arXiv:2302.00711v1 [math.OC] 1 Feb 2023

生成用于数值实验的线性、半正定和二阶锥优化问题

Mohammadhossein Mohammadisiahroudi[a]、Ramin Fakhimi[a]、
BrandonAugustino[a] 和 Tam´as Terlaky[a]
[a]I利哈伊大学工业与系统工程系，美国宾夕法尼亚州伯利恒 A

文章历史
编译于2023年2月3日

摘要
算法的数值性能可以通过测试集或生成此类问题的程序来研究。本文提出了多种生成线性、半定和二阶锥优化问题的方法。具体而言，我们关注需要已知最优解、已知最优划分、特定内点解或同时具备上述条件的问题实例。在所提出的问题生成器中，可以选择优化问题的不同特征，包括维度、规模、条件数、退化、最优划分和稀疏性，以便开展全面的计算实验。我们还开发了生成具有最大互补最优解且具有预定最优划分的实例的程序，用于生成具有挑战性的半定和二阶锥优化问题。生成的实例使我们能够评估用于锥优化问题的高效内点法。

关键词
问题生成器；锥优化；线性优化；半定优化；二阶锥优化

1. 引言

优化只是众多领域之一，在这些领域中，算法的实证分析高度依赖所提供测试实例的质量。学者依据这些测试问题评估算法的优缺点，而这些问题在可测量的特征或属性方面必须具有无偏性、代表性和多样性。然而，许多基准测试问题并不具备这些理想品质，因为它们往往基于有限的一组现实问题，或沿用自早期研究，而这些问题如今可能已经过时 [2]。

另一种用于优化实验的方法是使用随机测试问题生成器。虽然其设计必须经过仔细考量，但简单随机生成方法的一个优势是能够生成具有可预测特征的问题。因此，科学家们提倡使用高度参数化的生成器，为实验生成受控程度适当的数据[11]。作为这一领域最早的尝试之一，Todd [24]研究了随机生成的可行多面体的性质。Pilcher 和 Rardin [17] 提出了一种

通讯作者：M. M.，mom219@利哈伊大学.edu

用于纯整数优化问题的生成器，其方法是引入随机割，并以已知的部分多面体为基础。然而，这种方法仅适用于旅行商问题，且未明确考虑松弛问题的解或结构特征。由于无法改变所关注的特征，这些生成器在实验中的应用范围仅限于特定的问题领域。

有时，要以能够适当改变所关注性质的方式开发实例生成器，可能颇具挑战。某些特征（如图的密度）通常可以在生成过程中直接控制，但其他属性则更难预先定义或明确控制。同一问题实例的许多可测量特征可能高度相关，这可能是由于相互作用的约束所致，也可能仅仅是随机生成过程的结果。通过迭代局部搜索，可以获得特征组合不那么常见的实例：该方法逐步修改实例，直至其具备所需性质。尽管这类实例空间搜索技术的计算开销高于参数化生成器，但它们为生成具有特定目标特征的实例提供了可靠的方法 [2]。

这类应用中最常见的搜索技术是进化算法。Chakraborty 和 Choudhury [5] 以及 Cotta 和 Moscato [8] 采用这种方法，对算法性能进行统计意义上的平均情况和最坏情况分析。近来，这一方向的探索主要聚焦于拓展实例难度谱系以及所测量特征的多样性 [20]。这些技术在组合优化中的成功，引发了人们对若干问题的探讨：能否将类似方法用于线性优化（LO）和混合整数优化；能否采用更全面的搜索算法来获取难以设计的实例；以及如何构造搜索空间以实现高效性能。

为了开发具有可控性质的线性优化测试问题实例生成技术，Bowly 等人 [2] 比较了朴素随机生成器与高度参数化生成器，展示了每种方法能够有效控制哪些特征值。他们还研究了迭代搜索方法，以寻找难以设计或很少由生成器生成的实例。这些方法使从业者能够探索线性优化问题（LOPs）空间中感兴趣的区域，而此前已在这些区域发现了具有挑战性的实例。使用静态测试集或朴素随机生成方法无法做到这一点，因为它们对特征的控制有限。此外，大规模线性优化问题广泛存在于经济学、工业、物流、统计学、量子物理学及其他领域。与任何现实应用一样，目标都是高效地获得高质量的解，而这需要高性能计算系统和并行算法。因此，Sokolinsky 和 Sokolinskaya [21]认为，开发用于生成线性优化问题的新并行算法并改进现有算法十分必要。

开发用于求解大规模线性优化问题的新算法，需要在基准问题和随机问题上进行测试。有时，针对特定用途构建线性优化和整数优化实例生成器是合理的。NETGEN 生成器 [12] 及其后继 MNETGEN 可生成参数化的多商品流、运输和指派问题。因此，所用参数针对的是底层网络，而非可行集。Netlib-LP [10] 是知名的线性优化问题基准库之一。然而，在调试线性优化求解器时，通常需要生成具有特定特征的随机线性优化问题，例如稀疏性、系数矩阵的条件数或已知的最优划分。

Charnes 等人 [6] 提出了最早用于生成随机线性优化问题的方法之一

，这些问题具有已知解。该方法可以生成规模任意、数值特征范围广泛的测试问题。其主要思路如下：以一个具有已知解的线性优化问题为基础，然后随机修改该问题，使其解保持不变。这种方法的主要缺点是，预先固定最优解会显著限制生成的线性优化问题的随机性。

Arthur 和 Frendewey [1] 介绍了 GENGUB 生成器，该生成器可构造具有已知解且具备指定特征的随机线性优化问题（LOP），例如问题规模、系数矩阵的密度、紧约束不等式的数量或退化状态。GENGUB 的一项显著特点是能够引入广义上界约束，其定义为一组约束（或其子集），其中每个变量至多出现一次（即至多有一个非零系数）。这种方法与 [6] 中介绍的生成器存在相似的缺点：由于预先固定了最优解，所得 LOP 的随机性受到显著限制。

Castillo 等人 [4] 提出了一种生成随机 LOP 的方法，其解类型预先选定：有界或无界，唯一或多个。每种结构都通过随机整数向量生成，其取值范围可以作为已知条件。接下来，求得满足所需条件的目标函数，即能够导出预期类型解的目标函数。该线性优化问题生成器主要用于教学，而非测试新的线性优化算法。Okolinsky 和 Sokolinskaya [21] 提出了随机 LOP 生成器 FRaGenLP（可行线性规划随机生成器），并将其实现为适用于集群计算系统的并行程序。Calamai 等人 [3] 介绍了一种生成凸、严格凹以及不定（二次型可为双线性或非双线性）二次优化问题的新技术。

在半定优化文献中，学者们关注的是复杂问题。他们从不同方向探究如何界定半定优化问题的难度，例如，不存在严格互补解 [16], ，或不存在对偶间隙非零的解 [22]。Wei 和 Wolkowicz [25] 提出了一种生成不具备严格互补解的半定优化问题的程序。我们以这些思路为基础，开发高度参数化的生成器。

1.1. 贡献

本文回顾并提出了若干程序，用于生成具有指定最优解、内点解或同时具有这两者的随机线性优化问题、半定优化问题和二阶锥优化问题。我们还开发了半定优化问题和二阶锥优化问题生成器，可生成具有特定最大互补解的问题，从而预先确定最优划分。

生成具有特定内点解的半定优化问题和二阶锥优化问题，可确保生成的问题满足强对偶，且最优解集有界。借助预定义的内点解，研究人员能够分析优化算法（例如可行内点法）在不同初始内点解下的性能。

生成具有已知最优解的问题，可确保生成的问题具有有界最优值，并有助于分析算法与最优解特征之间的关系。这些过程将进一步增强学者检验算法的能力，使他们能够改变输入数据的不同特征，例如维度、稀疏性、条件数、解的规模（这对不可行内点法的性能起着关键作用）以及许多其他特征，此外还可改变预先设定的性质

以及最优解的性质。所提出过程的另一种可能应用是算法的平均情形复杂度分析。

本文其余部分的组织如下。第 2 节将简要回顾线性优化理论，然后介绍几种线性优化问题生成器，这些生成器能够生成具有特定最优解、特定内点解或同时具有二者的实例。随后，我们将在第 3 节和第 4 节分别为半定优化和二阶锥优化开发类似的生成器。第 5 节将讨论所提出实例生成器的实现，第 6 节对全文进行总结。

2. 线性优化

本节将简要回顾线性优化理论，然后介绍三种用于随机生成线性优化测试问题的不同算法。

2.1. 线性优化问题

在线性优化问题中，我们寻求最小化两个 n 维向量的内积

$$c^\top x = \sum_{i=1}^{n} c_i \cdot x_i,$$

对于常数向量 $c \in \mathbb{R}^n$ 和变量向量 $x \in \mathbb{R}^n$，在这一最小化问题中，变量 x 必须满足以下形式的线性约束

$$Ax = b,$$

对于给定的矩阵 $A \in \mathbb{R}^{m\times n}$ 和向量 $b \in \mathbb{R}^m$。此外，我们要求 x 逐元素非负，记作 $x \geq 0$。

因此，我们关注随机生成以下形式的线性优化问题

$$z_{LO}^P = \min_x \left\{ c^\top x : Ax = b, x \geq 0 \right\}, \tag{LOP-P}$$

并将 (LOP-P) 称为原问题。给定原问题 (LOP-P)，我们还关注一个称为 (LOP-P) 对偶问题的第二个问题，其标准形式如下：

$$z_{LO}^D = \max_{(y,s)} \left\{ b^\top y : A^\top y + s = c, s \geq 0, y \in \mathbb{R}^m \right\}, \tag{LOP-D}$$

其中 $s = c - A^\top y$ 是对偶松弛变量。

当 x 和 (y, s) 分别满足原始问题和对偶问题的约束时，我们称它们为可行解。因此，原始-对偶可行解的集合定义为

$$\mathcal{PD}_{LO} = \left\{ (x, y, s) \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^n : Ax = b, A^\top y + s = c, (x, s) \geq 0 \right\}.$$

类似地，所有可行内点解的集合为

$$\mathcal{PD}_{LO}^{0} = \{(x, y, s) \in \mathcal{PD}_{LO} : (x, s) > 0\}.$$

线性优化的一个关键性质是弱对偶性；对于任何满足 (LOP-D) 的 (y, s)，其值为 $c^\top x$ 的任意满足 (LOP-P) 的 x 提供一个下界 $b^\top y$，即:

$$b^\top y \leq c^\top x,$$

对于任意 $(x, y, s) \in \mathcal{PD}_{LO}$。反过来，任何满足 (LOP-P) 的 x 都为满足 (LOP-D) 的任意 y 的 $b^\top y$ 提供一个上界 $c^\top x$，我们将非负量 $c^\top x - b^\top y = x^\top s$ 称为对偶间隙。

只要 $(x, y, s) \in \mathcal{PD}$ 且满足 $c^\top x = b^\top y$，或者等价地满足 $x^\top s = 0$，那么 x 就是 (LOP-P) 的最优解，而 (y, s) 是 (LOP-D) 的最优解。在这种情况下，线性优化问题满足强对偶性，即如果原始问题和对偶问题都有可行解，那么二者都有目标值相等的最优解。在强对偶性成立时，所有最优解（如果存在）都属于集合 $\mathcal{PD}_{LO}^{*}$，该集合定义为

$$\mathcal{PD}_{LO}^{*} = \left\{(x, y, s) \in \mathcal{PD}_{LO} : x^\top s = 0\right\}.$$

