基于统计学习和优化方法的企业原材料的订购与运输决策问题

摘要

本文旨在解决生产企业的原材料的订购与决策问题。本文的模型综合了各种手段，针对性地建立了各种统计学习模型，并且基于最优化方法建立了规划问题进行求解。

在问题一中，解决供应商的重要性评级问题。在数据分析和特征工程的基础上，基于集成学习的思想，集成了供应量优先模型、熵权模型和聚类模型，给出重要性的候选供应商用于后续问题的求解。

在问题二中，解决企业订购方案和转运方案的多步骤决策问题。本文建立了加权移动平均模型用于回归供应量和订购量的关系，基于 ARIMA、HMM 等模型预测了对于转运商未来的转运损耗率。在上述数学模型的基础上，将问题拆解为多步骤决策问题，首先基于剪枝和贪心的算法选择候选供应商，再建立概率优化模型衡量优化中的不确定性因素确定订购方案，最后将转运方案的决策问题转化为 0-1 规划求解。

在问题三中，解决企业多目标的决策问题。本文首先给出该问题在数学上的优化表达式，之后基于供应商的统计模型的建立以及鼓励稀疏性的范数惩罚策略的运用，将多目标决策问题化归为问题二中的单目标决策问题。

在问题四中，解决企业最大产能的寻求问题。对于这个表面上的优化问题，本文将其转化为存在性判定问题。在由参数估计给出的供应商的概率模型和线性规划模型的基础上，改进问题三的解法，并且使用二分搜索算法计算答案。

对于模型给出的结果，采用蒙特卡罗方法进行模拟，验证了给出结果的可靠性。最后，我们对模型进行了稳健性分析和误差分析，并且进行了模型的优缺点对比和模型推广的可行性探究。

关键词: 集成学习　加权移动平均　时间序列　概率优化　0-1 规划　蒙特卡罗

一、问题重述

1.1 问题背景

对于生产企业，其用于生产的原材料的订购及运输是企业决策中关键的问题。在满足企业正常生产与运转的前提下，企业应进行明智地决策以合理选择不同的供应商和转运商，从而最大限度地减小企业成本。该问题是极具代表性同时又具有重要实际应用价值的问题。

1.2 问题重述

在该问题中，涉及两大重要的环节。一是企业对于不同供应商和转运商的评价体系问题，由于生产企业不可能知道供应商和转运商的所有信息，在此种不完全信息条件下的决策，同上需要生产企业对供应商和转运商的重要特征进行数据分析、特征提取和数学模型的构建，在此基础上建立合理的评价体系从而辅助企业决策。二是企业在决策中的规划问题，在满足企业运转所需要的各项约束的前提下，如何合理地分配资源，选择最优的策略从而使得企业达到其最优解。在上述两大环节中，为了更好地分析数据，把握整体的体系，往往需要建立相应的数学模型。我们建立了时间序列模型、加权移动平均值模型等数学模型以解决问题。

二、问题分析

2.1 问题一的分析

问题一给出了 402 家供应商的供货特征，要求建立数学模型衡量供应商的重要性，本质上为评价体系的构建问题，解决该问题的关键在于从题目给出的数据中挖掘数据的特征，并且根据统计学习等学科知识定义并且计算供应商的重要性。

对于该问题，首先需要对数据进行可视化分析，把握数据的基本性特征，并且对离群点和异常值进行检查和排除，寻找数据分布的规律。在上述数据分析的基础上，针对数据的特点和问题的痛点，我们提出基于的集成学习模型。集成学习模型的优点在于不同的子模型考虑到了问题的不同方面，从而使得集成学习的结果更加全面，具有更高的可信度；集成学习对于某些特殊情况，具有更高的鲁棒性。根据集成模型，从企业对每家供应商每周的订购量以及每家供应商的供应量出发，模型给出了企业的重要性得分，从而用于企业的规划决策中，便于后续几个问题的解决。我们所提出的集成学习模型共包含了：供应量优先模型，熵权模型，基于不同聚类方法的聚类模型。

2.2 问题二的分析

问题二要求解决企业的决策问题，企业的目标是最小化总成本，但直接将总成本作为优化目标较为困难。同时，企业主要需要考虑的决策变量有：供应商的选择、订购方案的选择，转运商的选择，同时对所有决策变量进行优化也是相当困难的。因此在问题二中，考虑逐步进行决策，且在每一步决策中仅考虑某一个较为易于求解的子目标进行优化，而非优化最终的总成本。

具体来讲，主要分为以下几个子决策：

1. 候选供应商的选择。考虑企业在现实决策中，不可能将所有的供应商列入考虑范围内，企业会通常在供应商的重要性的基础上（该部分可以基于问题一的解），对剩余的供应商进行规

划。在企业的乐观估计下，只需选择最少的供应商就可以满足企业的生产运转要求，这也是该问题中所考虑的。该子决策任务的优化目标应为最小化候选供应商的数量，使用最少的供应商使得其生产的需求得到相应的满足。

2. 订购方案的制定。在上一步决策的基础上，已经划定了候选供应商的范围，因此后续的决策任务只需在该较小的范围内进行考虑。该简化基于现实决策的理解，同时也使得该数学模型更易于求解。该子决策任务，目的在于为企业决策未来 24 周的订购方案选择，其优化目标应为：在满足企业生产的约束前提下，使得企业的运转最为经济（原材料订购成本和储存成本的总和最小）。

3. 转运方案的制定。基于企业的订购方案，该步决策任务为为每一周的供应商选择相应的转运商，其中要满足企业的基本原则性条件：每周的每家供应商尽量由一家转运商转运。优化目标为：最小化总材料的损耗量。

经过上述转化，一个复杂的规划问题，被拆解为几个易于求解的小的规划问题。但在每一步的规划中，由于企业的不完全信息和现实生活中的不确定因素，企业需要建立相应的数学模型填补其信息空白。主要有以下几个部分：

1. 订购量和供应量的相关性模型。对于企业的订单，供应商的满足率在 1 的附近波动，因此订购量和供应量之间的相关性模型的构建非常重要。对此，我们建立基于加权移动平均的回归模型。同时，基于回归模型，给定未来 24 周的订购量，可以给出其供应量的预测结果，用于企业的决策过程中。

2. 转运商的损耗率模型。转运商的损耗率是一个重要的不确定因素，在附件中给出了历史 5 年中每一家转运商的损耗率，对于这一明显具有时序信息的序列，基于不同转运商的不同特点的基础上。我们分别建立了差分整合移动平均自回归模型（ARIMA）、隐马尔可夫模型（HMM）、指数分布逼近模型对其未来的转运的损耗率进行相应的预测。

建立了上述几个数学模型，我们成功地量化了决策中难以把握的不确定性因素，将其数学化。并且将上述模型纳入规划的求解当中，使得求解的优化结果综合考虑了各种不确定性因素，因此所给出的结果将具有更高的鲁棒性。加入不确定性因素后，采用的规划模型也由确定性模型修改为概率优化模型。

2.3 问题三的分析

在问题二中，已经建立了详尽的数学模型和行之有效的解决方案。在问题三中考虑化归为问题二进行求解。问题三并没有要求企业逐步进行决策，因此为了进行整体的分析，首先针对其建立端到端的多目标的规划模型。但在实际求解中，考虑到转运商的损耗与供应商的供货量独立的假设，仍然可以沿用问题二中多步骤规划的求解思路。

相较于问题二，问题三有以下几个关键点，下面展开叙述并且给出我们的解决思路：

1. 问题三为多目标规划，而非问题二的单目标规划。考虑将多目标规划问题转化为加入范数惩罚的单目标规划模型。与问题二相似，供应商的选择应该越少越好，因此采用鼓励稀疏性的 1 范数惩罚。

2. 问题二建立在选择候选供应商的基础上进行规划求解，此时选择出的候选供应商的特征单一，而问题三供应商的可选范围为所有的 402 家供应商，如果单纯沿用问题二的模型不能刻画不同供应商不同的供应特点。针对上述观察，需要改进问题二所定义的概率优化模型。

3. 与问题二相比，问题三的求解复杂度更高。完全沿用问题二的求解方法不一定能在有限时间内求出精确解。解决的思路有：一是基于合理的假设更改问题二中的某些限制条件从而降低规划模型的求解复杂度，二是更改求解方法，寻求误差范围内的近似最优解。

2.4 问题四的分析

问题四假设企业的产能有提升的潜力，首先要求求解企业可以提升的最大产能。分析问题可以发现，企业的产能并不是可以无限制地提升的，企业需要考虑供应商和转运商的实际情况。对此，需要对供应商和转运商建立其数学模型，从而估计出供应商和转运商的潜力，在此基础上确定企业的产能的提升的潜力。建立起企业决策规划模型后，企业的最大产能必须保证建立的规模模型存在可行解。

在对所有供应商建立概率统计模型的基础上，我们得到了所有 402 家供应商的供应潜力的估计，据此使用二分搜索的算法得到企业可以提升的最大产能。问题四依托于问题三，但相比起问题三，对于产能较大的企业，其决策模式不能沿用之前产能较低的情况，因此问题四的一大难点对于问题三中的规划问题需要进行修正。

三、模型假设

1. 假设转运商的损耗与供货量独立。根据该假设，可以将订购方案和转运方案的决策看作是两个相对独立的过程。

2. 假设同一家供应商或转运商的统计模型相对稳定。根据该假设，对供应商或者转运商建立相应的统计模型之后，在企业决策的过程中可以将其看作恒定的模型。

3. 每家供应商总是在其能力范围内尽可能地满足企业的订购需求。该假设符合现实生活中供应商盈利的本质，也使得后续的建模过程更为简单。

4. 无特定说明，默认随机变量满足正态分布。如：通常假设残差满足均值为 0 的正态分布。

四、符号说明

符号	说明	单位/公式
$O_i(t)$	企业对第 i 家供应商第 t 周的订购量	m^3
$S_i(t)$	第 i 家供应商第 t 周的供货量	m^3
$R_i(t)$	供货量与订购量之间的残差项	$S_i(i) - O_i(t)$
$T_i(t)$	第 i 家转运商第 t 周所需转运的供应量	m^3
$T_i'(t)$	第 i 家转运商第 t 周的被企业的接收量	m^3
$\alpha_i(t)$	第 i 家转运商第 t 周的损耗率	$\frac{T_i(t)-T_i'(t)}{T_i(t)}$
c	企业每周所需要的产能	m^3
p_i	第 i 类原材料的采购单价 ($i = 1, 2, 3$)	元/m^3
q_i	第 i 类原材料的运输单价 ($i = 1, 2, 3$)	元/m^3
r_i	第 i 类原材料的储存单价 ($i = 1, 2, 3$)	元/m^3
t	每家转运商每周的转运能力 ($t = 6000$)	m^3

五、基于集成学习的供应商重要性度量模型（问题一）

5.1 数据预处理

附件一中给出了每一家供应商前五年（240 周）内每周的订购量 O 和供应量 S。且由附件中的表格可以看出，每一家供应商负责供应 ABC 三类原材料中的的一种，根据题目条件，ABC 三类原材料具有不同的效用（单位体积的原材料对于差产能的贡献），其具体数值如下表所示：

原材料类型	企业每单位产能所消耗的材料 (m^3)
A	0.6
B	0.66
C	0.72

给定企业每单位产能所消耗的材料 m，则 ABC 三类原材料的效用 u 可以定义为其倒数 $u = \frac{1}{m}$，此时 ABC 三类原材料的效用比为 $\frac{1}{0.6} : \frac{1}{0.66} : \frac{1}{0.72}$。对于附件中给出的订购量和供应量，实际上需要根据其不同材料的效用比作为比例系数，得到实际上该订购量和供应量对于产能的等效贡献值。也即使用如下公式：

$$\hat{S} = u \times S$$

$$\hat{O} = u \times O$$

得到等效后的订购量 $\hat{O}$ 和供应量 $\hat{S}$.

