业产品标准化生产认定检测抽样方案[C]//中国缝制机械协会.中国缝制机械协会八届三次理事会论文集.[出版者不详],2008:1.

[3]王淼,孙晓峰.基于二项分布的优化序贯截尾检验方法分析计算[J].海军航空工程学院学报,2013,28(04):421-424.

[4]裘镓荣.产品装配质量控制及预测方法研究[D].沈阳理工大学,2023.DOI:10.27323/d.cnki.gsgyc.2023.000566.

[5]彭钊.财务分析为企业经营决策提供参考的措施探析[J].商展经济,2024,(11):173-176.DOI:10.19995/j.cnki.CN10-1617/F7.2024.11.173.

[6]郑宽宽,谭激扬.企业生产的动态控制和优化[J].运筹与管理,2024,33(07):228-233

附录

附录 1 第一问.py

介绍：该代码是 python 语言编写的，解决问题一

```
import os
import sys
import subprocess
from scipy.stats import binom
import warnings

sys.stderr = open(os.devnull, 'w')
warnings.filterwarnings("ignore")

def install_packages(packages, mirror):
    for package in packages:
        subprocess.check_call([sys.executable, '-m', 'pip', 'install', package, '-i', mirror])

install_packages(['seaborn', 'scipy'], 'https://pypi.tuna.tsinghua.edu.cn/simple')

zhixinshuiping = [0.95, 0.90]
jaincelv = 0.10

def jisuanlv(sample_size, zhixinshuiping, jaincelv):
    jieshoushuiping = binom.ppf(1-zhixinshuiping[1], sample_size, jaincelv)
    jujueshuiping = binom.ppf(zhixinshuiping[0], sample_size, jaincelv)
    return jieshoushuiping, jujueshuiping

yangbendaxiao1 = 100
yangbendaxiao2, jujuelv22 = jisuanlv(yangbendaxiao1, zhixinshuiping, jaincelv)

print("第一次检测：")
```

```
print(f"如果检测到次品数量大于{jujuelv22:.0f},则拒收")
print(f"如果检测到次品数量小于{yangbendaxiao2:.0f},则接收")

second_sample_size = 350
second_accept_threshold, second_reject_threshold = jisuanlv(second_sample_size, zhixinshuiping, jaincelv)

print("\n 第二次检测： ")
print(f"如果检测到次品数量大于{second_reject_threshold:.0f},则拒收")
print(f"如果检测到次品数量小于{second_accept_threshold:.0f},则接收")
```

