对生产企业原材料订购与转运方案的研究

摘要

生产企业需要提前制定订购与运输原材料的方案以保证企业的正常运转。本文通过建立数学模型，对生产企业的方案制定问题进行分析，给出最优的生产原材料订购与运输方案。

针对问题一，对供应商的供货特征进行**定性与定量**分析。使用 Python 作图分析供应商收到的订货量及其供货量的特点，提取供应商影响企业生产的四个特征，即供货量、对企业需求的满足度、每周供货量的稳定性和历史供货次数，并对 402 家企业的这四个特征进行统计，最后运用**主成分分析法**获得综合得分在前 50 名的企业。

针对问题二，首先采用**时间序列模型**，通过对过去 10 个周期的分析，预测了 402 家企业在未来 24 周中每周的供货量。再运用 **0-1 整数规划模型**，使用 matlab 求解能满足生产需求的最少供应商数，最终解得为 **37** 家。此后，再次使用 0-1 规划模型，得到在满足每周生产需求且预留两周库存量的条件下，订购费用最小的原材料供货方案。接着进一步通过分析历史缺货情况，反推求得与之对应的订货方案。考虑到一家供应商的材料由一家转运商运输以及转运商有运输能力上限，使用 **0-1 背包模型**设计转运方案，降低转运过程中造成的损失。最后根据 24 周每周库存量是否合理及转运损失是否减少来评判订购方案和转运方案，发现方案实施效果较好。

针对问题三，首先查阅资料了解材料的订购、转运和仓储费用，得知订购每种材料的**总成本**。再一次运用 0-1 整数规划模型，得到在满足生产需求的条件下成本最小的原材料供货方案，按照前述方法求得相应的供货方案。为了降低转运**损耗率**，采取尽可能让损耗率小的转运商转运更大体积的货物的方案，并利用 0-1 背包模型设计转运方案。最后根据未来 24 周预估总成本是否比历史平均成本减少，以及转运损耗率是否降低来评估订购方案和转运方案，最后发现节约了约 **31.8%**的成本，损耗率降低了约 **57.6%**。方案实施效果较好。

针对问题四，首先分析确定了有两步，需要进行**双层线性规划**。第一层是通过对现有原材料的供应商和转运商的情况分析，建立线性规划模型，从而确定企业的产能上限。第二层是将上述的产能上限作为新的约束引入混合规划模型，以利润最大为目标，确定企业相应的产能和订货计划。最终求得产能为 **34855m^3/周**。随后按照和前问类似的方式构造 0-1 背包模型，求解最优的转运方案。

最后，给出每个模型的优缺点和评价。

关键词：时间序列，0-1 整数规划，0-1 背包模型，主成分分析法，双层规划模型

一、问题重述

某生产企业所用的原材料分为 A,B,C 三种类型。该企业的产能要求是 2.82 万立方米/周，其中每生产一立方米产品需要消耗 0.6 立方米的 A 材料，或 0.66 立方米的 B 材料，或 0.72 立方米的 C 材料。

该企业每年生产期为 48 周，需要提前制定 24 周的原材料订购和转运方案，即根据产能要求确定每周的原材料供应商和订货量，再安排将供应商的供货量转运到企业仓库的转运商。

由于供应商的实际供货量可能与订货量有差异，且企业原材料存货量尽量保证满足大于等于两周的生产需求，因此该企业总是全部收购供应商提供的原材料。其中材料 A,B 的购买单价分别是材料 C 的 1.2 倍和 1.1 倍，三类原材料运输和存储的单位费用相同。

转运商转运过程中也会损耗原材料。每家转运商每周的运输上限是 6000 立方米，通常一家供应商每周提供的原材料由一家转运商运输。

问题一，已知该企业之前 240 周共 402 家供应商的订货量和供货量，以供应商对于保证企业生产的重要性为标准，确定 50 家最重要的供应商。

问题二，在问题一的基础上，计算可以满足企业生产需求的供应商的最小数量。在确定这些供应商的基础上，给出接下来 24 周的最经济的原材料订购方案和损耗最少的转运方案，并分析方案效果。

问题三，制定新的订购和转运方案，以减少运输和存储的成本（多采购材料 A，少采购材料 C）并尽量降低转运材料损耗率，并分析方案效果。

问题四，计算该企业每周可以提高多少产能，并制定接下来 24 周的订购和转运方案。

二、问题分析

2.1 问题一分析

问题一需要参照该企业过去 240 周里向 402 家供应商的订购数量和供应商每周的供货数量，因此需要对表格数据先进行分析。分析表格可以得出：在过去 240 周的大部分时间内，企业对于生产材料的固定需求量（每周完成目标产能所需要的原材料的量）都大于所有供应商每周的供货量；且企业几乎每周的订单量都大于供货量。因此可以得出结论：供货量较多的供应商，对于企业的正常生产更为重要；供货量和企业订单量接近的供货商，即更能满足企业需求的供货商，也更为重要。

同时，考虑到企业每周都需要生产固定量的产品，因此供货量更为稳定的供应商更有利于企业的生产。该指标可以根据各个供应商在前 240 周供货量的方差来衡量。

最后，企业更为倾向和能够持续供货的供应商进行合作，因此供货次数更多的供应商与企业存在更稳定、良好的合作关系，对于企业生产更为重要。

因此决定供货商能否保障企业生产的因素有：

1) 供货量
2) 对企业需求的满足度
3) 每周供货量的稳定性
4) 历史供货次数

将这四种因素具体量化，可以分别用供货商五年的供货总量、供货量与订货量的差值比、每周供货量的方差、供货商的历史供货次数来衡量。这样每家供货商都具备了四个数字化的因素。

对于这四个因素，首先将其标准化，再采取主成分分析法建立模型，找到最后综合评分在前 50 的企业即可。

3.2 问题二分析

问题二需要参考问题一来作答。本题分为两个小题，其一是找到满足企业生产的最少的供应商数量；其二是在只向这些供应商订购产品的基础上，为企业未来 24 周每周分别制定最经济的订购方案和损耗最少的转运方案。

对于这两个小题，可以分为以下六个步骤来解答：

1) 预估供货商的未来 24 周的供货量；
2) 找到最少的供货商数量；
3) 确定未来 24 周每周向哪些供货商订货；
4) 确定具体的订货方案；
5) 确定损耗最少的转运方案；
6) 分析两个方案的实施效果。

在问题一里我们已经找到了最能保障企业生产的前 50 家供应商，因此第一问我们要在这 50 家里选择出正好满足企业未来 24 周生产的供应商，使得供应商总数最小。首先使用时间序列模型，根据 50 家供应商过去 240 周每周的供货量，预测其未来 24 周每周的供货量。

获得企业未来的供货量后，采取 0-1 整数规划模型来决定保留哪些供应商，目标函数是供应商数量达到最小值，约束条件是供应商每周的供货量是否能保证企业该周的产量，同时预留两个星期的原材料库存。最后可以解得最少需要的供应商的具体名单。

选出供应商后，可以再次采取 0-1 整数规划模型，对 24 个周每周分别选择哪些供应商进行计算。对于每个周，都需要保证从上一问解出的最少供应商中选择的供应商可以满足该周的产能需求，这也是线性规划模型中的约束条件。而为了实现经济性，则应该把每周订购费用作为目标函数，使其达到最小值。使用 matlab 求解。

随后，需要在已知每个周选择的供应商和其供货量的基础上求出订货方案。已经给出过去 240 周每周的订货量和供货量，因此本题是一个回归问题，可以通过分析历史缺货情况，反推求得与之对应的订货方案。

对于转运方案的确立，首先利用转运商的平均损耗率、高损耗频次和接收订单频次这三个因素对八家转运商进行排序，由于转运商有容量上限等限制，因此可以利用 0-1 背包算法确定每家供应商使用哪家转运商进行转运。

最后需要评价订购方案与转运方案的实施效果。对于订购方案，在选择订购商家时，已经考虑到了材料 A,B,C 的价值差异，因此可以分析使用新的订购方案后，未来 24 周每周的预计库存，若库存稳定且基本满足两周生产需求，则说明订购方案实施效果较好。对于转运方案，可以将未来 24 周预计的损耗价值与过去 5 年平均每 24 周的损耗价值进行对比，若未来 24 周损耗较小，则说明转运方案实施效果较好。

3.3 问题三分析

问题三和问题二较为类似，都是制定订购方案和转运方案并分析实施效果。步骤同样为：

1) 确定未来 24 周每周向哪些供货商订货；

2) 确定具体的订货方案；
3) 确定损耗率最少的转运方案；
4) 分析两个方案的实施效果。

首先根据 402 家企业未来 24 周的供货量，采取 0-1 整数规划模型来决定接下来 24 周每周选择哪些供应商。对于每个周，都需要保证供货量可以满足该周的产能需求，这也是线性规划模型中的约束条件。而为了减少转运和仓储的成本，则需要尽可能多的采购材料 A，少采购材料 C，因此应该结合问题二，把每周的订购、转运、仓储费用之和作为目标函数，使其达到最小值。

随后同问题二，需要在已知每个周选择的供应商和其供货量的基础上求出订货方案，是一个回归问题。

转运方案同问题二，使用 0-1 背包算法进行计算，但本期谋求的是损耗率减少，因此本题希望排名更靠前的转运商装载更多材料，而不是装载价值更高的材料。

本题需要考虑到转运和仓储的成本，因此可以计算未来 24 周与过去 5 年平均每 24 周的购买成本、转运费用、仓储成本，若未来 24 周所有成本的和小于过去 5 年平均，则说明订购方案实施效果较好。转运方案则可以分析转运商平均损耗率，若新方案下 8 家转运商的平均损耗率小于过去 5 年 8 家转运商的平均损耗率，则说明转运方案实施效果较好。

3.4 问题四分析

本题的产能是一个变量，需要根据现有产能、402 家原材料供应商的供应能力、8 家转运商的转运能力进行约束。利用时间序列模型，获得考虑库存量的产能可行的取值范围后，再建立与产能、原材料成本、原材料以外的成本有关的线性规划模型，求解获利最大条件下的最优产能和供货方案。

这是一个双层线性规划。第一层是通过对现有原材料的供应商和转运商的情况分析，建立线性规划模型，从而确定企业的产能上限。第二层是将上述的产能上限作为新的约束引入混合规划模型，以利润最大为目标，确定企业相应的产能和订货计划。

最后构造 0-1 背包模型，求解最优的转运方案。

三、模型的基本假设

结合实际情况，为简化问题作出以下合理假设：

a) 假设过去的 240 周中的第一周，企业储存着正好满足两周生产的原材料。
b) 假设该企业生产商品为密度板，主原材料为枝桠材。
c) 假设 $1m^3$ 的枝桠材的质量为 100kg。
d) 假设企业原材料运输距离为 200km，仓储费用为 2 元/ m^3 • 天。
e) 假设企业生产的纤维板售价为 1500 元/m^3。
f) 假设企业原材料的订购与运输的所有开销为：购买成本、运输费用和仓储费用，无其他开销。
g) 假设该企业产能的提高上限仅与原材料供应商与转运商的实际情况有关，与企业自身因素无关。

四、模型符号说明

符号	符号说明	单位
x_i	选择第 i 个供应商	
x_{ij}	在第 j 周选择第 i 个供应商	
e_{ij}	第 i 个供应商在第 j 周的供货量（全转换为材料 C）	m^3
a_{ij}	第 i 个材料 A 的供应商在第 j 周的供货量	m^3
b_{ij}	第 i 个材料 B 的供应商在第 j 周的供货量	m^3
c_{ij}	第 i 个材料 C 的供应商在第 j 周的供货量	m^3
f_{ij}	第 i 个材料供应商在第 j 周的供货量	m^3
Q	每周至少需要供应的材料 C 的量	m^3
g_i	第 i 个转运商转运的供货量	m^3
h_i	第 i 个转运商运输完后的企业接收量	m^3
d_i	第 i 个转运商的损耗率	%
P	企业新的产能	m^3/周

五、模型的建立与求解

5.1 问题一的建模与求解

5.1.1 数据的处理

首先需要将表格里的数据进行可视化。可以计算出企业对于材料 A,B,C 的 240 周周均订货量与 402 家供应商对材料 A,B,C 的周均供货量对比图。

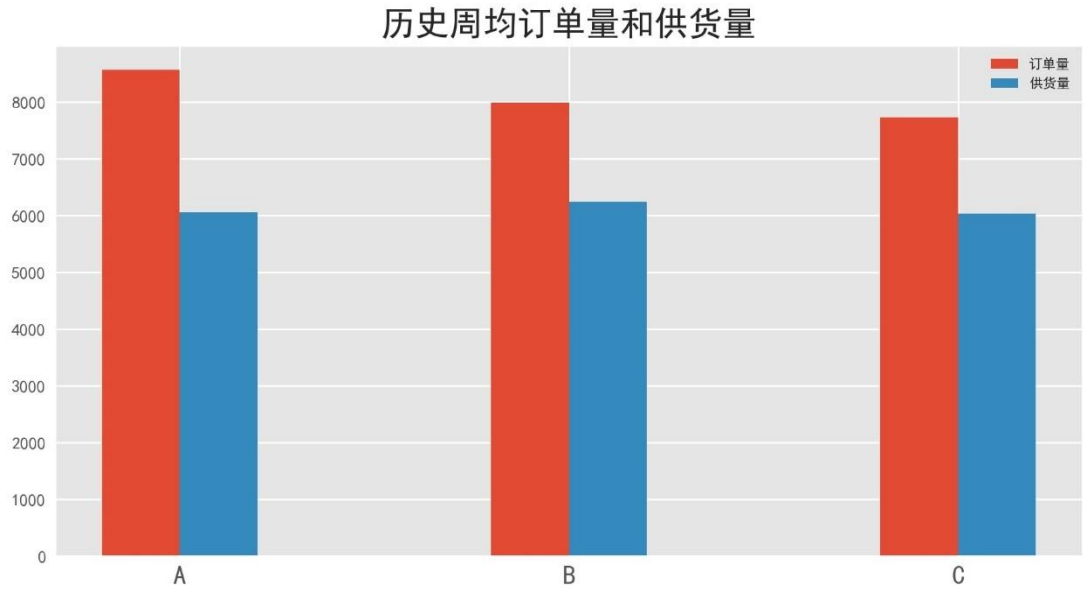

图 1 供应商历史周均订货量与供货量

可以看出，企业对于三种材料的需求度排行是 A>B>C，而 402 家商家提供三种材料周平均数量相差较小。基于后者，在本题的数据统计中，我们将材料 A,B 的量替换为能生产等体积产品的材料 C 的量。