5.2 数据分析

首先绘制图表，可视化对应的数据，把握数据的规律。该数据最主要的规律为稀疏性，大多数供应商的订购量和供应量都为 0 值。以供应量为例，总共有 $240 \times 402 = 96480$ 条数据，但其中仅有 25784 的非零元素，非零元素占总元素的比值仅有 14.2 %. 造成上述稀疏性的同时有客观上和主观上的原因。其主要的客观原因在于：重要的企业数目并非很多，且企业仅需要相对于可供选择的供应商总数较少的供应商便可以满足其正常生产运转的需求。在主观层面，考虑实际生活，该稀疏性也可以找到很多合理的解释，例如：生产企业通常会和供应商进行一定的长期合作关系，在一段时间内企业会选择较为固定的几家供应商。

5.3 特征工程

附件一中给出了每一家供应商前五年（240 周）内每周的订购量 O 和供应量 S。在此基础上提取以下几个特征，且提取的理由如下所述：

1. 平均供应量：$MEAN(\hat{S})$, 供应商的供应量是供应商重要性的直接衡量，对于更重要的供应商，企业会加大其订购，从而在一般情况下也会具有较大的供应量。而不用平均订购量衡量该指标的原因是，存在某些供应商由于自身能力等限制，尽管企业加大其订购量，但供应商却不能提供相应的供应量。考虑到该部分供应商的存在，采用每周的平均供应量衡量可以综合考虑到企业对供应商的重视程度以及供应商自身的响应能力。

2. 平均满足率：$MEAN(r), r = \frac{\hat{O}}{\hat{S}}$, 其中满足率定义为实际供应量与企业订购量的比例，含义为供应商对于企业的订购的满足率，由于数据中供应量可能大于订购量，也可能小于订购量，该满足率可能小于 1，也可能大于 1。较优的满足率应该维持在 1 的附件，甚至绍大于 1，说明企业的订单可以很好地得到满足。在对满足率进行特征提取的时候，对于 $S = 0$ 的情况，设置其相应的满足率 r 为特殊值 NAN 不予考虑，不考虑在均值、方差等统计量的计算之中。

3. 稳定性：$s = 1 - VAR(r)$, 衡量供应商的稳定性，一个好的供应商应该对于企业维持稳定的供应，也即供应商的满足率 r 应该维持在一个相对较高且波动较小的水平，这样的供应商会更加容易得到企业的长期信任。而稳定性用方差与 1 的差衡量，方差越小说明该值越接近 1，稳定性越高。

5.4 供应量优先模型

对于上述构建的三个主要特征，显然第一个特征：平均供应量 $MEAN(\hat{S})$ 是衡量企业重要性的最主要特征。如果没有较高的平均供应量作为保证，其他的两个特征就算再理想对不应使得企业的重要性更高。例如：有些企业常年保持在极低的供应量（接近于 0），此类企业显然是极不重要的企业，尽管其供应的稳定性很高。同理，就算某些企业具有更高的满足率，但如果本身的供应量就很低，满足率也没有其应有的意义。

根据上述分析，构建供应量优先模型。首先，在仅考虑供应量 $\hat{S}$ 的前提下，对所有的 402 家供应商进行排序，绘制散点图后可以发现数据的规律：大部分的供应商的平均供应量都接近于 0，该部分供应商为不重要的企业；而少部分供应商的平均供应量很大，说明该部分供应商为很重要的企业；同时，不同供应商的平均供应量之间存在几个差距比较大的明显的分隔阈值，可以根据几个明显的分隔阈值将供应商分为几类。

绘制箱线图和散点图观察数据的规律，可以发现存在着几个区分较为明显的阈值 τ。为了寻找能够合理地区分出 50 家较为重要的企业的阈值 τ，应该在 $\frac{50}{402} = 12.4\%$ 分位数的左右选取。根据观察法，可以选择 $\tau = 10000$ 的时候，接近 12.4% 分位数，且此时可见为一个明显的区分点。该区分点也可以通过调用大津算法（OSTU [1]）计算得到，其结果也在该值附近，说明该值的选取较为合理。OSTU 算法是数字图像处理中二值化问题的常见算法，在企业重要性的评价问题中，如果将重要和不重要的企业的区分视作一个二分类问题，本质上也可以视作一个二值化问题，因此使用 OSTU 算法也具有其内在的合理性。在该基准阈值的基础上，综合考虑阈值 τ 附近的供应商的其他特征，并且对阈值进行微调，当 $\tau = 14000$ 的时候，可以选取出 50 家较为重要的企业，而其他企业为较不重要的企业。

上述模型称为供应量优先模型，可以给出一个较为具有参考意义的基准答案。该模型给出的 50 家企业序号为：[78 208 189 292 273 74 114 3 218 210 244 86 294 80 346 55 367 364 338 40 365 31 284 126 374 37 247 395 307 201 143 194 352 348 306 268 356 330 308 131 139 340 329 275 151 282 108 361 140 229], 其中根据平均供应量的大小升序排列。

5.5 熵权模型

供应量优先模型的合理性如上所述，同时其也给出了较为令人信服的结果。但供应量优先模型的弊端在于对于除了平均供应量以外的其他特征，考虑得很少。因此可以考虑使用熵权模型，对不同的特征赋予不同的权重，通过加权得到最终的得分，而权重采用信息熵计算。当某个特征的信息熵更大，说明其包含了更多的信息，该特征对于该分类任务的重要性就会更大。

使用熵权模型，计算得到特征工程中所提取的三个特征的权重分别为: [0.21196796, 0.37585979, 0.41217225]. 得到权重后，需要对特征进行归一化，后根据权重进行加权组合得到最终表示重要性的指标。对该权重进行分析可以发现，熵权模型给出的权重对于本应较为重要的第一个特征 $\hat{S}$ 给出了相对较低的权重，主要是由于数据的稀疏性造成的，因为平均供应量有大量零元素或者接近零元素的存在，导致其数据的二极化较为严重，从而该特征的信息熵却反而较小。

利用上述的熵权模型，可以给出 50 家重要性较大的企业为：[76 129 189 307 201 98 3 86 338 348 291 114 314 150 7 123 55 244 139 346 80 367 40 294 218 364 140 143 365 31 247 284 266 374 308 194 352 330 356 306 268 131 151 340 329 275 282 108 361 229]，其中同样按照分值升序排列（分值越大表明重要性越高）。

熵权模型所给出的 50 家供应量，与简单使用供应量优先模型给出的 50 家供应商的结果，总有 41 家完全相同的企业，且这些企业大多为重要性较高的企业。说明两种模型都有其一定和合理性和一致性，从某种意义上说明方法的有效性。

5.6 聚类分析模型

由于重要的企业通常和企业有更紧密的联系和合作，这种正反馈机制下，重要和不重要的企业的差异会被拉大，从而导致不同重要程度的企业之间的特征差异较为明显。根据上述观察和分析，可以考虑对数据进行聚类分析。

首先，采用层次聚类法 [2]，从下而上地把小的类合并聚集，聚类过程中得到一棵树形结构。考虑采用两种不同的层次聚类的距离定义方式，average-linkage（即计算两个聚簇中各自数据点的两两距离的平均值）和 complete-linkage（即选择两个最远的一对数据点的距离作为类的距离，两种方式的层次聚类，选择聚簇数为 4，都得到了相同的聚类结果：[108 131 139 143 151 194 201 268 275 282 306 307 308 329 330 340 348 352 356 361 395 140 229]. 并且观察上述结果，发现上述聚类给出的 23 家企业都同时在熵权模型和供应量优先模型所给出的答案中，因此该聚类的确将非常重要的企业聚成了一类。虽然上述聚类结果并不能直接得到 50 家重要的企业解决该问题，但却可以作为一个重要的参考，同时聚类结果给出的 23 家企业应当是优先度很高的极为重要的企业。

再者，考虑到层次聚类法，需要手动选择阈值，而考虑使用基于迭代方法的 K-Means 聚类法。同样选取聚簇数为 4，可以得到较重要的企业在某一类中，选取该类结果，得到 43 家供应商：[3 7 31 40 55 80 108 114 123 131 139 140 143 150 151 194 218 229 244 247 266 268 275 282 284 291 294 306 308 314 329 330 338 340 346 348 352 356 361 364 365 367 374]。该 43 家供应商同样与之前模型给出的所有结果都有较大的交集，说明 K-Means 所得到的结果也是某种意义上合理的。例如：K-Means 算法给出的 43 家供应商，全部落在熵权模型所给出的 50 家供应商之中，说明此部分企业的确较为具有区分度。

5.7 集成学习模型

根据上述的分析，上述几个模型尽管在某种意义上有很大的一致性，但都各有其优缺点。考虑到上述情况，建立集成学习模型，由几个较弱的分类器出发，基于 Bagging 的方式得到一个较强的分类器。本节的代码采用 R 和 Python 完成，其中统计机器学习相关的 Python 包使用了 sklearn，代码详见附录。

采用集成学习模型得到最终的 50 家重要供应商为：[126 98 37 3 86 291 395 338 314 307 201 114 150 7 123 244 80 266 218 55 294 346 367 364 40 348 365 31 284 374 247 143 194 352 306 356 268 139 308 330 131 340 329 151 275 282 108 140 361 229]。

六、基于统计模型和最优化方法的多步骤企业规划（问题二）

6.1 候选供应商的选择

在企业的乐观估计下，只需要选择最少数目的供应商就可以满足企业的需求。该决策问题的困难点在于，企业并不知道供应商的供应潜力，因此企业需要基于问题一中对供应商的供货特征

进行分析的结果进行评估。而上述结果已经反映在问题一所选出的 50 家较为重要的供应商中，因此在该决策任务，也仅限定在上述 50 家企业中进行选择。

6.1.1 基于剪枝和贪心算法的候选供应商选择

考虑问题的限制条件，可以给出一些剪枝条件，作为一种过滤的手段，过滤出困可能出现在候选解的供应商，从而降低该问题的复杂度。

比对项	比例关系 (A:B:C)
效用	$\frac{1}{0.6}:\frac{1}{0.66}:\frac{1}{0.72}$
订购成本	1.2: 1.1 : 1.0
运输储存成本	1: 1: 1

1. 原材料类型的优先级。考虑 ABC 三种原材料的效用、订购、运输、储存成本可以发现，其具有如上表所示的关系。通过简单的计算和对比可以发现，在 ABC 三种原材料中，对于订购成本该项的贡献，A 和 C 是等价的，且 AC 两种原材料显然优于材料 B。因此在考虑选择不同的供应商的时候，应该优先考虑选择供应 A 或 C 这两类原材料的供应商。因此，可以优先排除性价比低的 B 类供应商，而在 A 和 C 两类供应商之间，由于其性价比相同但是单位产能所需要的 A 类原材料低于 C 类原材料，也应该优先考虑 A 类供应商。

2. 最大转运限制。根据转运决策的要求，每一家供应商所供应的原材料应该尽可能由同一家转运公司所承担。因此，该转运决策的限制实际上也限制了供应商的供应限制，也不应该超过上述每周 $6000m^3$ 转运限制。由于订购量和供应量密切相关，因此有理由认为，企业决策订购方案的时候，可以加上最大订购量不超过 $6000m^3$ 的限制条件。

3. 供应潜力限制。分析供应商 5 年之内的供货特征，可以发现很多供应商并不具备大量供货的潜力。考虑用供应商 5 年内的最大历史供货量作为其供货潜力的一个衡量，若其最大历史供货量不能超过 $6000m^3$，则有理由继续认为其在未来 24 周内也不可能进行如此大规模的供货。考虑该供应潜力的限制，可以利用其作为剪枝筛选掉最大历史供货量不超过 $6000m^3$ 的企业。该剪枝的理由有两方面：一方面，此类公司不具备较大的供货潜力，可以说明其通常不是重要的公司。令一方面，实验结果证明，在上述剪枝之在剩余的公司中选择仍然可以满足企业的需求（存在可行解），因此上述剪枝并不会对解造成过多的影响。

基于上述观察，可以提出适用于该决策任务的贪心算法：在经过剪枝算法过滤掉部分供应商之后，在剩余的供应商中预先计算得到对每一家供应商进行最大订购时该供应商对总体优化目标产生的代价，每次贪心地选择产生代价最小而同时更重要的的一家供应商，直到所选择出来的供应商可以维持企业的生产需求为止。将上述贪心算法给出的企业数目，作为最小满足条件的企业数目，给出的解为：[201, 307, 395], 可以看出给出的都为 A 类原材料的供应商。

6.1.2 基于加权移动平均值的候选供应商选择

上述基于剪枝和贪心算法的优点是算法实现简单且效率较高，但其缺点是剪枝和贪心的过程中对实际情况做了较多的假设。例如：上述贪心算法并没有考虑到供应商满足率的不确定性问题：供应商所提供的原材料不一定正好是企业的订购量。同时，贪心算法中简单的使用企业的历史最大供货量定义其供货潜力在某种程度上时过于乐观的，比如该算法所生成的解中包含了 S201 这家供应商。但观察该供应商的历史供货特点可以发现，该供应商对于企业并不是持续稳定的供货，而是在某几个少数时间节点上面存在极大宗的供货（$30000m^3$ 的供应量）。贪心算法的解将该供应商纳入可行解中，忽略了该供应商巨大的不确定性因素，某种程度来说，对于企业（特别是决策者为风险厌恶类型的企业）是不利的结果。

基于上述考虑，此问题中我们使用加权移动平均值来建立每个供应商的订货量-供货量曲线关系，并进一步计算具体的供应商选择。据观察题中给定的 $O_i(t)$ 与 $S_i(t)$ 数据点，我们认为时间变量 t 的影响可以忽略,(S_i, O_i) 可以认为是离散的数据点；其近似满足恒等关系，但不同供应商在各自不同的区间上会出现严重的供货能力下降。对于某一给定的企业 i，其收到订购量 o 后，供货量取为

$$S_i(o) = \frac{\sum_{i=1}^{402} S_i w_i}{\sum_{i=1}^{402} w_i}$$

其中权值 w_i 的计算方法为：

$$w_i(o) = \exp(-\frac{(O_i - o)^2}{\sigma^2})$$

其中，σ 控制距某位置 o 同一距离的数据点对 o 位置的影响。较大的 σ 值使曲线变得平滑，但将脱离实际数据点；较小的 σ 使曲线贴近实际数据点而导致全局的不光滑。此处 σ 取为 500，对于距 o 点 1000 单位的数据点，相应权值小于 2%。

该模型给出了一个有实用价值的建立 $S(O)$ 关系的方法，它有如下特征：

1. 对于数据点稠密的、线性关系明显的区域，算法效果近似线性插值，表示供应情况正常，供应量基本满足订购量。

2. 对于数据点稀少的区域，由于距有效数据点距离较大，值近似为 0，表示供应商无力承担此订购量。

3. 在数据点分布突变、以及整体数据点稀疏分布的情况下，算法给出的预测值相对于给出供货情况的一种合理推测，包括一段区域内行为的适量外延，或模拟供应情况正常时的小波动。