附录 2 第二问.py

介绍：该代码是 python 语言编写的，解决问题二

```
import numpy as np
import matplotlib.pyplot as plt
from itertools import product

from matplotlib import rcParams
rcParams['font.sans-serif'] = ['SimHei']
rcParams['axes.unicode_minus'] = False

cases = [
    {'零配件 1 次品率': 0.1, '零配件 1 购买单价': 4, '零配件 1 检测成本': 2, '零配件 2 次品率': 0.1, '零配件 2 购买单价': 18,
     '零配件 2 检测成本': 3, '成品次品率': 0.1, '成品检测成本': 3, '装配成本': 6, '市场售价': 56, '调换损失': 6, '拆解费用': 5},
    {'零配件 1 次品率': 0.2, '零配件 1 购买单价': 4, '零配件 1 检测成本': 2, '零配件 2
```

```
次品率': 0.2, '零配件 2 购买单价': 18,
    '零配件 2 检测成本': 3, '成品次品率': 0.2, '成品检测成本': 3, '装配成本': 6, '市场售价': 56, '调换损失': 6, '拆解费用': 5},
    {'零配件 1 次品率': 0.1, '零配件 1 购买单价': 4, '零配件 1 检测成本': 2, '零配件 2 次品率': 0.1, '零配件 2 购买单价': 18,
    '零配件 2 检测成本': 3, '成品次品率': 0.1, '成品检测成本': 3, '装配成本': 6, '市场售价': 56, '调换损失': 30, '拆解费用': 5},
    {'零配件 1 次品率': 0.2, '零配件 1 购买单价': 4, '零配件 1 检测成本': 1, '零配件 2 次品率': 0.2, '零配件 2 购买单价': 18,
    '零配件 2 检测成本': 1, '成品次品率': 0.2, '成品检测成本': 2, '装配成本': 6, '市场售价': 56, '调换损失': 30, '拆解费用': 5},
    {'零配件 1 次品率': 0.1, '零配件 1 购买单价': 4, '零配件 1 检测成本': 8, '零配件 2 次品率': 0.2, '零配件 2 购买单价': 18,
    '零配件 2 检测成本': 1, '成品次品率': 0.1, '成品检测成本': 2, '装配成本': 6, '市场售价': 56, '调换损失': 10, '拆解费用': 5},
    {'零配件 1 次品率': 0.05, '零配件 1 购买单价': 4, '零配件 1 检测成本': 2, '零配件 2 次品率': 0.05, '零配件 2 购买单价': 18,
    '零配件 2 检测成本': 3, '成品次品率': 0.05, '成品检测成本': 3, '装配成本': 6, '市场售价': 56, '调换损失': 10, '拆解费用': 40}
]

# 计算总成本、合格率和调换成本的函数
def zongjisuan(case, detect_parts1, detect_parts2, detect_final, dismantle):
    N = 100  # 固定生产的成品数

    # 购买零配件数量
    peijian1 = N / (1 - case['零配件 1 次品率'] * (1 - detect_parts1))
    peijian2 = N / (1 - case['零配件 2 次品率'] * (1 - detect_parts2))

    # 成本 1：购买成本
    cost_purchase = peijian1 * case['零配件 1 购买单价'] + peijian2 * case['零配件 2 购买单价']

    # 成本 2：零配件 1 和零配件 2 检测成本
```

```
    jiancechengben1 = peijian1 * case['零配件 1 检测成本'] * detect_parts1
    jiancechengben2 = peijian2 * case['零配件 2 检测成本'] * detect_parts2

    # 成本 3: 装配成本
    zhuangpeichengben = N * case['装配成本']

    # 成本 4: 成品检测成本
    jiancechengben3 = N * case['成品检测成本'] * detect_final

    # 成本 5: 调换成本
    diaohuanchengben = (1 - detect_final) * N * case['成品次品率'] * case['调换损失']

    # 成本 6: 拆解成本
    chaijiechengben = N * case['成品次品率'] * dismantle * (case['拆解费用'] - 0.6 * case['市场售价'])

    # 总成本
    zongchengben = cost_purchase + jiancechengben1 + jiancechengben2 + zhuangpeichengben + jiancechengben3 + diaohuanchengben + chaijiechengben

    # 合格率
    cipinlv1 = 1 - case['零配件 1 次品率'] * (1 - detect_parts1)
    cipinlv2 = 1 - case['零配件 2 次品率'] * (1 - detect_parts2)
    cipinlv3 = cipinlv1 * cipinlv2 * (1 - case['成品次品率'] * (1 - detect_final))

    return zongchengben, cipinlv3, diaohuanchengben

# 生成策略
strategies = list(product([1, 0], repeat=4))  # (detect_parts1, detect_parts2, detect_final, dismantle)

def celuejieguoshuchu(celue1):
    detect_parts1, detect_parts2, detect_final, dismantle = strategies[celue1]
```

```
    explanation = f"策略 {celue1 + 1}: "
    explanation += "检测零配件 1" if detect_parts1 else "不检测零配件 1"
    explanation += "，检测零配件 2" if detect_parts2 else "，不检测零配件 2"
    explanation += "，检测成品" if detect_final else "，不检测成品"
    explanation += "，拆解不合格成品" if dismantle else "，不拆解不合格成品"
    return explanation

costs = []
defective_rates = []
exchanges = []

for case_idx, case in enumerate(cases):
    print(f"\n 情况 {case_idx + 1}:")

    qingkuanghuafei = []
    qingkuangcipinlv = []
    jiaohuanqingkuang = []

    for celue1, strategy in enumerate(strategies):
        zongchengben, cipinlv3, exchange = zongjisuan(case, *strategy)

        qingkuanghuafei.append(zongchengben)
        qingkuangcipinlv.append(cipinlv3)
        jiaohuanqingkuang.append(exchange)

        print(f"{celuejieguoshuchu(celue1)}: 总成本 = {zongchengben:.2f}, 调换成
本 = {exchange:.2f}, 成品合格率 = {cipinlv3:.2f}")

    costs.append(qingkuanghuafei)
    defective_rates.append(qingkuangcipinlv)
    exchanges.append(jiaohuanqingkuang)

costs = np.array(costs)
defective_rates = np.array(defective_rates)
```

```
exchanges = np.array(exchanges)

plt.figure(figsize=(15, 8))
colors = plt.cm.viridis(np.linspace(0, 1, costs.shape[1]))  # 使用渐变色
for i in range(costs.shape[1]):
    plt.plot(range(1, costs.shape[0] + 1), costs[:, i], marker='o', linestyle='-.', linewidth='2',
             color=colors[i], label=f"策略 {i + 1} - 总成本", alpha=0.8)

plt.xlabel("情况", fontsize=14, fontweight='bold')
plt.ylabel("总成本", fontsize=14, fontweight='bold')
plt.title("不同情况和策略下的成本比较", fontsize=16, fontweight='bold')
plt.legend(loc='best', fontsize=12)
plt.grid(True, which='both', linestyle='--', linewidth=0.7)
plt.tight_layout()
plt.show()
```