令 $[n]$ 表示集合 $\{1, 2, \ldots, n\}$。根据 Roos 等人 [18],，线性优化问题具有最优划分 $\mathcal{N} \cup \mathcal{B} = [n]$ 和 $\mathcal{B} \cap \mathcal{N} = \emptyset$，其中

$$\mathcal{B} = \{i : \exists (x^*, y^*, s^*) \in \mathcal{PD}_{LO}^{*} \text{ with } x_i^* > 0\},$$
$$\mathcal{N} = \{i : \exists (x^*, y^*, s^*) \in \mathcal{PD}_{LO}^{*} \text{ with } s_i^* > 0\}.$$

如果 $(x^*, y^*, s^*) \in \mathcal{PD}_{LO}^{*}$，且对所有 $i \in \mathcal{B}$ 都有 $x_i^* > 0$，对所有 $i \in \mathcal{N}$ 都有 $s_i^* > 0$，那么我们有 $x^* + s^* > 0$，最优解对 (x^*, y^*, s^*) 称为严格互补。在本节中，我们用 $(\mathcal{B}, \mathcal{N})$ 表示最优划分，用 (B, N) 表示算法中的指标集划分。介绍每种算法后，我们将说明预定义划分 (B, N) 何时等于最优划分 $(\mathcal{B}, \mathcal{N})$。

2.2. 线性优化问题实例生成器

在本节剩余部分中，我们回顾三种主要生成器。它们可以根据预先指定（或随机选取）的内点解、预先指定（或随机选取）的最优解（可能严格互补，也可能不是），或两者兼有，生成线性优化实例。每种线性优化问题生成器都允许用户控制参数（A、b、c）的特征，包括但不限于条件数、稀疏性和范数。此外，用户可以调整最优解的特征，以考察其算法的性能。

在以下算法中，“生成”一词应作宽泛理解。它可以指随机地生成相应数据，也可以表示出于某种特定目的来构造数据，例如，获得具有某种特定结构（如稀疏性或条件数）的矩阵。

2.2.1. 具有预定义内部解的线性优化问题

为了研究内点法应用于线性优化问题时的性能，通常需要具有特定内点解的实例；生成具有所需内点解的线性优化问题的一种常用方法如算法 1 所示。

算法 1 生成具有特定内点解的线性优化问题

选择维数 $m < n$ 选择或生成 (x^0, s^0)，使得 $x_i^0 > 0$ 且对于所有 $i \in [n]$ 均有 $s_i^0 > 0$ 生成 $A \in \mathbb{R}^{m \times n}$ 生成 $y^0 \in \mathbb{R}^m$ 计算 $b = Ax^0$ 和 $c = A^\top y^0 + s^0$ 返回线性优化问题 (A, b, c)，其内点解为 (x^0, y^0, s^0)

备注 1. 假设我们希望内点解 (x^0, s^0) 具有 $x^{0\top} s^0 = n\mu$ 的对偶间隙，其中 $\mu > 0$ 是某个标量。那么，在算法 1 的步骤 1 中，我们生成 $x_i^0 > 0$，并针对 $i \in [n]$ 计算 $s_i^0 = \frac{\mu}{x_i^0}$。

上述评论指出了与内点法相关的一点：在内点法的背景下，常数 μ（称为中心路径参数）起着至关重要的作用。内点法从某个初始内点解 $(x^0, s^0) \in \mathcal{PD}^0$ 开始，且

$$\frac{x^{0\top} s^0}{n} = \mu^0 > 0,$$

随着算法逐步求得具有所需互补间隙的线性优化问题的解，每次迭代都会减小 μ。与我们对线性优化对偶性的讨论一致，很容易看出，当 $\mu \to 0$ 时，我们将趋近于原始-对偶对 (LOP-P)-(LOP-D) 的最优解。

评论 2. 算法 1 便于生成具有任意所需性质的系数矩阵 A，例如稀疏性、结构或病态性。

评论 3. 随机生成满行秩系数矩阵 A，并使其以概率一满足条件，需要满足若干条件 [, 参见例如 7]。

2.2.2. 具有预定义最优解的线性优化问题

算法2描述了一种生成具有已知最优解的线性优化问题的常用方法。

算法2 生成具有特定最优解的线性优化问题

选择维度 $m < n$
将指标集 $[n]$ 划分为 B 和 N，其中包含 $B \cap N = \emptyset$ 和 $B \cup N = [n]$
生成 x^*，使得 $x^*_i > 0$ 对于 $i \in B$ 和 $x_i^* = 0$ 对于 $i \in N$
生成 s^*，使得 $s^*_i > 0$ 对于 $i \in N$ 和 $s_i^* = 0$ 对于 $i \in B$
生成 $A \in \mathbb{R}^{m \times n}$
生成 $y^* \in \mathbb{R}^m$
计算 $b = Ax^*$ 和 $c = A^\top y^* + s^*$
返回线性优化问题 (A, b, c)，其最优解为 (x^*, y^*, s^*)

备注4。由于算法2生成的最优解（x^*, y^*, s^*）是严格互补的，因此最优划分（$\mathcal{B}, \mathcal{N}$）等于（B, N）。

备注 5. 划分 (B, N) 可以随机生成，也可以满足某些所需性质，例如原始退化或对偶退化，或两者兼有，或者具有唯一的最优基解。

备注 6. 令 $A = [A_B \ A_N]$。如果 $|B| = m$ 和 A_B 是非奇异的，那么 x^* 和 s^*给出唯一的最优基解。

备注 7. 如果我们修改算法2，生成 x^* 使得 $x_i^* \geq 0$ 对于 $i \in B$ 成立，且$x_i^* = 0$ 对于 $i \in N$ 成立，并生成 s^* 使得 $s_i^* \geq 0$ 对于 $i \in N$ 成立，且 $s_i^* = 0$ 对于 $i \in B$成立，那么 B 和 N 不一定给出最优划分。虽然 x^* 和 s^* 是互补解，但它们不一定严格互补。

2.2.3. 具有预定义的最优解和内点解的线性优化问题

Charnes 等人 [6] 讨论了生成具有特定最优解或内点解的问题的步骤。在此，我们提出一种新方法，用于生成具有特定最优解（x^*, y^*, s^*）和特定内点解（x^0, y^0, s^0）的线性优化问题，如算法 3 所示。其基本思路是先使用算法2生成一个具有最优解（x^*, y^*, s^*）的问题，然后通过添加一个变量和一个约束来扩展该问题，使内点（x^0, y^0, s^0）对新问题可行。采用这一方案，我们可以生成具有任意一般预定义最优解和内点解的线性优化问题，其唯一的附加条件是

$$(x^0 - x^*)^\top (s^0 - s^*) = 0. \tag{1}$$

公式 (1) 规定的条件是一种自然性质；对于任意可行解对，都有 $(x^* - x^0) \in \text{Lin}^\perp(A)$ 和 $(s^* - s^0) \in \text{Lin}(A)$，其中 $\text{Lin}(A)$ 表示 A 的线性空间。换言之，预定义解 $x^0 - x^*$ 与 $s^0 - s^*$ 之差必须正交，而算法 3 的步骤 4 和 5 确保了这一性质成立。

算法 3 生成具有特定最优解和内点解的线性优化问题

1：选择 $m < n$ （生成的线性优化问题有 $m+1$ 个约束和 $n+1$ 个变量。） ˆ ˆ 2：使用算法 2 生成线性优化问题 $(A, b, \hat{c})$，其最优解为 $(\hat{x}, \hat{y}, \hat{s})$，划分为 (B, N) 3：生成 $x^0, s^0 \in \mathbb{R}^n$，使得 $x^0, s^0 > 0$ 4：令 $\delta = (x_B^0 - \hat{x}_B)^\top s_B^0 + (s_N^0 - \hat{s}_N)^\top x_N^0$，生成 $x_{n+1}^0 > 0$ 和 $s_{n+1}^0 > 0$ 5：计算 $\hat{s}_{n+1} = \frac{\delta}{x_{n+1}^0} + s_{n+1}^0$ 并令 $\hat{x}_{n+1} = 0$ 6：构建 $x^* = \begin{pmatrix} \hat{x}_B \\ \hat{0} \\ 0 \end{pmatrix}$ 和 $x^0 = \begin{pmatrix} x_B^0 \\ x_N^0 \\ x_{n+1}^0 \end{pmatrix}$ 7：构建 $s^* = \begin{pmatrix} s_B \\ \hat{s}_N \\ \hat{\hat{s}}_{n+1} \end{pmatrix}$ 和 $s^0 = \begin{pmatrix} s_B^0 \\ s_N^0 \\ s_{n+1}^0 \end{pmatrix}$ 8：随机生成 $y^0 = \begin{pmatrix} y_{1:m}^0 \\ y_{m+1}^0 \end{pmatrix} \in \mathbb{R}^{m+1}$，使得 $y_{m+1}^0 \neq 0$ 9：构建 $y^* = \begin{pmatrix} \hat{y} \\ 0 \end{pmatrix}$ 10：计算

$$\hat{a}_{n+1} = \frac{1}{x_{n+1}^0}(\hat{A}_B(\hat{x}_B - x_B^0) - \hat{A}_N x_N^0)$$
$$d_B = \frac{1}{y_{m+1}^0}(\hat{A}_B^\top(\hat{y} - y_{1:m}^0) - s_B^0)$$
$$d_N = \frac{1}{y_{m+1}^0}(\hat{A}_N^\top(\hat{y} - y_{1:m}^0) + s_N^* - s_N^0)$$
$$d_{n+1} = \frac{1}{x_{n+1}^0}(d_B^\top(\hat{x}_B - x_B^0) - d_N^\top x_N^0)$$

11: 构造 $A_{(m+1)\times(n+1)} = \begin{pmatrix} \hat{A}_B & A_N & a_{n+1} \\ d_B^\top & d_N^\top & d_{n+1} \end{pmatrix}$ ˆ ˆ 12: 计算 $b = \begin{pmatrix} b \\ \hat{d}_B^\top \hat{x}_B \end{pmatrix}$ 和 $c = \begin{pmatrix} \hat{c} \\ y + s_{n+1}^* \end{pmatrix}$ ˆ $a_{n+1}^\top$ ˆ 13: 返回线性优化问题 (A, b, c)，其最优解为 (x^*, y^*, s^*)，内点解为 (x^0, y^0, s^0)

定理 2.2 指出，(x^0, y^0, s^0) 和 (x^*, y^*, s^*) 的相关性质确实成立。在陈述并证明定理 2.2 之前，我们需要验证所生成解的正交性。

引理 2.1. 对于由算法 3 生成的 (x^0, y^0, s^0) 和 (x^*, y^*, s^*)，我们有

$$(x^0 - x^*)^\top (s^0 - s^*) = 0.$$

证明. 根据构造，我们有

$$\begin{aligned}(x^0 - x^*)^\top (s^0 - s^*) =& (x_B^0)^\top s_B^0 + (x_B^*)^\top s_B^* - (x_B^0)^\top s_B^* - (x_B^*)^\top s_B^0 (x_N^0)^\top s_N^0 \\ &+ (x_N^*)^\top s_N^* - (x_N^0)^\top s_N^* - (x_N^*)^\top s_N^0 (x_{n+1}^0)^\top s_{n+1}^0 \\ &+ (x_{n+1}^*)^\top s_N^* - (x_{n+1}^0)^\top s_{n+1}^* - (x_{n+1}^*)^\top s_{n+1}^0 \\ =& -\delta + (x_{n+1}^0)^\top s_{n+1}^0 - (x_{n+1}^0)^\top s_{n+1}^* = 0.\end{aligned}$$