4. 在供应量在近似于 0 和订货量之间有较大波动的情况下，我们认为模型给出的值表示特定订货量下，供应商供货能力的一种期望。

在得到每个供应商的订货-供货关系后，考虑到各供应商独立，相当于构成了多个单元函数最大值问题。此处考虑到加权平均值的平滑性，可以简单地遍历可选区域内的订货量值变量求解。此过程中的两个约束基本与前一方法相同：只取 A,C 原材料的供应商；订货量可选范围为 $[0, \min(6000,$ 各供应商历史最大供应量$)]$。

基于此最终选择出 4 家供应商：[374 307 395 201]。

6.2 基于概率优化模型的订购方案制定

本节订购方案将根据上述选出的 4 家供应商进行规划求解，考虑到供应量的不确定性，尝试在优化中加入概率的元素。为了符号的简单，将上述 4 家供应商编号 [374 307 395 201] 重编号为 [1 2 3 4]，其中第 1 家供应 C 类原材料，其余 3 家供应 A 类原材料。下面列出本节需要的符号：

符号	说明	单位
x_{ij}	第 j 周对第 i 家供应商的订购量	m^3
y_{ij}	第 i 家供应商商第 j 周的供应量	m^3
z_j	第 j 周订购原材料的生产量	m^3
ϵ_i	第 i 家供应商与订购量的误差	m^3
$\sigma_{\epsilon i}$	ϵ_i 的方差	
$store_j$	第 j 周的库存所能生产产品的总量	m^3
μ_j	$store_j$ 的期望	m^3
σ_j	$store_j$ 的方差	

有了上述思想和符号准备后，现在开始建立模型。

假设：

1. 假设初始库存恰好为 2 周生产需求，即 $store_0 = 56400$。

2. 每周固定生产 $28200m^3$ 产品。

3. 供应量在订购量附近波动，误差 ϵ_i 服从均值为 0 的高斯分布，且方差保持时不变，假定标准差均为 100，这个数值将在后续稳健性分析中进行更深入研究。

规划目标：最小化采购成本，即

$$\sum_{j=1}^{24} x_{1j} + 1.2 \sum_{j=1}^{24} \sum_{i=2}^{4} x_{ij}$$

约束条件：

1. 订购量非负，即 $x_{ij} \geq 0$

2. 由假设 3，供应量与订购量的关系为

$$y_{ij} = x_{ij} + \epsilon_i \sim N(x_{ij}, \sigma_{\epsilon i}^2)$$

3. 本周库存等于上一周库存加上本周供应量减去固定的生产数量（假设 2），即

$$store_j = store_{j-1} + \frac{1}{0.72} y_{1j} + \frac{1}{0.6} \sum_{i=2}^{4} y_{ij} - 28200$$

4. 库存满足 2 周生产需求的概率大于给定数值，由假设 1，只需

$$\mathbb{P}\left[store_j \geq 56400\right] \geq 0.945$$

经过简单计算，可以将上述约束改写为 MATLAB cvx 可以识别的数学形式：

$$
\begin{aligned}
\textbf{min}\ & \sum_{j=1}^{24} x_{1j} + 1.2 \sum_{j=1}^{24} \sum_{i=2}^{4} x_{ij} \\
\textbf{subject to}\ & \forall i = 1, \cdots, 4, j = 1, \cdots, 24 \\
& x_{ij} \geq 0 \\
& z_j = \frac{1}{0.72} x_{1j} + \frac{1}{0.6} \sum_{i=2}^{4} x_{ij} \\
& \mu_j = \mu_{j-1} + z_j - 28200 \\
& \sigma_j = j \left(\frac{\sigma_{\epsilon 1}}{0.6} + \sum_{i=2}^{4} \frac{\sigma_{\epsilon i}}{0.72} \right) \\
& \mu_j - 56400 \geq 1.6 \sigma_j
\end{aligned}
$$

利用 cvx 求解上述优化问题得到最优解，代码和结果详见附录。

6.3 转运商损耗率的时间序列模型

由于历史数据没有给出每一周每家转运商的运输量，难以分析供应对损耗的影响，因而作出假设 1 简化分析。根据此假设，我们可以对转运商的损耗率独立地建立模型。

首先总体观察题目所给附件 2 中的数据，除了 0 作为缺失值外，我们注意到一个特殊现象：数据中存在大量整数 5，与其他数值较小的多位小数极不匹配。我们认为 5 不是合理的数据，一并作为缺失值处理。接着我们考察每个转运商在 240 周中缺失值的数量，得到下表结果：

T1	T2	T3	T4	T5	T6	T7	T8
4	1	124	159	191	35	41	49

因为 T1、T2、T6、T7、T8 这 5 个序列缺失值数量较少，因而直接用平均值填充来进行分析，而 T3、T4、T5 序列的缺失值较多，应在后续的模型建立过程中额外考虑。

现在对 8 个序列绘图并观察，由图 1所示。

根据图像，8 个序列形态各异，因此考虑分别建立模型进行预测。

6.3.1 T1、T2、T6、T7、T8 序列基于 ARIMA 模型的预测

以 T1 序列为例，观察 T1 序列图像，由图 2所示，可以隐约感受到其作为时间序列的周期性与平稳性特征，因此尝试使用 ARIMA 模型 [3] 进行建模。

为此，计算序列的自相关与偏自相关并可以得到，T1 的自相关周期振荡，而偏自相关在 1 之后都接近于 0，这提示我们 T1 序列可能符合模型 MA(1)。拟合该模型并得到后 24 周的预测值，见图 3。

类似地，我们可以建立对 T2、T6、T7、T8 序列的 ARIMA 模型并得到预测。

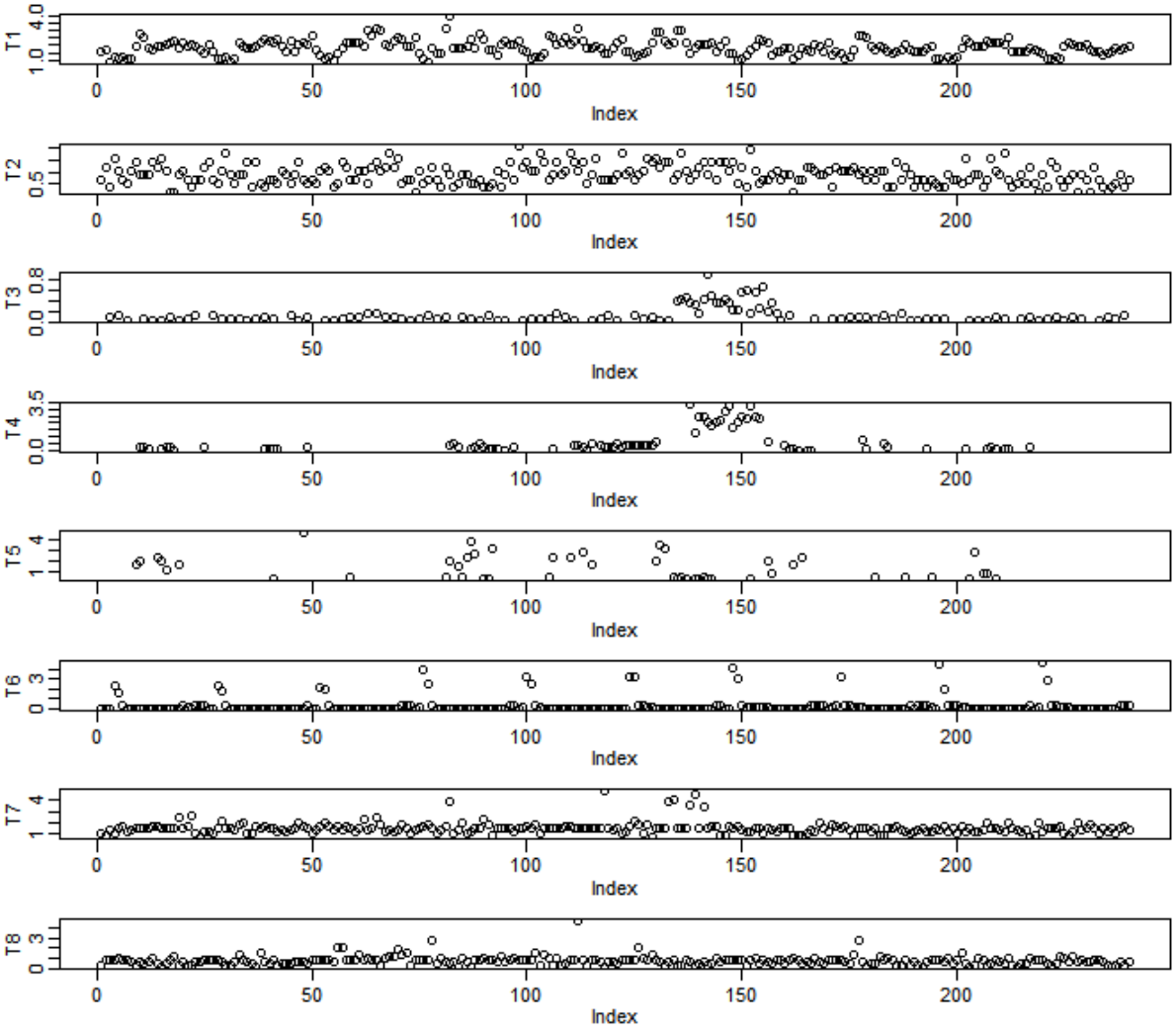

图 1: T1 至 T8 序列图像

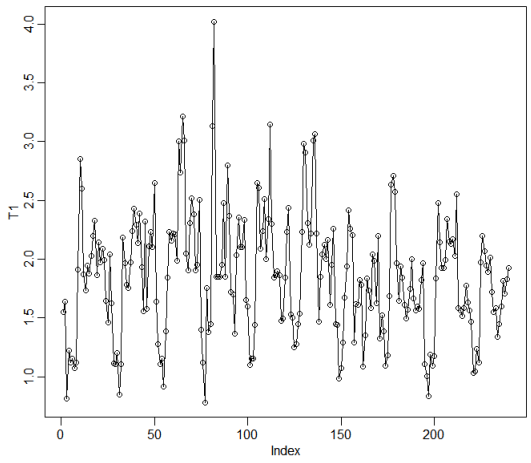

图 2: T1 序列

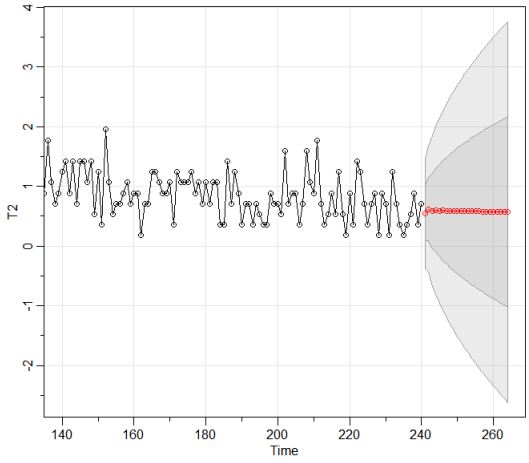

图 3: T1 后 24 周预测

6.3.2 T3、T4 序列基于 HMM 模型的预测

以 T3 为例，仍然首先观察序列图像，由图 4所示。

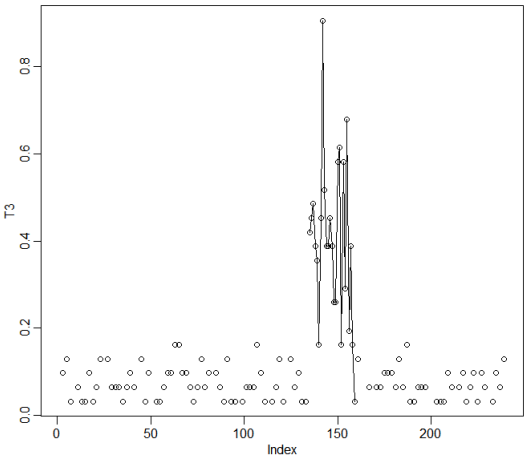

图 4: T3 序列

对于该序列（T4 具有类似特征），有 2 个注意点：

1. 序列具有较多缺失值，正因为如此，图像中只有中间部分由折线连接。

2. 序列在 150 周附近时值突然变得较大。

为解释第 2 点，设想 T3 转运商可能在第 150 周附近时某种状态发生了改变，导致其运输损耗率偏高。根据这一设想，我们可以使用隐马尔可夫模型（HMM）[3] 进行建模，将这种状态作为隐状态，并假设在每一状态下损耗率服从高斯分布。这样，该模型可以较好地解释 T3 转运商损耗率为何发生突变。不仅如此，HMM 作为状态-空间模型的一种，可以应对缺失值情形，由此可见使用 HMM 建模是较为合理的。