附录 3 第三问.py

介绍：该代码是 python 语言编写的，解决问题三

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from matplotlib.colors import LinearSegmentedColormap

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

shoumaicishu = 10
```

```
konu = 0.02    #次品率阈值
cipinlvjiangdi = 0.6

# 计算总成本函数
def jisuanzongchengben (df, defects, detection_col, loss_func):
    zonghuafei1 = 0
    juece = [] #决策方案

    for i in range(len(df)):
        if df.loc[i, '检测成本'] < loss_func(i, defects):
            zonghuafei1 += df.loc[i, '检测成本']
            juece.append(1)  # 1 表示检测
        else:
            zonghuafei1 += loss_func(i, defects)
            juece.append(0)  # 0 表示不检测
    return zonghuafei1, juece

# 计算零配件、半成品和成品的总成本
def jisuanzonghuafei11(bujia1, df_halfparts, df_final, shoumaicishu=shoumaicishu,
cipinlvjiangdi=cipinlvjiangdi, konu=konu):
    zonghuafei1 = 0
    cycle = 0  # 售卖次数

    jaince11 = bujia1['次品率'].copy()
    halfjaince11 = df_halfparts['次品率'].copy()
    zongjaince11 = df_final['次品率'][0]

    juece = {'售卖次数': [], '零配件检测决策': [], '半成品检测决策': [], '成品检测决策
': []}

    while cycle < shoumaicishu and (jaince11.max() > konu or halfjaince11.max() >
konu or zongjaince11 > konu):
        cycle += 1
```

```
        # 计算零配件的检测成本
        lingpeijianjiance1dd = lambda i, defects: defects[i] * bujia1.loc[i, '购买单价']
        bufenhaufeia, bujia2 = jisuanzongchengben(bujia1, jaince11, '检测成本',
lingpeijianjiance1dd)
        zonghuafei1 += bufenhaufeia

        # 计算半成品的检测成本
        banchengben55 = lambda i, defects: defects[i] * (df_halfparts.loc[i, '装配成本
'] + df_halfparts.loc[i, '拆解费用'])
        banchengben66, halfpart_juece = jisuanzongchengben(df_halfparts,
halfjaince11, '检测成本', banchengben55)
        zonghuafei1 += banchengben66

        # 计算成品的检测成本
        zongdajianchegben = zongjaince11 * (df_final.loc[0, '装配成本'] +
df_final.loc[0, '市场售价'] + df_final.loc[0, '拆解费用'])
        if df_final.loc[0, '检测成本'] < zongdajianchegben:
            zonghuafei1 += df_final.loc[0, '检测成本']
            wupinjuece = 1
        else:
            zonghuafei1 += zongdajianchegben
            wupinjuece = 0


        juece['售卖次数'].append(cycle)
        juece['零配件检测决策'].append(bujia2)
        juece['半成品检测决策'].append(halfpart_juece)
        juece['成品检测决策'].append(wupinjuece)


        jaince11 *= (1 - cipinlvjiangdi)
        halfjaince11 *= (1 - cipinlvjiangdi)
        zongjaince11 *= (1 - cipinlvjiangdi)
```