即：每周供应商等价能提供的材料 C 的体积=提供的材料 A 的体积*1.2

每周供应商等价能提供的材料 C 的体积=提供的材料 B 的体积*1.09

每周企业等价订购的材料 C 的体积=订购的材料 A 的体积*1.2

每周企业等价订购的材料 C 的体积=订购的材料 B 的体积*1.09

基于此，我们可以得到企业 240 周每周的总订购量和 402 家供应商每周的总供货量的折线图。图中可以看到在绝大部分情况下，企业订购量是多于供应商供货量的。

图 2 240 周企业对材料 C（等价后）的订购量与供应商供货量折线图

企业每周的产能为 2.82 万立方米，相当于消耗 2.0304 万立方米的材料 C。因为企业是一个持续运作的状态，因此我们假设在过去 240 周的第一周，企业有着正好满足两周生产需求的材料库存。全部转换成材料 C 即为 4.0608 万立方米。即为了企业正常的运作和生产，此后 240 周每周的需求量（基于材料 C 换算）均为 2.0304 万立方米。因此我们也可以得到企业每周的需求量（基于材料 C 换算）与 402 家供应商每周的供货量的对比柱状图。

图 3 企业每周的需求量（基于材料 C 换算）与供应商供货量的对比柱状图

将二者分别累计相加，易得企业的总需求量随着周数的增加，是一个一次函数递增的状态，而供应商的总供货量随着周数增加也在不断递增。经过计算可得，240 周的总供货量与企业总需求量基本维持平衡。因此我们可以推断，在接下来的 24 周开始时，企业仍然保持储藏着满足两周生产需求的原材料。

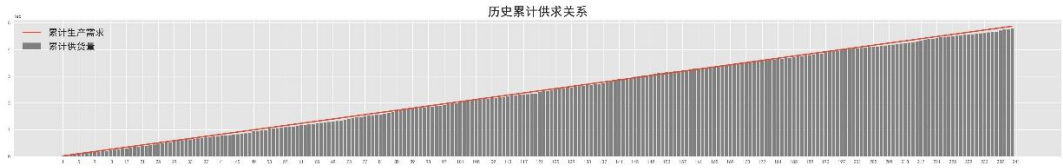

图 4 企业的需求量（基于材料 C 换算）与供应商总供货量的对比图

5.1.2 模型的建立

经过以上数据分析，我们可以得出以下结论：

a) 企业几乎每周的订单量都大于供应商的供货量；

b) 大部分时间内，企业对于生产材料的需求量都大于所有供应商每周的供货量；

c) 企业对于生产材料的总需求量与供应商的总供货量基本保持平衡。

而我们知道，即使是在供货量多于订单量的情况下，企业对于供货商提供的材料也总是全部收购。

因此可以做出如下推断：

a) 供货量较多的供应商，对于企业的正常生产更为重要；

b) 供货量和企业订单量接近的供货商，即更能满足企业需求的供货商，也较为重要。

同时，鉴于企业每周的固定产能为 2.82 万立方米，相当于每周消耗 2.0304 万立方米的材料 C，因此周与周之间供货量相差较小，即更为稳定的供应商更有利于企业的生产。该指标可以通过过去 240 周供应商供货量的方差来衡量。

最后，企业更为倾向和能够持续供货的供应商进行合作，因此供货次数更多的供应商对于企业生产更为重要。

因此决定供货商能否保障企业生产的因素有：

a) 供货量

b) 对企业需求的满足度

c) 每周供货量的稳定性

d) 历史供货次数

针对供货量，可以将表格数据进行统计，取一家供货商五年以来的总供货量（等价为材料 C）代表。

针对供应商对企业的需求满足度，采用以下公式来计算：

$$需求满足度 = \frac{总订货量 - 总供货量}{总订货量}$$

针对供应商的供货稳定性，可以计算 240 周的供货量方差。考虑到存在企业未向供应商订购原材料导致供应商供货量为 0 的情况，因此我们剔除 240 周中供应商收到的订货量为 0 的周，再计算剩余周数的供货量方差。

针对历史供货次数，我们用 240 周内供货量为 0 的周数来表示。

这样，我们获得了四种影响企业生产的因素的量化表示方法。要基于这四种因素选出 50 家最重要的企业，可以使用主成分分析法。

5.1.3 模型的求解

主成分分析法的求解一般步骤为：

1) 假设进行主成分分析的变量有 m 个（本题中为 4 个），共有 n 个评价对象（本题中为 402 个），设第 i 个评价对象的第 j 个指标的取值为 m_{ij}。首先将各指标值 m_{ij} 转换成标准指标 n_{ij}：

$$n_{ij} = \frac{m_{ij} - \overline{m_j}}{s_j}, (i = 1,2, \dots, n; j = 1,2, \dots, m)$$

其中：

$$\overline{m_j} = \frac{1}{n}\sum_{i=1}^{n} m_{ij}\ , s_j = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}\left(m_{ij} - \overline{m_j}\right)^2}, (j = 1,2, \dots, m)$$

另外，称

$$\widetilde{m_i} = \frac{m_i - \overline{m_i}}{s_i}, (i = 1,2, \dots, m)$$

为标准化指标变量。

2) 计算相关系数矩阵 R：

$$R = (r_{ij})_{m\times m}$$

其中 r_{ij} 是第 i 个指标与第 j 个指标的相关系数。

3) 相关系数矩阵 R 的特征值$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_m \geq 0$，和它对应的特征向量$u_1, u_2, \cdots, u_m$，其中$u_j = (u_{1j}, u_{2j}, \cdots, u_{nj})^T$，由特征向量组成 m 个新的指标变量：

$$\begin{cases} y_1 = u_{11}\widetilde{m_1} + u_{21}\widetilde{m_2} + \cdots + u_{n1}\widetilde{m_n} \\ y_2 = u_{12}\widetilde{m_1} + u_{22}\widetilde{m_2} + \cdots + u_{n2}\widetilde{m_n} \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ y_m = u_{1m}\widetilde{m_1} + u_{2m}\widetilde{m_2} + \cdots + u_{nm}\widetilde{m_n} \end{cases}$$

其中 y_1 是第一主成分，y_2 是第二主成分，……，y_m 是第 m 主成分。

4) 选择 p (p≤m)个主成分，计算综合评价值，选择依据为该主成分的累计贡献率是否接近 1。综合得分 Z 计算如下：

$$Z = \sum_{j=1}^{p} b_j y_j$$

其中：

$$b_j = \frac{\lambda_j}{\sum_{k=1}^{m}\lambda_k}(j = 1,2,\cdots,m)$$

基于以上步骤，我们求得四个指标的标准化，计算出主成分。求得 402 家供货商每一家的综合得分，最后选择出综合得分在前五十的供货商，分别为：

表 1 供应商得分及排名

排名	供货商	材料类别	排名	供货商	材料类别
1	S229	A	26	S365	C
2	S140	B	27	S040	B
3	S361	C	28	S364	B
4	S108	B	29	S367	B
5	S282	A	30	S201	A
6	S151	C	31	S055	B
7	S275	A	32	S346	B
8	S329	A	33	S080	C
9	S340	B	34	S294	C
10	S139	B	35	S244	C
11	S131	B	36	S218	C
12	S308	B	37	S007	A
13	S330	B	38	S266	A
14	S356	C	39	S123	A
15	S268	C	40	S338	B
16	S306	C	41	S114	A
17	S348	A	42	S150	A
18	S352	A	43	S314	C
19	S194	C	44	S291	A
20	S143	A	45	S086	C
21	S247	C	46	S037	C
22	S374	C	47	S003	C
23	S284	C	48	S098	B
24	S307	A	49	S395	A
25	S031	B	50	S076	C

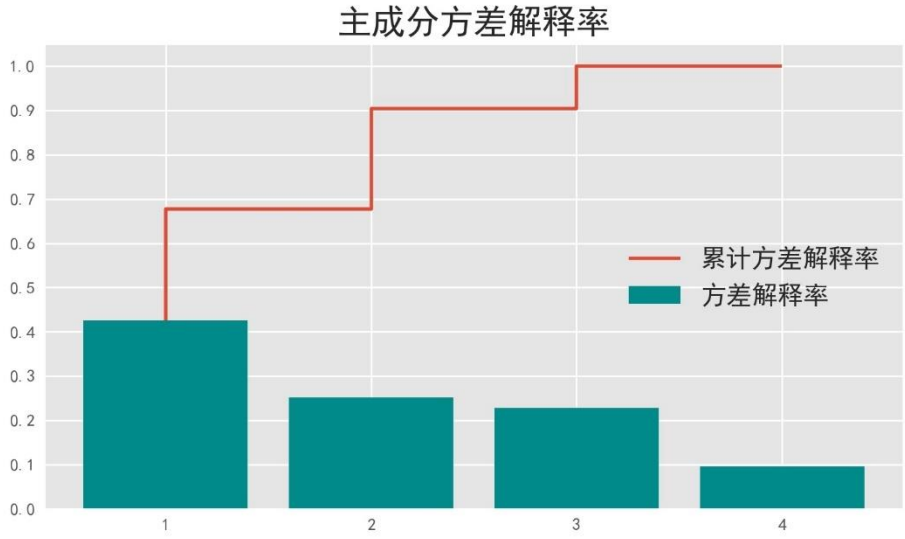

图 5 主成分方差解释率

5.2 问题二的建模与求解

5.2.1 时间序列模型的建立与求解

在问题一里我们已经找到了最能保障企业生产的前 50 家供应商，因此第一问我们要在这 50 家里选择出正好满足企业未来 24 周生产的供应商，使得供应商总数最小。首先使用时间序列模型，根据 402 家供应商过去 240 周每周的供货量，预测其未来 24 周每周的供货量，同时也获得了 50 家供应商的预计供货量。

由于 50 家供应商过去 240 周的供货量是按时间顺序的一组数字序列，是真实的一组数据，反映了供应商供应能力的一般规律，同时也具备一定的周期性，因此我们可以采用时间序列模型来预测 50 家供应商未来 24 周每周的供货量，从而计算出可以满足企业生产的供应商的最小数。

时间序列建模基本步骤是：

1) 取得观察值时间序列

2) 进行预处理，检验观察值序列的平稳性和纯随机性。平稳性可以通过时序图，单位根，自相关系数和偏相关系数来检验。若不平稳，可以利用差分运算将序列数据转换为平稳序列。

3) 对于平稳非纯随机序列，可以进行时间序列建模。根据其自相关系数和偏相关系数，选择 AR/MA/ARMA/ARIMA 模型，估计模型中未知参数的值并进行参数检验，最后进行模型检验。

本题使用了 ARMA 模型，即自回归滑动平均模型，最终得到了所有供应商未来 24 周每周的供货量（见支撑材料中：“未来 24 周 402 家供应商供货量预计.xlsx”）。

5.2.2 0-1 整数规划模型的建立与求解

5.2.2.1 求供应商最小数

时间序列能够得到第 i 个供应商在第 j 周的供货量（等价替换为材料 C）e_{ij}，本题需要找到数量最少的供应商，因此可以令$x_i(1 \le i \le 50)$代表是否选择第 i 个供应商，$x_i = 1$代表选择，$x_i = 0$代表不选择。

则可建立以下目标函数：

$$min\sum_{i=1}^{50} x_i$$

每周选择的供应商的供货量需要满足该周的生产需求，记能满足企业生产需求的最小供货量为 Q，则可以得到其表达式：

$$Q = 28200 * 0.72 * 101\%$$

这里 28200 代表每周生产量为 28200m^3，0.72 代表每 1 m^3 的生产品需要消耗 0.72m^3 的材料 C。统计近五年 8 家转运商的损耗率平均值，可以计算出损耗率约为 0.99%，因此乘 101%代表供应商需要供应的值。同时上一周剩余的原材料也可以用于下一周的生产，因此有以下约束：

$$\begin{cases} \sum_{i=1}^{50} e_{i1}x_i \geq Q \\ \sum_{i=1}^{50} e_{i1}x_i + \sum_{i=1}^{50} e_{i2}x_i \geq 2Q \\ \cdots\cdots\cdots\cdots\cdots \\ \sum_{i=1}^{50}\sum_{j=1}^{24} e_{ij}x_i \geq 24Q \end{cases}$$

求解这一 0-1 整数规划模型，可以得到选择的供应商（见下表）：

表 2 选择的最少数目的供货商列表

S031	S194	S306	S356
S040	S201	S307	S361
S055	S229	S308	S364
S086	S244	S329	S365
S108	S247	S330	S367
S131	S268	S338	S374
S139	S275	S340	S395
S140	S282	S346	
S143	S284	S348	
S151	S294	S352	

供应商最小数为 37。制作接下来 24 周每周满足企业生产需求的最小供货量 Q 与每周选出的 37 家供应商的供货量之和对比图，可以发现二者十分接近，可见这 37 家供应商是能实现满足生产需求的最小数量供应商。

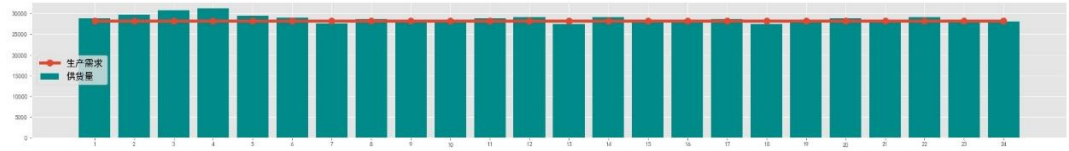

图 6 选中的 37 家供应商供货量之和与生产需求关系

5.2.2.2 求每周选择哪些供应商

将每个提供 A,B 原材料的供应商的供货量由等价替换成的材料 C 的供货量重新转换成材料 A,B 的供货量。这 37 家供应商中，提供 A 材料的供应商有 10 个，提供 B 材料的供应商有 14 个，提供 C 材料的供应商有 13 个。

将 37 家供应商重新排序，材料 A 供应商排在最前，其后是材料 B 供应商，材料 C 供应商排在最后。

设第 i 个供应商在第 j 周的供货量为 f_{ij}。要求得最经济的原材料订购方案，则需要建立一个能反映总采购费用的目标函数。即采购 A,B,C 材料的费用之和。则可建立以下目标函数：

$$\min 1.2*i=110j=124fijxij+1.1*$$
$$i=1124j=124fijxij+i=2537j=124fijxij\min 1.2*\sum_{i=1}^{10}\sum_{j=1}^{24}f_{ij}x_{ij}+1.1*$$
$$\sum_{i=11}^{24}\sum_{j=1}^{24}f_{ij}x_{ij}+\sum_{i=25}^{37}\sum_{j=1}^{24}f_{ij}x_{ij}$$