在对历史数据进行拟合后，可以得到其对应的状态，如图 5所示。

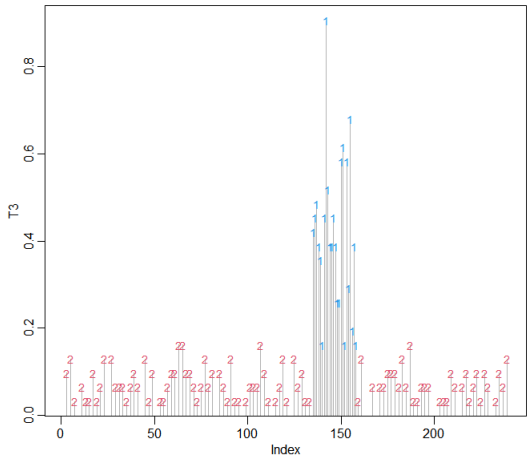

图 5: T3 序列状态

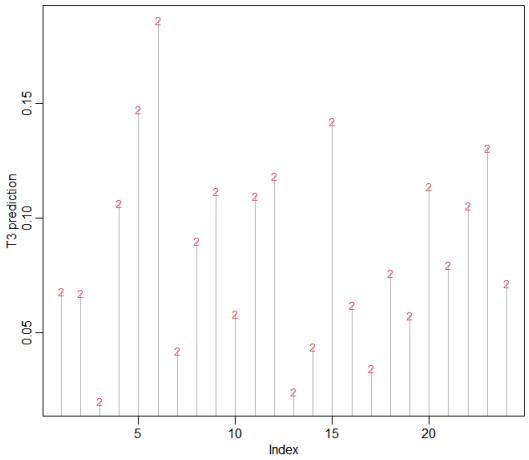

图 6: T3 序列后 24 周预测

在对历史数据的拟合中，模型能够完美将 150 周附近的数据识别为状态 2，其余数据识别为状态 1，符合预期解释。后续 24 周预测见图 6。

对 T4 序列的建模与预测是完全类似的，这里将不作展示。

6.3.3 T5 序列分析

这里将 T5 序列单独列出，是由于其特征过于独特。首先，该序列缺失值数量达到 191，缺失占比接近 80%；其次，观察其图像（7），难以识别具体的模式，具有较强的随机性。这样，很难找到合适的时间序列模型来拟合它，因此我们将其看做是一个随机分布来分析。

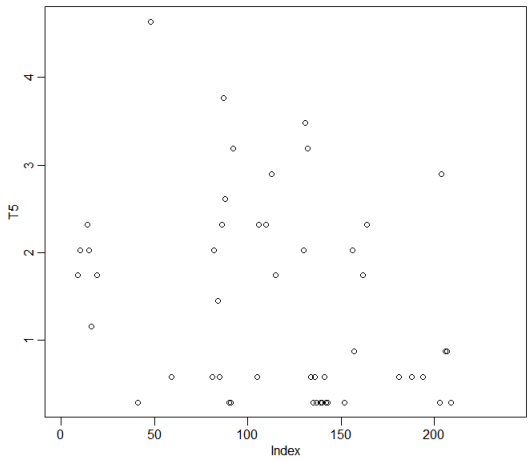

图 7: T5 序列

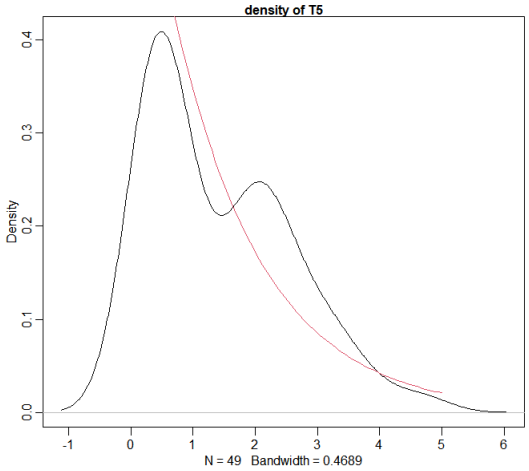

图 8: T5 序列核密度估计与指数分布逼近

现研究去除缺失值后的核密度估计，见图 8黑线。因数值一定大于 0，考虑用指数分布来逼近，容易计算参数 λ 的极大似然估计 $\lambda = 0.701448$，逼近效果如图 8红线所示。后 24 周预测则简单使用随机数生成器生成。

6.4 转运方案的 0-1 规划问题求解

根据上述决策过程，已经确定了：未来 24 周的订购方案，转运商的转运损耗率。在上述前提下，需要确定供应商和转运商的指派关系。考虑该指派矩阵，为 0-1 矩阵，因此可以将该优化问题视作 0-1 规划问题求解。

符号	说明	单位
a_{kj}	第 k 家转运商第 j 周的损耗率	%
S_{ij}	第 i 家供应商商第 j 周的供应量	m^3
T_{ijk}	第 i 家供应商商第 j 周与第 k 家转运商之间的指派关系	-

定义了上述符号后，考虑该 0-1 规划问题的约束条件：

1. 指派矩阵为 0-1 矩阵。该条件等价于 $T_{ijk} \in \{0, 1\}$.

2. 每一家供应商所供应的原材料由一家转运商转运。该条件等价于：$\sum_k T_{ijk} = 1$.

3. 每一家转运商的转运量不超过其最大转运限制。该条件等价于：$\sum_i Tijk \leq 6000$.

规划的优化目标为：最小化损耗量, 因此总体的 0-1 规划为:

$$\begin{aligned} \textbf{min} \quad & \sum_{ijk} S_{ij} \times T_{ijk} \times a_{kj} \\ \textbf{subject to} \quad & \sum_{k} T_{ijk} = 1 \\ & \sum_{i} T_{ijk} \times S_{ij} \leq 6000 \\ & T_{ijk} \in \{0, 1\} \end{aligned}$$

对于上述 0-1 规划问题，采用 LINGO 软件包进行求解，求解部分的代码详见附录。显然该问题的全局最优解存在，且 LINGO 确实找到了全局最优解。

6.5 问题二的效果分析与蒙特卡罗模拟检验

对于问题 2，我们分别得到了两方面的实施方案：对于 4 家供应商每星期的订购量；将这 4 家供应商的供应原材料分配至各转运商的具体安排。可以认为，整个建模和求解的过程基本合理，4 家供应商选择合理、订购量在合理范围；对于各转运商损耗率的预测合理，在其上得到了满足需求而综合损耗率最低的方案。

此处将进一步地考虑相关的定量分析。此问题中的两个方案均是在考虑了外部随机因素的情况下得到的，例如供应商具体供应量的波动、根据转运商的损耗率的可能变化进行选择。由于随机因素的存在，可能难以确定性的分析给出的方案的具体性质。因此，我们选择蒙特卡罗模拟的方法来检验方案的实施效果。根据本题题目的需要，我们进行两次相对独立的蒙特卡罗模拟：首先是不考虑运输损耗量，研究供应量的波动对于整体生产过程的影响；第二部分是固定各月供应量，计算按照转运方案在转运过程中的损失率。

6.5.1 供应量波动对生产过程的影响

此处延续" 基于概率优化模型的订购方案制定'' 一节中的相关假设，并代入对应的求解结果。认为各周、各公司供货量呈独立的正态分布，供货量均值是求解结果中对应量，标准差同样认定为定值 100（将在后续内容中进一步讨论）。此子问题中不需要考虑储存、运输成本。模拟程序首先初始化变量 $cost = 0, store = 56400$，对应初始 0 成本、两周库存。此处 “库存” 实际上是 A，B，C 原材料按照各自比例生产产品的量，因为不需要考虑储存从而不需要考虑 A，B，C 分别的储存量。此后对于每一周进行模拟：

1. 从库存中扣除每周产量 28200

2. 按照 4 家供应商的供货情况取正态分布随机数，出现小于 0 或大于 6000 则置为各自边界量；成本按照比例加入 *cost*。

3. 按 A，B，C 材料比例增加 *store* 库存变量

4. 若出现 *store* 低于 56400，则报错退出。

我们对于此程序进行了一万次循环，其中没有出现任何一次报错退出；即完全满足了库存量相关约束。同时，统计了 24 周结束后的总成本和总储存量，如图 9。考虑到每周的产量固定，显然总成本与总储存量线性相关。总成本均值为 5.0497×10^5，方差为 1.1252×10^3 ，相对稳定。综上，可以认方案有实施效果，且极具稳定性。

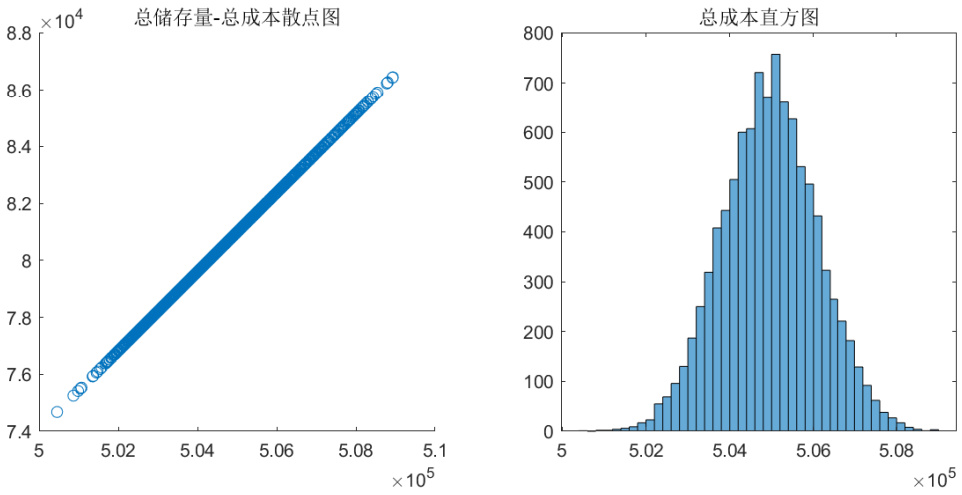

图 9: 蒙特卡罗模拟得到数据

6.5.2 转运方案综合损失率的计算

类似地，建立在求解过程中各种假设的基础上，我们进行一万次了对各转运商、各周损耗率的波动的模拟。

在波动模拟过程中，考虑到 ARIMA 模型的预测是稳定且收敛的，为节省计算，使用该模型的 5 条序列损耗率不再模拟。而 T3、T4 的 HMM 模型及 T5 的指数随机分布，因为随机性较强，可以进行模拟。对于 T5，只需随机生成随机数；对于 T3 和 T4，根据初始状态和转移概率矩阵，则可以不断生成不同的 Markov 链，从而完成模拟。

而认定供应量完全按照订购方案的理想情况供应的基础上，我们计算出了 4 家供应商 24 周由转运损耗导致的损耗量的统计数值（详见附录）。相对计划供应损失从-5 到-75 不等，各自有一定波动；每个供应商的平均损耗量为

$$\mu = [-12.4215, -20.5948, -20.3157, -19.2731]$$

对于目前的结果，可以认为总损耗量相比原供应量较小，可以接受；但损耗量的总数是不可忽略的，在问题三中，将考虑相关问题，进行优化。

七、基于稀疏性模型的多目标凸概率优化（问题三）

7.1 端到端的规划问题建模

在问题二中，建立了多步骤的企业规划模型，在问题三中我们尝试将其推广为端到端的规划模型。也即从问题二的先确定订购方案的决策变量，再确定转运方案的决策变量，改为同时确定订购方案和转运方案的决策变量，端到端的规划模型，由于决策变量的自由度更高，很有可能找到更优的解，但相应的求解难度也增大了。未了降低求解难度，在端到端的规划问题中去掉了概率规划产生的不确定性度量，也即假设供应商的满足率恒为 1，也即 $S = O$。

定义如下的符号：

符号	说明	单位
a_{kj}	第 k 家转运商第 j 周的损耗率	%
S_{ij}	第 i 家供应商第 j 周的供应量	m^3
R_{ij}	第 i 家供应商第 j 周供应的原材料经过损耗后的剩余量	m^3
T_{ijk}	第 i 家供应商商第 j 周与第 k 家转运商之间的指派关系	-
d_{ij}	第 j 周企业储存的第 i 家供应商的原材料等效的产能	m^3
b_{ij}	第 j 周企业储存的第 i 家供应商的损耗的产能	m^3
w_i	每家供应商对应的原材料的采购单价比	-
u_i	每家供应商对应的原材料所等效的产能比	-
$StoreCost$	总储存成本	元
$TransportCost$	总运输成本	元
$OrderCost$	总订购成本	元
α, β, γ	权重	元

最终的优化目标为最小化企业的成本，共包括原材料的订购、转运、储存成本。即最小化：

$$\alpha OrderCost + \beta TransportCost + \gamma StoreCost$$

由于题目中没有给定订购、转运、储存成本之间的比例，可以设置 $\alpha = \beta = \gamma = 1$。

根据约束条件和优化目标，有：

$$
\begin{aligned}
&\textbf{minimize } \alpha OrderCost + \beta TransportCost + \gamma StoreCost \\
&\textbf{subject to } OrderCost = \sum_{ij} w_i S_{ij} \\
&\qquad TransportCost = \sum_{ij} S_{ij} \\
&\qquad StoreCost = \sum_{ij} \frac{d_{ij}}{u_i} \\
&\qquad R_{ij} = \sum_{k} S_{ij} \times T_{ijk} \times (1 - a_{kj}) \\
&\qquad \sum_{k} T_{ijk} = 1, T_{ijk} \in \{0, 1\} \\
&\qquad \sum_{i} T_{ijk} \times S_{ij} \leq 6000 \\
&\qquad \sum_{i} d_{ij} \geq 56400 \\
&\qquad d_{ij} = d_{i,j-1} + u_i R_{ij} - b_{ij}, j > 1 \\
&\qquad \sum_{i} b_{ij} = 28200
\end{aligned}
$$

上述规划问题为混合整数二阶问题（MIQP），且决策变量过多，比较难找到在限定时间内求解的求解器，单仍然不失为一个参考的基准模型。下面基于假设 1 的独立性假设，拆分企业的决策规划过程，在问题二的基础上建立改进版的多步骤规划模型。

7.2 基于范数惩罚和改进概率规划的多步骤规划

这一节，我们仍沿用问题二的流程与方法，即先求解订购方案的规划问题，再据此方案进行转运商的 0-1 规划。但在问题二的基础上，针对供货量这个不确定因素，我们据 402 家企业的情况建立统计模型，对原有的方法进行改进。而对于该多目标规划问题，我们采用 1 范数惩罚的方式，改进原本的优化目标。