证明完毕。 □

利用引理 2.1，下面的定理表明，生成问题满足所需的性质。

定理 2.2. 设 (x^0, y^0, s^0) 和 (x^*, y^*, s^*) 由算法 3 生成。则，

$$x^* \geq 0,\ s^* \geq 0,\ x^0 > 0,\ s^0 > 0, \tag{2a}$$
$$(x^*)^\top s^* = 0, \tag{2b}$$
$$Ax^* = b, \tag{2c}$$
$$A^\top y^* + s^* = c, \tag{2d}$$
$$Ax^0 = b, \tag{2e}$$
$$A^\top y^0 + s^0 = c. \tag{2f}$$

也就是说，(x^0, y^0, s^0) 和 (x^*, y^*, s^*) 分别是生成的线性优化问题 (A, b, c) 的内点解和最优解。

证明。注意，由构造可知 (2a) 成立。等式 (2b) 指的是 (x^*, y^*, s^*) 的互补性，这是因为

$$x^{*\top} s^* = \hat{x}_B^\top 0 + 0^\top \hat{s}_N + 0 s_{n+1}^* = 0.$$

为说明等式 (2c) 成立，即最优解满足原始可行性，注意到

$$Ax^* = \begin{pmatrix} \hat{A}_B & \hat{A}_N & \hat{a}_{n+1} \\ d_B^\top & d_N^\top & d_{n+1} \end{pmatrix} \begin{pmatrix} \hat{x}_B \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} \hat{A}_B \hat{x}_B \\ d_B^\top \hat{x}_B \end{pmatrix} = \begin{pmatrix} \hat{b} \\ d_B^\top \hat{x}_B \end{pmatrix} = b.$$

同样，由于最优解满足对偶可行性，

$$A^\top y^* + s^* = \begin{pmatrix} \hat{A}_B^\top & d_B \\ \hat{A}_N^\top & d_N \\ \hat{a}_{n+1}^\top & d_{n+1} \end{pmatrix} \begin{pmatrix} \hat{y} \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ \hat{s}_N \\ s_{n+1}^* \end{pmatrix} = \begin{pmatrix} \hat{c} \\ \hat{a}_{n+1}^\top \hat{y} + s_{n+1}^* \end{pmatrix} = c.$$

也就是说，方程 (2d) 成立。

内点解 (x^0, y^0, s^0) 是原始可行的，因为

$$\begin{aligned}
Ax^0 &= \begin{pmatrix} \hat{A}_B & \hat{A}_N & \hat{a}_{n+1} \\ d_B^\top & d_N^\top & d_{n+1} \end{pmatrix} \begin{pmatrix} x_B^0 \\ x_N^0 \\ x_{n+1}^0 \end{pmatrix} \\
&= \begin{pmatrix} \hat{A}_B x_B^0 + \hat{A}_N x_N^0 + \hat{a}_{n+1} x_{n+1}^0 \\ d_B^\top x_B^0 + d_N^\top x_N^0 + d_{n+1} x_{n+1}^0 \end{pmatrix} \\
&= \begin{pmatrix} \hat{A}_B x_B^0 + \hat{A}_N x_N^0 + (\hat{A}_B(\hat{x}_B - x_B^0) - \hat{A}_N x_N^0) \\ d_B^\top x_B^0 + d_N^\top x_N^0 + (d_B^\top(\hat{x}_B - x_B^0) - d_N^\top x_N^0) \end{pmatrix} \\
&= \begin{pmatrix} \hat{A}_B \hat{x}_B \\ d_B^\top \hat{x}_B \end{pmatrix} = \begin{pmatrix} \hat{b} \\ d_B^\top \hat{x}_B \end{pmatrix} = b,
\end{aligned}$$

这就证明了 (2e)。我们还可以验证内点解的对偶可行性:

$$\begin{aligned}
A^\top y^0 + s^0 &= \begin{pmatrix} \hat{A}_B^\top & d_B \\ \hat{A}_N^\top & d_N \\ \hat{a}_{n+1}^\top & d_{n+1} \end{pmatrix} \begin{pmatrix} y_{1:m}^0 \\ y_{m+1}^0 \end{pmatrix} + \begin{pmatrix} s_B^0 \\ s_N^0 \\ s_{n+1}^0 \end{pmatrix} = \begin{pmatrix} \hat{A}_B^\top y_{1:m}^0 + d_B y_{m+1}^0 + s_B^0 \\ \hat{A}_n^\top y_{1:m}^0 + d_N y_{m+1}^0 + s_N^0 \\ \hat{a}_{n+1}^\top y_{1:m}^0 + d_{n+1} y_{m+1}^0 + s_{n+1}^0 \end{pmatrix} \\
&= \begin{pmatrix} \hat{A}_B^\top y_{1:m}^0 + (\hat{A}_B^\top(\hat{y} - y_{1:m}^0) - s_B^0) + s_B^0 \\ \hat{A}_n^\top y_{1:m}^0 + (\hat{A}_N^\top(\hat{y} - y_{1:m}^0) + s_N^* - s_N^0) + s_N^0 \\ \alpha \end{pmatrix} = \begin{pmatrix} \hat{A}_B^\top \hat{y} \\ \hat{A}_N^\top \hat{y} + s_N^* \\ \alpha \end{pmatrix} \\
&= \begin{pmatrix} \hat{c} \\ \hat{a}_{n+1}^\top \hat{y} + s_{n+1}^* \end{pmatrix} = c,
\end{aligned}$$

其中 $\alpha = \hat{a}_{n+1}^\top y_{1:m}^0 + d_{n+1} y_{m+1}^0 + s_{n+1}^0$。

最后，为了证明方程 (2f) 也成立，我们还需要证明 $\alpha = \hat{a}_{n+1}^\top \hat{y} + s_{n+1}^*$。通过直接计算可得

$$\begin{aligned}
\alpha &= \frac{1}{x_{n+1}^0}(\hat{A}_B(\hat{x}_B - x_B^0) - \hat{A}_N x_N^0)^\top y_{1:m}^0 + \frac{y_{m+1}^0}{x_{n+1}^0}(d_B^\top(\hat{x}_B - x_B^0) - d_N^\top x_N^0) + s_{n+1}^0 \\
&= \frac{1}{x_{n+1}^0}\Big((\hat{A}_B(\hat{x}_B - x_B^0) - \hat{A}_N x_N^0)^\top y_{1:m}^0 + (\hat{A}_B^\top(\hat{y} - y_{1:m}^0) - s_B^0)^\top(\hat{x}_B - x_B^0) - (\hat{A}_N^\top(\hat{y} - y_{1:m}^0) + s_N^* - s_N^0)^\top x_N^0\Big) + s_{n+1}^0 \\
&= \frac{1}{x_{n+1}^0}\Big({y_{1:m}^0}^\top \hat{A}_B \hat{x}_B - {y_{1:m}^0}^\top \hat{A}_B x_B^0 - {y_{1:m}^0}^\top \hat{A}_N x_N^0 + \hat{x}_B^\top \hat{A}_B^\top \hat{y} - \hat{x}_B^\top \hat{A}_B^\top y_{1:m}^0 - \hat{x}_B^\top s_B^0 - {x_B^0}^\top \hat{A}_B^\top \hat{y} + {x_B^0}^\top \hat{A}_B^\top y_{1:m}^0 + {x_B^0}^\top s_B^0 \\
&\quad - {x_N^0}^\top \hat{A}_N^\top \hat{y} + {x_N^0}^\top \hat{A}_N^\top y_{1:m}^0 - {x_N^0}^\top s_N^* + {x_N^0}^\top s_N^0\Big) + s_{n+1}^0 \\
&= \frac{1}{x_{n+1}^0}\Big(\hat{x}_B^\top \hat{A}_B^\top \hat{y} - {x_B^0}^\top \hat{A}_B^\top \hat{y} - {x_N^0}^\top \hat{A}_N^\top \hat{y} - \hat{x}_B^\top s_B^0 + {x_B^0}^\top s_B^0 - {x_N^0}^\top s_N^* + {x_N^0}^\top s_N^0\Big) + s_{n+1}^0 \\
&= \frac{(\hat{x}_B^\top \hat{A}_B^\top - {x_B^0}^\top \hat{A}_B^\top - {x_N^0}^\top \hat{A}_N^\top)}{x_{n+1}^0}\hat{y} + \frac{-\hat{x}_B^\top s_B^0 + {x_B^0}^\top s_B^0 - {x_N^0}^\top s_N^* + {x_N^0}^\top s_N^0 + s_{n+1}^0 x_{n+1}^0}{x_{n+1}^0} \\
&= \hat{a}_{n+1}^\top \hat{y} + \frac{(x^0 - x^*)^\top(s^0 - s^*) + s_{n+1}^* x_{n+1}^0}{x_{n+1}^0} = \hat{a}_{n+1}^\top \hat{y} + s_{n+1}^*.
\end{aligned}$$

The proof is complete. □

注 8. 由于算法 3 生成的最优解 (x^*, y^*, s^*) 是严格互补的，因此最优划分 $(\mathcal{B}, \mathcal{N})$ 等于 (B, N)。如果我们修改算法 3，使得 $x_i^* \geq 0$ 用于 $i \in B$，而 $s_i^* \geq 0$ 用于 $i \in N$，那么 B 和 N 不一定给出最优划分。虽然 x^* 和 s^* 是互补解，但它们不一定是严格互补的。

注 9. 我们可以通过设置以下内容来简化算法 3

$$x_B^0 = \hat{x}_B, s_N^0 = \hat{s}_N, s_{n+1}^* = s_{n+1}^0, and\ y_{1:m}^0 = \hat{y}.$$

不难验证，对于这些选择，条件 1 得以满足。一个更简单的情形是选择

$$x_N^0 = e, s_B^0 = e, \text{ and } y_{m+1}^0 = x_{n+1}^0 = s_{n+1}^0 = 1.$$

在下一节中，我们将扩展这些问题生成器，以生成半定优化问题。

3. 半定优化

现在，我们将注意力转向半定优化。与上一节一样，我们首先回顾问题设定和重要性质，然后介绍这一类优化问题的实例生成器。

3.1. 半定优化问题

在半定优化中，我们旨在最小化两个 $n \times n$对称矩阵的内积:

$$C \bullet X = \operatorname{tr}(CX) = \sum_{i=1}^{n} \sum_{j=1}^{n} C_{ij} X_{ij},$$

其中 $C \in \mathcal{S}^n$ 为对称常数矩阵， $X \in \mathcal{S}^n$为矩阵变量。注意，$\mathcal{S}^n$表示 $n \times n$对称矩阵组成的空间，我们用 $\mathcal{S}_+^n (\mathcal{S}_{++}^n)$ 表示对称半正定（对称正定）矩阵锥。