```
    return zonghuafei1, cycle, juece

data1_0parts = {
    '零配件': ['零配件 1', '零配件 2', '零配件 3', '零配件 4', '零配件 5', '零配件 6', '零配件 7', '零配件 8'],
    '次品率': [0.10, 0.10, 0.10, 0.10, 0.10, 0.10, 0.10, 0.10],
    '购买单价': [2, 8, 12, 2, 8, 12, 8, 12],
    '检测成本': [1, 1, 2, 1, 1, 2, 1, 2]
}

data2_halfparts = {
    '半成品': ['半成品 1', '半成品 2', '半成品 3'],
    '次品率': [0.10, 0.10, 0.10],
    '装配成本': [8, 8, 8],
    '检测成本': [4, 4, 4],
    '拆解费用': [6, 6, 6]
}

data3_finalproducts = {
    '成品': ['成品'],
    '次品率': [0.10],
    '装配成本': [8],
    '检测成本': [6],
    '拆解费用': [10],
    '市场售价': [200],
    '调换损失': [40]
}

shuchuges1 = pd.DataFrame(data1_0parts)
shuchuges2 = pd.DataFrame(data2_halfparts)
shuchuges3 = pd.DataFrame(data3_finalproducts)

# 计算总成本
zonghuafei1, cycle_count, decision_results = jisuanzonghuafei11(shuchuges1,
```

```
shuchuges2, shuchuges3)

# 输出结果
print(f"单个成品总成本: {zonghuafei1:.2f}")
print(f"总售卖次数: {cycle_count}")
for cycle in range(cycle_count):
    print(f"第 {decision_results['售卖次数'][cycle]} 次循环： ")
    print(f"  是否检测零配件: {decision_results['零配件检测决策'][cycle]}")
    print(f"  是否检测半成品: {decision_results['半成品检测决策'][cycle]}")
    print(f"  是否检测成品: {decision_results['成品检测决策'][cycle]}")

def ziongjueceshuchu11(bujia2, bujia1, cycles_labels):
    fig, ax = plt.subplots(figsize=(12, 8))

    # 使用'coolwarm'色图，并设置合适的色标范围
    cmap = plt.get_cmap('OrRd')
    im = ax.imshow(bujia2.T, cmap=cmap, vmin=0, vmax=1)

    # 添加标题和标签
    ax.set_title("是否检测零配件", fontsize=16, fontweight='bold')
    ax.set_xticks(np.arange(len(cycles_labels)))
        ax.set_xticklabels(cycles_labels,  rotation=45,  ha='right',  fontsize=12,
fontweight='bold')
    ax.set_yticks(np.arange(len(bujia1)))
    ax.set_yticklabels(bujia1['零配件'], fontsize=12, fontweight='bold')
    ax.set_ylabel('(0 不检测，1 检测)', fontsize=14, fontweight='bold')



    for i in range(len(bujia1)):
        for j in range(len(bujia2)):
            value = bujia2[j, i]
            text = ax.text(j, i, f'{value:.2f}',
                           ha="center", va="center", color="white" if value > 0.5 else
"black",
```

```
                fontsize=10, fontweight='bold')


    cbar = fig.colorbar(im, ax=ax, orientation='vertical')
    cbar.set_label('检测概率', fontsize=14, fontweight='bold')


    ax.grid(which='both', color='black', linestyle='-', linewidth=0.5)

    plt.tight_layout()
    plt.show()

import matplotlib.pyplot as plt
import numpy as np

def ziongjueceshuchu22(banpart_juece_juzhen, final_product_juece, cycle_labels):
    fig, ax = plt.subplots(figsize=(12, 7))
    x = np.arange(len(cycle_labels))
    width = 0.35

    rects1 = ax.bar(x - width/2, np.sum(banpart_juece_juzhen, axis=1), width,
                    color=plt.cm.viridis(np.linspace(0.2, 0.8, len(cycle_labels))),
                    edgecolor='black', alpha=0.8, label='检测半成品')


    ax.plot(x, final_product_juece, color='blue', marker='o', markersize=10,
            markerfacecolor='cyan', markeredgewidth=2, markeredgecolor='blue',
            linestyle='-', linewidth=2, alpha=0.7, label='检测成品')

    ax.set_ylabel('(0 不检测, 1 检测)', fontsize=14, fontweight='bold')
    ax.set_title('是否进行半成品和成品检测', fontsize=16, fontweight='bold')
    ax.set_xticks(x)
    ax.set_xticklabels(cycle_labels, rotation=45, fontsize=12)
```