系数 1.2 和 1.1 指材料 A,B 相对于材料 C（单位 1）的价格，连续的两个求和符号分别计算了材料 A,B,C 的购买量。

令材料 A,B,C 的消耗率为生产每立方米产品需要消耗的对应材料的体积。每周选择的供应商的供货量需要满足该周的生产需求，即提供材料 A,B,C 的供应商的供货量，除以各自材料的消耗率的和，再相加，要大于企业每周的生产量与 1-损耗率（转运过程中的损耗量与供货量的比）的比，这里我们用Q_1表示。

企业每周生产量为 28200m^3。统计近五年 8 家转运商的损耗率平均值，可以计算出损耗率约为 1.38%，为了便于计算估计为 100/101。因此$Q_1 = 28200 * 1.01$。

则可以得到以下约束：

$$\begin{cases} \frac{1}{0.6}\sum_{i=1}^{10}f_{i1}x_{i1}+\frac{1}{0.66}\sum_{i=11}^{24}f_{i1}x_{i1}+\frac{1}{0.72}\sum_{i=25}^{37}f_{i1}x_{i1}\geq Q_1 \\ \frac{1}{0.6}\sum_{i=1}^{10}f_{i2}x_{i2}+\frac{1}{0.66}\sum_{i=11}^{24}f_{i2}x_{i2}+\frac{1}{0.72}\sum_{i=25}^{37}f_{i2}x_{i2}\geq Q_1 \\ \cdots\cdots\cdots\cdots\cdots\cdots \\ \frac{1}{0.6}\sum_{i=1}^{10}f_{i,24}x_{i,24}+\frac{1}{0.66}\sum_{i=11}^{24}f_{i,24}x_{i,24}+\frac{1}{0.72}\sum_{i=25}^{37}f_{i,24}x_{i,24}\geq Q_1 \end{cases}$$

求解这一 0-1 整数规划模型，可以得到接下来 24 周每周选择的供货商。

5.2.3 根据历史缺货统计情况计算未来订货量

求解得到每周所选择的供货商列表后，对应前述预测的未来 24 周供货量，可以得知未来的供货情况。但是根据历史情况可得知，一周的订货量一般超出实际供货量许多，这个情况可以称为“缺货”，是供货商受到季节和行情影响而产生的。

因此，对于过去 5 年的 10 个周期（24 周为一周期）计算各周期的平均缺货率，得到缺货率向量gap，其中 o 代表平均订货向量，s 代表平均供货向量,下标 h 表示历史统计量，p 代表未来预测量：

$$gap = \frac{o_h - s_h}{s_h}$$

则可以求得未来应该发给供货商的订单量O_p，其中 O 是订单矩阵，S 是供货矩阵，G 是以gap为行的缺货率矩阵，E 是全 1 矩阵，∘是元素乘法。

$$O_p = (G + E) \circ S_p$$

5.2.4 0-1 背包模型的建立与求解

（1）对转运商进行排序

为了获得损耗最少的转运方案，首先将转运商的平均损耗率、高损耗频次（达到损耗率 5%极限值的次数）、无订单频次（损耗率为 0 的次数）分别进行排名，赋分 1-8。之后将平均损耗率、高损耗频次、无订单频次的 3 个单项得分以 60%、20%、20%的比例折算相加，计算得转运商最终评分，并制定转运商选取的排序。由于排名越靠前赋分越小，因此总分也以小者为优。具体得分及排名如下表所示。

表 3 转运商得分与排名

转运商	5 年平均损耗率	高损耗频次	无订单频次	平均损耗率得分	高损耗频次得分	无订单频次得分	总得分	排名
T1	1.90	0	4	6	1	3	4.6	5
T2	0.92	0	1	3	1	1	2.2	2
T3	0.19	123	1	1	6	1	1.5	1
T4	1.57	138	21	5	7	6	5.5	6
T5	2.89	157	34	8	8	7	7.7	8
T6	0.54	25	11	2	4	4	2.8	3
T7	2.08	0	41	7	1	8	6.7	7
T8	1.01	37	12	4	5	5	4.4	4

因此，本题中转运商的优先级顺序是 T3、T2、T6、T8、T1、T4、T7、T5。

（2）利用 0-1 背包为每家供应商选择转运商

对于转运商的限制有：

a) 每家转运商的运输能力为 6000 立方米/周。

b) 通常情况下，一家供应商每周供应的原材料尽量由一家转运商运输。

在已经确定了每周选择哪些供应商，以及供应商每周会供货的量的基础上，可以采取 0-1 背包算法来解决选择转运商的问题。具体思路为：

1) 按照转运商的优先级顺序，分别设置八个“背包”，其中第一个背包为优先级最高的转运商，即为 T3。

2) 利用 0-1 背包算法，计算哪些供应商的供货量可以放进“背包”里，即用第一个转运商进行运输。每个“物品”的重量为供应商的供货量 $a_{ij}/b_{ij}/c_{ij}$，“价值”为 A,B,C 材料的单位价值。

3) 如果进行一次 0-1 背包计算后，仍有供应商的原材料没有转运商运输，则把按优先级顺序的下一位转运商作为新的“背包”，继续装填。

材料 A,B,C 的单位价值采取如下计算方法：

每立方米产品需要消耗 0.6 立方米的材料 A，每立方米材料 A 需要花费 1.2x 元（x 代表材料 C 的单价），则每立方米产品用材料 A 来制作，需花费 0.72x 元，即用 1 元购买材料 A，可以制造 1/0.72x 立方米的产品。同理，材料 B、C 可以制造 1/0.726x，1/0.72x 立方米的产品。

则材料 A、B、C 的价值比为：$\frac{1}{0.72}$：$\frac{1}{0.726}$：$\frac{1}{0.72}$。

根据这些条件，可以解出所有供应商使用哪家转运商进行材料运输及其运输量。

5.2.5 方案实施效果分析

5.2.5.1 订购方案实施效果分析

制定好订购方案后，我们可以计算出新订购方案下 24 周每周的预计库存，将其与过去 240 周平均到 24 周的历史库存，和正好满足企业两个周生产需求的安全库存线进行对比，可以得到一个对比图。从图中可以看出，由于 0-1 线性规划约束条件的限制，总体上库存保持在较为稳定的水平，即能够满足两周生产需求的货物存量且库存不过多。相比历史上的 10 个周期的均值（以 24 周为一个周期），保持了更为稳定的库存水平，不论是对于保障企业生产进行还是节省仓储管理费用都是有益的。

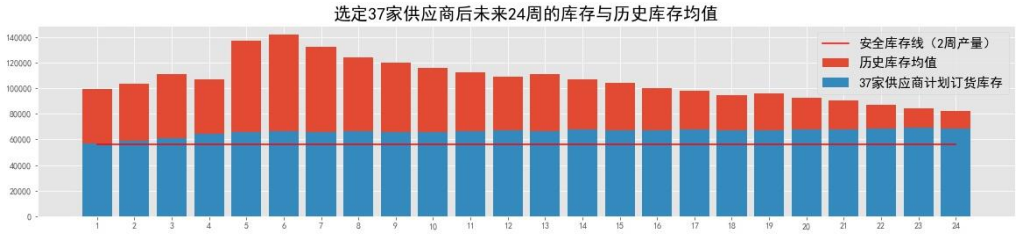

图 7 新的订购方案的 24 周预计库存与历史平均库存对比图

同时，由于限定了选择 37 家供应商，订货的选择空间受到了局限，因此在三种商品的选择上受到限制，又由于三种商品的性价比也有所区别，故而无法达成全局的订货方案（包括 402 家供应商的情况）最优。

因此，在本题的理想模型中，当供应商的选择空间受限时，优化过的未来订货方案的成本无论是与历史水平保持了一致，还是有所提高，都是可以理解的。而在现实中，显然，和数量相对有限的供应商保持长期的稳固合作关系对于生产企业来说显然是更有利的选择，能够一定程度上降低商业合作、企业管理等多方面的成本。最重要的是，生产企业很有可能从和上游供应商的长期、稳固、规模性的合作关系中得到特别优惠。因此，本题中对商业合作伙伴的重要程度的衡量是相当具有价值的行为。

5.2.5.2 转运方案实施效果分析

要求给出损耗最少的转运方案，因此要分析转运方案的实施效果，就要计算损耗的量。此处将新制定的方案与过去 5 年的转运数据（折算为 24 周）进行对比。由于过去 5 年的数据仅有总供货数量、没有 8 家转运商各自的转运数量，因此假设各转运商平均分配货物。

设原材料 A、B、C 的价值分别为 1.2x、1.1x 和 x。根据转运数量、平均损耗率可以计算和对比两种转运方案的损耗量，以及损耗的价值，如下表所示。发现未来 24 周所采用的的转运方式相较过去五年在损耗数量上明显下降，损耗价值明显更少，凸显新出转运方案的优秀实施效果。

表 4 转运方案损耗数量和损耗价格明细（单位：元）

转运方案	转运数量			损耗数量			损耗价值
	A	B	C	A	B	C	
未来 24 周	136028.6	225189.5	86210.0	814.6	1348.5	516.2	2977.0x
过去 5 年（折算为 24 周）	205690.3	191724.1	185527.9	1231.7	1148.1	1111.0	3851.9x

5.3 问题三的建模与求解

5.3.1 0-1 整数规划模型的建立与求解

现有文献中，我们可以了解到常用建筑和装饰板材的分类[1]。由于密度板（纤维板）的原料为“木质纤维或其他植物纤维”，与该企业所购的原材料“木质纤维和其他植物素纤维”吻合，因此可以假设该企业生产产品即为密度板。

表 5 常用建筑和装饰板材的分类[1]

板材	**原料**	**特点**
实木板	完整木材	原木纹路、自然耐用
装饰面板	实木板经过切割获得的薄木皮	美观，可用于单面装饰
细木工板（大芯板）	单板、拼接木板	强度高，横向抗弯抗压
刨花板	木材碎料、添加剂等	强度低，格极低廉
密度板（纤维板）	**木质纤维或其他植物纤维**	质地柔软

防火板	钙质材料或硅质材料等	防火
三聚氰胺板	纸、三聚氰胺树脂胶粘剂等	具有装饰作用

其中，密度板（纤维板）又可分为高密度、中密度、低密度三个级别。中密度纤维板应用最为广泛，产量最高，在 2013 年已达 7600 万立方米。且中密度纤维板是国家鼓励的木材综合利用产品，因此假设该企业生产产品为中密度纤维板。

中密度纤维板的原材料主要是木质纤维，也即木材采伐剩余物，通常称为枝桠材。查阅文献，列出各地区枝桠材的采购价格[2]。

表 6 2014 年 9 月各地区枝桠材采购价格

地区	山东	河北	河南	江苏	福建	两广	云南
采购价（元/吨）	480-490	520	470	480	440-460	450-520	450

根据上表，可发现密度板原材料枝桠材的采购价为 480 元/吨左右，因此设置原材料 C 采购价为 480 元/吨。相应的，B 为 528 元/吨、A 为 576 元/吨。假设 1m^3 枝桠材的质量为 100kg，则原材料价格为：A 为 57.6 元/m^3，B 为 52.8 元/m^3，C 为 48.0 元/m^3。

查阅文献可知，2020 年 12 月，我国大中城市平均公路货运价格约为 0.49 元/吨·公里[3]。

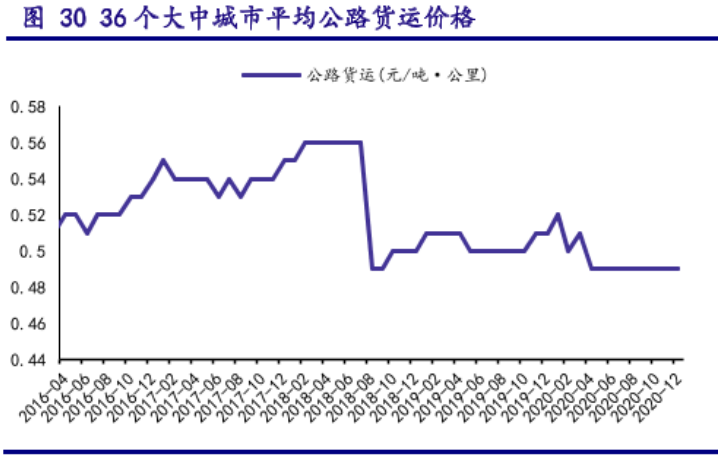

图 8 平均公路货运价格变化趋势[3]

通常，木材使用企业在木材产区附近布局较为便利，因此假设该企业的原材料运输距离为 200km[4]。假设仓储费用为 2 元/m^3，备货两周，因此货物仓储 14 天。根据以上信息，可以计算出原材料 A,B,C 的各项成本。

表 7 三种原材料的成本计算

原材料分类	**原材料单价**	**200km 运输费用**	**14 天仓储费用**	**每购买 1 立方米原料的成本**	**每制成 1 立方米产品的成本**
A	57.6	9.8	28	95.4	57.2
B	52.8	9.8	28	90.6	59.8
C	48.0	9.8	28	85.8	61.8
单位	元/m^3	元/m^3	元/m^3	元	元

本题需要在 402 家供应商的基础上进行分析，因此在使用时间序列模型，获得每个供应商未来 24 周的等价材料 C 的供货量后，再将提供 A,B 原材料的供应商的供货量由等价替换成的材料 C 的供货量重新转换成材料 A,B 的供货量。这 402 家供应商中，提供 A 材料的供应商有 146 家，提供 B 材料的供应商有 134

家，提供 C 材料的供应商有 122 家。将 37 家供应商重新排序，材料 A 供应商排在最前，其后是材料 B 供应商，材料 C 供应商排在最后。

设第 i 个供应商在第 j 周的供货量为 f_{ij}。要求得最经济的原材料订购方案，则需要建立一个能反映总费用的目标函数。即采购、运输与储存 A,B,C 材料的费用之和。则可建立以下目标函数：

$$\min 95.4 * \sum_{i=1}^{146}\sum_{j=1}^{24} f_{ij}x_{ij} + 90.6 * \sum_{i=147}^{280}\sum_{j=1}^{24} f_{ij}x_{ij} + 85.8 * \sum_{i=281}^{402}\sum_{j=1}^{24} f_{ij}x_{ij}$$

系数指每购买 $1m^3$ 材料 A,B,C 的成本，连续的两个求和符号分别计算了材料 A,B,C 的购买量。

同问题二，令材料 A,B,C 的消耗率为生产每立方米产品需要消耗的对应材料的体积。每周选择的供应商的供货量需要满足该周的生产需求，即提供材料 A,B,C 的供应商的供货量，除以各自材料的消耗率的和，再相加，要大于企业每周的生产量与 1-损耗率（转运过程中的损耗量与供货量的比）的比，这里我们用Q_1表示。