与上一节研究的线性优化问题类似，变量 X 必须满足如下形式的线性约束:

$$A_i \bullet X = b_i, \quad \forall i \in [m],$$

其中 A_1，…， $A_m \in \mathcal{S}^n$ 为给定的对称矩阵，且 $b \in \mathbb{R}^m$。鉴于 $C \bullet X$ 是关于 X的线性函数，若就此为止，得到的只会是一个线性优化问题，其变量由矩阵 X的各个元素构成。实际上，我们还加入了一个非线性（但为凸）约束，规定 X 必须是半正定矩阵，记作 $X \succeq 0$。更一般地，符号 $U \succeq V$ 表示 $U - V$ 是对称半正定矩阵，这等价于说明 $U - V \in \mathcal{S}_+^n$。同样，当不等式为严格不等式，即 $U \succ V$时，可得 $U - V \in \mathcal{S}_{++}^n$，因此 $U - V$为对称正定矩阵。由上述讨论不难看出，半定优化是线性优化的一种推广：它将（线性优化问题-对偶）中的逐元素非负约束 $x \geq 0$ 替换为关于锥 $\mathcal{S}_+^n$的锥不等式。

A因此，本节将探讨如何生成如下形式的问题:

$$z_{SDO}^P = \inf_X \{C \bullet X : A_i \bullet X = b_i,\ \forall i \in [m], X \succeq 0\}, \qquad \text{(SDOP-P)}$$

该问题对应的对偶问题为

$$z_{SDO}^{D} = \sup_{(y,S)} \left\{ b^{\top} y : \sum_{i=1}^{m} y_i A_i + S = C,\ S \succeq 0, y \in \mathbb{R}^m \right\}, \qquad \text{(SDOP-D)}$$

其中， $S = C - \sum_{i=1}^{m} y_i A_i$ 是对偶问题的松弛矩阵。不失一般性，我们可以假设矩阵 A_1、.....、 A_m 线性无关。

如果 X 和（y， S）分别满足原始约束和对偶约束，则称它们为可行解，并将（SDOP-P）和（SDOP-D）的可行集记为:

$$\mathcal{P}_{SDO} = \{X \in \mathcal{S}^n : A_i \bullet X = b_i,\ i \in [m], X \succeq 0\}$$
$$\mathcal{D}_{SDO} = \left\{ (y, S) \in \mathbb{R}^m \times \mathcal{S}^n \ : \sum_{i=1}^{m} y_i A_i + S = C, S \succeq 0 \right\}.$$

相应地，可行内点解的集合为

$$\mathcal{P}_{SDO}^{0} = \{X \in \mathcal{P}_{SDO} : X \succ 0\},$$
$$\mathcal{D}_{SDO}^{0} = \{(y, S) \in \mathcal{P}_{SDO} : S \succ 0\}.$$

为便于表示，我们采用

$$\mathcal{PD}_{SDO} = \mathcal{P}_{SDO} \times \mathcal{D}_{SDO} \qquad \mathcal{PD}_{SDO}^{0} = \text{和的写法。}$$

$\mathcal{P}_{SDO}^{0}$与线性优化（LO）的情形一样，将内点法（IPMs）应用于半定优化问题（SDOPs）时，通常假设存在严格可行原始-对偶对 X 和（y， S），其中（X， S） $\succ 0$。由严格可行初始解（X^0， S^0） $\succ 0$的存在可知，内点条件（IPC）成立 [9], ，从而保证原始和对偶最优集

$$\mathcal{P}_{SDO}^{*} = \left\{ X \in \mathcal{P}_{SDO} : C \bullet X = z_{SDO}^{P} \right\},$$
$$\mathcal{D}_{SDO}^{*} = \left\{ (y, S) \in \mathcal{D}_{SDO} : b^{\top} y = z_{SDO}^{D} \right\},$$

非空且有界，存在一个零对偶间隙的原始-对偶最优对，即强对偶成立。也就是说，对于最优解 $(X^*, y^*, S^*) \in \mathcal{PD}_{SDO}^{*}$，其中 $\mathcal{PD}_{SDO}^{*} = \mathcal{P}_{SDO}^{*} \times \mathcal{D}_{SDO}^{*}$，我们有

$$C \bullet X^* - b^{\top} y^* = X^* \bullet S^* = 0,$$

这意味着 $X^* S^* = S^* X^* = 0$ ，因为 X^* 和 S^* 是对称半正定矩阵。

3.2. 半定优化问题实例生成器

与我们在线性优化方面的工作类似，我们提出了三种生成器，用于生成具有预定义内点解、最优解或同时具有两者的半定优化实例。每种生成器都经过设计，使用户能够控制条件数、稀疏性、矩阵结构和规模等参数的特征。此外，用户还可以调整最优解的特征，以评估其算法的性能。

3.2.1. 具有预定义内部解的半定优化问题

为了研究内点法应用于半定优化时的性能，拥有具有特定内点解的实例会很有帮助。通常，部分用户可能需要生成具有内点解的问题，以确保强对偶成立，即最优时对偶间隙为零。为此，我们对算法 1 进行调整，以生成具有已知内点解的半定优化实例，如算法 4 所示。

算法 4 生成具有特定内点解的半定优化问题

1: 选择维数 m ，n 其中 $m < \frac{n(n+1)}{2}$
2: 生成 (X^0, S^0)，使得 $X^0 \succ 0$ 并且 $S^0 \succ 0$
3: 为 $i \in [m]$生成 $A_i \in \mathcal{S}^n$
4: 生成 $y^0 \in \mathbb{R}^m$
5: 计算 $b_i = A_i \bullet X^0$，用于 $i \in [m]$ 和 $C = \sum_{i=1}^m y_i^0 A_i + S^0$
6: 返回具有内点解 (X^0, y^0, S^0) 的半定优化问题 $(A_1, \ldots, A_m, b, C)$

与算法 1 相比，以一般方式生成 X^0 和 S^0，使得对于 $\mu > 0$ 有 $X^0 S^0 = \mu I$，计算上要复杂得多；我们需要先随机地生成 $X^0 \succ 0$，随后将 S^0 计算为 $S^0 = \mu(X^0)^{-1}$。不过，如果我们对其结构作出额外假设（例如，假设它们是对角矩阵），就可以轻松地为指定的 μ值生成 X^0 和 S^0。我们还可以生成矩阵 $A_1, \ldots, A_m$，使其具有所需的性质，例如稀疏性、条件数，或满足某个范数界。附录 A 讨论了几种生成随机半正定矩阵的方法。

3.2.2. 具有预定义块对角最优解的半定优化问题

算法 5 可视为算法2 在半定优化问题上的推广。在这些问题中，生成的最优解明确具有与最优划分相对应的块对角结构。在介绍实例生成器之前，我们先回顾半定优化背景下最优划分的记号。

我们关注最优解 (X^*, y^*, S^*) 具有零对偶间隙的问题，即 $X^* S^* = 0$。因此，最优对 (X^* 和 S^*) 的谱分解形式为

$$X^* = Q \Sigma Q^\top \text{ and } S^* = Q \Lambda Q^\top,$$

其中 Q 是正交归一的，矩阵 Σ 和 Λ 是对角矩阵，分别包含 X^* 和 S^*的特征值。令 $\sigma_i = \Sigma_{i,i}$ 和$\lambda_i = \Lambda_{i,i}$，则对于所有 $i \in [n]$，当且仅当 $\sigma_i \lambda_i = 0$ 时，$X^* S^* = 0$成立。原始-对偶最优解（X^*，y^*，S^*）$\in \mathcal{PD}^*_{SDO}$ 称为最大互补的，如果 $X^* \in \text{ri}(\mathcal{P}^*_{SDO})$且（$y^*$，$S^*$）$\in \text{ri}(\mathcal{D}^*_{SDO})$。最大互补最优解（$X^*$，$y^*$，$S^*$）称为严格互补的，如果 $X^* + S^* \succ 0$。令 $\mathcal{B} := \mathcal{R}(X^*)$和 $\mathcal{N} := \mathcal{R}(S^*)$，其中（$X^*$，$y^*$，$S^*$）是最大互补最优解，$\mathcal{R}(.)$表示值域空间。我们定义 $n_\mathcal{B} := \dim(\mathcal{B})$和 $n_\mathcal{N} := \dim(\mathcal{N})$。于是，对于所有（$X$，$y$，$S$）$\in \mathcal{PD}^*_{SDO}$，有 $\mathcal{R}(X) \subseteq B$和 $\mathcal{R}(S) \subseteq \mathcal{N}$。根据互补条件，子空间 $\mathcal{B}$ 和 $\mathcal{N}$ 正交，这意味着 $n_\mathcal{B} + n_\mathcal{N} \leq n$；在严格互补的情况下，$n_\mathcal{B} + n_\mathcal{N} = n$。否则，存在子空间 $\mathcal{T}$，它是 $\mathcal{B} + \mathcal{N}$的正交补。类似地，有 $n_\mathcal{T} := \dim(\mathcal{T})$，因此 $n_\mathcal{B} + n_\mathcal{N} + n_\mathcal{T} = n$[15]。$\mathbb{R}^n$ 的划分（$\mathcal{B}$，$\mathcal{N}$，$\mathcal{T}$）称为半定优化问题的最优划分。

在线性优化问题中，我们知道 $\mathcal{T}$ 为空；但一般而言，在半定优化问题中， $\mathcal{T}$ 可以非空 [9]。

在算法 5 中，我们使用划分 (B, N, T) 生成半定优化问题，其最优解呈现块对角结构；该划分可能不同于生成问题的最优划分 $(\mathcal{B}, \mathcal{N}, \mathcal{T})$。

算法 5 生成具有特定最优解的半定优化问题

1: 选择维数 m , n 其中 $m < \frac{n(n+1)}{2}$
2: 选择 $n_B, n_N \in [n]$ ，其中 $n_B + n_N \leq n$
3: 生成正定矩阵 $X_B \in \mathcal{S}^{n_B}_{++}$
4: 生成正定矩阵 $S_N \in \mathcal{S}^{n_N}_{++}$
5: 构建[1] $X^* = \begin{pmatrix} X_B & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$ 以及 $S^* = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & S_N \end{pmatrix}$
6: 为 $i \in [m]$生成 $A_i \in \mathcal{S}^n$
7: 生成 $y^* \in \mathbb{R}^m$
8: 为 $i \in [m]$ 和 $C = \sum_{i=1}^m y_i^* A_i + S^*$计算 $b_i = A_i \bullet X^*$
9: 返回具有最优解（X^*, y^*, S^*）的半定优化问题（$A_1, \ldots, A_m, b, C$）

备注 10. 算法 5 生成的集合（B, N, T）不一定是所生成半定优化问题（$A_1, \ldots, A_m, b, C$）的最优划分（$\mathcal{B}, \mathcal{N}, \mathcal{T}$）。一般来说，我们只能得到

$$B \subseteq \mathcal{B}, N \subseteq \mathcal{N}, \text{ and } \mathcal{T} \subseteq T.$$

备注 11. 如果要求半定优化问题具有严格互补最优解，则我们令 $n_N = n - n_B$。此时，最优划分预先设定为 $\mathcal{B} = B$、$\mathcal{N} = N$ 和 $\mathcal{T} = \emptyset$。

容易验证，算法 5 生成的解（X^*, y^*, S^*）对于半定优化问题（$A_1, \ldots, A_m, b, C$）是可行的，并且由于 $X^* S^* = 0$，该解也是最优的。此外，矩阵 A_i 和 C 可以按特定的稀疏性、条件数或范数生成，而且我们还可以控制原始和/或对偶退化。