```
    ax.grid(True, which='major', linestyle='--', linewidth=0.5, alpha=0.7)
    ax.set_facecolor('#f7f7f7')
        ax.legend(loc='upper left', fontsize=12, frameon=True, shadow=True, fancybox=True)

    for rect in rects1:
        height = rect.get_height()
        ax.annotate(f'{height:.2f}',
                    xy=(rect.get_x() + rect.get_width() / 2, height),
                    xytext=(0, 3),
                    textcoords="offset points",
                    ha='center', va='bottom', fontsize=10, color='black')

    plt.tight_layout()
    plt.show()

def ziongjueceshuchu33(bujia1, df_halfparts, df_final, cycle_count):
    fig, ax = plt.subplots(figsize=(12, 7))

    # 计算每个周期的总成本
        cost_per_cycle = [jisuanzonghuafei11(bujia1, df_halfparts, df_final, shoumaicishu=i+1)[0] for i in range(cycle_count)]


     ax.plot([f"售卖次数 {i+1}" for i in range(cycle_count)], cost_per_cycle, marker='o', color='darkblue', linestyle='-', linewidth=2, markersize=8, label='单个产品总成本')

    ax.set_title('单个产品总成本变化', fontsize=16, fontweight='bold')
    ax.set_xlabel('售卖次数', fontsize=14, fontweight='bold')
    ax.set_ylabel('单个成品总成本', fontsize=14, fontweight='bold')
```

```
    ax.grid(True, linestyle='--', color='gray', alpha=0.6)



    ax.legend(fontsize=12)

    plt.xticks(fontsize=12, rotation=45)
    plt.yticks(fontsize=12)
    plt.tight_layout()

    plt.show()

part_juece_juzhen = np.array(decision_results['零配件检测决策'])
banpart_juece_juzhen = np.array(decision_results['半成品检测决策'])
final_product_juece = decision_results['成品检测决策']

cycle_labels = [f"售卖次数 {i+1}" for i in range(cycle_count)]