企业每周生产量为 $28200m^3$。统计近五年 8 家转运商的损耗率平均值，可以计算出损耗率约为 0.99%，即 100/101。因此$Q_1 = 28200 * 1.01$。

则可以得到以下约束：

$$\begin{cases} \frac{1}{0.6}\sum_{i=1}^{146} f_{i1}x_{i1} + \frac{1}{0.66}\sum_{i=147}^{280} f_{i1}x_{i1} + \frac{1}{0.72}\sum_{i=281}^{402} f_{i1}x_{i1} \geq Q_1 \\ \frac{1}{0.6}\sum_{i=1}^{146} f_{i2}x_{i2} + \frac{1}{0.66}\sum_{i=147}^{280} f_{i2}x_{i2} + \frac{1}{0.72}\sum_{i=281}^{402} f_{i2}x_{i2} \geq Q_1 \\ \cdots\cdots\cdots\cdots\cdots\cdots \\ \frac{1}{0.6}\sum_{i=1}^{146} f_{i,24}x_{i,24} + \frac{1}{0.66}\sum_{i=147}^{280} f_{i,24}x_{i,24} + \frac{1}{0.72}\sum_{i=281}^{402} f_{i,24}x_{i,24} \geq Q_1 \end{cases}$$

求解这一 0-1 整数规划模型，可以得到接下来 24 周每周选择的供货商。

5.3.2 根据历史缺货统计情况计算未来订货量

同问题二，不再多加赘述。

5.3.3 0-1 背包模型的建立与求解

（1）对转运商进行排序

本题需要使得转运商的损耗率尽可能少，因此我们将转运商的 5 年平均损耗率进行排名，结果同下表：

表 8 转运商排名

转运商	5 年平均损耗率	排名
T1	1.90	6
T2	0.92	3
T3	0.19	1
T4	1.57	5
T5	2.89	8
T6	0.54	2
T7	2.08	7
T8	1.01	4

因此转运商的优先级顺序是 T3、T6、T2、T8、T4、T1、T7、T5。

（2）利用 0-1 背包为每家供应商选择转运商

本题同样可以使用 0-1 背包问题进行转运商的选择，但与问题二有所不同的是本题要求出损耗率最少的转运方案，设第 i 家转运商转运的供货量为g_i，转运后的接受量为h_i，240 周平均损耗率为d_i，则可得到总损耗率的计算公式：

$$总损耗率 = \frac{\sum_{i=1}^{8} g_i - \sum_{i=1}^{8} h_i}{\sum_{i=1}^{8} g_i} = \frac{\sum_{i=1}^{8} g_i * d_i}{\sum_{i=1}^{8} g_i}$$

此时，货物损耗率考虑的是损耗货物的数量，而非单位价值。易得要使总损耗率最少，则要尽可能使得损耗率低的转运商转运更多的货物（d_i小的转运商g_i大）。因此仍然选用 0-1 背包模型，但将 A、B、C 三种原材料的单位价值设置为 1:1:1。从损耗率低的企业开始安排转运原材料，装尽可能多的原材料后，再安排损耗率第二低的企业。

5.3.4 方案实施效果分析

5.3.4.1 订购方案实施效果分析

制定好新的订购方案后，同样可以计算出新订购方案下 24 周每周的预计库存，将其与过去 240 周平均到 24 周的历史库存，和正好满足企业两个周生产需求的标准库存进行对比，可以得到一个对比图。图中可以看出，新的订购方案的原材料库存在基本维持在满足两个周生产需求的同时，比起过去 5 年的历史库存有了大幅度的减少，仓储费用也随之减少，因此新的订购方案在保障企业生产的同时，也节约了仓储管理费用。

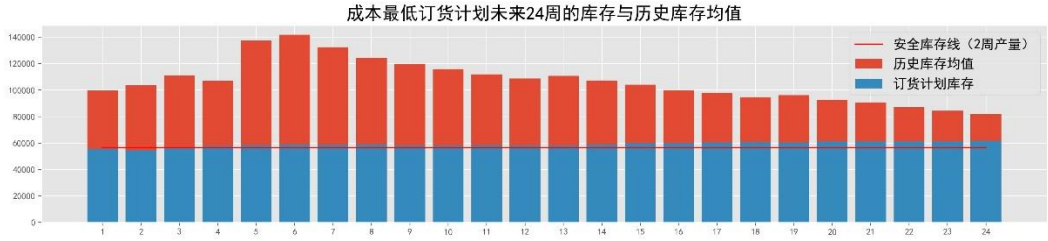

图 9 新的订购方案的 24 周预计库存与历史平均库存对比图

同时可以计算在得到新的订购方案下未来 24 周的购买成本与运费开销，虽然比过去 5 年的历史开销略大，但仓储费用却大幅度减小，总的开销也随之大幅减少，节约了约 31.8%的成本。因此本题选择的订购方案具有较强的实施效果。

表 9 订购方案的花费明细（单位：元）

	购买成本与运费之和	仓储费用	总价
未来 24 周预计	28004658.83	39677059.39	67681718
过去 5 年历史（转换为 24 周）	27546327.16	71696279	99242606

5.3.4.2 转运方案实施效果分析

此处将新制定的方案与过去 5 年的转运损耗率进行对比。由于过去 5 年的数据仅有总供货数量、没有 8 家转运商各自的转运数量，因此假设各转运商平均分配货物，也即损耗率为 8 家转运商的平均值，并将过去 5 年的损耗率转换为 24 周与未来 24 周进行比较，结果如下表所示。

发现未来 24 周所采用的的转运方式相较过去五年在损耗率上明显下降，降低了 57.6%的损耗率，凸显新出转运方案的优秀实施效果。

表 10 转运方案的平均损耗率

转运方案	转运数量								平均损耗率
	T1	T2	T3	T4	T5	T6	T7	T8	
未来 24 周	0	115992.4	138181.4	0	0	133012.3	0	53920.4	0.588%
过去 5 年	72867.8								1.388%

5.4 问题四的建模与求解

5.4.1 模型的假设

本问的优化目标是企业调整产能，以获得最高的利润，因此需要确定产品的售价、生产产品的总成本。

查阅资料得，2006 年至 2016 年间，主要纤维板的销售价格在 1500 元/m^3 左右浮动，因此假设本企业的产品售价为 1500 元/m^3[5]。

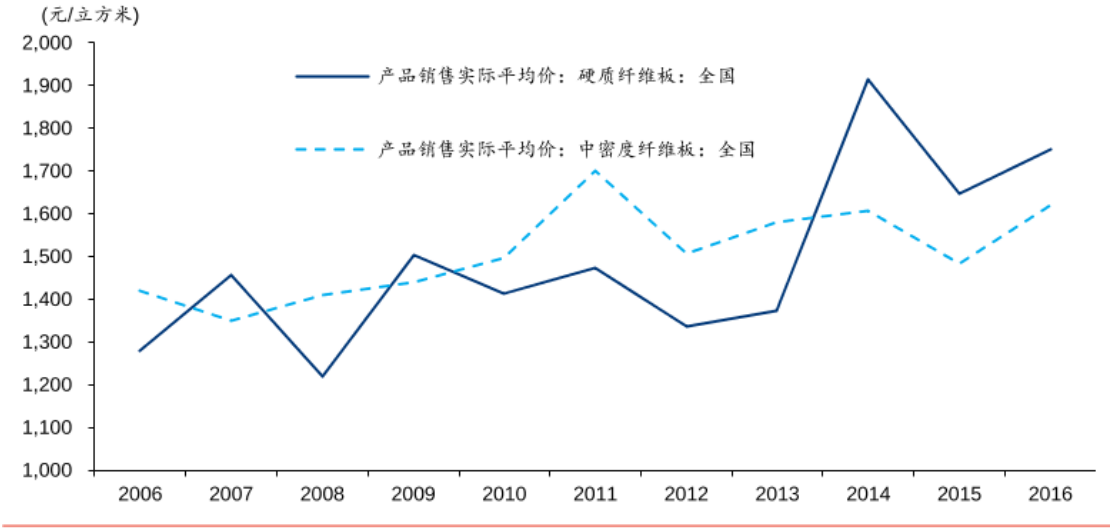

图 10 主要纤维板销售价格[5]

为确定生产产品的总成本，查阅相关文献。目前已知 A、B、C 三种木质原材料的价格，通过木质原料占原材料的 41%可以推算原料成本[6]。

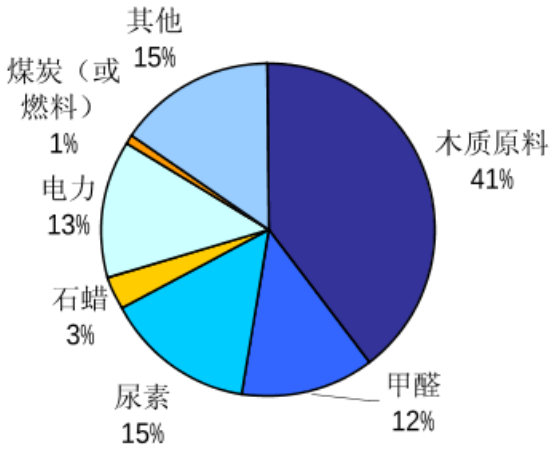

图 11 纤维板原材料的单位成本占比[6]

表 11 原料总成本计算

木质原材料分类	每 1m³ 木质原料的成本（元）	每 1m³ 木质原料所需其他原料的成本（元）	每 1m³ 木质原料所需的原料总成本（元）	每 1m³ 产品所需的原料总成本（元）
A	95.4	137.3	232.7	139.6
B	90.6	130.4	221.0	145.9
C	85.8	123.5	209.3	150.7

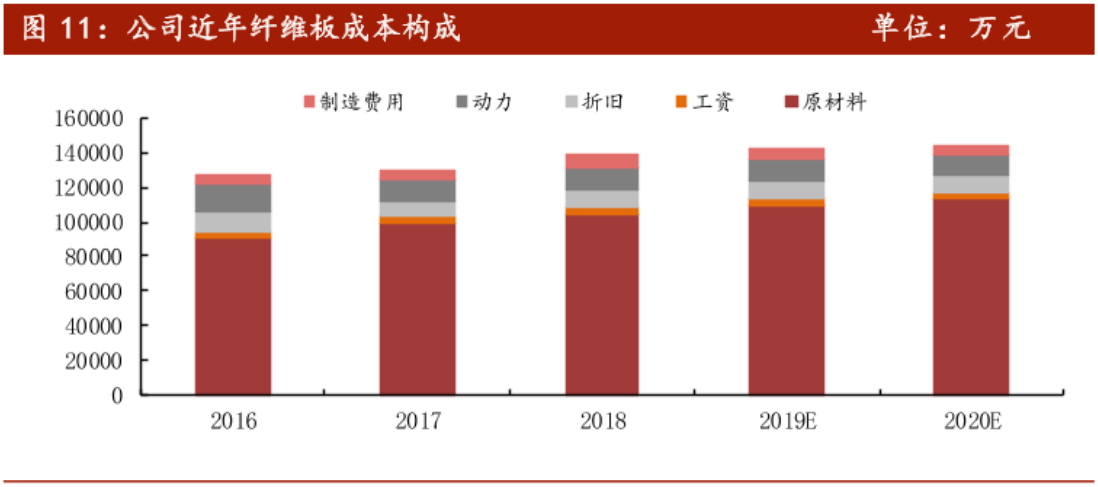

图 12 纤维板成本构成[7]

查阅文献，原材料以外的成本（包括制造费用、动力费用、折旧费用、工人工资等）与原材料成本成本比例约为 3:7[7]。由于每 1m³ 产品所需的原料总成本平均为 145.3 元，因此设每 1m³ 产品的原材料以外的成本为 62.3 元。

5.4.2 线性规划模型的建立与求解

本题的产能是一个变量，需要根据现有产能、402 家原材料供应商的供应能力、8 家转运商的转运能力进行约束。利用时间序列模型，获得考虑库存量的产能可行的取值范围后，再建立与产能、原材料成本、原材料以外的成本有关的线性规划模型，求解获利最大条件下的最优产能。

（1）求解产能的取值范围

技术改造带来产能提升，因此新产能 P 大于原产能 28200m³/周。产能提升不应过剩，即需要满足：

a)P 应当小于每周最大转运量(48000m³/周)；

b)对任意一周，P 应当小于最大到货量与产品库存之和。即：

目标函数：$\max S$

约束条件：

$$\begin{cases} \frac{1}{0.6}\sum_{i=1}^{146} f_{i1}x_{i1} + \frac{1}{0.66}\sum_{i=147}^{280} f_{i1}x_{i1} + \frac{1}{0.72}\sum_{i=281}^{402} f_{i1}x_{i1} \geq -0.5S \\ \frac{1}{0.6}\left(\sum_{i=1}^{146} f_{i1}x_i + \sum_{i=1}^{146} f_{i2}x_i\right) + \frac{1}{0.66}\left(\sum_{i=147}^{280} f_{i1}x_i + \sum_{i=147}^{280} f_{i2}x_i\right) + \frac{1}{0.72}\left(\sum_{i=281}^{402} f_{i1}x_i + \sum_{i=281}^{402} f_{i2}x_i\right) \geq S \\ \cdots\cdots\cdots\cdots \\ \frac{1}{0.6}\sum_{i=1}^{146}\sum_{j=1}^{24} f_{ij}x_{ij} + \frac{1}{0.66}\sum_{i=147}^{280}\sum_{j=1}^{24} f_{ij}x_{ij} + \frac{1}{0.72}\sum_{i=281}^{402}\sum_{j=1}^{24} f_{ij}x_{ij} \geq (24 * 1.5 - 2)S \end{cases}$$

求解出的 P 取值范围为(28200,34855]。

（2）建立线性规划模型

本题的目标是获得最高的利润，因此目标函数为产品售价（每 $1m^3$ 售价为 1500 元），减去 A、B、C 产品换算得到的原材料成本，减去原材料以外的其他成本。

目标函数：

$$max1500P - 232.7*\sum_{i=1}^{146}\sum_{j=1}^{24} f_{ij}x_{ij} - 211*\sum_{i=147}^{280}\sum_{j=1}^{24} f_{ij}x_{ij} - 209.3*\sum_{i=281}^{402}\sum_{j=1}^{24} f_{ij}x_{ij} - 62.3P$$

约束条件除产能自身限制以外，每一周的供应量加仓库储量应当大于本周消耗量。即：供+余-消耗≥0。其中，第 1 周初始库存设为 2P，第 1-24 周过程中的每周库存最低限制设为 P，以减小存储费用、尽可能多地创造利润。前一周的剩余库存均会传递至下一周。