3.2.3. 具有预定义块对角最优解和内点解的半定优化问题

通过生成具有特定内点解和最优解的半定优化问题，我们可以研究各种求解方法的性能。例如，可以分析可行内点法从预定义的内点解出发，将互补性降至最优解所需的效率；或者考察算法性能对所提供的起始点，以及最优解特征或划分变化的稳健性。为此，本文提出了若干算法，用于提供最优解或最大互补解。

本节关注这样一种情形：用户希望预先定义一个内点解和一个最优解，而二者不一定是最大互补的。因此，对于生成的最优解具有块对角结构的情形，算法 6 将算法 3 推广到了半定优化。我们同样希望尽可能一般地生成一个最优解（ X^*, y^*, S^* ）和一个内点解（X^0, y^0, S^0），但需要施加一些额外要求。令$\mathcal{L}$ =张成空间$\{A_1, \ldots, A_m\}$, 则有 $X^0 - X^* \in \mathcal{L}^\perp$ 和 $S^0 - S^* \in \mathcal{L}$, 因此,

生成的解必须满足正交条件

$$(X^0 - X^*) \bullet (S^0 - S^*) = 0. \tag{3}$$

在算法 6 中，步骤 7 和 8 旨在确保生成的解 (X^*, y^*, S^*) 和 (X^0, y^0, S^0) 确实满足正交性。

算法 6 生成具有特定内点和最优解的半定优化问题

1: 选择维度 m, n with $m < {}^{n}_{(n+1)}$ 22: 选择 $n_B, n_N \in [n]$ where $n_B + n_N \leq n$ ˆ ˆ ˆ ˆ ˆ3: 生成半定优化问题 $(A_1, \ldots, A_m, b, C)$，其最优解为 $(X, \hat{y}, \hat{s})$，使用

g

算法 5 4: 随机生成 $X_B^0 \succ 0$, $X_T^0 \succ 0$,$X_N^0 \succ 0$,$X_{n+1}^0 > 0$ 5: 构建 $X^*_{(n+1)\times(n+1)} = \begin{pmatrix} \hat{X} & 0 \\ 0 & 0 \end{pmatrix}$ 和 $X^0_{(n+1)\times(n+1)} = \begin{pmatrix} X_B^0 & 0 & 0 & 0 \\ 0 & X_T^0 & 0 & 0 \\ 0 & 0 & X_N^0 & 0 \\ 0 & 0 & 0 & X_{n+1}^0 \end{pmatrix}$ 6: 随机

生成 $S_T^0 \succ 0$, $S_B^0 \succ 0$, $S_N^0 \succ 0$ 7: 计算 $\delta = (X_B^0 - \hat{X}_B) \bullet S_B^0 + X_T^0 \bullet S_T^0 + X_N^0 \bullet (S_N^0 - \hat{S}_N)$ 8: 生成 $S_{n+1}^0 > (\frac{-\delta}{X_{n+1}^0})^+$ 并计算 $\hat{S}_{n+1} = \frac{\delta}{X_{n+1}^0} + S_{n+1}^0$ 9: 构建 $S^*_{(n+1)\times(n+1)} = \begin{pmatrix} \hat{S} & 0 \\ 0 & \hat{S}_{n+1} \end{pmatrix}$ 和 $S^0_{(n+1)\times(n+1)} = \begin{pmatrix} S_B^0 & 0 & 0 & 0 \\ 0 & S_T^0 & 0 & 0 \\ 0 & 0 & S_N^0 & 0 \\ 0 & 0 & 0 & S_{n+1}^0 \end{pmatrix}$ 10:

随机生成 $y^0 \in \mathbb{R}_{m+1}$，使得 $y_{m+1}^0 \neq 0$ 11: 构建 $y^* = \begin{pmatrix} \hat{y} \\ 0 \end{pmatrix} \in \mathbb{R}_{m+1}$ 12: 对 $i \in [m]$ 计算 $\alpha_i = \frac{1}{X_{n+1}^0}(\hat{A}_{i_B} \bullet (X_B - X_B^0)) - (\hat{A}_{i_N} \bullet X_N^0) - (\hat{A}_{i_T} \bullet X_T^0))$ 13: 对 $i \in [m]$ 构建 $A_i = \begin{pmatrix} \hat{A}_i & 0 \\ 0 & \alpha_i \end{pmatrix}$ 14: 构建 $A_{n+1} = \sum_{i=1}^{m} \hat{} \begin{pmatrix} -S_B^0 & 0 & 0 & 0 \\ 0 & -S_T^0 & 0 & 0 \\ 0 & 0 & S_N - S_N^0 & 0 \\ 0 & 0 & 0 & \hat{} \hat{} S_{n+1} - S_{n+1}^0 \end{pmatrix}$ 15: 计算 $\theta = \frac{y_i - y_i^0}{y_{m+1}^0} A_i + \frac{1}{y_{m+1}^0}$

$\hat{S}_{n+1} + \sum_{i=1}^m \hat{y}_i \alpha_i$ 并构建 $C = \begin{pmatrix} \hat{C} & 0 \\ 0 & \theta \end{pmatrix}$ 16: 计算 $\beta = A_{n+1} \bullet X^*$ 并构建 $b = \begin{pmatrix} \hat{} \\ b \\ \beta \end{pmatrix}$ 17: 返回半定优化问题 $(A_1, \ldots, A_m, b, C)$，其最优解为 (X^*, y^*, S^*)，且内点

r

解 (X^0, y^0, S^0)

在证明算法 6 的正确性之前，下一个结果将验证所生成解的正交性质。

引理 3.1. 对于算法 6 生成的任意 (X^0, y^0, S^0) 和 (X^*, y^*, S^*)，有

$$(X^0 - X^*) \bullet (S^0 - S^*) = 0.$$

证明. 与引理 2.1 的证明类似，通过代入并利用步骤 7 和 8 即可证明。

□

利用引理 3.1，下面的定理表明生成问题满足所需的性质。

定理 3.2. 设 (X^0, y^0, S^0) 和 (X^*, y^*, S^*) 是由算法 6 生成的解。那么，

$$X^* \succeq 0,\ S^* \succeq 0,\ X^0 \succ 0,\ S^0 \succ 0, \tag{4a}$$
$$X^* \bullet S^* = 0, \tag{4b}$$
$$A_i \bullet X^* = b_i, \tag{4c}$$
$$\sum_{i=1}^{n} y_i^* A_i^\top + S^* = C, \tag{4d}$$
$$A_i \bullet X^0 = b_i, \tag{4e}$$
$$\sum_{i=1}^{n} y_i^0 A_i^\top + S^0 = C. \tag{4f}$$

证明. 与线性优化的情形一样，定理 3.2 的所有部分都很容易根据算法 6 的步骤验证，除了关于 $i = n + 1$ 的等式 (4e)。按照定理 2.2 的证明，所述结果可由 α 的定义和等式 (3) 得出。

□

与生成具有最优解的半定优化问题类似，我们也可以生成同时具有特定严格互补最优解和特定内点解的问题。

注 12. 一个特殊情形是 $n_B + n_N = n$、$T = \emptyset$ 和

$$X_B^0 = \hat{X}_B, X_N^0 = I, X_T^0 = I, S_B^0 = I, S_T^0 = I, S_N^0 = \hat{S}_N, X_{n+1}^0 = 1, S_{n+1}^0 = 1.$$

3.2.4. 具有预定义最优解的半定优化问题（一般结构）

我们还关注所需最优解不呈现块结构的情形。有些方法可以利用最优解的结构性质，例如，当它呈现块对角结构或具有稀疏性时。为了生成具备特定所需性质的最优解，我们采用特征值分解的逆过程。首先，我们生成对角矩阵 Σ 和 Λ，其对角元素分别为 X^* 和 S^* 的特征值。然后，可通过使用随机生成的正交归一矩阵 Q 对这些对角矩阵进行掩蔽，来计算 X^* 和 S^*；附录 B 讨论了生成正交矩阵的方法。总体方案在下文的算法 7 中形式化给出。

算法 7 生成具有特定最优解的半定优化问题

选择维数 m ，n 其中 $m < \frac{n}{(n+1)}$
选择 $n_B, n_N \in [n]$，其中 $n_B + n_N \leq n$
为 $i \in [n_B]$ 生成 $\sigma_i > 0$，并为 $i \in [n_N]$生成 $\lambda_i > 0$
生成正交归一矩阵 Q
构造 $X^* = Q \begin{pmatrix} \text{diag}(\sigma) & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} Q^\top$ 以及 $S^* = Q \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & \text{diag}(\lambda) \end{pmatrix} Q^\top$
为 $i \in [m]$随机地生成 $A_i \in \mathcal{S}^n$
随机地生成 $y^* \in \mathbb{R}^m$
计算 $b_i = A_i \bullet X^*$，其中 $i \in [m]$ 和 $C = \sum_{i=1}^m y_i^* A_i + S^*$
返回半定优化问题 $(A_1, \ldots, A_m, b, C)$，其最优解为 (X^*, y^*, S^*)

虽然算法 7 生成的最优解比算法 5 更一般，但由于需要进行多次矩阵乘法并生成一个正交矩阵，其计算开销更大。附录 B 讨论了生成正交矩阵的一些方法。容易验证，(X^*, y^*, S^*) 是最优的，因为 $X^* S^* = Q \Sigma \Lambda Q^\top = 0$。然而，与备注 10 类似，生成的最优解可能并非半定优化问题 $(A_1, \ldots, A_m, b, C)$ 的最大互补解。因此，生成的半定优化问题的最优划分 $(\mathcal{B}, \mathcal{N}, \mathcal{T})$ 可能不同于 (B, N, T)，使得 $\dim(\mathcal{B}) \geq n_B$ 和 $\dim(\mathcal{N}) \geq n_N$；也就是说，满足 $\sigma_i = \lambda_i = 0$ 的指标集合可能大于划分 T。下一节将讨论如何生成具有预定义最优分划的半定优化问题。

3.2.5. 具有预定义的最大互补解的半定优化问题（一般结构）

算法 7 生成的半定优化问题，其最优划分可能与输入划分不同，因为指定的最优解可能不是最大互补解。本节将开发一种生成具有特定最优划分的半定优化问题的方法，并由此生成具有特定最大互补解的半定优化问题。

算法 8 生成具有特定最大互补解的半定优化问题

选择维度 m、n，满足 $m < n \quad \frac{(n+1)}{}$
2 2：选择 $n_B, n_N \in [n]$，其中 $n_B + n_N \leq n$3：为 $i \in [n_B]$生成 $\sigma_i > 0$，并为 $i \in [n_N]$生成 $\lambda_i > 0$ 4：生成正交归一矩阵 Q 5：构造 $X^* = Q \begin{pmatrix} \text{diag}(\sigma & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} Q^\top$ 和 $S^* = Q \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0\,\text{diag}(\lambda) \\ 0 & 0 & \end{pmatrix} Q^\top$ 6：生成 $A_1 = Q \Gamma Q^\top$，使得 $\Gamma = \text{diag}(\gamma)$，其中