ziongjueceshuchu11(part_juece_juzhen, shuchuges1, cycle_labels)
ziongjueceshuchu22(banpart_juece_juzhen, final_product_juece, cycle_labels)
ziongjueceshuchu33(shuchuges1, shuchuges2, shuchuges3, cycle_count)
```

附录 4 第四问 1.py

介绍：该代码是 python 语言编写的，解决问题四第一问

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm

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

data = {
```

```
    '零配件 1 次品率': [0.10, 0.20, 0.10, 0.20, 0.10, 0.05],
    '零配件 1 购买单价': [4, 4, 4, 4, 4, 4],
    '零配件 1 检测成本': [2, 2, 2, 1, 8, 2],
    '零配件 2 次品率': [0.10, 0.20, 0.10, 0.20, 0.20, 0.05],
    '零配件 2 购买单价': [18, 18, 18, 18, 18, 18],
    '零配件 2 检测成本': [3, 3, 3, 1, 1, 3],
    '成品次品率': [0.10, 0.20, 0.10, 0.20, 0.10, 0.05],
    '成品装配成本': [6, 6, 6, 6, 6, 6],
    '成品检测成本': [3, 3, 3, 2, 2, 3],
    '市场售价': [56, 56, 56, 56, 56, 56],
    '调换损失': [6, 6, 30, 30, 10, 10],
    '拆解费用': [5, 5, 5, 5, 5, 40]
}

df = pd.DataFrame(data)

xunhuancishu = 10  # 最多售卖次数
yuzhi = 0.01  # 次品率阈值
dijainbili1 = 0.8  # 零配件 1 次品率递减比例
dijainbili2 = 0.8  # 零配件 2 次品率递减比例
dijainbili3 = 0.9  # 成品次品率递减比例
yunxuwucha = 0.02  # 允许的误差

# 置信度 Z 值
zhixinzhi = {
    95: norm.ppf(0.975),
    90: norm.ppf(0.95)
}

# 抽样检测次品率估计函数
def cipinlvgujizhi(p_estimate, confidence, wucha1=yunxuwucha):
    zhixindu1 = zhixinzhi[confidence]
    yangbenshuliang = (zhixindu1**2 * p_estimate * (1 - p_estimate)) / (wucha1**2)
      yangbencipinlv = np.random.binomial(int(yangbenshuliang), p_estimate) /
```

```
yangbenshuliang
    return yangbencipinlv

# 决策分析函数
def juece(row, confidence=95):
    zhixindu1 = zhixinzhi[confidence]



    cipinlvguji11 = cipinlvgujizhi(row['零配件 1 次品率'], confidence)
    cipinlvguji22 = cipinlvgujizhi(row['零配件 2 次品率'], confidence)
    cipinlvguji33 = cipinlvgujizhi(row['成品次品率'], confidence)

    costs = {
        '零配件 1': (row['零配件 1 购买单价'], row['零配件 1 检测成本']),
        '零配件 2': (row['零配件 2 购买单价'], row['零配件 2 检测成本']),
        '成品': (row['成品装配成本'], row['市场售价'], row['成品检测成本']),
        '拆解': (row['调换损失'], row['拆解费用'])
    }

    totalcost1, cycles = 0, 0
    cipinlishi = {'零配件 1': [], '零配件 2': [], '成品': []}

    while cycles < xunhuancishu and any(rate > yuzhi for rate in [cipinlvguji11, cipinlvguji22, cipinlvguji33]):
        cycles += 1

        cipinlishi['零配件 1'].append(cipinlvguji11)
        cipinlishi['零配件 2'].append(cipinlvguji22)
        cipinlishi['成品'].append(cipinlvguji33)

        # 计算零配件与成品的检测或替换成本
        jaincetihuanchengben1 = min(costs['零配件 1'][1], cipinlvguji11 * costs['零配件 1'][0])
        jaincetihuanchengben2 = min(costs['零配件 2'][1], cipinlvguji22 * costs['零配
```

```
件 2'][0])
        product_cost = min(costs['成品'][2], cipinlvguji33 * (costs['成品'][0] + costs['
成品'][1]))

            shoumaihaufei = jaincetihuanchengben1 + jaincetihuanchengben2 +