7.2.1 订购方案的制定

总体求解思想仍与问题二相同，即在优化模型中加入概率元素。对于规划目标，题目要求尽可能多地订购 A 而尽可能少地订购 B。对于企业而言，成本必然是首要优化的目标，因此考虑在采购成本的基础上加上对 B 和 C 类原材料采购的惩罚项，即类似于 lasso 的方法。就惩罚项的形式而言，我们使用每周 B/C 类订购总量组成的 24 维向量的 1 范数，因为 1 范数更倾向于导致某些系数为 0，即不做采购。

对于约束条件，首先放开 4 家供应商的限制，将采购范围扩大到全部 402 家。此外，这 402 家供应商的地位显然是不同的，为了在优化模型中体现这一点，我们需要对每家供应商的采购量进

行限制，并在保证模型简单性的同时对相应的供应量作出更好的估计。基于此，沿用问题二制定订购方案时的符号，模型需要符合如下条件：

1. 误差项 ϵ_i 仍服从高斯分布。

2. 使用每家供应商历史订购量的 0.95 分位数作为上界，以保证订购量较少的那些供应商不会喧宾夺主。

3. 根据历史数据对供应商进行分类，供应量较少的供应商为一类，供应量较多的为另一类。对于前一类供应商，将其供应量求和后视作单独一个供应商，并计算相应的方差。

4. 对于供货量的估计，使用 $y_i = x_i + shift_i + \epsilon_i$ 作为估计，其中 $shift_i$ 是供应量对订购量的偏移，其值由历史数据中供应与订购残差的平均值来计算得到；ϵ_i 的方差 $\sigma_{\epsilon i}$ 由残差的方差来计算得到。

假定问题二中的优化目标为 Obj，加入范数惩罚后的优化目标为 $Obj' = Obj + \lambda\|O_C\|_1 + \mu\|O_B\|_1(\lambda > \mu)$. 其中 O_B 和 O_C 分别表示两类原材料的订购量矩阵，在该问题的求解中，设 $\lambda = 0.1, \mu = 0.05,$。经过实验验证，该参数的设置对解影响很小，对于该惩罚项参数的稳健性详见模型的稳健性分析一节。

由于优化问题与问题二极为相似，这里不再列出，详见附录代码。本节数据的预处理使用了统计软件 R，而优化问题则利用 MATLB 的 cvx 工具箱成功求解。

7.2.2 转运方案的制定

给定订购方案之后，求解与问题二中相同的 0-1 规划即可得到转运方案。但考虑到求解 402 家企业的转运方案的全局最优解的复杂度较高，考虑在 LINGO 求解器中加入时间限制，求解器给出的 0-1 规划的下界为 $129188.9m^3$, 而当前找到的最优的可行解为 $129191.1\ m^3, 2m^2$ 的误差实际上对结果并没有过大的影响，该结果距离求解器给出的最优下界仅有 0.0015% 的误差，在精度要求内可以忽略不计，因此可以认为求解器找到了最优解。

7.3 问题三的方案效果分析与蒙特卡罗模拟检验

类似于问题 2，本节中对于问题 3 也同样采用蒙特卡罗方法进行检验；但此处由于同时考虑了订购与转运两方面的内容，将需要在同一次模拟中模拟两方面内容的随机性。基于前文的假设，此处利用建模时对于供货商的分类、关于供货量变化的假设，以及问题二中已得到的 10000 条损耗率波动模拟数据。共进行 10000 次试验，依次取损耗率波动数据中的一条，并生成供货量数据。对于这两方面的处理基本与问题二对应部分相同。但有所不同的时此处分别考虑了 A、B、C 原料的储存和运输，仅在生产时、计算采购成本时用对应系数修正。另外，还需要考虑有多种材料库存时，该用哪一种材料的问题；实际上由于储存成本的性质，只需要按照生产等量产品消耗量顺序，依此使用 C、B、A 即可。

实际程序编写中，供货量的生成过程中以订购量-$shift$ 为均值，解题过程中统计标准差作为标准差。而对于解题中进行的小供应商合并为大供应商的情况，此处将统计的 n 个小供应商的合

计标准差 σ 平均分拆给各小供应商，标准差变为 $\frac{\sigma}{\sqrt{n}}$。考虑到小供应商的供应量很少，这样的做法合理。

模拟的具体流程为：

1. 每次外层循环中，提前生成 24 周的供应量、载入损耗率模拟数据。库存数量设为 56400*0.6 单位的 A 材料，即恰能满足库存约束。
2. 计入库存费用；
3. 从库存中扣除每周产量 28200；库存按照 C、B、A 顺序依此扣去；
4. 供应量的成本按照比例加入 *cost*。；
5. 按 A，B，C 材料比例增加 *store* 库存变量；
6. 若出现 *store* 低于 56400，则报错退出。

本次试验共进行 10000 次；其中出现了 536 次未满足库存条件而退出的情况，即方法的可靠率为 94.54%。总成本的均值为 5.2179×10^6，方差为 3.0547×10^5。下图 10分别展示了 24 周生成生产结束后的总成本与各库存结余情况，以及总成本的分布。生产中主要订购 A, 与总成本有着显著的线性关系，但有少部分的 B、C 分布。基于此结果，可以认为我们得到的方法比较稳健，且即使考虑到了供货量波动和随机转运损耗，仍可以有一定的库存结余。

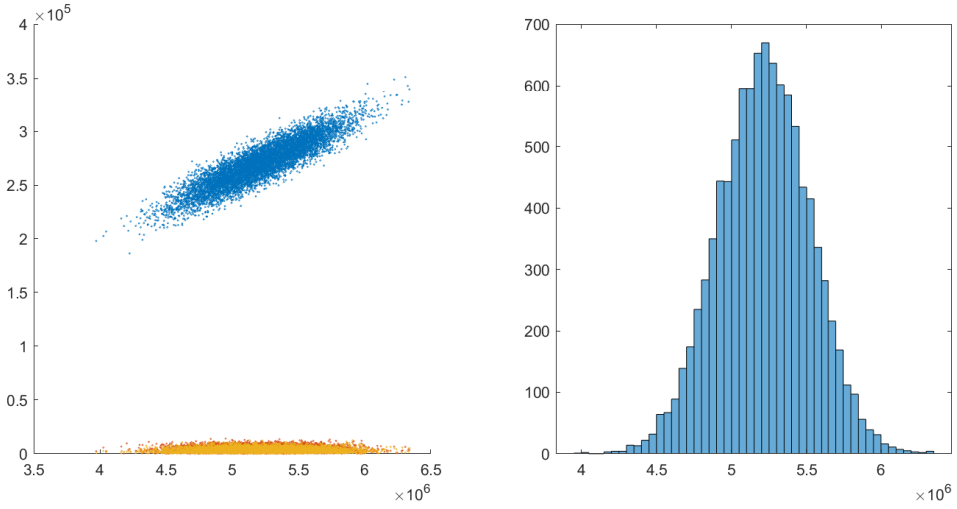

图 10: 问题三整体生产过程的蒙特卡罗模拟

八、基于供应商潜力估计的二分搜索算法（问题四）

8.1 基于二分法的求解方法

利用二分法的思想，可以将问题四转化为问题三。根据转运损耗率与供应量无关的假设，可以将订购方案的决策和转运方案的决策视作两个独立的决策过程，仍然考虑多阶段规划模型。问

题四要求寻找企业可以提升的最大产能，本质上可以转化为一个存在性问题，等价于寻找使得问题二和问题三中的订购规划问题存在可行解的最大产能。显然，该产能满足二分法中的单调性条件，因此只要给定二分法的上下限，便可以使用问题二或问题三中的规划模型作为二分法的存在性判断的子程序。

二分法的下限为当前的产能 $M = 28200$，而其上限为一先验条件。简单地，假设最大的产能可以提升为当前产能的 N 倍，则二分法的上限为 MN. 设规划模型的复杂度为 T，则该二分法只需要在 $\log N \times T$ 的时间内就可以完成。

8.2 基于供应商统计模型的供应商潜力估计

供应商的供应量，为企业产能提高的限制之一，因此对于供应商供应能力的估计在该问题中较为关键。上述基于二分的方法，在求解最优的订购方案的时候，沿用了问题三中给出的供应商供应量的概率模型：本质上是将其 5 年内的供应量的经验分布函数作为其概率分布的估计。由于经验分布函数依概率收敛于真实分布函数，因此上述方法有其合理性。但使用经验分布函数将导致推断出的供应商最大的可能的供应量不可能超过历史的最大供应量，在某种程度上是对供应商的供应潜力的一种限制。因此在本节中，考虑重新为供应商的供应量建立统计模型。

改进后的统计模型建立在问题三的模型的基础上，也即供应商已经按照其供货特征被分为几类，下面分别对其建立不同的统计模型：

1. 对于极其不重要的供应商，供应量矩阵具有很大的稀疏性（零元很多），由于供应量为非负数且均为离散值，符合泊松分布的特性。因此，将这部分供应商用泊松分布建模。

2. 对于其他的供应商，假设其供应量服从高斯分布。

采用极大似然估计可以得到分布函数中的参数，从而给出所有 402 家供应商的统计模型。该改进手段的优点是使得最大供应量可以突破历史最大供应量的限制，从而使得对供应商的潜力的估计更为合理。

给定置信度 α, 选取每家供应商的分布函数的上 $1-\alpha$ 分位数作为其在该置信度下的供应潜力估计。将新的限制条件代入问题三的概率优化模型中求解，同样使用二分法寻找最大产能，可以得到若供应商的潜力发生改变，在给定置信度下的最大产能。调用二分方法可以发现，当企业的产能为 7.64 万 m^3 的时候不存在可行解，但当产能为 7.65 万 m^3 的时候规划模型可以找到相应的解，因此二分法给出的最大产能为 7.64 万 m^3.

8.3 基于线性规划的转运方案制定

问题二和问题三中转运方案的制定为 0-1 规划模型，但在问题四中，由于企业的产能得到提升，问题二和问题三中假设每一家供应商所供应的原材料仅由一家转运商转运的条件不应再成立。根据上面的订购方案的求解结果，存在着某些供应商的单周供应量超过每家转运商的 $6000m^3$ 的最大转运容量限制，需要由多家转运商进行转运。因此在问题四的转运方案的制定过程中，应该去除该限制，该改动使得原本的 0-1 规划问题变为更易于求解的线性规划问题。

在问题二和问题三所定义的符号的基础上，将原本的指派矩阵 T 重新定义为转运商的决策矩阵，也即矩阵元素 T_{ijk} 表示第 i 家供应商在第 j 周由第 k 家转运商所承担的转运量。

根据上述分析，问题四所需要求解的线性规划为：

$$\begin{aligned} \textbf{minimize } & T_{ijk} \times a_{kj} \\ \textbf{subject to } & \sum_{k} T_{ijk} = S_{ij} \\ & \sum_{i} T_{ijk} \leq 6000 \end{aligned}$$

使用 LINGO 求解，代码与结果详见附录和支撑材料。

九、模型的分析

9.1 稳健性分析

在上述建立的数学模型中，对于问题中没有给定的数值或者超参数，我们经过分析进行相对合理地赋值。但这部分超参数会影响模型的表现能力，但在这一节中我们验证了我们所建立的数学模型对这些超参数是不敏感的，因此我们所建立的模型是鲁棒的。

9.1.1 问题二订购方案的稳健性

在问题二订购方案的优化模型中，标准差被设定为 100，这里我们将变化这个值进而考察最优值并计算平均每周最优解与基准模型的 frobenius 范数来衡量最优解的差距。计算结果如下表所示，可以看到最优值（采购成本）随着标准差的增加而增加。这是符合预期的，因为标准差越大，为了保证库存达到需求的概率，就必须增加订购量。而对于最优解的差距，考虑到每周约 1.7 万的订购量，即便是 1000 情形的 600，相对波动约为 3.5%，并不是很大，表明模型相对稳健。

序号	标准差	最优值	范数
1	10	489062	60.12
2	100	504960	0
3	200	522624	66.81
4	500	575616	267.23
5	1000	663936	601.28

9.1.2 问题三订购方案的稳健性

相对于问题二，问题三的优化模型中加入了对订购 B、C 类材料的的惩罚项，这里将变化惩罚项前的系数来分析模型的稳健性。同样考察最优值和最优解与基准模型的差距，结果如下表所示。可以看到最优值几乎不改变，最优解差距仅为个位数，表明模型具有极强的稳健性。

序号	系数	最优值	范数
1	0.1,0.05	600526	0
2	0.05,0.1	600527	10.22
3	0.5,0.1	600526	3.18
4	1,0.5	600526	8.30
5	10,1	600526	6.95

9.2 误差分析

1. 对于订购方案的决策，将问题转化为凸优化问题，找到的解即为全局最优解，所得解与理论最优解的误差仅仅来自于取决于迭代方法的停止准则产生的数值误差。

2. 对于 0-1 规划问题，问题二中制订的转运方案找到了全局最优解，而问题三中给出的解与算法给出的下界之间仅有 0.0015% 的最大误差，可以忽略不计。

3. 尽管建立的订购量-供应量的相关模型和转运商的损耗率模型（如 HMM 模型）等存在随机变量产生的误差，但经过蒙特卡罗方法模拟可见上述误差很小，对解的可行性几乎没有影响。

十、模型的评价与推广

10.1 模型的优点

1. 使用集成学习模型衡量企业重要性，有效降低了模型的偏差。

2. 分别使用 ARIMA、HMM、指数分布模型等具有针对性的模型预测不同特点的转运商的损耗率。

3. 所建立的模型对于超参数的设置不敏感，详见稳健性分析一节。

4. 创新型地建立了移动加权平均模型用于供应量的回归，本质上类似于高斯去噪过程 [1]，有效地降低了噪声数据的影响。

5. 采用概率优化模型，定量地衡量优化中的不确定性因素。

6. 使用蒙特卡罗方法，利用统计模拟实验有力地说明结果的有效性。

10.2 模型的缺点

1. 在优化模型中，为了适应概率的复杂度，在估计供应量方面简单地使用一些线性的方法，并且误差项均假设为同方差的高斯噪声，这与实际不尽相符。

2. ARIMA 模型的预测一般会收敛到一个值，因此不太适合做长时间的预测。

3. 0-1 规划为指数型复杂度，不适用于推广到供应商数目过多的情况，在该情况下应该优先使用基于剪枝和贪心的算法寻找候选供应商。

10.3 模型的推广

1. 本文所建立的模型针对的问题是极具代表性的，可以推广到各类企业的决策过程中。

2. 模型对于不确定因素的量化，对于随机环境下的企业决策具有参考意义。

3. 对供应商、转运商建立的统计模型可以运用到其他相似的具有时序特征、相关关系、稀疏性特点的数据中。

参考文献

[1] Rafael C Gonzale. Digital image processing third edition, 2002.