$\gamma_B = 0$、$\gamma_T > 0$和 $\gamma_N \in \mathbb{R}^{n_N}$7：生成 $A_i \in \mathcal{S}^n$ $i \in [m]$，使得对于 $i \in [m]$，$A_i Q_B$ 线性无关 8：生成 $y^* \in \mathbb{R}^m$9：计算 $i \in [m]$ 的 $b_i = A_i \bullet X^*$ 和 A_1的 $C = \sum_{i=1}^m y_i^* A_i + S^*$ 10：返回半定优化问题 $(A_1, \ldots, A_m, b, C)$，其最大互补解为 (X^*, y^*, S^*)

如我们所见，使用算法 8 生成半定优化问题时的自由度较小，当

与算法 7 相比。这可归因于矩阵 A_1 经过指定，以确保指定的最优解是最大互补的，而我们无法直接改变其特征。下一个定理将证明该生成器的正确性。

定理 3.3。对于算法 8 生成的问题（$A_1,\dots,A_m,b,C$），（X^*,y^*,S^*）是最大互补最优解。

证明。该结论可由反证法得出，此证明改编自 [25]。~假设（X^*,y^*,S^*）不是最大互补解，而（$X,\tilde{y},S$）是最大互补解。由于 $XS^*=0$，我们有

$$\mathcal{R}(X^*)\subseteq\mathcal{R}(\tilde{X})\subseteq\mathcal{R}(S^*)^{\perp}.$$

因此，我们可以写成

$$\tilde{X}=Q\begin{pmatrix}D_B & 0 & 0\\ 0 & D_T & 0\\ 0 & 0 & 0\end{pmatrix}Q^{\top}.$$

~由于 X 和 X^* 均可行，因此

$$0=A_1\bullet(\tilde{X}-X^*)=A_1\bullet Q\begin{pmatrix}D_B-\Lambda_B & 0 & 0\\ 0 & D_T & 0\\ 0 & 0 & 0\end{pmatrix}Q^{\top}=\Gamma\bullet\begin{pmatrix}D_B-\Lambda_B & 0 & 0\\ 0 & D_T & 0\\ 0 & 0 & 0\end{pmatrix}$$
$$=\Gamma_T\bullet D_T.$$

~鉴于 $\Gamma_T>0$，可得 $D_T=0$，这意味着 $\mathcal{R}(X^*)=\mathcal{R}(X)$。
接下来，我们需要证明 $\mathcal{R}(S^*)=\mathcal{R}(\tilde{S})$。再次，根据对偶可行性，我们有

$$\sum_{i=1}^{m}A_i(y_i^*-\tilde{y}_i)=-(S^*-\tilde{S}).$$

~根据 Q_B 和 S^*-S 的正交性，可以观察到

$$\sum_{i=1}^{m}A_iQ_B(y_i^*-\tilde{y}_i)=-(S^*-\tilde{S})Q_B=0.$$

由于对于 $i\in[m]$，矩阵 A_iQ_B 线性无关，因此可得 $y_i^*=\tilde{y}_i$ 和 $S^*=S$，从而 (X^*,y^*,S^*) 是最大互补的。因此，我们得到了矛盾，证明完毕。

□

推论 3.4。对于由算法 8 生成的半定优化问题，其最优划分 ($\mathcal{B},\mathcal{N},\mathcal{T}$) 等于 ($B,N,T$)。

注 13. 令 $\Pi_i=A_iQ_B$ 对于 $i\in[m]$。可以生成矩阵 Π_i 对于 $i\in\{2,\dots,m\}$，使得矩阵集合 Π_i 对于 $i\in[m]$ 线性无关，并计算 $A_i=\Pi_iQ_B^{\top}$。因此，矩阵 A_iQ_B 以概率为 1 线性无关。

当 $B \neq \emptyset$ 和 $N \neq \emptyset$时，我们所描述的框架是正确的。对于 $B = \emptyset$ 和/或 $N = \emptyset$的情况，可以构造一个简单的流程，例如算法 9 中所述的流程，以生成具有预定最优划分的问题。

算法 9 生成具有特定最大互补解的半定优化问题，当 $B = \emptyset$

选择维度 m ，n 其中 $m < \frac{n(n+1)}{2}$
选择 $n_N \in [n]$
为 $i \in [n_N]$生成 $\lambda_i > 0$
生成正交归一矩阵 Q
构造 $X^* = 0$ 和 $S^* = Q \begin{pmatrix} 0 & 0 \\ 0 & \text{diag}(\lambda) \end{pmatrix} Q^\top$
生成 $A_i = Q\Gamma_i Q^\top$，使得 $\Gamma_i = \text{diag}(\gamma_i)$，其中 $\gamma_{Ti} = 0$，且 $\gamma_{Ni} \in \mathbb{R}^{n_N}$对于 $i \in [m]$
生成 $y^* \in \mathbb{R}^m$
令 $b = 0$ 并计算 $C = \sum_{i=1}^m y_i^* A_i + S^*$
返回半定优化问题 $(A_1, \ldots, A_m, b, C)$，其最优解为 (X^*, y^*, S^*)

3.2.6. 具有预定义的最优解和内点解的半定优化问题（一般结构）

我们还可以将算法 7 推广，以生成具有内点解的半定优化问题，所得方案如算法10所示。在这里，通过使用特征值分解的逆，生成的最优解和内点解都具有一般结构；从整体上看，该方案可以视为算法6和算法7的组合。

算法10 生成具有特定内点解和最优解的半定优化问题

选择维数 m, n，满足 $m < n_{(n+1)}$（生成的半定优化问题的维数：$m+1$, $n+1$）选择 $n_B, n_N \in [n]$，其中 $n_B + n_N \leq n$ 定义集合

$$B = \{1, \ldots, n_B\}, T = \{n_B + 1, \ldots, n - n_N\}, \text{ and } N = \{n - n_N + 1, \ldots, n\}$$

为 $i \in B$ 生成 $\sigma_i > 0$，并构造 $\Sigma_B = \mathsf{diag}(\sigma)$

为 $i \in N$ 生成 $\lambda_i > 0$，并构造 $\Lambda_N = \mathsf{diag}(\lambda)$

生成正交归一矩阵 $\hat{Q}_{n \times n}$

Build $\hat{X} = \hat{Q} \begin{pmatrix} \Sigma_B & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \hat{Q}^\top$ and $\hat{S} = \hat{Q} \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \Lambda_N \end{pmatrix} \hat{Q}^\top$

为 $i \in [m]$ 生成 $\hat{y} \in \mathbb{R}^m$ 和 $A_i \in \mathcal{S}^n$

计算 $i \in [m]$ 的 $b_i = \hat{A}_i \bullet X^*$ 以及 $C = \sum_{i=1}^m \hat{y}_i \hat{A}_i + \hat{S}$ 构造 $Q_{(n+1)\times(n+1)} = \begin{pmatrix} \hat{Q} & 0 \\ 0 & 1 \end{pmatrix}$ 生成正对角矩阵 $\Sigma_B^0, \Sigma_T^0, \Sigma_N^0$ 以及数值 $\sigma_{n+1}^0 > 0$ 生成正对角矩阵 Λ_B, Λ_T 以及 Λ_N 计算 $\delta = \sum_{i \in B} (\sigma_i - \sigma_i^0) \lambda_i^0 + \sum_{i \in N} \sigma_i^0 (\lambda_i^0 - \lambda_i)$ 生成 $\lambda_{n+1}^0 > (\frac{\delta}{\sigma_{n+1}^0})^+$，并计算

Build $X^* = Q \begin{pmatrix} \Sigma_B & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} Q^\top$ and $X^0 = Q \begin{pmatrix} \Sigma_B^0 & 0 & 0 & 0 \\ 0 & \Sigma_T^0 & 0 & 0 \\ 0 & 0 & \Sigma_N^0 & 0 \\ 0 & 0 & 0 & \sigma_{n+1}^0 \end{pmatrix} Q^\top$

Build $S^* = Q \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \Lambda_N & 0 \\ 0 & 0 & 0 & \lambda_{n+1} \end{pmatrix} Q^\top$ and $S^0 = Q \begin{pmatrix} \Lambda_B^0 & 0 & 0 & 0 \\ 0 & \Lambda_T^0 & 0 & 0 \\ 0 & 0 & \Lambda_N^0 & 0 \\ 0 & 0 & 0 & \lambda_{n+1}^0 \end{pmatrix} Q^\top$

随机地生成 $y^0 \in \mathbb{R}^{m+1}$，使得 $y_{m+1}^0 \neq 0$，并构建 $y_{(m+1)}^* = \begin{pmatrix} \hat{y} \\ 0 \end{pmatrix}$

Calculate $\alpha_i = \frac{1}{\sigma_{n+1}^0} \operatorname{tr} \left(\hat{A}_i \hat{Q} \begin{pmatrix} \Sigma_B - \Sigma_B^0 & 0 & 0 \\ 0 & -\Sigma_T^0 & 0 \\ 0 & 0 & C - \Sigma_N^0 \end{pmatrix} \hat{Q}^\top \right)$ for $i \in [m]$

为 $i \in [m]$ 构造 $A_i = \begin{pmatrix} A_i & 0 \\ 0 & \alpha_i \end{pmatrix}$ 计算 $C = \begin{pmatrix} 0 & \sum_{i=1}^m \hat{y}_i \alpha_i + \lambda_{m+1} \end{pmatrix}$ 计算 $b_i = b_i$，使其满足 $i \in [m]$ 和 $b_{m+1} = \operatorname{tr}(\Lambda_{B_{m+1}} X^*)$ 返回 SDOP (A_i, b, C)，其最优解为 (X^*, y^*, S^*)，内点解为 (X^0, y^0, S^0)

$$A_{m+1} = \frac{1}{y_{m+1}^0} \left(\sum_{i=1}^m (y_i - y_i) A_i + Q \begin{pmatrix} \Lambda_B^0 - \Lambda_B & 0 & 0 & 0 \\ 0 & -\Lambda_T^0 & 0 & 0 \\ 0 & 0 & \Lambda_T - \Lambda_T^0 & 0 \\ 0 & 0 & 0 & \Lambda_{n+1} - \Lambda_{n+1}^0 \end{pmatrix} Q^\top \right)$$

对于生成的解 (X^0, y^0, S^0) 和 (X^*, y^*, S^*)，步骤 14 和 15 确保

$$\sum_{i\in B}(\sigma_i^0 - \sigma_i)\lambda_i^0 + \sum_{i\in T}\sigma_i^0\lambda_i^0 + \sum_{i\in N}\sigma_i^0(\lambda_i^0 - \lambda_i) + \sigma_{n+1}^0(\lambda_{n+1}^0 - \lambda_{n+1}) = 0, \qquad (5)$$

因此，正交条件 (3) 得以满足。与块对角情形类似，定理 3.5 确立了所生成的半定优化问题及其最优解和内点解的正确性。

定理 3.5. 设 (X^0, y^0, S^0) 和 (X^*, y^*, S^*) 是由算法10生成的解。那么，

$$\begin{aligned} X^* \succeq 0,\ S^* \succeq 0,\ X^0 \succ 0,\ S^0 \succ 0,& \\ X^* \bullet S^* &= 0, \\ A_i \bullet X^* &= b_i, \\ \sum_{i=1}^{n} y_i^* A_i^\top + S^* &= C, \\ A_i \bullet X^0 &= b_i, \\ \sum_{i=1}^{n} y_i^0 A_i^\top + S^0 &= C. \end{aligned}$$