product_cost

        # 如果需要拆解，计算额外成本
        if costs['拆解'][1] < costs['拆解'][0]:
            shoumaihaufei += costs['拆解'][1] + cipinlvguji11 * costs['零配件 1'][0] +
cipinlvguji22 * costs['零配件 2'][0] + costs['成品'][0]

        totalcost1 += shoumaihaufei

        # 次品率递减
        cipinlvguji11 *= (1 - dijainbili1)
        cipinlvguji22 *= (1 - dijainbili2)
        cipinlvguji33 *= (1 - dijainbili3)

    return {
        '检测零配件 1': costs['零配件 1'][1] < cipinlvguji11 * costs['零配件 1'][0],
        '检测零配件 2': costs['零配件 2'][1] < cipinlvguji22 * costs['零配件 2'][0],
        '检测成品': costs['成品'][2] < cipinlvguji33 * (costs['成品'][0] + costs['成品
'][1]),
        '拆解': costs['拆解'][1] < costs['拆解'][0],
        '总成本': totalcost1,
        '售卖次数': cycles,
        '零配件 1 次品率历史': cipinlishi['零配件 1'],
        '零配件 2 次品率历史': cipinlishi['零配件 2'],
        '成品次品率历史': cipinlishi['成品']
    }

# 对所有情况进行决策分析
zhixin95 = df.apply(juece, axis=1, confidence=95)
```

```
zhixin90 = df.apply(juece, axis=1, confidence=90)

biaozhun95 = pd.DataFrame(zhixin95.tolist())
biaozhun90 = pd.DataFrame(zhixin90.tolist())
biaozhun1 = pd.DataFrame(zhixin95.tolist(), columns=['检测零配件 1', '检测零配件 2', '检测成品', '拆解', '总成本', '售卖次数'])
biaozhun2 = pd.DataFrame(zhixin90.tolist(), columns=['检测零配件 1', '检测零配件 2', '检测成品', '拆解', '总成本', '售卖次数'])

print("\n" + "=" * 50)
print("95% 置信度下的决策分析结果：")
print("=" * 50)
print(biaozhun1[['检测零配件 1', '检测零配件 2', '检测成品', '拆解', '总成本', '售卖次数']])
print("=" * 50)

print("\n" + "=" * 50)
print("90% 置信度下的决策分析结果：")
print("=" * 50)
print(biaozhun2[['检测零配件 1', '检测零配件 2', '检测成品', '拆解', '总成本', '售卖次数']])
print("=" * 50)


def lishishujukeshihua(df_results, confidence_levels):
    num_plots = len(df_results)
    cols = 3
    rows = (num_plots + cols - 1) // cols

     fig, axes = plt.subplots(rows, cols, figsize=(15, 5 * rows), sharex=True, sharey=True)



    axes = axes.flatten()
```

```
    for i, (ax, (index, row)) in enumerate(zip(axes, df_results.iterrows())):
        cycles = range(1, len(row['零配件 1 次品率历史']) + 1)
        ax.plot(cycles, row['零配件 1 次品率历史'], label='零配件 1 次品率', marker='.', linestyle='--', color='red')
        ax.plot(cycles, row['零配件 2 次品率历史'], label='零配件 2 次品率', marker='.', linestyle='--', color='darkblue')
        ax.plot(cycles, row['成品次品率历史'], label='成品次品率', marker='.', color='orange')
        ax.set_title(f"情况 {index+1} - {confidence_levels[i]}")
        ax.set_xlabel('售卖次数')
        ax.set_ylabel('次品率')
        ax.grid(True)
        ax.legend()

    for j in range(i + 1, len(axes)):
        axes[j].axis('off')

    plt.tight_layout()
    plt.show()

lishishujukeshihua(biaozhun95, ["95% 置信度"] * len(biaozhun95))
lishishujukeshihua(biaozhun90, ["90% 置信度"] * len(biaozhun90))
```