约束条件：

$$\begin{cases} 28200 < P \le 34855 \\ \frac{1}{0.6}\sum_{i=1}^{146} f_{i1}x_{i1} + \frac{1}{0.66}\sum_{i=147}^{280} f_{i1}x_{i1} + \frac{1}{0.72}\sum_{i=281}^{402} f_{i1}x_{i1} + P \ge 0 \\ \frac{1}{0.6}\left(\sum_{i=1}^{146} f_{i1}x_i + \sum_{i=1}^{146} f_{i2}x_i\right) + \frac{1}{0.66}\left(\sum_{i=147}^{280} f_{i1}x_i + \sum_{i=147}^{280} f_{i2}x_i\right) + \frac{1}{0.72}\left(\sum_{i=281}^{402} f_{i1}x_i + \sum_{i=281}^{402} f_{i2}x_i\right) \ge 0 \\ \cdots\cdots \\ \frac{1}{0.6}\sum_{i=1}^{146}\sum_{j=1}^{24} f_{ij}x_{ij} + \frac{1}{0.66}\sum_{i=147}^{280}\sum_{j=1}^{24} f_{ij}x_{ij} + \frac{1}{0.72}\sum_{i=281}^{402}\sum_{j=1}^{24} f_{ij}x_{ij} - 22P \ge 0 \end{cases}$$

解得的最优产能为 $34855m^3$/周。

根据该新产能计算库存变化情况，如下图。在约束条件下，库存从两倍产能起始，随供应量的上升而略有上下浮动，最后随约束稳步下降，直至与产能逐渐接近。该库存变化情况存储量小、仓储费用较低，同时与较高的产能相配合，共同达成获取最高利润的目标。

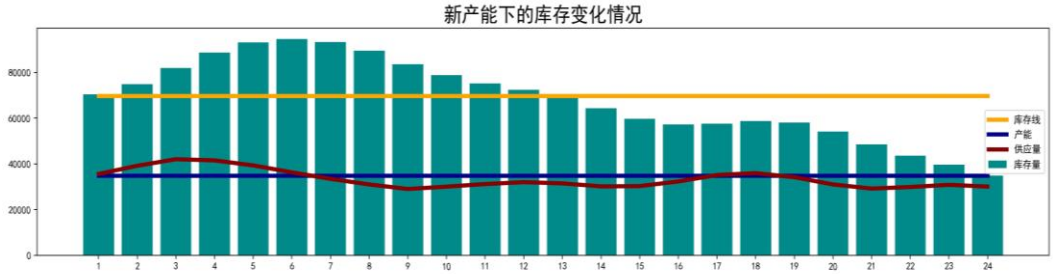

图 13 新产能下的库存变化情况

5.4.3 线性回归模型的建立与求解

同问题二、三。

5.4.4 0-1 背包模型的建立与求解

本题的转运规划目标同样是获得损耗最少的转运方案，因此同问题二，将平均损耗率、高损耗频次、无订单频次以 60%、20%、20%的比例折算相加，计算

得转运商最终评分，并制定转运商选取的排序。因此，本问与第二问中选取转运商的顺序同为 T3、T2、T6、T8、T1、T4、T7、T5。

仍然选用 0-1 背包模型，设原材料 A、B、C 的价值分别根据每 $1m^3$ 产品所需的原料总成本设为 1/139.6、1/145.9、1/150.7。根据转运数量、平均损耗率可以计算和对比两种转运方案的平均损耗率，如下表所示。在该问下采用的转运方式相较过去五年在损耗数量上同样存在明显下降，为 0.684%。该值相较于第 3 问的 0.588%稍显逊色，原因是产能上升、运转原材料数变多，而不得不选取损耗率次之的企业。由于损耗率差异极小，因此本转运规划仍然符合利润最大化的目标。

表 12 转运方案的平均损耗率

转运方案	转运数量								平均损耗率
	T1	T2	T3	T4	T5	T6	T7	T8	
未来 24 周	0	115992.4	138181.4	0	0	133012.3	0	53920.4	0.684%
过去 5 年	72867.8								1.388%

六、模型的评价、改进与推广

6.1 模型优点：

（1）模型得出的结果与预期较为符合；

（2）详略得当，在面对比较复杂的规划问题时，模型能够准确把握关键的约束条件和变量关系，保持较好精度。另一方面，模型假设合理，在建模中适当忽略次要问题，保证模型精度的同，求解难度适中；

（3）模型灵活，对于问题缺少的数据，有充分和多样的的分析和预测手段。同时也不滥用模型，对于复杂但效果不好的回归方法果断予以舍弃。

6.2 模型缺点：

（1）在进行求解的过程中，涉及大量大规模矩阵运算和求解，求解有一定难度；

（2）对于未来数据的预测缺少验证和检验，准确性缺少保证，且后续分析都建立在这些预测上，更进一步增加了模型的不确定性；

（3）优化结果较为理想化，缺少现实方面的考量，有很多小数订单，合理性有待讨论。

6.3 模型改进：

（1）模型主要的改进在于提高预测的准确度，通过对过去时间序列进行分割产生训练集和验证集，从而能够对模型预测能力进行量化；

（2）另一方面，选择性能更强的预测方法，通过提取更多维度数据，也能增强模型预测能力；

（3）优化方面，运用更高级的多目标规划方法，能够更好地解决问题。

6.3 模型推广：

（1）模型具有一定的预测和优化能力，对于类似的多分类物流问题都有一定的处理分析能力；

（2）模型具有复杂的自下而上进行优化预测的能力，适用于各种数据较为丰富的预测和优化问题，可以给出直观、可解释的结果

七、参考文献

[1]李备.家庭装修中经常使用的板材及选用所遵循的原则[J].现代装饰(理论),2012(06):28.

[2]证券时报,金九银十纤维板期货继续低迷,2014.https://www.bmlink.com/news/938071.html

[3]王靖添,共同富裕下快递行业发展环境改善，积极把握公路货运数字化趋势机遇——2021 年 8 月行业动态报告[R].中国银河证券研究院,2021(08):16.

[4]肖武,陈昱霏,蒋瑞.我国木材行业物流运输现状分析及对策[J].全国流通经济,2017(22):16-17.

[5]李斌,邱乐园,孙雪琬.传统主业稳健，外延布局锂电+稀土[R].威华股份,2021:8.

[6]周海晨.丰林集团—合理股价为 13.5－15.7 元，预计首日价格在 14 元左右[R].轻工制造,2011:4.

[7]李帅华,马妍.纤板为盾，锂业为矛[R].威华股份,2019:9.

[8]吴美琼,陈秀贵.基于主成分分析法的钦州市耕地面积变化及其驱动力分析[J].地理科学,2014,34(01):54-59.

[9]刘潇,薛莹,纪毓鹏,徐宾铎,任一平.基于主成分分析法的黄河口及其邻近水域水质评价[J].中国环境科学,2015,35(10):3187-3192.

[10]林秋红.整数规划在数学建模中的应用[J].大众科技,2010(05):21-22+30.

[11]胡明.整数规划在数学建模竞赛中的应用初探[J].江苏教育学院学报(自然科学版),2009,26(02):1-4.

附录

1.初始数据处理 Python 代码

```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns
plt.rcParams['font.family'] = ['sans-serif']
plt.rcParams['font.sans-serif'] = ['SimHei']
sup = pd.read_csv('supplyer.csv')
sup.columns
sup_weeks_sum = sup.drop(['供应商 ID', '材料分类'],
axis=1).agg('sum')
sup_weeks_sum
plt.figure(figsize=[30,4])
plt.xticks(range(1,242,4))
plt.plot(range(1,241), sup_weeks_sum)

order = pd.read_csv('orders.csv')
order.columns
ord_weeks_sum = order.drop(['供应商 ID', '材料分类'],
axis=1).agg('sum')
ord_weeks_sum
plt.figure(figsize=[30,4],dpi=200)
plt.style.use('ggplot')
plt.xticks(range(1,242,4),fontsize=10)
plt.title('历史订货供货量', fontsize=20)
plt.plot(range(1,241), sup_weeks_sum, label='supply')
plt.plot(range(1,241), ord_weeks_sum, label='order')
# plt.bar(range(1,241), sup_weeks_sum, label='supply')
# plt.bar(range(1,241), ord_weeks_sum, label='order')
plt.legend(fontsize=15)
plt.savefig('历史订货供货量.jpg')
sup_agg_cat = sup.drop('供应商 ID', axis=1).groupby('材料分类
').agg('sum')
sup_agg_cat = sup_agg_cat.T
plt.figure(figsize=[30,4])
sns.lineplot(data=sup_agg_cat)
sns.heatmap(sup_agg_cat[['A','B','C']].corr(), cmap='rainbow')
(np.cov(sup_agg_cat['A'],sup_agg_cat['C'])[0,1],
np.cov(sup_agg_cat['A'],sup_agg_cat['B'])[0,1],
np.cov(sup_agg_cat['B'],sup_agg_cat['C'])[0,1])
ord_agg_cat = order.drop('供应商 ID', axis=1).groupby('材料分类
```

```
').agg('sum')
ord_agg_cat = ord_agg_cat.T
plt.figure(figsize=[30,4])
sns.lineplot(data=ord_agg_cat)
plt.savefig('ord_agg_cat.jpg')
ord_agg_cat.columns = ['orderA', 'orderB', 'orderC']
sup_agg_cat.columns = ['supplyA', 'supplyB', 'supplyC']
n_df = pd.concat([ord_agg_cat, sup_agg_cat], axis=1)
n_df
sns.pairplot(n_df)
plt.savefig('pair.jpg')
order = pd.read_csv('orders.csv')
sup = pd.read_csv('supplyer.csv')
order = order.drop([402,403], axis=0)
order
gap = sup.drop(['供应商 ID', '材料分类'], axis=1)-order.drop(['供
应商 ID', '材料分类'], axis=1)
gap =gap.T
gap
gap_array = np.array(gap.sum(axis=1)).reshape(10,24).sum(axis=0)
plt.figure(figsize=[30,4])
plt.bar(x=range(24),
height=np.array(gap.sum(axis=1)).reshape(10,24).sum(axis=0))
sup_week  =  np.array(sup.drop(['供应商 ID',  '材料分类'],
axis=1).agg('sum'))
sup_week = sup_week.reshape(10,24).sum(axis=0)
sup_week
ratio = gap_array/sup_week
pd.DataFrame(ratio).to_csv('gap_ratio.csv')
a_1 = pd.read_csv('A_1.csv')

a_1 = a_1.drop(['mat','id'], axis=1)
a_1
i=0
for col in a_1.columns:
    a_1[col] = np.array(a_1[col]*(1-ratio[i]))
    i+=1
a_1.to_csv('A_1.csv', index=False)
binned_gap = gap.copy()
def binned(lis):
    return [-1 if x<0 else 0 for x in lis]
for col in binned_gap.columns:
    collis = binned_gap[col].values
    binned_gap[col] = binned(collis)
```

```
sns.distplot(binned_gap)

gap_ratio = gap / order.drop(['供应商 ID', '材料分类'], axis=1).T

sns.distplot(gap_ratio)
ord_agg_cat
cat_sum = ord_agg_cat.agg('sum')
plt.bar(x=['A','B','C'],height=cat_sum.values/240,color=['dark
cyan','darkblue','gray'])
plt.title('History Orders Count')
plt.savefig('History Orders Count.jpg')

sup_sum = sup_agg_cat.agg('sum')
plt.bar(x=['A','B','C'],height=sup_sum.values/240,color=['dark
cyan','darkblue','gray'])
plt.title('History Supply Count')
plt.savefig('History Supply Count.jpg')
plt.style.use('ggplot')
plt.figure(dpi = 200, figsize =[9,5] )
x_index = np.arange(3)   #柱的索引
x_data = ('A', 'B', 'C')
y1_data = cat_sum.values/240
y2_data = sup_sum.values/240
bar_width = 0.2  #定义一个数字代表每个独立柱的宽度
plt.rcParams['font.family'] = ['sans-serif']
plt.rcParams['font.sans-serif'] = ['SimHei']
rects1 = plt.bar(x_index, y1_data, width=bar_width ,label='订单
量')           #参数：左偏移、高度、柱宽、透明度、颜色、图例
rects2    =    plt.bar(x_index    +    bar_width,    y2_data,
width=bar_width,label='供货量') #参数：左偏移、高度、柱宽、透明度、
颜色、图例
plt.yticks(fontsize=10)
plt.xticks(x_index + bar_width/2, x_data, fontsize=15)
plt.legend()
plt.title('历史周均订单量和供货量', fontsize = 20)
plt.tight_layout()

plt.savefig('历史周均订单量和供货量.jpg')
plt.figure(figsize=[30,4])
plt.title('Gap-Time Curve')
sns.lineplot(x=range(240), y=abs(gap.T.sum()))
plt.savefig('Gap-Time Curve.jpg')
order_eqC = order.T
for col in order_eqC.columns:
```

```
    colu = order_eqC[col]
    cat = colu[1]
    if cat == 'A':
        order_eqC[col][2:] = colu[2:]*1.2
    elif cat == 'B':
        order_eqC[col][2:] = colu[2:]*1.091

order_eqC = order_eqC.T
order_eqC.to_csv('order_eqC.csv', index=False)
order_eqC.head()
supply_eqC = sup.T
for col in supply_eqC.columns:
    colu = supply_eqC[col]
    cat = colu[1]
    if cat == 'A':
        supply_eqC[col][2:] = colu[2:]*1.2
    elif cat == 'B':
        supply_eqC[col][2:] = colu[2:]*1.091

supply_eqC = supply_eqC.T
supply_eqC.to_csv('supply_eqC.csv', index=False)
supply_eqC.head()

avg_need_eqC = (2.82*10**4)*0.72

trans = pd.read_csv('trans.csv')
def replace0(x):
    if x == 0:
        return np.nan
    else:
        return x

notnull = 8*240 - trans.isnull().sum().sum()
avg_los = (trans.sum().values[1:].sum()/notnull)*0.01



plt.style.use('ggplot')
plt.figure(dpi=200)
need_array_eqC = np.array([avg_need_eqC for i in range(240)])
supply_eqC = pd.read_csv('supply_eqC.csv')
sup_weeks_sum_eqC = supply_eqC.drop(['供应商 ID', '材料分类'],
axis=1).agg('sum')
sup_weeks_sum_eqC    =    np.array(sup_weeks_sum_eqC.values*(1-
```

```
avg_los))
plt.figure(figsize=[30,4])
plt.xticks(range(1,242,4),fontsize=10)

plt.rcParams['font.family'] = ['sans-serif']
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.yticks(fontsize=10)
plt.title('历史供求关系', fontsize = 20)
plt.bar(x=range(1,241), height=need_array_eqC, label='生产需求')
plt.bar(x=range(1,241), height=sup_weeks_sum_eqC, label='供货量
')

plt.legend(fontsize = 15)

plt.savefig('历史供求关系.jpg')
plt.style.use('ggplot')
need_cum_eqC = need_array_eqC.cumsum()
sup_cum_eqC = sup_weeks_sum_eqC.cumsum()

plt.figure(figsize=[30,4],dpi=200)
plt.xticks(range(1,242,4))

plt.plot(range(1,241), need_cum_eqC,  label='累计生产需求')
plt.bar(x=range(1,241),                      height=sup_cum_eqC,
color='gray',label='累计供货量')
plt.legend(fontsize=15)
plt.title('历史累计供求关系', fontsize=20)
plt.savefig('历史累计供求关系.jpg')
storage_eqC = sup_cum_eqC - need_cum_eqC + 2*avg_need_eqC

plt.figure(figsize=[30,4],dpi=200)
plt.xticks(range(1,242,4))
plt.bar(x=range(1,241), height=storage_eqC, label='storage_eqC',
color='darkred')
plt.legend()
plt.title('storage_eqC')
plt.savefig('storage_eqC.jpg')
reshape_storage = storage_eqC.reshape(5,48)
annual_storage = [reshape_storage[:,i] for i in range(48)]

avg_annual_storage = [sum(i)/5 for i in annual_storage]
```

```
avg_annual_storage = np.array(avg_annual_storage)
avg_annual_storage += avg_need_eqC*2

plt.figure(figsize=[30,4],dpi=200)
plt.xticks(range(1,49))
plt.bar(x=range(1,49),                height=avg_annual_storage,
label='avg_annual_storage', color='darkred')
plt.legend()
plt.title('avg_annual_storage')
plt.savefig('avg_annual_storage.jpg')
```