证明。与基于算法10的步骤和条件 (3) 对定理 3.2 的证明类似。

□

3.2.7. 具有预定义内部解和最大互补解的半定优化问题（一般结构）

为了获得预定最优划分，我们开发了算法 11，用于生成具有特定内点解和最大互补解的半定优化问题，如下所示。

算法 11 生成具有特定内点解和最大互补解的半定优化问题

选择维度 m, n，满足 $m < \frac{n(n+1)}{2}$（生成的半定优化问题的维度：$m+1$, $n+1$）选择 $n_B, n_N \in [n]$，满足 $n_B + n_N \leq n$ 定义集合

$$B = \{1, \ldots, n_B\}, T = \{n_B + 1, \ldots, n - n_N\}, \text{ and } N = \{n - n_N + 1, \ldots, n\}$$

为 $i \in B$ 生成 $\sigma_i > 0$，并构造 $\Sigma_B = \mathbf{diag}(\sigma)$

为 $i \in N$ 生成 $\lambda_i > 0$，并构造 $\Lambda_N = \mathbf{diag}(\lambda)$

生成正交归一矩阵 $\hat{Q}_{n \times n}$

构造 $\hat{X} = \hat{Q} \begin{pmatrix} \Sigma_B & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \hat{Q}^\top$ 和 $\hat{S} = \hat{Q} \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \Lambda_N \end{pmatrix} \hat{Q}^\top$ 生成

$A_1 = Q \Gamma Q^\top$，使得 $\Gamma = \mathbf{diag}(\gamma)$，其中 $\gamma_B = 0$、$\gamma_T > 0$ 和 $\gamma_T \in \mathbb{R}^q$ 为

$i \in \{2, \ldots, m\}$ 生成 $\hat{y} \in \mathbb{R}^m$ 和 $\hat{A}_i \in \mathcal{S}^n$ 计算 $i \in [m]$ 的 $\hat{b}_i = \mathbf{tr}(\hat{A}_i X^*)$，以及 $\hat{C} = \sum_{i=1}^m \hat{y}_i \hat{A}_i + \hat{S}$ 构造 $Q_{(n+1)\times(n+1)} = \begin{pmatrix} \hat{Q} & 0 \\ 0 & 1 \end{pmatrix}$ 生成正对角矩阵

Σ_B^0、Σ_T^0、Σ_N^0 和数值 $\sigma_{n+1}^0 > 0$

Λ_B Λ_T Λ_N：生成正对角矩阵，，和：计

算 $\delta = \sum_{i \in B} (\sigma_i^0 \lambda_i^0 + \sum_{i \in T} \sigma_i^0 \lambda_i^0 + \sum_{i \in N} \sigma_i^0 (\lambda_i^0 - \lambda_i)$ 生成 $\lambda_{n+1}^0 > (\frac{-\delta}{\sigma_{n+1}^0})^+$，

Build $X^* = Q \begin{pmatrix} \Sigma_B & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_{n+1} \end{pmatrix} Q^\top$ and $X^0 = Q \begin{pmatrix} \Sigma_B^0 & 0 & 0 & 0 \\ 0 & \Sigma_T^0 & 0 & 0 \\ 0 & 0 & \Sigma_N^0 & 0 \\ 0 & 0 & 0 & \sigma_{n+1}^0 \end{pmatrix} Q^\top$，

并计算 $\lambda_{n+1} = \frac{\delta}{\sigma_{n+1}^0} + \lambda_{n+1}^0$ 生成 $y^0 \in \mathbb{R}^{m+1}$，使得 $y_{m+1}^0 = 0$，并构造

$y_{(m+1)}^* =$ y 计算 $i \in [m]$ 的 $\alpha_i = \frac{1}{\sigma_{n+1}^0} \mathbf{tr} \left(\hat{A}_i \hat{Q} \begin{pmatrix} \Sigma_B^0 & 0 & 0 \\ 0 & -\Sigma_T^0 & 0 \\ 0 & 0 & -\Sigma_N^0 \end{pmatrix} \hat{Q}^\top \right)$

0 为 $i \in [m]$ 构造 $A_i = \begin{pmatrix} A_i & 0 \\ 0 & \alpha_i \end{pmatrix}$ 计算 $C = \begin{pmatrix} \hat{C} & 0 \\ 0 & 0 \end{pmatrix} - \sum_{i=1}^m y_i^0 A_i$

Build $S^* = Q \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \Lambda_N & 0 \\ 0 & 0 & 0 & \lambda_{n+1} \end{pmatrix} Q^\top$ and $S^0 = Q \begin{pmatrix} \Lambda_B^0 & 0 & 0 & 0 \\ 0 & \Lambda_T^0 & 0 & 0 \\ 0 & 0 & \Lambda_N^0 & 0 \\ 0 & 0 & 0 & \lambda_{n+1}^0 \end{pmatrix} Q^\top$

$y_i \alpha_i + \lambda_{m+1}$ 计算 $i \in [m]$ 的 $b_i = b_i$ 以及 $b_{m+1} = \mathbf{tr}(A_{m+1} X^*)$ 返回半定

优化问题 $(A_1, \ldots, A_m, b, C)$，其最优解为 (X^*, y^*, S^*)，内点解为 (X^0, y^0, S^0)

$$A_{m+1} = \frac{1}{y_{m+1}^0} \left(\sum_{i=1}^m (\hat{y}_i - y_i^0) \hat{A}_i + Q \begin{pmatrix} -\Lambda_B^0 & 0 & 0 & 0 \\ 0 & -\Lambda_T^0 & 0 & 0 \\ 0 & 0 & \Lambda_T - \Lambda_T^0 & 0 \\ 0 & 0 & 0 & \Lambda_{n+1} - \Lambda_{n+1}^0 \end{pmatrix} Q^\top \right)$$

定理 3.6。对于算法 11，解 (X^*, y^*, S^*) 是生成问题 $(A_1, \ldots, A_m, b, C)$ 的最大互补最优解。

证明。该证明与定理 3.3 的证明基本相同；唯一的区别在于，我们通过添加一行和一列扩展了矩阵 A_i 。构造矩阵 A_1 时，新增的特征值 $\gamma_{n+1} = \alpha_1$ 和 $Q_{n+1} = (0, 0, 0, \ldots, 0, 1)^\top$ 属于划分 N ，对此我们没有任何限制。因此，将定理 3.3 的证明调整用于本定理十分直接。 □

在所有提出的半定优化问题生成器中，算法 11 能生成最复杂的半定优化问题，并以一般方式得到最大互补解和内点解；它还提供了调整最优解的特征、最优划分、矩阵 A_i、 C以及向量 b 的机会，从而能够通过细致而复杂的分析研究求解方法的性能。然而，与其他提出的生成器相比，该算法所需的计算复杂得多。

4. 二阶锥优化

在结束之前，我们将用于线性优化和半定优化的技术调整后，应用于二阶锥（或洛伦兹锥）上的线性优化问题。

4.1. 二阶锥优化问题

二阶锥定义如下

$$\left\{(x_1, x_2, \ldots, x_n) \in \mathbb{R}^n : x_1^2 - \sum_{i=2}^{n} x_i^2 \geq 0, x_1 \geq 0\right\}.$$

请注意，根据上述定义，$(x_1, x_2, \ldots, x_n)$ 是二阶锥，当且仅当矩阵

$$\begin{pmatrix} x_1 & x_{2:n}^\top \\ x_{2:n} & x_1 I_{n-1} \end{pmatrix}$$

是半正定的，其中 $x_{2:n}^\top \equiv (x_2, x_3, \ldots, x_n)$，且 I_{n-1} 是 $n-1$ 阶单位矩阵。因此，原始或对偶二阶锥优化问题（SOCOP）可视为半定优化 [19] 的一个特例。

在二阶锥优化问题中，我们旨在可行域上最小化线性目标函数；该可行域由仿射空间与 p 个维度为 n_i 的二阶锥的笛卡尔积的交集定义，其定义如下

$$\mathbb{L}^n = \mathcal{L}^{n_1} \times \cdots \times \mathcal{L}^{n_p}, \quad n = \sum_{i=1}^{p} n_i,$$

其中

$$\mathcal{L}^{n_i} = \{x^i = (x_1^i, \ldots, x_{n_i}^i)^\top \in \mathbb{R}^{n_i} : x_1^i \geq \|x_{2:n_i}^i\|\}, \quad i \in [p].$$

显然，线性优化问题是二阶锥优化问题的特例，其中 $n_i = 1$ 适用于 $i \in [p]$。

标准形式的原始和对偶二阶锥优化问题表示为

$$z^P_{SOCO} = \inf_x \{c^\top x : Ax = b\ ,\ x \in \mathbb{L}^n\},$$
$$z^D_{SOCO} = \sup_{(y,s)} \{b^\top y : A^\top y + s = c\ ,\ s \in \mathbb{L}^n\},$$

其中 $b \in \mathbb{R}^m$, $A = (A_1, \ldots, A_p)$, $x = (x^1; \ldots; x^p)$, $s = (s^1; \ldots; s^p)$ 和 $c = (c^1; \ldots; c^p)$, 其中 $A_i \in \mathbb{R}^{m \times n_i}$、$s^i \in \mathbb{R}^{n_i}$ 和 $c^i \in \mathbb{R}^{n_i}$ 对于 $i \in [p]$。原始和对偶可行解集定义为

$$\mathcal{PD}_{SOCO} = \{(x, y, s) \in \mathbb{L}^n \times \mathbb{R}^m \times \mathbb{L}^n : Ax = b, A^\top y + s = c\}.$$

Let

$$\mathcal{L}^{n_i}_+ = \{x^i \in \mathcal{L}^{n_i} : x^i_1 > \|x^i_{2:n_i}\|\}, \quad i \in [p],$$

于是，我们可以将原始和对偶内点可行解集定义为

$$\mathcal{PD}^0_{SOCO} = \{(x, y, s) \in \mathbb{L}^n_+ \times \mathbb{R}^m \times \mathbb{L}^n_+ : Ax = b, A^\top y + s = c\}.$$

与线性优化和半定优化一样，通常假设存在内点可行的原始-对偶解。若存在严格可行解，则可知内点条件（IPC）成立 [14], ，从而保证 $z^P_{SOCO} = z^D_{SOCO}$ 以及原始-对偶最优解集

$$\mathcal{PD}^*_{SOCO} = \left\{(x, y, s) \in \mathcal{PD}_{SOCO}\ :\ c^\top x = z^P_{SOCO} = b^T y = z^D_{SOCO}\right\},$$

非空且有界。因此，存在一个对偶间隙为零的原始-对偶最优对。也就是说，对于最优解 x^* 和 (y^*, s^*)，有

$$x^* \circ s^* = (x^1 \circ s^1, \ldots, x^p \circ s^p) = 0, \tag{7}$$

其中，约旦乘积“$\circ$”定义为

$$x^i \circ s^i = \begin{pmatrix} (x^i)^\top s^i \\ x^i_1 s^i_{2:n_i} + s^i_1 x^i_{2:n_i} \end{pmatrix}. \tag{8}$$