[2] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning*, volume 112. Springer, 2013.

[3] Robert H Shumway, David S Stoffer, and David S Stoffer. *Time series analysis and its applications*, volume 3. Springer, 2000.

附录

附录 A：提交结果

- 附件 A 订购方案数据结果.xlsx
- 附件 B 转运方案数据结果.xlsx

附录 B：支撑材料列表

- data: 文件夹下存放数据文件，包括程序的 IO 数据和中间结果
- prepocess: 文件夹下存放数据格式转换，数据预处理相关的文件
- LINGO: 存放.lg4 文件，详见附录 C
- R: 存放.R 文件，详见附录 C
- MATLAB: 存放.m 程序文件和.mat 矩阵数据，详见附录 C
- Python: 存放.py 文件，详见附录 C

附录 C：代码

运行环境说明

- .R 文件在 R 4.0.3 版本上可以运行
- .m 文件在 matlab R2020a 版本上可以运行
- .py 文件在 Python3.8 版本上可以运行
- .lg4 文件在 LINGO18.0 版本上可以运行

代码文件说明

- p1-cap.py: 可视化历史 5 年内的原材料储存量曲线。

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

#读取数据并输出

sExcelFile_1 = 'data\data1.xlsx'
sExcelFile_2 = 'data\data2.xlsx'
df_order = pd.read_excel(sExcelFile_1,sheet_name=0,engine='
    openpyxl')
df_supply = pd.read_excel(sExcelFile_1,sheet_name=1,engine='
    openpyxl')
df_transport = pd.read_excel(sExcelFile_2,sheet_name=0,engine='
    openpyxl')

array_order = np.array(df_order)[:-2,:]
# 出现两行NAN值, 预先去除
array_supply = np.array(df_supply)
array_transport = np.array(df_transport)

# ABC三类原材料
#print(array_supply[:,1])

print(array_supply.shape) #(402,242)

# 第一列: 名称
# 第二列: 类别

array_cap = np.zeros(240)

cap = 0
for j in range(2,242):
    for i in range(0,402):
        ty = array_supply[i,1]
        if ty == 'A':
            cap += array_supply[i,j] / 0.6
        elif ty == 'B':
            cap += array_supply[i,j] / 0.66
        elif ty == 'C':
```

```
            cap += array_supply[i,j] / 0.72
    cap -= 28200
    array_cap[j-2] = cap
print(array_cap)
plt.plot(array_cap)
plt.show()

cap = 0
for j in range(2,242):
    for i in range(0,402):
        ty = array_order[i,1]
        if ty == 'A':
            cap += array_order[i,j] / 0.6
        elif ty == 'B':
            cap += array_order[i,j] / 0.66
        elif ty == 'C':
            cap += array_order[i,j] / 0.72
    cap -= 28200
    array_cap[j-2] = cap
print(array_cap)
plt.plot(array_cap)
plt.show()
```

- C1-clustering.R：问题一中衡量供应商重要性的层次聚类模型。

```
d <- read.table('DS/data/d.txt', header = F)
s <- read.table('DS/data/s.txt', skip = 1, header = F)

sA <- s[s[2]=='A',]
sB <- s[s[2]=='B',]
sC <- s[s[2]=='c',]
sA.w <- sA[,c(-1,-2)]/0.6
sB.w <- sB[,c(-1,-2)]/0.66
sC.w <- sC[,c(-1,-2)]/0.72
s.w <- s
s.w[s[2]=='A',c(-1,-2)] <- sA.w
s.w[s[2]=='B',c(-1,-2)] <- sB.w
s.w[s[2]=='C',c(-1,-2)] <- sC.w

feature <- function(s) {
```

```
 s.mean <- apply(s[,c(-1,-2)], 1, mean)
 s.rate <- s[,c(-1,-2)]/d[,c(-1,-2)]
 s.rate[is.nan(as.matrix(s.rate))] <- NA
 s.rate.mean <- apply(s.rate, 1, mean, na.rm = T)
 s.rate.var <- apply(s.rate, 1, var, na.rm = T)
 return(cbind(s.mean, s.rate.mean, s.rate.var))
}
s.w.feature <- feature(s.w)

X <- scale(s.w.feature)
X[1] <- X[1]
hc.complete <- hclust(dist(X), method="complete")
hc.average <- hclust(dist(X), method="average")
opar <- par(no.readonly = T)
par(mfrow=c(1,2))
plot(hc.complete, main="Complete Linkage",
     xlab="", sub="", cex=.9)
plot(hc.average, main="Average Linkage",
     xlab="", sub="", cex=.9)
par(opar)
class.c <- cutree(hc.complete, 2)
table(class.c)
aggregate(s.w.feature, by=list(cluster=class.c), median)

class.a <- cutree(hc.average, 3)
table(class.a)
aggregate(s.w.feature, by=list(cluster=class.a), mean)
```

- p1-imp.py: 使用集成学习模型对企业重要性进行评价，代码中包括了供应量优先模型、熵权模型、聚类模型的实现，机器学习模型部分使用了 sklearn 机器学习包。

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
#读取数据并输出

sExcelFile_1 = 'data\data1.xlsx'
sExcelFile_2 = 'data\data2.xlsx'
df_order = pd.read_excel(sExcelFile_1,sheet_name=0,engine='
```

```
    openpyxl')
df_supply = pd.read_excel(sExcelFile_1,sheet_name=1,engine='
    openpyxl')
df_transport = pd.read_excel(sExcelFile_2,sheet_name=0,engine='
    openpyxl')

# 出现两行NAN值, 预先去除
array_order = np.array(df_order)[:-2,:]
array_supply = np.array(df_supply)
array_transport = np.array(df_transport)

#根据种类进行归一化
def scale_by_type(array):
    for i in range(0,402):
        ty = array[i,1]
        if ty == 'A':
            array[i,2:] = array[i,2:] / 0.6
        elif ty == 'B':
            array[i,2:] = array[i,2:] / 0.66
        elif ty == 'C':
            array[i,2:] = array[i,2:] / 0.72
    return array

array_order = scale_by_type(array_order)
array_supply = scale_by_type(array_supply)

def min_max_scale(a):
    a = (a - np.min(a)) / (np.max(a) - np.min(a))
    return a

#供应量优先模型: 根据总供应平均供应量确定企业重要性
def method_sum_supply():
    # 根据总供应量选择重要的企业
    sum_supply = np.mean(array_supply[:,2:],1)
    sum_supply = min_max_scale(sum_supply)
    sort_score = np.argsort(sum_supply)
    imp = sort_score[-50:] + 1
    return sum_supply, imp
```

```
def get_features():

    # 特征提取
    rate = array_supply[:,2:] / (array_order[:,2:] + 1e-12)
    rate[rate>1e8] = None

    mrate = np.mean(rate,1)
    # 用方差衡量稳定性
    var = 1 - np.var(rate,1)
    msupply = np.mean(array_supply[:,2:],1)

    print(msupply.shape)

    # 可以选择做聚类...

    mrate = min_max_scale(mrate)
    var = min_max_scale(var)
    msupply = min_max_scale(msupply)

    #生成特征矩阵

    features = np.zeros((402,3))
    features[:,0] = msupply
    features[:,1] = mrate
    features[:,2] = var

    return features

#基于熵确定权重
def method_end():
    # 特征提取

    features = get_features()
    #print(features)
    k=1/np.log(3)
    yij=features.sum(axis=0)
    pij=features/yij

    #计算pij
```

```
    test=pij*np.log(pij + 1e-5)
    test=np.nan_to_num(test)
    ej=-k*(test.sum(axis=0))
    #计算每种指标的信息熵
    wi=(1-ej)/np.sum(1-ej)
    #计算每种指标的权重

    #wi = np.array([1,0,0])
    scores = np.sum(features * wi,1)
    #print(scores)
    sort_score = np.argsort(scores)
    imp = sort_score[-50:] + 1

    return scores, imp

def method_cluster():
    features = get_features()
    kmeans = KMeans(n_clusters=3)
    kmeans.fit(features)
    y_kmeans = kmeans.predict(features)

    index = np.argwhere(y_kmeans==y_kmeans[229-1])
    index = index.squeeze() + 1
    print(index)
    return index

def intersect(a,b):
    ans = 0
    for aa in a:
        for bb in b:
            if aa == bb:
                ans += 1
                break
    return ans

if __name__ == '__main__':
    sc1,imp1 = method_sum_supply()
    sc2,imp2 = method_end()
```

```
    sc = sc1 + sc2
    imp = np.argsort(sc)[-50:] + 1
    print(imp)

    #print(intersect(imp1,imp2))

    # 比较两种方法的的得分
    # 相等的共有41家企业

    #imp3 = method_cluster()
    #print(imp3)
    #print('number of imp3',len(imp3))
    #print(intersect(imp2,imp3))
```

- p2-num.py: 问题二中最少供应商数目的求解，基于剪枝过滤后贪心算法的代码实现。

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.mixture import GaussianMixture as GMM

#读取数据并输出

sExcelFile_1 = 'data\data1.xlsx'
sExcelFile_2 = 'data\data2.xlsx'
df_order = pd.read_excel(sExcelFile_1,sheet_name=0,engine='
    openpyxl')
df_supply = pd.read_excel(sExcelFile_1,sheet_name=1,engine='
    openpyxl')
df_transport = pd.read_excel(sExcelFile_2,sheet_name=0,engine='
    openpyxl')

# 出现两行NAN值，预先去除
array_order = np.array(df_order)[:-2,:]
array_supply = np.array(df_supply)
array_transport = np.array(df_transport)

#根据种类进行归一化
```

```
def scale_by_type(array):
    for i in range(0,402):
        ty = array[i,1]
        if ty == 'A':
            array[i,2:] = array[i,2:] / 0.6
        elif ty == 'B':
            array[i,2:] = array[i,2:] / 0.66
        elif ty == 'C':
            array[i,2:] = array[i,2:] / 0.72
    return array

#进行等效化
array_order = scale_by_type(array_order)
array_supply = scale_by_type(array_supply)

def min_max_scale(a):
    a = (a - np.min(a)) / (np.max(a) - np.min(a))
    return a

#scale过后就不需要考虑type信息,先将type信息记录在相应的数组中
array_type = array_supply[:,1]
#print('type',array_type)
array_order = array_order[:,2:]
array_supply = array_supply[:,2:]

def greedy_search(sort_max_supply,max_valid_supply):
    i = 401
    s = 0
    selected_company = []
    while s < 28200:
        index = sort_max_supply[i]
        s += max_valid_supply[index]
        i -=1
        selected_company.append(index+1)

    #print(selected_company)
    return selected_company

def greedy_method():
```

```
# 数据稀疏性分析
#print(np.sum(array_supply[:,2:]>=0))
#print(np.sum(array_supply[:,2:]>0))

mean_valid_supply = np.zeros(402)
for i in range(0,402):
    s = 0
    cnt = 0
    for j in range(0,240):
        if array_supply[i,j]>0:
            s += array_supply[i,j]
            cnt += 1
    #print(cnt)
    mean_valid_supply[i] = s / cnt

# 尝试使用贪心算法求解，使用有效的均值和最大值分别衡量

#print(np.sort(mean_valid_supply))
#print(np.sort(max_supply))

max_valid_supply = np.max(array_supply,1)
maxA = 6000 / 0.6
maxB = 6000 / 0.66
maxC = 6000 / 0.72

#最大供应量需要受到限制
for i in range(402):
    ty = array_type[i]
    if ty == 'A':
        if max_valid_supply[i] > maxA:
            max_valid_supply[i] = maxA
    elif ty == 'B':
        if max_valid_supply[i] > maxB:
            max_valid_supply[i] = maxB
    elif ty == 'C':
        if max_valid_supply[i] > maxC:
            max_valid_supply[i] = maxC
#print(max_valid_supply)
```

```
    sort_mean_supply = np.argsort(mean_valid_supply)
    sort_max_supply = np.argsort(max_valid_supply)

    # 贪心算法对于max_supply求解
    selected_company = greedy_search(sort_max_supply,
        max_valid_supply)
    print(selected_company)

    # 按照最大的求解: [201, 307, 395]
    # 但没有考虑订购量和供应量的关系

    selected_company = greedy_search(sort_mean_supply,
        mean_valid_supply)
    print(selected_company)

if __name__ == '__main__':
    # 贪心算法
    greedy_method()
```

- zy-prediction.R: 实现问题二中关于转运商损耗率预测模型，数出'zy-pred.csv'。