2.解问题一 Python 代码

```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns
plt.rcParams['font.family'] = ['sans-serif']
plt.rcParams['font.sans-serif'] = ['SimHei']
n_sup = pd.DataFrame()
n_sup['id'] = supply_eqC['供应商 ID']
n_sup['supply_amount'] = supply_eqC.drop(['供应商 ID','材料分类'],
axis=1).sum(axis=1)
n_sup['avg_gap']                                                    =
gap.T.sum(axis=1).values/n_sup['supply_amount']

non_zero_count = []
for col in order_eqC.T.columns:
    array = np.array(order_eqC.T[col])
    zeros = np.zeros(len(array))
    not_zero = zeros!=array
    non_zero_count.append(not_zero.sum()-2)
supply_zero_count = []
for col in supply_eqC.T.columns:
    array = np.array(supply_eqC.T[col])
    zeros = np.zeros(len(array))
    is_zero = zeros==array
    supply_zero_count.append(is_zero.sum())
cnt=0
var = []
for col in supply_eqC.T.columns:
    array = np.array(supply_eqC.T[col])[2:]
    avg = array/non_zero_count[cnt]
```

```
    var.append(sum((array-avg)**3)/non_zero_count[cnt])
    # var.append(sum((array-avg))/non_zero_count[cnt])
    cnt += 1

n_sup['annul_var'] = var
n_sup['supply_zero_count'] =supply_zero_count
n_sup.to_csv('supply_features.csv', index=False)
import sklearn
from    sklearn.model_selection    import    train_test_split,
cross_val_score
from sklearn.preprocessing import StandardScaler, MinMaxScaler
from sklearn.linear_model import LogisticRegression
from sklearn.decomposition import PCA
x_train = n_sup.drop('id', axis=1)
sc = StandardScaler()
x_train_std = sc.fit_transform(x_train)
cov_matrix = np.cov(x_train_std.T)
eigen_val, eigen_vec = np.linalg.eig(cov_matrix)
eigen_val
plt.style.use('ggplot')
plt.figure(dpi = 200, figsize =[9,5] )

var_ratio = sorted(eigen_val/(eigen_val.sum()), reverse=True)
cum_var_ratio = np.cumsum(var_ratio)
plt.step(x = range(1,5), y = cum_var_ratio,
       label = '累计方差解释率')
plt.bar(x = range(1,5), height = var_ratio, color='darkcyan' ,
       label = '方差解释率')
plt.xticks([1,2,3,4], fontsize=10)
plt.yticks(np.arange(0,1.1,0.1), fontsize=10)

plt.title('主成分方差解释率',fontsize = 20)
plt.legend(loc = 'right',fontsize=15)
plt.savefig('主成分方差解释率.jpg')

eigen_pairs = [(np.abs(eigen_val[i]),
               eigen_vec[:,i]) for i in range(len(eigen_val))]

eigen_pairs.sort(key=lambda k:k[0], reverse=True)
w = np.c_[eigen_pairs[0][1],eigen_pairs[1][1]]

x_train_pca = x_train_std.dot(w)

n_sup['pc']      =      (x_train_pca[:,0]*var_ratio[0]      +
```

```
x_train_pca[:,1]*var_ratio[1])/(var_ratio[0]+var_ratio[1])
n_sup = n_sup.set_index('pc').sort_index()
n_sup.to_csv('n_sup.csv')
```

3.解问题二 Python 代码

```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns

pred_eqC = supply_eqC.drop(['供应商 ID','材料分类'], axis=1).T

from fbprophet import Prophet
df = pd.read_csv('pred_data.csv')
df['ds'] = pd.to_datetime(df['ds'])

pred_result = pd.DataFrame()
for col in pred_eqC.columns:
    print(col/402*100)
    y = pred_eqC[col].values
    df['y'] = y
    m = Prophet(
               yearly_seasonality=True
               )
    m.fit(df)
    future = m.make_future_dataframe(
                                    periods = 24,
                                    freq='W'
                                    )
    forecast = m.predict(future)
    result = forecast['yhat'][240:].values
    pred_result[col] = result
pred_result.to_csv('TSA_pred.csv')


df['y'] = pred_eqC[6].values
m = Prophet(yearly_seasonality=True)

m.fit(df)
future = m.make_future_dataframe(
                                periods = 24,
```

```
                                freq='W'
                                )

forecast = m.predict(future)
m.plot(forecast)

x = forecast['yhat'][240:].values


from statsmodels.tsa.arima_model import ARIMA
import pmdarima as pm


model = pm.auto_arima(df.y, start_p=1, start_q=1,
                      information_criterion='aic',
                      test='adf',           # use adftest to find
optimal 'd'
                      max_p=4, max_q=4, # maximum p and q
                      m=24,              # frequency of series
                      d=None,           # let model determine 'd'
                      seasonal=True,   # No Seasonality
                      start_P=0,
                      D=0,
                      trace=True,
                      error_action='ignore',
                      suppress_warnings=True,
                      stepwise=True)

print(model.summary())

# Forecast
n_periods = 24
fc,      confint      =      model.predict(n_periods=n_periods,
return_conf_int=True)
index_of_fc = np.arange(len(df.y), len(df.y)+n_periods)

# make series for plotting purpose
fc_series = pd.Series(fc, index=index_of_fc)
lower_series = pd.Series(confint[:, 0], index=index_of_fc)
upper_series = pd.Series(confint[:, 1], index=index_of_fc)

# Plot
plt.plot(df.y)
plt.plot(fc_series, color='darkgreen')
```

```
plt.fill_between(lower_series.index,
                 lower_series,
                 upper_series,
                 color='k', alpha=.15)

import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns
pred_result = pd.read_csv('TSA_decided.csv')
pred_result['id'] = pred_result['Unnamed: 0']+1

pred_result.set_index('id',inplace=True)
pred_result.drop('Unnamed: 0',axis=1, inplace=True)

pred_result
candidates = pd.read_csv('n_sup_decided.csv')
candidates['n_id']     =     [int(x[1:])     for     x     in
candidates['id'].values]
selected = candidates.loc[range(50)]
selected
selected.supply_amount.values.sum()/5
avg_need_eqC = (2.82*10**4)*0.72
'每周生产需求(eqC)：'+str((2.82*10**4)*0.72)
trans = pd.read_csv('trans.csv')
def replace0(x):
    if x == 0:
        return np.nan
    else:
        return x

notnull = 8*240 - trans.isnull().sum().sum()
avg_los = (trans.sum().values[1:].sum()/notnull)*0.01

'有运输时的平均货损率：'+str(avg_los)

selected_pred = pred_result.loc[selected['n_id'].values]
selected_pred
month_sup_50 = selected_pred.sum(axis=0)
month_sup_50.sum()
need_array_eqC = np.array([avg_need_eqC for i in range(24)])
plt.figure(figsize=[30,4])
plt.xticks(range(1,25))
```

```
plt.bar(x=range(1,25),                      height=month_sup_50,
label='month_sup_50', color='darkred')
plt.bar(x=range(1,25),                    height=need_array_eqC,
label='need_array_eqC', color='darkgray',alpha=0.7)

plt.legend()
plt.title('50 Need & Supply Barplot')
plt.savefig('50 Need & Supply Barplot.jpg')
selected_pred.sort_index(inplace=True)
selected_pred.to_csv('for_matlab.csv')
import numpy as np
from scipy import optimize as op
selected_pred = selected_pred.T

selected_pred
len(selected_pred)
selected_pred[3].sum()
for i in range(len(selected_pred)):
    if i<=22:
        selected_pred.iloc[i+1]   =   selected_pred.iloc[i+1]   +
selected_pred.iloc[i]
selected_pred
np.array(-selected_pred).shape
import cvxpy as cp

c = np.array([1 for i in range(50)])

a = np.array(selected_pred)

b = np.array([avg_need_eqC*i for i in range(1,25)])

x = cp.Variable(50,integer=True)

obj = cp.Minimize(c*x)

cons =[a*x>=b, x>=0, x<=1]

prob =cp.Problem(obj, cons)

prob.solve( solver='GLPK_MI', verbose=True)

print('opt_obj', prob.value)
print('opt_x', x.value)
```

```
selected_pred = selected_pred.T
selected_pred['opt_res'] = x.value
selected_pred
opt_res1_df = pd.DataFrame()
opt_res1_df['opt'] = x.value
opt_res1_df['id'] = selected_pred.index
opt_res1_df.to_csv('opt_res1_df.csv')
opt_res1_df
order_eqC = pd.read_csv('supply_eqC.csv')


order_eqC['id'] = [int(x[1:]) for x in order_eqC['供应商
ID'].values]
selected_eqC = order_eqC.set_index('id').loc[opt_res1_df.id]
selected_eqC
selected_eqC = selected_eqC['材料分类']
selected_eqC
recat_selected                                                    =
pd.concat([selected_eqC,pred_result.loc[opt_res1_df.id]],
axis=1)
idx = recat_selected.index

recat_selected = recat_selected.set_index('材料分类')
recat_selected.loc['A'] = recat_selected.loc['A']/1.2
recat_selected.loc['B'] = recat_selected.loc['B']/1.091

recat_selected['id'] = idx
recat_selected
# recat_selected.to_csv('recat_selected_pred.csv')
n_selected = recat_selected.copy()
n_selected.loc['A'] = n_selected.loc['A']/0.6
n_selected.loc['B'] = n_selected.loc['B']/0.66
n_selected.loc['C'] = n_selected.loc['C']/0.72
n_selected['id'] = idx
n_selected['mat'] =n_selected.index
n_selected

n_selected['selected2'] = x.value
nn_selected = n_selected.set_index('selected2').loc[1]
nn_selected = nn_selected.set_index('mat')
nn_selected.sort_index(inplace=True)
nn_selected
nn_selected.to_csv('2_plan_data.csv')
a_temp = np.array(nn_selected.drop('id', axis=1)).T
```

```
a_temp.shape
len((list(a_temp[0,:]) + [0 for i in range(37*(23-0))]))
a_temp
aa_temp=[[0 for i in range(888)] for i in range(24)]
len(aa_temp[0])

for i in range(24):
    aa_temp[i] = (list(a_temp[i,:]) + [0 for i in range(37*(23-
i))])
    aa_temp[i] = ([0 for i in range(37*(23-(23-i)))] + aa_temp[i])
len(aa_temp[0])
aaa_array = np.array(aa_temp)
aaa_array.shape
c_temp             =             np.array(nn_selected.drop('id',
axis=1).loc['A']).flatten()*0.6*1.2
c_temp.shape
c_temp   =   np.append(c_temp,   np.array(nn_selected.drop('id',
axis=1).loc['B']).flatten()*0.66*1.1)
c_temp.shape
c_temp   =   np.append(c_temp,   np.array(nn_selected.drop('id',
axis=1).loc['C']).flatten()*0.72)
c_temp.shape
b = np.array([avg_need_eqC/0.72*1.01 for i in range(24)])
b
x_df = pd.DataFrame(aaa_array)
x_df.to_csv('a.csv')
x_df = pd.DataFrame(b)
x_df.to_csv('b.csv')
x_df = pd.DataFrame(c_temp)
x_df.to_csv('c.csv')
# c = c_temp

# a = aaa_array

# b = np.array([avg_need_eqC*1.01 for i in range(24)])

# x = cp.Variable(336,integer=True)

# obj = cp.Minimize(c*x)

# cons =[a*x>=b, 1>=x, x>=0]

# prob =cp.Problem(obj, cons)
```

```
# prob.solve( solver='GLPK_MI', verbose=True)

# print('opt_obj', prob.value)
# print('opt_x', x.value)
# x_df = pd.DataFrame(c)
# x_df.to_csv('for_matlab.csv')

import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns
res = pd.read_csv('2_oder_plan.csv')
res = np.array(res)
res.resize(24,37)
res = res.T
res = pd.DataFrame(res)
res
data = pd.read_csv('2_plan_data.csv')
data
res['id'] = data['id']
res.set_index('id', inplace=True)
res
sup14 = np.array(data.drop(['id','mat'], axis=1))
sup14.shape
supply_14 = res*sup14
supply_14.to_csv('supply14.csv')
supply_14
plot_sup = np.array(supply_14.sum(axis=0))


avg_need_eqC = (2.82*10**4)
'每周生产需求(eqC): '+str((2.82*10**4))
need_array_eqC = np.array([avg_need_eqC for i in range(24)])

# plt.figure(figsize=[30,4])
# plt.xticks(range(1,25))

#         plt.bar(x=range(1,25),         height=need_array_eqC,
label='need_eqC', color='darkgray')
#  plt.bar(x=range(1,25),  height=plot_sup,  label='supply_eqC',
color='darkred', alpha=0.6)

# plt.legend()
# plt.title('sup14')
```

```
# plt.savefig('sup14.jpg')
need_cum_eqC = need_array_eqC.cumsum()
sup_cum_eqC = plot_sup.cumsum()

plt.figure(figsize=[30,4],dpi=200)
plt.xticks(range(1,25))

#          sns.lineplot(x=range(1,25),          y=need_cum_eqC,
label='need_cum_eqC', color='darkred')
plt.bar(x=range(1,25),                        height=sup_cum_eqC-
need_cum_eqC+2*avg_need_eqC,                label='sup_cum_eqC',
color='darkred')
plt.legend()
# plt.title('Need & Supply Barplot(cumulative)')
# plt.savefig('Need & Supply Barplot(cumulative).jpg')
filler = pd.DataFrame(np.zeros([402,24]))
filler.set_index(np.arange(1,403))
for col in filler.columns:
    filler[col] = [np.nan for i in range(402)]
filler

filler.loc[supply_14.index] = supply_14
filler.replace(0, np.nan,inplace = True)
filler.isnull().sum()


filler.to_csv('A_0.csv')

supply_14['mat'] = data['mat'].values
supply_14
calc_storage = supply_14.groupby('mat').agg('sum')
sup_eqP          =          calc_storage.loc['A']/0.6          +
calc_storage.loc['B']/0.66 +calc_storage.loc['C']/0.72
sup_eqP
calc_price = supply_14.groupby('mat').agg('sum')
calc_price

calc_price = calc_price.sum(axis=1)
calc_price
opt_price = calc_price[0]*1.2 + calc_price[1]*1.1 + calc_price[2]
'预计未来订货费用：'+str(opt_price)
supply = pd.read_csv('supplyer.csv')
hist_price    =    np.array(supply.groupby('   材   料   分   类
').agg('sum').sum(axis=1))
```

```
hist_price
hist_price = hist_price[0]*1.2 +hist_price[1]*1.1+hist_price[2]
hist_price
hist_calc = np.array(supply.groupby('材料分类').agg('sum'))
hist_calc.shape

hist_calc = np.array(supply.groupby('材料分类').agg('sum'))
hist_calc[0,:] = hist_calc[0,:]/0.6
hist_calc[1,:] = hist_calc[1,:]/0.66
hist_calc[1,:] = hist_calc[1,:]/0.72
hist_calc.shape
hist_calc = hist_calc.sum(axis=0)

hist_calc.shape
hist_calc = np.array(hist_calc.reshape(10,24))
hist_calc = [hist_calc[:,i].mean() for i in range(24)]
hist_calc = np.array(hist_calc)
plt.style.use('ggplot')
plt.figure(dpi=200)
hist_plot_cum = hist_calc.cumsum()
sup_cum_eqP = sup_eqP.cumsum()
need_array_eqC = np.array([avg_need_eqC for i in range(24)])

plt.figure(figsize=[20,4])
plt.xticks(range(1,25))

#          sns.lineplot(x=range(1,25),          y=need_cum_eqC,
label='need_cum_eqC', color='darkred')
plt.bar(x=range(1,25),                    height=hist_plot_cum-
need_cum_eqC+2*avg_need_eqC, label='历史库存均值')
plt.bar(x=range(1,25),                      height=sup_cum_eqP-
need_cum_eqC+2*avg_need_eqC, label='37 家供应商计划订货库存', )
sns.lineplot(x=range(1,25), y=2*avg_need_eqC, label='安全库存线
（2 周产量）', color = 'red')
plt.rcParams['font.family'] = ['sans-serif']
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.title('选定 37 家供应商后未来 24 周的库存与历史库存均值',
fontsize = 20)

plt.legend(fontsize = 15)

plt.savefig('选定 37 家供应商后未来 24 周的库存与历史库存均值.jpg')
```