附录 5 第四问 2.py

介绍：该代码是 python 语言编写的，解决问题四第二问

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm

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

```
data = {
    '零配件': ['零配件 1', '零配件 2', '零配件 3', '零配件 4', '零配件 5', '零配件 6', '零配件 7', '零配件 8'],
    '购买单价': [2, 8, 12, 2, 8, 12, 8, 12],
    '检测成本': [1, 1, 2, 1, 1, 2, 1, 2]
}
banchengpin = {
    '半成品': ['半成品 1', '半成品 2', '半成品 3'],
    '装配成本': [8, 8, 8],
    '检测成本': [4, 4, 4],
    '拆解费用': [6, 6, 6]
}
chengpin = {
    '成品': ['成品'],
    '装配成本': [8],
    '检测成本': [6],
    '拆解费用': [10],
    '市场售价': [200],
    '调换损失': [40]
}

parts_df = pd.DataFrame(data)
semi_df = pd.DataFrame(banchengpin)
zuizhongjuedingshuchu = pd.DataFrame(chengpin)

shoumaicishu = 10  # 最大循环次数
yuzhi = 0.001  # 次品率阈值
dijianbili = 0.7  # 次品率递减比例
chouyangwucha = 0.02  # 抽样误差

# 估计次品率的抽样函数
def gujicipinlv(p_estimate, confidence_level):
    zhixinzhi = norm.ppf((1 + confidence_level) / 2)
```

```
    yangbenliang = (zhixinzhi**2 * p_estimate * (1 - p_estimate)) / chouyangwucha**2
    jaincezhi = np.random.binomial(int(yangbenliang), p_estimate) / yangbenliang
    return jaincezhi

# 定义总成本计算函数
def calculate_bufenchengben11(parts_df, semi_df, zuizhongjuedingshuchu, confidence_level=0.95, shoumaicishu=shoumaicishu, threshold=yuzhi):
    bufenchengben11 = 0
    cycles = 0

    # 初始次品率估计
        cipinlv1 = np.array([gujicipinlv(0.1, confidence_level) for _ in range(len(parts_df))])
        cipinlv2 = np.array([gujicipinlv(0.1, confidence_level) for _ in range(len(semi_df))])
    cipinlv3 = gujicipinlv(0.1, confidence_level)

    jiancelishi = {'parts': [], 'semi': [], 'final': []}
    juecelicheng = []

    while cycles < shoumaicishu and (cipinlv1.max() > threshold or cipinlv2.max() > threshold or cipinlv3 > threshold):
        cycles += 1
        # 记录次品率
        jiancelishi['parts'].append(cipinlv1.copy())
        jiancelishi['semi'].append(cipinlv2.copy())
        jiancelishi['final'].append(cipinlv3)

        # 计算零配件检测成本
        jiancechengben11 = 0
        jiancec11 = []
        for i, row in parts_df.iterrows():
            if row['检测成本'] < cipinlv1[i] * row['购买单价']:
```

```
                jiancechengben11 += row['检测成本']
                jiancec11.append(1)  # 进行检测
            else:
                jiancechengben11 += cipinlv1[i] * row['购买单价']
                jiancec11.append(0)  # 不检测
        bufenchengben11 += jiancechengben11

        # 计算半成品检测成本
        banchengpinchengben = 0
        benchengpinfenxi1 = []
        for i, row in semi_df.iterrows():
            if row['检测成本'] < cipinlv2[i] * (row['装配成本'] + row['拆解费用']):
                banchengpinchengben += row['检测成本']
                benchengpinfenxi1.append(1)
            else:
                banchengpinchengben += cipinlv2[i] * (row['装配成本'] + row['拆解
费用'])
                benchengpinfenxi1.append(0)
        bufenchengben11 += banchengpinchengben

        # 计算成品检测成本
        zuizhonghuafei = 0
            if zuizhongjuedingshuchu.loc[0, '检测成本'] < cipinlv3 *
(zuizhongjuedingshuchu.loc[0, '装配成本'] + zuizhongjuedingshuchu.loc[0, '市场售
价'] + zuizhongjuedingshuchu.loc[0, '拆解费用']):
            zuizhonghuafei += zuizhongjuedingshuchu.loc[0, '检测成本']
            final_decision = 1
        else:
            zuizhonghuafei += cipinlv3 * (zuizhongjuedingshuchu.loc[0, '装配成本'] +
zuizhongjuedingshuchu.loc[0, '市场售价'] + zuizhongjuedingshuchu.loc[0, '拆解费用
'])
            final_decision = 0
        bufenchengben11 += zuizhonghuafei
```

```
        juecelicheng.append({
            'cycle': cycles,
            '零配件检测决策': jiancec11,
            '半成品检测决策': benchengpinfenxi1,
            '成品检测决策': final_decision
        })


        cipinlv1 *= (1 - dijianbili)
        cipinlv2 *= (1 - dijianbili)
        cipinlv3 *= (1 - dijianbili)

    return bufenchengben11, cycles, juecelicheng, jiancelishi

bufenchengben11_95, cycles_95, decisions_95, defects_95 = calculate_bufenchengben11(parts_df, semi_df, zuizhongjuedingshuchu, confidence_level=0.95)
bufenchengben11_90, cycles_90, decisions_90, defects_90 = calculate_bufenchengben11(parts_df, semi_df, zuizhongjuedingshuchu, confidence_level=0.90)

# 输出结果
def print_results(confidence, bufenchengben11, cycles, decisions, defects):
    print(f"\n{confidence}% 置信度下的决策分析结果：")
    print("=" * 40)
    print(f"单个产品总成本: {bufenchengben11:.2f}")
    print(f"总循环次数: {cycles}")
    print("=" * 40)

    print("决策详情：")
    print("-" * 40)

    for decision in decisions:
```

```
        print(f"第 {decision['cycle']} 次循环：")
        print(f"  零配件检测决策: {decision['零配件检测决策']}")
        print(f"  半成品检测决策: {decision['半成品检测决策']}")
        print(f"  成品检测决策: {decision['成品检测决策']}")
        print("-" * 40)

    print("=" * 40)

print_results(95, bufenchengben11_95, cycles_95, decisions_95, defects_95)
print_results(90, bufenchengben11_90, cycles_90, decisions_90, defects_90)

def plot_jiancelishi(defects, confidence):
    cycles = range(1, len(defects['parts']) + 1)
    plt.figure(figsize=(10, 6))
     plt.plot(cycles, np.array(defects['parts'])[:, 0], label='零配件 1 次品率', color='orange', marker='o')
     plt.plot(cycles, np.array(defects['parts'])[:, 1], label='零配件 2 次品率', color='yellow', marker='o')
     plt.plot(cycles, np.array(defects['parts'])[:, 2], label='零配件 3 次品率', color='red', marker='o')
     plt.plot(cycles, np.array(defects['parts'])[:, 3], label='零配件 4 次品率', color='blue', marker='o')
     plt.plot(cycles, np.array(defects['parts'])[:, 4], label='零配件 5 次品率', color='black', marker='o')
     plt.plot(cycles, np.array(defects['parts'])[:, 5], label='零配件 6 次品率', color='green', marker='o')
      plt.plot(cycles, np.array(defects['semi'])[:, 0], label='半成品次品率', color='purple', marker='o')
    plt.plot(cycles, defects['final'], label='成品次品率', color='pink', marker='o')
    plt.xlabel('循环次数')
    plt.ylabel('次品率')
    plt.title(f'{confidence}% 置信度下的次品率历史')
    plt.legend()
    plt.grid(True)
```

```
    plt.show()

plot_jiancelishi(defects_95, 95)
plot_jiancelishi(defects_90, 90)
```