```
zy <- read.table('DS/data/zy.txt', header = T)
zy <- zy[-1]
zy[zy == 5] <- 0
zy[zy == 0] <- NA

# 3,4,5 have many NA
apply(zy, 1, function(x) {
  sum(is.na(x))
})

# use mean to fill NA exclude T3,T4,T5
zy.mean <- apply(zy, 1, mean, na.rm = T)
for (i in c(1, 2, 6, 7, 8)) {
  zyi <- zy[i, ]
  zy[i, is.na(zyi)] <- zy.mean[i]
}

# data frame to vector
```

```
T1 <- as.vector(as.matrix(zy[1, ]))
T2 <- as.vector(as.matrix(zy[2, ]))
T3 <- as.vector(as.matrix(zy[3, ]))
T4 <- as.vector(as.matrix(zy[4, ]))
T5 <- as.vector(as.matrix(zy[5, ]))
T6 <- as.vector(as.matrix(zy[6, ]))
T7 <- as.vector(as.matrix(zy[7, ]))
T8 <- as.vector(as.matrix(zy[8, ]))
# plot
par(mfrow = c(8,1))
plot(T1)
plot(T2)
plot(T3)
plot(T4)
plot(T5)
plot(T6)
plot(T7)
plot(T8)
par(mfrow = c(1,1))

## T1
# arima model
library(astsa)
# ma(1)
plot(T1,type = "o")
acf2(T1)
T1.pred <- sarima.for(T1, 24, 0, 0, 1)

## T2
# arima(1,1,0)
plot(T2,type = "o")
acf2(T2)
T2.d <- diff(T2)
acf2(T2.d)
T2.pred <- sarima.for(T2, 24, 1, 1, 0)

## T3
# hmm
plot(T3,type = "o")
```

```
library(depmixS4)
hmm.T3 <-
  depmix(T3 ~ 1,
         nstates = 2,
         data = data.frame(T3),
         family = gaussian())
set.seed(574846)
summary(fm <- fit(hmm.T3))

plot(
  T3,
  main = "",
  ylab = 'T3',
  type = 'h',
  col = gray(.7)
)
text(
  T3,
  col = 6 * posterior(fm)[, 1] - 2,
  labels = posterior(fm)[, 1],
  cex = .9
)
# parameters
para = as.vector(getpars(fm))
mtrans = matrix(para[3:6], byrow = TRUE, nrow = 2)
mu = para[c(7, 9)]
sigma = para[c(8, 10)]
generate.sample = function(n, m, mu, sigma, Mtrans, ostate)
{
  # n  length
  # m  # of states
  # Mtrans  transition matrix
  # ostate  origin state
  mvect = 1:m
  state = numeric(n)
  state[1] = ostate
  for (i in 2:n)
    state[i] = sample(mvect , 1, prob = Mtrans[state[i - 1] ,])
  y = rnorm(n, mu[state], sigma[state])
```

```
  list(y = y, state = state)
}
set.seed(1234567)
T3.pred <- generate.sample(24, 2, mu, sigma, mtrans, 2)
T3.pred
plot(
  T3.pred$y,
  main = "",
  ylab = 'T3 prediction',
  type = 'h',
  col = gray(.7)
)
text(
  T3.pred$y,
  col = 6 * posterior(fm)[, 1] - 2,
  labels = T3.pred$state,
  cex = .9
)

## T4
# hmm
plot(T4,type = "o")
hmm.T4 <-
  depmix(T4 ~ 1,
         nstates = 2,
         data = data.frame(T4),
         family = gaussian())
set.seed(574846)
summary(fm <- fit(hmm.T4))

plot(
  T4,
  main = "",
  ylab = 'T4',
  type = 'h',
  col = gray(.7)
)
text(
  T4,
```

```
  col = 6 * posterior(fm)[, 1] - 2,
  labels = posterior(fm)[, 1],
  cex = .9
)

# parameters
para = as.vector(getpars(fm))
mtrans = matrix(para[3:6], byrow = TRUE, nrow = 2)
mu = para[c(7, 9)]
sigma = para[c(8, 10)]
# prediction
set.seed(684586844)
T4.pred <- generate.sample(24, 2, mu, sigma, mtrans, 2)
T4.pred

## T5
# seems no pattern
plot(T5)
T5.narm <- na.exclude(T5)
plot(density(T5.narm),main = 'density of T5')
# exponential distribution
mu <- mean(T5, na.rm = T)
sigma <- var(T5, na.rm = T)
lam <- 1 / mu
x <- seq(0,5,length.out = 100)
lines(x,dexp(x,rate=lam),col = 2)
# prediction
T5.pred <- rexp(24, lam)

## T6
# ar(1)
plot(T6,type = "o")
acf2(T6)
T6.pred <- sarima.for(T6, 24, 1, 0, 0)

## T7
# arima(1,1,0)
plot(T7,type = "o")
acf2(T7)
```

```
T7.d <- diff(T7)
acf2(T7.d)
T7.pred <- sarima.for(T7, 24, 1, 1, 0)

## T8
# arima(1,1,0)
plot(T8,type = "o")
acf2(T8)
T8.d <- diff(T8)
acf2(T8.d)
T8.pred <- sarima.for(T8, 24, 1, 1, 0)

zy.pred <- rbind(
  T1.pred$pred,
  T2.pred$pred,
  T3.pred$y,
  T4.pred$y,
  T5.pred,
  T6.pred$pred,
  T7.pred$pred,
  T8.pred$pred
)
row.names(zy.pred) <- c('T1', 'T2', 'T3', 'T4',
                        'T5', 'T6', 'T7', 'T8')
colnames(zy.pred) <- 1:24

write.csv(zy.pred,
          'zy_pred.csv')
```

- zy-sim.R: 根据对损耗率已建立的模型模拟 10000 次，输出'zy-simulation.txt'.

```
# data
zy <- read.table('DS/data/zy.txt', header = T)
zy <- zy[-1]
zy[zy == 5] <- 0
zy[zy == 0] <- NA

# 3,4,5 have many NA
apply(zy, 1, function(x) {
  sum(is.na(x))
```

```
})

# use mean to fill NA exclude T3,T4,T5
zy.mean <- apply(zy, 1, mean, na.rm = T)
for (i in c(1, 2, 6, 7, 8)) {
  zyi <- zy[i,]
  zy[i, is.na(zyi)] <- zy.mean[i]
}

# data frame to vector
T1 <- as.vector(as.matrix(zy[1,]))
T2 <- as.vector(as.matrix(zy[2,]))
T3 <- as.vector(as.matrix(zy[3,]))
T4 <- as.vector(as.matrix(zy[4,]))
T5 <- as.vector(as.matrix(zy[5,]))
T6 <- as.vector(as.matrix(zy[6,]))
T7 <- as.vector(as.matrix(zy[7,]))
T8 <- as.vector(as.matrix(zy[8,]))

# simulation
library(depmixS4)
# T3
hmm.T3 <-
  depmix(T3 ~ 1,
         nstates = 2,
         data = data.frame(T3),
         family = gaussian())
fm <- fit(hmm.T3)
para3 = as.vector(getpars(fm))
mtrans3 = matrix(para3[3:6], byrow = TRUE, nrow = 2)
mu3 = para3[c(7, 9)]
sigma3 = para3[c(8, 10)]

# T4
hmm.T4 <-
  depmix(T4 ~ 1,
         nstates = 2,
         data = data.frame(T4),
         family = gaussian())
```

```
fm <- fit(hmm.T4)
para4 = as.vector(getpars(fm))
mtrans4 = matrix(para4[3:6], byrow = TRUE, nrow = 2)
mu4 = para4[c(7, 9)]
sigma4 = para4[c(8, 10)]


generate.sample = function(n, m, mu, sigma, Mtrans, ostate)
{
  # n  length
  # m  # of states
  # Mtrans  transition matrix
  # ostate  origin state
  mvect = 1:m
  state = numeric(n)
  state[1] = ostate
  for (i in 2:n)
    state[i] = sample(mvect , 1, prob = Mtrans[state[i - 1] , ])
  y = rnorm(n, mu[state], sigma[state])
  list(y = y, state = state)
}

mu <- mean(T5, na.rm = T)
lam <- 1 / mu
for (i in 1:10000) {
  T3.pred <- generate.sample(24, 2, mu3, sigma3, mtrans3, 2)
  T4.pred <- generate.sample(24, 2, mu4, sigma4, mtrans4, 2)
  T5.pred <- rexp(24, lam)
  zy.pred <- rbind(
    T1.pred$pred,
    T2.pred$pred,
    abs(T3.pred$y),
    abs(T4.pred$y),
    T5.pred,
    T6.pred$pred,
    T7.pred$pred,
    T8.pred$pred
  )
  row.names(zy.pred) <- c('T1', 'T2', 'T3', 'T4',
```

```
                                'T5', 'T6', 'T7', 'T8')
  colnames(zy.pred) <- 1:24

  write.table(zy.pred,
              'zy_simulation.txt',
              sep = ',',
              col.names = F,
              append = T)
}
```

- C2-oreder-program-new.m: 在给定 4 家候选供应商的基础上，实现问题二含概率元素的优化。

```
w = [0.72
     0.6
     0.6
     0.6];
sigma2 = [10000
          10000
          10000
          10000];
sigma = sqrt(sigma2);
%%
cvx_clear
cvx_begin
     variable x(4,24)
     expressions mu(24,1) sig(24,1) z(24,1)
     minimize(sum(x(1,:))+1.2*sum(x(2,:)+x(3,:)+x(4,:)))
     subject to
     x>=0;

     for j=1:24
          for i=1:4
               z(j)=z(j)+x(i,j)/w(i);
          end
          if(j==1)
               mu(j)=z(j)-28200;
          else
               mu(j)=mu(j-1)+z(j)-28200;
          end
     end
```

```
    sig=zeros(24,1);
    for j=1:24
        sig(j)=j.*sum(sigma./w);
    end
    mu-1.6*sig>=0

cvx_end
```

- p2.lg4: 问题二订购方案的 LINGO 求解代码。(对于问题三订购方案的求解代码详见 p3.lg4, 由于该部分代码除了数据规模、数组大小、文件 IO 路径等几乎完全一致, p3.lg4 不在附录中展示, 可详见支撑材料的代码部分)

```
model:
sets:
supplyers/1..4/:;
weeks/1..24/:;
transporters/1..8/:;
rate(transporters,weeks): a;
assignment(supplyers,weeks): S;
decision(supplyers,weeks,transporters): T,ans;
endsets

data:
a = @ole('C:\Users\HONOR\Desktop\C\data\p2_trans_pred.xlsx','
    pred_data');
S = @ole('C:\Users\HONOR\Desktop\C\data\p2_supply_prob.xlsx','
    pred_data');
@ole('C:\Users\HONOR\Desktop\C\res\p2_method_prob.xlsx','
    pred_data') = ans;
enddata
min = @sum(transporters(k):@sum(supplyers(i): @sum(weeks(j): S(i,
    j)*T(i,j,k)*a(k,j))));
@for(supplyers(i): @for(weeks(j): @sum(transporters(k): T(i,j,k))
    =1));
@for(transporters(k): @for(weeks(j): @sum(supplyers(i): (T(i,j,k)
    )*S(i,j))<=6000));
@for(transporters(k):@for(supplyers(i): @for(weeks(j): @bin(T(i,j
    ,k)))));
@for(transporters(k): @for(weeks(j): @for(supplyers(i): ans(i,j,k
    ) = T(i,j,k) * S(i,j))));
```

```
end
```

- C3-order-program-limit6000.m：问题三订购方案的概率优化模型，在问题二的概率优化模型的基础上加入了转运两 6000 的最大容量限制等。

```
w=zeros(402,1);
w(label=='A')=0.6;
w(label=='B')=0.66;
w(label=='C')=0.72;
sigma=100*ones(402,1);

aindex=find(label=='A');
bindex=find(label=='B');
cindex=find(label=='C');

load 'res.txt'
load 'upbound.txt'

upboundd=upbound*ones(1,24);
shift=res(:,1);
sigma=res(:,2);
sig=zeros(24,1);
for j=1:24
    sig(j)=j.*sum(sigma./w);
end
sig=sig+158.0332; % r.sd in R files 'C3_preprocess'
%%
cvx_clear
cvx_begin
    variable x(402,24)
    expressions mu(24,1) sig(24,1) z(24,1)
    sa=sum(x(aindex,:));sb=sum(x(bindex,:));sc=sum(x(cindex,:));
    minimize(1.2*sum(sa)+1.1*sum(sb)+sum(sc)+0.1*norm(sc,1)+0.05*
        norm(sb,1))
    %minimize(1.2*sum(sa)+1.1*sum(sb)+sum(sc))
    subject to
    x>=0;
    x<=upboundd;
    x<=6000;
```

```
    for j=1:24
        for i=1:402
            z(j)=z(j)+(x(i,j)+shift(i))/w(i);
        end
        if(j==1)
            mu(j)=z(j)-28200;
        else
            mu(j)=mu(j-1)+z(j)-28200;
        end
    end

    mu-1.6*sig>=0

cvx_end
```

- p3-end2end.lg4: 问题三端到端的多目标规划求解程序代码。

```
model:
sets:
supplyers/1..402/: u,w;
weeks/1..24/:;
transporters/1..8/:;

rate(transporters,weeks): a;
assignment(supplyers,weeks): S,R,d,b;
decision(supplyers,weeks,transporters): T,ans;
endsets

data:
alpha = 1;
beta = 1;
gamma = 1;

a = @ole('C:\Users\HONOR\Desktop\C\data\p2_trans_pred.xlsx','
    pred_data');
@ole('C:\Users\HONOR\Desktop\C\res\p3_method_S.xlsx','S') = S;
@ole('C:\Users\HONOR\Desktop\C\res\p3_method_T.xlsx','T') = ans;

enddata
```

```
min = alpha * TransCost + beta * StoreCost + gamma * OrderCost;