4.解问题三 Python 代码

```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns
avg_need_eqC = (2.82*10**4)
'每周生产需求(eqC)：'+str((2.82*10**4))
tsa = pd.read_csv('TSA_decided.csv')
tsa['id'] = range(1,403)
tsa.drop('Unnamed: 0', axis=1, inplace=True)
tsa
supply = pd.read_csv('supplyer.csv')
tsa['mat'] = supply['材料分类']
tsa
tsa.set_index('mat', inplace=True)
tsa.sort_index(inplace=True)
idx = tsa.id
tsa
tsa.loc['A'] = tsa.loc['A']/1.2/0.6
tsa.loc['B'] = tsa.loc['B']/1.091/0.66
tsa.loc['C'] = tsa.loc['C']/0.72
tsa['id'] = idx
tsa
a_temp = np.array(tsa.drop('id', axis=1)).T
a_temp.shape
aa_temp=[[0 for i in range(9648)] for i in range(24)]


for i in range(24):
    aa_temp[i] = (list(a_temp[i,:]) + [0 for i in range(402*(23-
i))])
    aa_temp[i]  =  ([0  for  i  in  range(402*(23-(23-i)))]  +
aa_temp[i])
len(aa_temp[0])
aaa_array = np.array(aa_temp)
aaa_array.shape
c_temp                 =                np.array(tsa.drop('id',
axis=1).loc['A']).flatten()*0.6*95.4/0.4
c_temp.shape
c_temp     =     np.append(c_temp,     np.array(tsa.drop('id',
axis=1).loc['B']).flatten()*0.66*90.6/0.4)
c_temp.shape
```

```
c_temp = np.append(c_temp, np.array(tsa.drop('id',
axis=1).loc['C']).flatten()*0.72*85.8/0.4)
c_temp.shape
b = np.array([avg_need_eqC*1.01 for i in range(24)])
x_df = pd.DataFrame(aaa_array)
x_df.to_csv('a1.csv')
x_df = pd.DataFrame(b)
x_df.to_csv('b1.csv')
x_df = pd.DataFrame(c_temp)
x_df.to_csv('c1.csv')
res = pd.read_csv('3_oder_plan.csv')
res = np.array(res)
res = res.reshape(24,402)
res = res.T
res = pd.DataFrame(res)
res
res['id'] = tsa['id'].values
res.set_index('id', inplace=True)
res
temp = np.array(tsa.drop('id', axis=1))
sup402 = res*temp
sup402['mat'] = tsa.index
sup402
sup_save = sup402.copy()
sup_save.sort_index(inplace=True)
sup_save.replace(0,np.nan,inplace=True)
sup_save.to_csv('A_1.csv')
agg = sup402.groupby('mat').agg('sum')
agg = agg.sum(axis=1)
agg
agg1 = tsa.groupby('mat').agg('sum')
agg1 = agg1.sum(axis=1)
agg1
opt_price = agg[0]*(57.6+9.8)+agg[1]*(52.8+9.8)+agg[2]*(48+9.8)
'单价和运费',opt_price
eqP = sup402.set_index('mat')
eqP.loc['A'] = eqP.loc['A']/0.6
eqP.loc['B'] = eqP.loc['B']/0.66
eqP.loc['C'] = eqP.loc['C']/0.72
eqP = eqP.sum(axis=0)
eqP
supply = pd.read_csv('supplyer.csv')
hist_price = np.array(supply.groupby('材料分类
').agg('sum').sum(axis=1))
```

```
hist_price
hist_price              =              hist_price[0]*(57.6+9.8)
+hist_price[1]*(52.8+9.8)+hist_price[2]*(48+9.8)
hist_price = hist_price*0.1
hist_price
hist_calc = np.array(supply.groupby('材料分类').agg('sum'))
hist_calc[0,:] = hist_calc[0,:]/0.6
hist_calc[1,:] = hist_calc[1,:]/0.66
hist_calc[1,:] = hist_calc[1,:]/0.72
hist_calc.shape
hist_calc = hist_calc.sum(axis=0)

hist_calc.shape
hist_calc = np.array(hist_calc.reshape(10,24))
hist_calc = [hist_calc[:,i].mean() for i in range(24)]
hist_calc = np.array(hist_calc)
plt.style.use('ggplot')

hist_plot_cum = hist_calc.cumsum()
sup_cum_eqP = eqP.cumsum()
need_array_eqC = np.array([avg_need_eqC for i in range(24)])
need_cum_eqC = need_array_eqC.cumsum()

plt.figure(figsize=[20,4],dpi=200)
plt.xticks(range(1,25))
plt.rcParams['font.family'] = ['sans-serif']
plt.rcParams['font.sans-serif'] = ['SimHei']
#          sns.lineplot(x=range(1,25),          y=need_cum_eqC,
label='need_cum_eqC', color='darkred')
plt.bar(x=range(1,25),                    height=hist_plot_cum-
need_cum_eqC+2*avg_need_eqC, label='历史库存均值')
plt.bar(x=range(1,25),                      height=sup_cum_eqP-
need_cum_eqC+2*avg_need_eqC, label='订货计划库存')
#          plt.bar(x=range(1,25),          height=hist_plot_cum,
label='hist_plot', color='darkgray')
#           plt.bar(x=range(1,25),           height=sup_cum_eqP,
label='sup_cum_eqC', color='darkred')
sns.lineplot(x=range(1,25), y=2*avg_need_eqC, label='安全库存线
（2 周产量）', color='red')



plt.legend(fontsize=15)
plt.title('成本最低订货计划未来 24 周的库存与历史库存均值
```

```
',fontsize=20)
plt.savefig('成本最低订货计划未来 24 周的库存与历史库存均值.jpg')
opt_storage = np.array(sup_cum_eqP-need_cum_eqC+2*avg_need_eqC)
hist_storage              =              np.array(hist_plot_cum-
need_cum_eqC+2*avg_need_eqC)
hist_storage_price = 0
opt_storage_price = 0
for i in range(len(opt_storage)):
    if opt_storage[i]>0:
        opt_storage_price+=opt_storage[i]*28
    if hist_storage[i]>0:
        hist_storage_price+=hist_storage[i]*28
print('优化总价：',opt_storage_price+opt_price)
print('历史均价：',hist_storage_price+hist_price)

opt_storage_price
hist_storage_price
sup_cum_eqP
res = res.sort_index()
res
temp = np.array(tsa.drop('id', axis=1))
sup402 = res*temp
sup402
order_pred = pred*np.array(res)
smaller_than_sup = order_pred<sup402
smaller_than_sup
order_pred.shape
for col in smaller_than_sup.columns:
    array =  np.array(smaller_than_sup[col])
    for i in range(len(array)):
        if array[i]==True:
            order_pred[i,col] = np.array(sup402)[i,col]
order_pred
a_1 = order_pred*res
a_1.to_csv('A_1x.csv')
```

5.解问题四 Python 代码

```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns
```

```
sup = pd.read_csv('supplyer.csv')
sup
tsa = pd.read_csv('TSA_decided.csv')
tsa = tsa.drop('Unnamed: 0',axis=1)
tsa['mat'] = sup['材料分类']
tsa.set_index('mat',inplace=True)
tsa
tsa.loc['A'] = tsa.loc['A']/0.6
tsa.loc['B'] = tsa.loc['B']/0.66
tsa.loc['C'] = tsa.loc['C']/0.72

avg_sup = tsa.agg('sum')

avg_sup = np.array(avg_sup)
avg_sup
for i in range(23):
    avg_sup[i+1] = avg_sup[i+1] +avg_sup[i]

avg_sup

pd.DataFrame(avg_sup).to_csv('bs.csv')
a_s = [i*1.5-2 for i in range(1,25)]
a_s[23] = 23*1.5
pd.DataFrame(a_s).to_csv('as.csv')

avg_need_eqC = (2.82*10**4)
'每周生产需求(eqC): '+str((2.82*10**4))
avg_sup = tsa.agg('sum')

avg_sup
# hist_plot_cum = hist_calc.cumsum()
sup_cum_eqP = avg_sup.cumsum()
need_array_eqC = np.array([1.2*avg_need_eqC for i in range(24)])
need_cum_eqC = need_array_eqC.cumsum()

plt.figure(figsize=[30,4],dpi=200)
plt.xticks(range(1,25))

#          sns.lineplot(x=range(1,25),          y=need_cum_eqC,
label='need_cum_eqC', color='darkred')
#          plt.bar(x=range(1,25),          height=hist_plot_cum-
need_cum_eqC+2*avg_need_eqC,                label='hist_plot',
color='darkgray')
plt.bar(x=range(1,25),                      height=sup_cum_eqP-
```

```
need_cum_eqC+2*avg_need_eqC,                label='sup_cum_eqC',
color='darkred')
#          plt.bar(x=range(1,25),          height=hist_plot_cum,
label='hist_plot', color='darkgray')
#           plt.bar(x=range(1,25),           height=sup_cum_eqP,
label='sup_cum_eqC', color='darkred')
sns.lineplot(x=range(1,25),                    y=2*avg_need_eqC,
label='sup_cum_eqC', color='darkred')



plt.legend()
# plt.title('Need & Supply Barplot(cumulative)')
# plt.savefig('Need & Supply Barplot(cumulative).jpg')
supply = pd.read_csv('supplyer.csv')
tsa['mat'] = supply['材料分类'].values
tsa['id'] = np.arange(1,403)
tsa.set_index('mat', inplace=True)
tsa.sort_index(inplace=True)
idx = tsa.id
tsa
tsa.loc['A'] = tsa.loc['A']/1.2/0.6
tsa.loc['B'] = tsa.loc['B']/1.091/0.66
tsa.loc['C'] = tsa.loc['C']/0.72
tsa['id'] = idx
tsa
a_temp = np.array(tsa.drop('id', axis=1)).T
a_temp.shape
aa_temp=[[0 for i in range(9648)] for i in range(24)]
for i in range(24):
    aa_temp[i] = (list(a_temp[i,:]) + [0 for i in range(402*(23-
i))])
    aa_temp[i]  =  ([0  for  i  in  range(402*(23-(23-i)))]  +
aa_temp[i])
len(aa_temp[0])
aa_temp = np.array(aa_temp)
for i in range(23):
    aa_temp[i+1,:] = aa_temp[i,:]+aa_temp[i+1,:]
pd.DataFrame(aa_temp)
c_temp                  =                np.array(tsa.drop('id',
axis=1).loc['A']).flatten()*0.6*232.7
c_temp.shape
c_temp     =     np.append(c_temp,     np.array(tsa.drop('id',
axis=1).loc['B']).flatten()*0.66*221)
```

```
c_temp.shape
c_temp     =     np.append(c_temp,     np.array(tsa.drop('id',
axis=1).loc['C']).flatten()*0.72*209.3)
c_temp.shape
62.3-1500
ap = np.array([i-1 for i in range(24)]).reshape(24,1)
aa_temp.shape
aa_temp = np.hstack((aa_temp,ap))
x_df = pd.DataFrame(aa_temp)
x_df.to_csv('a3.csv')
x_df = pd.DataFrame(b)
x_df.to_csv('b3.csv')
x_df = pd.DataFrame(c_temp)
x_df.to_csv('c3.csv')

ratio = pd.read_csv('gap_ratio.csv')
ratio = np.array(ratio.drop('Unnamed: 0', axis=1))
a_2 = pd.read_csv('TSA_decided.csv')
a_2 = a_2.drop('Unnamed: 0', axis=1)

i=0
for col in a_2.columns:
    a_2[col] = np.array(a_2[col]*(1-ratio[i]))
    i+=1
a_2.to_csv('A_2.csv', index=False)
```