OrderCost = @sum(supplyers(i): @sum(weeks(j): S(i,j) * w(i)));
TransCost = @sum(supplyers(i): @sum(weeks(j): S(i,j)));
StoreCost = @sum(supplyers(i): @sum(weeks(j): d(i,j) * u(i)));

@for(supplyers(i): @for(weeks(j): @sum(transporters(k): T(i,j,k)
    * S(i,j) * (1-a(k,j)) ) = R(i,j)));
@for(supplyers(i): @for(weeks(j): @sum(transporters(k): T(i,j,k))
    =1));
@for(transporters(k):@for(supplyers(i): @for(weeks(j): @bin(T(i,j
    ,k)))));
@for(transporters(k): @for(weeks(j): @sum(supplyers(i): (T(i,j,k)
    )*S(i,j))<=6000));

@for(supplyers(i): @for(weeks(j)|j#gt#1: d(i,j) = d(i,j-1) + R(i,
    j) / u(i) - b(i,j)));
@for(weeks(j): @sum(supplyers(i): b(i,j)) = 28200);
@for(weeks(j): @sum(supplyers(i): d(i,j)) >= 56400);

@for(transporters(k): @for(weeks(j): @for(supplyers(i): ans(i,j,k
    ) = T(i,j,k) * S(i,j))));

end
```

- C4-bisection.m: 基于问题三中订购方案的 C3 的 Matlab 文件，使用改进版本的概率优化模型，并且利用二分法求上界。

```
w=zeros(402,1);
w(label=='A')=0.6;
w(label=='B')=0.66;
w(label=='C')=0.72;
sigma=100*ones(402,1);

aindex=find(label=='A');
bindex=find(label=='B');
cindex=find(label=='C');



load 'upbound_new.txt'
```

```
upbound=upbound_new(:,1)*ones(1,24);
sigma=upbound_new(:,2);
sig=zeros(24,1);
for j=1:24
    sig(j)=j.*sum(sigma./w);
end
sig=sig+158.0332; % r.sd in R files 'C3_preprocess'
%%
bound=76400; % solved
%bound=76500; % infeasible
cvx_clear
cvx_begin
    variable x(402,24)
    expressions mu(24,1) sig(24,1) z(24,1)
    sa=sum(x(aindex,:));sb=sum(x(bindex,:));sc=sum(x(cindex,:));
    minimize(1.2*sum(sa)+1.1*sum(sb)+sum(sc)+0.2*norm(sc,1)+0.1*
        norm(sb,1))
    %minimize(1.2*sum(sa)+1.1*sum(sb)+sum(sc))
    subject to
    x>=0;
    x<=upbound;
    sum(x)<=6000*8;
    for j=1:24
        for i=1:402
            z(j)=z(j)+x(i,j)/w(i);
        end
        if(j==1)
            mu(j)=z(j)-bound;
        else
            mu(j)=mu(j-1)+z(j)-bound;
        end
    end

    mu-1.6*sig>=0

cvx_end
```

- p4.lg4: 问题四订购方案的 LINGO 求解代码。

```
model:
sets:
supplyers/1..402/:;
weeks/1..24/:;
transporters/1..8/:;
rate(transporters,weeks): a;
assignment(supplyers,weeks): S;
decision(supplyers,weeks,transporters): T;
endsets

data:
a = @ole('C:\Users\HONOR\Desktop\C\data\p2_trans_pred.xlsx','
    pred_data');
S = @ole('C:\Users\HONOR\Desktop\C\data\p4_supply_prob.xlsx','
    pred_data');
@ole('C:\Users\HONOR\Desktop\C\res\p4_method_prob.xlsx','T') = T;
enddata

min = @sum(transporters(k):@sum(supplyers(i): @sum(weeks(j): T(i,
    j,k)*a(k,j))));
@for(supplyers(i): @for(weeks(j): @sum(transporters(k): T(i,j,k))
     = S(i,j) ));
@for(transporters(k): @for(weeks(j): @sum(supplyers(i): (T(i,j,k)
    ))<=6000));

end
```

- WMA.m: 进行对于已知 X,Y 列向量，求 x 处加权滑动均值的 Matlab 代码。

```
function [y,sigma2] = WMA(X,Y,x)
%WMA 已知X,Y, 计算x的对于加权移动平均
% X (N,1), x(n,1), Y(N,1)
N = size(X,1);
n = size(x,1);
% W(n,N)
W = exp(-((X'-x)./500).^2);
y = W*Y ./ sum(W,2);
sigma2 = sum(W.*((y-Y').^2),2) ./ sum(W,2); %用于试验的方差项; 未
    使用
```

```
end
```

- SB_ proxy.m: 基于 WMA.m 及已读取的订购量、供应量，拟合第 i 供应商的订-供曲线。

```
% 拟合第i供应商的订-供曲线
function [S,sigma2] = SB_proxy(i,B)
load datas.mat;
X = Book_matrix(i,:)';
[S,sigma2] = WMA(X,Supply_matrix(i,:)',B);
end
```

- proxy_ fit.m: 进行具体的试验，取 WMA 模型下，各供应商可取区间的最大值。最后得到 4 家供应商并绘图。

```
clc;
load datas.mat
K = 4; %试验得可取出4家公司的数量最优解
Xx = [1:10:6000]';

for i = 1:402
    if S_Class(i) == 'B' % 据理论分析，应该选择A或C供应商以最优化
        s_max(i)=-1;p_max(i)=-1;continue
    end
    [s_max(i),I] = max(SB_proxy(i,[1:10:min([6000,S_max(i)])]'));
    p_max(i) = Xx(I);
    [~,sigma2(i)] = SB_proxy(i,p_max(i));
end

[s_max,I] = sort(s_max,'descend');
p_max=p_max(I);
s_max = s_max(1:K);
p_max = p_max(1:K);
sigma2 = sigma2(1:K);
I = I(1:K);

s_max
sigma2
s_max' .* density_vec(I)

for i = 1:K
```

```
    figure(i);
    scatter(Book_matrix(I(i),:), ...
            Supply_matrix(I(i),:));
    hold("on");
    plot(Xx,SB_proxy(I(i),Xx));
    scatter(p_max(i),s_max(i));
end
```

- MonteCarlo2.m: 基于已解出的供应量，计算问题 2 第一部分蒙特卡罗模拟，统计相关数据。

```
% 基于已解出的供应量，计算问题2第一部分蒙特卡罗模拟

load datas.mat
load S1.mat

store = 56400;
tot_cost = 0;

global Wdata;
Wdata = [ ...
2228.722909     2091.471194     2086.687549     2090.396821
    2094.586691     2099.442788     2103.946976     2107.057604
        2108.235009     2107.664493     2106.234781
    2105.22956      2105.817512     2108.409079     2111.887995
        2113.042447     2107.463828     2092.775876
    2071.379274     2062.778145     2152.900943     2269.523075
        1369.960226     2160.012439;
5299.899551     5258.797363     5264.352687     5264.312022
    5263.755617     5262.802978     5261.746028     5260.872938
        5260.345076     5260.17704      5260.257568
    5260.409268     5260.463994     5260.317032     5259.949036
        5259.435157     5258.817379     5257.679365
    5255.429351     5253.281115     5253.793933     5262.869029
        5413.542291     5189.019956;
5299.899551     5258.797363     5264.352687     5264.312022
    5263.755617     5262.802978     5261.746028     5260.872938
        5260.345076     5260.17704      5260.257568
    5260.409268     5260.463994     5260.317032     5259.949036
        5259.435157     5258.817379     5257.679365
```

```
    5255.429351      5253.281115      5253.793933      5262.869029
        5413.542291      5189.019956;
5299.899551      5258.797363      5264.352687      5264.312022
    5263.755617      5262.802978      5261.746028      5260.872938
        5260.345076      5260.17704       5260.257568
    5260.409268      5260.463994      5260.317032      5259.949036
        5259.435157      5258.817379      5257.679365
    5255.429351      5253.281115      5253.793933      5262.869029
        5413.542291      5189.019956];
% 已求解的各供货商、各周的供应量（作为均值）。
% 其中行1为C供应商，2:4为A供应商

for time = 1:24
    store;

    target = 28200;
    store = store - target;
    if store <0
        error("Can't reach target at a time duration")
    end

    [a_in,b_in,c_in] = week_supply(time);

    store = store + a_in /0.6 + b_in / 0.66 + c_in / 0.72;
    tot_cost = tot_cost + 1.2*a_in+1.1*b_in+1*c_in;

    if store < 56400
        error("Can't reach target at a time duration")
    end
end

%disp("Finished")
tot_cost;

function [a_in,b_in,c_in] = week_supply(w)
    global Wdata;

    A_mean = Wdata(2:4,w);
    A_std = [100;100;100];
```

```
    C_mean = Wdata(1,w);
    C_std = 100;
    b_in = 0;
    a_in = normrnd(A_mean,A_std);
    c_in = normrnd(C_mean,C_std);
    a_in(a_in<0) = 0; a_in(a_in>6000) = 6000;
    c_in(c_in<0) = 0; c_in(c_in>6000) = 6000;
    a_in = sum(a_in);
    c_in = sum(c_in);
end
```

- run_MC_ 2.m: 在此处进行给定次数的 MonteCarlo2.m 并展示，以及执行另一部分的 MonteCarlo2_T.m

```
MC2_cost = []
MC2_store=[]
for i = 1:10000
    MonteCarlo2;
    MC2_cost=[MC2_cost,tot_cost];MC2_store=[MC2_store,store];
end

mean(MC2_cost)
std(MC2_cost)

MonteCarlo2_T;
```

- MonteCarlo3.m: 进行问题 3 的蒙特卡罗模拟；在其中进行 10000 次并统计。

```
load datas.mat
load MC.mat

K = 1;

u = zeros(402,1);
u(S_Class=='A') = 1/0.60;
u(S_Class=='B') = 1/0.66;
u(S_Class=='C') = 1/0.72;

w = zeros(402,1);
w(S_Class=='A') = 1.2;
w(S_Class=='B') = 1.1;
```

```
w(S_Class=='C') = 1.0;

C = zeros(402,3);
C(:,1) = S_Class=='A';
C(:,2) = S_Class=='B';
C(:,3) = S_Class=='C';

supply = P3_S; % 402x24 供应量
T = P3_T; % 402x24x8 转运关系
A = permute(MC2_T_sim,[1 3 2]); % （转置后）10000x24x8 带有蒙特卡
    罗模拟的损耗率

small_count = size(Low_ID,1);
small_std = 158.0332 ^ 0.5;

shift_ex = zeros(402,24);
shift_std_avr = shift_std;
shift_std_avr(Low_ID) = small_std/ small_count^0.5;
shift_std_ex = zeros(402,24);

for i = 1 : 24
    shift_ex(:,i) = shift(:);
    shift_std_ex(:,i) = shift_std_avr;
end

MC3_cost = [];
MC3_store = [];

MC3_err_count = 0;
for s = 1:10000
    As = A(s,:,:); % 1 24 8
    S_std = 100 .* ones(size(supply));

    supply_S = normrnd(supply-shift_ex,shift_std_ex);
    supply_S(supply_S<0) = 0;
    supply_S(supply_S>6000) = 6000;

    %supply_S = supply;
```

```
R = sum((1- 0.01.*As).* T .* reshape(supply_S,[402 24 1]),3);
R = reshape(R,[402 24]);
if any(R<0)
    error("R<0")
end
In = R'*C;

% In (24 3) 每周 A,B,C 接受量
tot_cost = sum(K.*(w .* supply_S)+supply_S,'all');
store = [56400*0.6 0 0 ];
%In %

err_flag = 0;
for week = 1:24

    tot_cost = tot_cost + sum(store);
    target = 28200;
    A_used = 0; B_used = 0; C_used = 0;
    if store(3)>0
        C_used = min(target, store(3)./0.72);
        target = target - C_used;
        store(3) = store(3) - C_used*0.72;
    end
    if target > 0 && store(2)>0
        B_used = min(target, store(2)./0.66);
        target = target - B_used;
        store(2) = store(2) - B_used*0.66;
    end
    if target > 0 && store(1)>0
        A_used = min(target, store(1)./0.60);
        target = target - A_used;
        store(1) = store(1) - A_used*0.60;
    end

    store = store + In(week,:);

    if store' * [1/0.6 1/0.66 1/0.72] < 56400
        disp("store < 56400");
        err_flag = 1;
```

```
            continue % 继续执行完当前生产过程
        end
    end

    MC3_cost = [MC3_cost tot_cost];
    MC3_store = [MC3_store; store];
    if err_flag == 1
        MC3_err_count = MC3_err_count + 1;
    end
end

disp("Finished")
```

- MonteCarlo2_ T.m: 进行问题 2 转运量部分的蒙特卡罗模拟。

```
%已得到转运损耗量波动模拟后, 计算问题2第二部分的蒙特卡罗模拟

load MC.mat

for i = 1:10000
    for j = 1:4
        MC2_T_result(i,j,:) = Wdata(j,:) .* sum( reshape(P2_T(j
            ,:,:),[8 24]) .* (1-0.01.*reshape(MC2_T_sim(i,:,:),[8
            24])));
    end
end

MC2_T_res = MC2_T_result- reshape(Wdata,[1,4,24]);
```