6.0-1 背包问题 Python 代码

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Sat Sep 11 17:16:52 2021

@author: xieyifan
"""
import pandas as pd
import numpy as np
import csv


def bag(n, c, w, v):
```

```
    #初始状态

    value = [[0 for j in range(c + 1)] for i in range(n + 1)]
    for i in range(1, n + 1):
        for j in range(1, c + 1):
            value[i][j] = value[i - 1][j]
            # 背包放当前物体，遍历前一状态
            if j >= w[i - 1] and value[i][j] < value[i - 1][j -
w[i - 1]] + v[i - 1]:
                value[i][j] = value[i - 1][j - w[i - 1]] + v[i -
1]
    return value
def show(n, c, w, value):
  #  print('最大价值为:', value[n][c])
    global x
    x=[]
    x = [False for i in range(n)]
    j = c
    for i in range(n, 0, -1):
        if value[i][j] > value[i - 1][j]:
            x[i - 1] = True
            j -= w[i - 1]
    print('hui=:',hui,'lun=:',lun)

    for i in range(n):
        if x[i]:
            flag1[lun].append(i)
            #print('第', i+1, '个,', end='')
#_____________________________________________________________
__


data    =    pd.read_csv('/Users/zhangke/Desktop/D    原    始    数
据.csv',header=None,nrows=804)
rowsB=[]
for hui in range(24):#################注意这里改成 24
    list1=data[hui].values.tolist()#csv 文件转成 list
    list25=data[24].values.tolist()#每一个供货商的货物 A,B,C 分类

#采用第一个转运商的供货商的数字编号，里面只有少数几个数字
    flag11=[]
    flag2=[]
    flag3=[]
    flag4=[]
```

```
    flag5=[]
    flag6=[]
    flag7=[]
    flag8=[]
    flag1=[flag2,flag3,flag4,flag5,flag6,flag7,flag8,flag11]

    list1A=[]#写入新 csv 文件第一个转运商的内容
    list1B=[]
    list1C=[]
    list1D=[]
    list1E=[]
    list1F=[]
    list1G=[]
    list1H=[]


    for i in range(402):
        list1A.append('')#这个写入新 csv 的值的列表要有 804 格
        list1B.append('')
        list1C.append('')
        list1D.append('')
        list1E.append('')
        list1F.append('')
        list1G.append('')
        list1H.append('')
#n=0
#for ss in range(len(list1)):
#    if list1[ss]!=0:
#        n=n+1
#print(n)

    n = 804#总的物品数量
    c = 6000#包的容量
    for lun in range(8):#8 运营商循环##################这里得把 1
换成 8
        w = []#每个物品的重量
        v = []#物品的价值，A\C 为 0.72,B 为 0.726

        for i in range(804):
            if i<402 and list1[i]>6000:
                list1[i+402]=list1[i]-6000#多的部分放到+402 的一个
数字上
                if list1[i+402]>6000:
                    print('有问题')
```

```
                list1[i]=6000#它自己保留 6000 的部分

            if list1[i]==0:
                w.append(100000)#如果供货量是 0，就给他一个很大的重
量，就放不进包里
            else:
                if (float(list1[i])-int(list1[i]))>=0.5:
                    w.append(int(list1[i]+1))
                else:
                    w.append(int(list1[i]))#否则它的重量就是自己的
值
            if  list25[i]=='A':#如果货物是 A，B 中价值就是 1/0.72，C
中为 1，D 中为 1/139.6
                v.append(1/139.6)
            if list25[i]=='B':
                v.append(1/145.9)#如果货物 B，B 中价值是 1/0.726，C
中为 1，D 中为 145.9
            if  list25[i]=='C':#如果货物是 C，B 中价值就是 1/0.72，C
中为 1，D 中为 150.7
                v.append(1/150.7)

            if i in flag1[0] or i in flag1[1] or i in flag1[2] or
i in flag1[3] or i in flag1[4] or i in flag1[5] or i in flag1[6]:
                w[i]=100000#如果已经选过了就不放进包里

        value = bag(n, c, w, v)#运行 01 背包
        show(n, c, w, value)#运行 01 背包


        for i in range(804):
            if i<402:
                if x[i]==True:
                    if lun==0:
                        list1C[i]=list1[i]#B 按照 32681475 的顺序写
入新 csv 的列表的值
                    if lun==1:
                        list1B[i]=list1[i]#C 按照 36284175
                    if lun==2:
                        list1F[i]=list1[i]
                    if lun==3:
                        list1H[i]=list1[i]
                    if lun==4:
                        list1A[i]=list1[i]
                    if lun==5:
```

```
                    list1D[i]=list1[i]
                if lun==6:
                    list1G[i]=list1[i]
                if lun==7:
                    list1E[i]=list1[i]

            else:
                if x[i]==True:
                    if lun==0:
                        list1C[i-402]=list1[i]#写入新csv的列表
的值
                    if lun==1:
                        list1B[i-402]=list1[i]
                    if lun==2:
                        list1F[i-402]=list1[i]
                    if lun==3:
                        list1H[i-402]=list1[i]
                    if lun==4:
                        list1A[i-402]=list1[i]
                    if lun==5:
                        list1D[i-402]=list1[i]
                    if lun==6:
                        list1G[i-402]=list1[i]
                    if lun==7:
                        list1E[i-402]=list1[i]
    if hui==0:
        list1A1 =[[],[],[],[],[],[],[],[]]
        list1A1[0:7]                                                =
[list1A,list1B,list1C,list1D,list1E,list1F,list1G,list1H]# 第一
周的8个转运商列
    if hui==1:
        list1A2 =[[],[],[],[],[],[],[],[]]
        list1A2[0:7]                                                =
[list1A,list1B,list1C,list1D,list1E,list1F,list1G,list1H]
    if hui==2:
        list1A3 =[[],[],[],[],[],[],[],[]]
        list1A3[0:7]                                                =
[list1A,list1B,list1C,list1D,list1E,list1F,list1G,list1H]
    if hui==3:
        list1A4 =[[],[],[],[],[],[],[],[]]
        list1A4[0:7]                                                =
[list1A,list1B,list1C,list1D,list1E,list1F,list1G,list1H]
    if hui==4:
        list1A5 =[[],[],[],[],[],[],[],[]]
```

```
        list1A5[0:7]                                                =
[list1A,list1B,list1C,list1D,list1E,list1F,list1G,list1H]
    if hui==5:
        list1A6 =[[],[],[],[],[],[],[],[]]
        list1A6[0:7]                                                =
[list1A,list1B,list1C,list1D,list1E,list1F,list1G,list1H]
    if hui==6:
        list1A7 =[[],[],[],[],[],[],[],[]]
        list1A7[0:7]                                                =
[list1A,list1B,list1C,list1D,list1E,list1F,list1G,list1H]
    if hui==7:
        list1A8 =[[],[],[],[],[],[],[],[]]
        list1A8[0:7]                                                =
[list1A,list1B,list1C,list1D,list1E,list1F,list1G,list1H]
    if hui==8:
        list1A9 =[[],[],[],[],[],[],[],[]]
        list1A9[0:7]                                                =
[list1A,list1B,list1C,list1D,list1E,list1F,list1G,list1H]
    if hui==9:
        list1A10 =[[],[],[],[],[],[],[],[]]
        list1A10[0:7]                                               =
[list1A,list1B,list1C,list1D,list1E,list1F,list1G,list1H]
    if hui==10:
        list1A11 =[[],[],[],[],[],[],[],[]]
        list1A11[0:7]                                               =
[list1A,list1B,list1C,list1D,list1E,list1F,list1G,list1H]
    if hui==11:
        list1A12 =[[],[],[],[],[],[],[],[]]
        list1A12[0:7]                                               =
[list1A,list1B,list1C,list1D,list1E,list1F,list1G,list1H]
    if hui==12:
        list1A13 =[[],[],[],[],[],[],[],[]]
        list1A13[0:7]                                               =
[list1A,list1B,list1C,list1D,list1E,list1F,list1G,list1H]
    if hui==13:
        list1A14 =[[],[],[],[],[],[],[],[]]
        list1A14[0:7]                                               =
[list1A,list1B,list1C,list1D,list1E,list1F,list1G,list1H]
    if hui==14:
        list1A15 =[[],[],[],[],[],[],[],[]]
        list1A15[0:7]                                               =
[list1A,list1B,list1C,list1D,list1E,list1F,list1G,list1H]
    if hui==15:
        list1A16 =[[],[],[],[],[],[],[],[]]
```

```
        list1A16[0:7]                                                   =
[list1A,list1B,list1C,list1D,list1E,list1F,list1G,list1H]
    if hui==16:
        list1A17 =[[],[],[],[],[],[],[],[]]
        list1A17[0:7]                                                   =
[list1A,list1B,list1C,list1D,list1E,list1F,list1G,list1H]
    if hui==17:
        list1A18 =[[],[],[],[],[],[],[],[]]
        list1A18[0:7]                                                   =
[list1A,list1B,list1C,list1D,list1E,list1F,list1G,list1H]
    if hui==18:
        list1A19 =[[],[],[],[],[],[],[],[]]
        list1A19[0:7]                                                   =
[list1A,list1B,list1C,list1D,list1E,list1F,list1G,list1H]
    if hui==19:
        list1A20 =[[],[],[],[],[],[],[],[]]
        list1A20[0:7]                                                   =
[list1A,list1B,list1C,list1D,list1E,list1F,list1G,list1H]
    if hui==20:
        list1A21 =[[],[],[],[],[],[],[],[]]
        list1A21[0:7]                                                   =
[list1A,list1B,list1C,list1D,list1E,list1F,list1G,list1H]
    if hui==21:
        list1A22 =[[],[],[],[],[],[],[],[]]
        list1A22[0:7]                                                   =
[list1A,list1B,list1C,list1D,list1E,list1F,list1G,list1H]
    if hui==22:
        list1A23 =[[],[],[],[],[],[],[],[]]
        list1A23[0:7]                                                   =
[list1A,list1B,list1C,list1D,list1E,list1F,list1G,list1H]
    if hui==23:
        list1A24 =[[],[],[],[],[],[],[],[]]
        list1A24[0:7]                                                   =
[list1A,list1B,list1C,list1D,list1E,list1F,list1G,list1H]


rowsZ                                                                   =
zip(list1A1[0],list1A1[1],list1A1[2],list1A1[3],list1A1[4],lis
t1A1[5],list1A1[6],list1A1[7],
list1A2[0],list1A2[1],list1A2[2],list1A2[3],list1A2[4],list1A2
[5],list1A2[6],list1A2[7],
list1A3[0],list1A3[1],list1A3[2],list1A3[3],list1A3[4],list1A3
[5],list1A3[6],list1A3[7],
list1A4[0],list1A4[1],list1A4[2],list1A4[3],list1A4[4],list1A4
```

```
[5],list1A4[6],list1A4[7],
list1A5[0],list1A5[1],list1A5[2],list1A5[3],list1A5[4],list1A5
[5],list1A5[6],list1A5[7],
list1A6[0],list1A6[1],list1A6[2],list1A6[3],list1A6[4],list1A6
[5],list1A6[6],list1A6[7],
list1A7[0],list1A7[1],list1A7[2],list1A7[3],list1A7[4],list1A7
[5],list1A7[6],list1A7[7],
list1A8[0],list1A8[1],list1A8[2],list1A8[3],list1A8[4],list1A8
[5],list1A8[6],list1A8[7],
list1A9[0],list1A9[1],list1A9[2],list1A9[3],list1A9[4],list1A9
[5],list1A9[6],list1A9[7],
list1A10[0],list1A10[1],list1A10[2],list1A10[3],list1A10[4],li
st1A10[5],list1A10[6],list1A10[7],
list1A11[0],list1A11[1],list1A11[2],list1A11[3],list1A11[4],li
st1A11[5],list1A11[6],list1A11[7],
list1A12[0],list1A12[1],list1A12[2],list1A12[3],list1A12[4],li
st1A12[5],list1A12[6],list1A12[7],
list1A13[0],list1A13[1],list1A13[2],list1A13[3],list1A13[4],li
st1A13[5],list1A13[6],list1A13[7],
list1A14[0],list1A14[1],list1A14[2],list1A14[3],list1A14[4],li
st1A14[5],list1A14[6],list1A14[7],
list1A15[0],list1A15[1],list1A15[2],list1A15[3],list1A15[4],li
st1A15[5],list1A15[6],list1A15[7],
list1A16[0],list1A16[1],list1A16[2],list1A16[3],list1A16[4],li
st1A16[5],list1A16[6],list1A16[7],
list1A17[0],list1A17[1],list1A17[2],list1A17[3],list1A17[4],li
st1A17[5],list1A17[6],list1A17[7],
list1A18[0],list1A18[1],list1A18[2],list1A18[3],list1A18[4],li
st1A18[5],list1A18[6],list1A18[7],
list1A19[0],list1A19[1],list1A19[2],list1A19[3],list1A19[4],li
st1A19[5],list1A19[6],list1A19[7],
list1A20[0],list1A20[1],list1A20[2],list1A20[3],list1A20[4],li
st1A20[5],list1A20[6],list1A20[7],
list1A21[0],list1A21[1],list1A21[2],list1A21[3],list1A21[4],li
st1A21[5],list1A21[6],list1A21[7],
list1A22[0],list1A22[1],list1A22[2],list1A22[3],list1A22[4],li
st1A22[5],list1A22[6],list1A22[7],
list1A23[0],list1A23[1],list1A23[2],list1A23[3],list1A23[4],li
st1A23[5],list1A23[6],list1A23[7],
list1A24[0],list1A24[1],list1A24[2],list1A24[3],list1A24[4],li
st1A24[5],list1A24[6],list1A24[7]
)

with open('/Users/zhangke/Desktop/D 的答案.csv', "w") as f:#写入
```

的过程

```
#rowsA                                                                    =
zip(list1A,list1B,list1C,list1D,list1E,list1F,list1G,list1H)#第
一周的 8 个转运商列
    writer = csv.writer(f)
    for row in rowsZ:
        writer.writerow(row)
```