如果 $x^* \in \mathrm{ri}(\mathcal{P}^*_{SOCO})$ 且 $(y^*; s^*) \in \mathrm{ri}(\mathcal{D}^*_{SOCO})$，则称最优解 (x^*, y^*, s^*) 为最大互补。此外，如果

$$x^* + s^* \in \mathbb{L}^n_+.$$

4.2. 二阶锥优化问题的实例生成器

受我们在线性优化问题和半定优化问题生成器方面工作的启发，我们进一步希望将这些思路用于生成二阶锥优化问题。由于二阶锥优化可被视为半定优化的一个特例，Sampourmahani等人 [19] 研究了半定优化问题与二阶锥优化问题之间的映射及其最优划分。开发二阶锥优化问题生成器

很直接，只需在所提出的半定优化问题生成器基础上加入适当的映射即可。不过，这种方法效率不高，因此我们另行提出了几种不依赖半定优化表示的二阶锥优化问题生成器。

4.2.1. 具有预定义内部解的二阶锥优化问题

生成具有内点解的二阶锥优化问题，也能确保问题存在一个对偶间隙为零的最优解。算法 12 是对算法 1 和算法 4 的修改，用于生成具有特定内点解的二阶锥优化问题。

算法 12 生成具有特定内点解的二阶锥优化问题

1: 选择维数 $m < n$ 2: 选择 $n_1, \ldots, n_p$，使得 $n = n_1 + \cdots + n_p$ 3: 生成 (x^0, s^0)，使得 $x^0 \in \mathbb{L}_+^n$ 且 $s^0 \in \mathbb{L}_+^n$ 4: 生成 $A \in \mathbb{R}^{m \times n}$ 5: 生成 $y^0 \in \mathbb{R}^m$ 6: 计算 $b = Ax^0$ 和 $c = A^\top y^0 + s^0$ 7: 返回具有内点解 (x^0, y^0, s^0) 的二阶锥优化问题 (A, b, c)

为了得到内点解 x^0，我们必须针对 $i = 1, \ldots, p$ 生成 $(x^0)^i$，使得 $((x^0)_2^i, \ldots, (x^0)_{n_i}^i) \in \mathbb{R}^{n_i - 1}$ 且 $(x^0)_1^i > \|(x^0)_{2:n_i}^i\|$。生成此类解的一种方法是生成 $(x^0)^i \in \mathbb{R}^{n_i}$，并根据以下规则对其进行更新

$$(x^0)_1^i = \|(x^0)_{2:n_i}^i\| + \|(x^0)_1^i\|.$$

与线性优化类似，如果矩阵 A 是随机生成的，那么 A 的所有行线性无关的概率为 1。此外，用户可以生成具有特定特征（如稀疏性、条件数和范数）的目标矩阵 A。

4.2.2. 具有预定义最优解的二阶锥优化问题

对于二阶锥优化问题，其最优划分比线性优化和半定优化的情形稍复杂。指标集 $[p]$ 被划分为集合 $(\mathcal{B}, \mathcal{N}, \mathcal{R}, \mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$，定义如下：

$$\mathcal{B} := \{i : x_1^i > \|x_{2:n_i}^i\|_2, \text{ 对于某个 } x \in \mathcal{P}_{SOCO}^*\},\ \mathcal{N} := \{i : x_1^i > \|x_{2:n_i}^i\|_2, \text{ 对于某个 } x \in \mathcal{D}_{SOCO}^*\},\ \mathcal{R} := \{i : x_1^i = \|x_{2:n_i}^i\|_2 > 0, x_1^i = \|x_{2:n_i}^i\|_2 > 0, \text{ 对于某个 } (x, y, x) \in \mathcal{P}_{SOCO}^* \times \mathcal{D}_{SOCO}^*\},\ \mathcal{T}_1 := \{i : x^i = x^i = 0, \text{ 对于所有 } (x, y, x) \in \mathcal{P}_{SOCO}^* \times \mathcal{D}_{SOCO}^*\},$$
$$\mathcal{T}_2 := \{i : x^i = 0, \text{ 对于所有 } (y, x) \in \mathcal{D}_{SOCO}^*, x_1^i = \|x_{2:n_i}^i\|_2 > 0, \text{ 对于某个 } x \in \mathcal{P}_{SOCO}^*\},$$
$$\mathcal{T}_3 := \{i : x^i = 0, \text{ 对于所有 } x \in \mathcal{P}_{SOCO}^*, \ x_1^i = \|x_{2:n_i}^i\|_2 > 0, \text{ 对于某个 } (y, x) \in \mathcal{D}_{SOCO}^*\}.$$

关于二阶锥优化问题中最优划分的进一步讨论，请参阅 [23]。由此，我们可以提出算法 13，它是算法 2 的推广，用于生成具有特定最优解的随机二阶锥优化问题。

算法 13 生成具有特定最优解的二阶锥优化问题

1: 选择维数 $m < n$2: 选择 $n_1, \ldots, n_p$，使得 $n = n_1 + \cdots + n_p$3: 将指标集 $[p]$ 划分为 (B, N, R, T_1, T_2, T_3) 4: 对于 $i \in B$，$(s^*)^i = 0$ 并生成 $(x^*)^i \in \mathbb{R}^{n_i}$，使得 $(x^*)^i_1 > \|(x^*)^i_{2:n_i}\|$5: 对于 $i \in N$，$(x^*)^i = 0$ 并生成 $(s^*)^i \in \mathbb{R}^{n_i}$，使得 $(s^*)^i_1 > \|(s^*)^i_{2:n_i}\|$6: 对于 $i \in T_1$，$(s^*)^i = 0$ 和 $(x^*)^i = 0$7: 对于 $i \in T_2$，$(s^*)^i = 0$ 并生成 $(x^*)^i \in \mathbb{R}^{n_i}$，使得 $(x^*)^i_1 = \|(x^*)^i_{2:n_i}\| > 0$8: 对于 $i \in T_3$，$(x^*)^i = 0$ 并生成 $(s^*)^i \in \mathbb{R}^{n_i}$，使得 $(s^*)^i_1 = \|(s^*)^i_{2:n_i}\| > 0$9: 对于 $i \in R$，生成 $(x^*)^i_{2:n_i} \in \mathbb{R}^{n_i - 1}$ 和 $\delta \in \mathbb{R}$ 并构造

$$(x^*)^i = \begin{pmatrix} \|(x^*)^i_{2:n_i}\| \\ (x^*)^i_{2:n_i} \end{pmatrix}, \text{ and } (s^*)^i = \delta \begin{pmatrix} \|(x^*)^i_{2:n_i}\| \\ -(x^*)^i_{2:n_i} \end{pmatrix}$$

10: 生成 $y^* \in \mathbb{R}^m$11: 生成 $A \in \mathbb{R}^{m \times n}$12: 计算 $b = Ax^*$ 和 $c = A^\top y^* + s^*$13: 返回二阶锥优化问题 (A, b, c)，其最优解为 (x^*, y^*, s^*)

注 14. 算法 13 给出一个具有最优解的二阶锥优化问题，但该最优解不一定是最大互补解。因此，生成问题的最优划分 $(\mathcal{B}, \mathcal{N}, \mathcal{R}, \mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$ 可能不同于 (B, N, R, T_1, T_2, T_3)，而我们只能得到

$$B \subseteq \mathcal{B},\ N \subseteq \mathcal{N},\ R \subseteq \mathcal{R},\ \mathcal{T}_1 \subseteq T_1,\ \mathcal{T}_2 \subseteq T_2 \cap T_1,\ \text{and } \mathcal{T}_3 \subseteq T_3 \cap T_1.$$

与线性优化问题和半定优化问题生成器类似，可以通过随机生成 A，使其具有特定特征。(x^*, y^*, s^*) 的范数和性质可直接控制。此外，可以通过适当且谨慎地缩放 (x^*, y^*, s^*)，预先设定 (b, c) 的范数，因为只要算法 13 第 11 行的方程成立，就可以同时确定所有参数和最优解的范数。容易看出，(x^*, y^*, s^*) 是最优解，因为 $x^* \circ s^* = 0$，且根据构造它是可行的。下一节将讨论如何生成具有最大互补解的二阶锥优化问题。

4.2.3. 具有预定义的最大互补解的二阶锥优化问题

由于最优划分会影响求解与半定优化问题类似的二阶锥优化问题的算法性能，我们希望生成具有预先确定的最优划分的问题。为此，我们将针对具有最大互补解的半定优化问题设计的实例生成器改编为算法 14 中的二阶锥优化。令 $A^p_{i,\,j}$ 表示 A 中与锥 p 对应的部分（列）中的第i 行、第 j 列元素。我们还用上标表示划分，例如，A^B 表示 A 中与划分 B对应的列。

算法 14 生成具有特定最大互补解的二阶锥优化问题

1: 选择维度 $m < n$2: 选择 $n_1, \ldots, n_p$，使得 $n = n_1 + \cdots + n_p$
3: 将指标集 $[p]$ 划分为 (B, N, R, T_1, T_2, T_3)，使得

$$|T_2| + 1 < m \leq |B| + |R| + |T_2|$$

4: 对于 $i \in B$，$(s^*)^i = 0$ 并生成 $(x^*)^i \in \mathbb{R}^{n_i}$，使得 $(x^*)^i_1 > \|(x^*)^i_{2:n_i}\|$5: 对于 $i \in N$，$(x^*)^i = 0$ 并生成 $(s^*)^i \in \mathbb{R}^{n_i}$，使得 $(s^*)^i_1 > \|(s^*)^i_{2:n_i}\|$6: 对于 $i \in T_1$，$(s^*)^i = 0$ 和 $(x^*)^i = 0$7: 对于 $i \in T_2$，$(s^*)^i = 0$ 并生成 $(x^*)^i \in \mathbb{R}^{n_i}$，使得 $(x^*)^i_1 = \|(x^*)^i_{2:n_i}\| > 0$8: 对于 $i \in T_3$，$(x^*)^i = 0$ 并生成 $(s^*)^i \in \mathbb{R}^{n_i}$，使得 $(s^*)^i_1 = \|(s^*)^i_{2:n_i}\| > 0$9: 对于 $i \in R$，生成 $(x^*)^i_{2:n_i} \in \mathbb{R}^{n_i - 1}$ 和 $\delta \in \mathbb{R}$ 并构造

$$(x^*)^i = \begin{pmatrix} \|(x^*)^i_{2:n_i}\| \\ (x^*)^i_{2:n_i} \end{pmatrix}, \text{ and } (s^*)^i = \delta \begin{pmatrix} \|(x^*)^i_{2:n_i}\| \\ -(x^*)^i_{2:n_i} \end{pmatrix}$$

10: 生成 $y^* \in \mathbb{R}^m$
11: 生成 $A \in \mathbb{R}^{m \times n}$，使得

- 第一行:

$$\begin{aligned} A^p_{1,1} > 0, A^p_{1,j} = 0 \text{ for } j = 2, \ldots, n_p, \text{ and } p \in T_1 \cup T_3 \\ A^p_{1,j} = 0 \text{ for } j = 1, \ldots, n_p, \text{ and } p \in B \cup R \cup T_2 \\ A^p_{1,j} \in \mathbb{R} \text{ for } j = 1, \ldots, n_p, \text{ and } p \in N \end{aligned}$$

- 第 2 行至 $|T_2| + 1$:

$$A^p_{p,1:n_p} = \begin{bmatrix} -1 & \frac{(x^{*k}_{2:n_p})^\top}{\|x^{*k}_{2:n_p}\|} \end{bmatrix}, A^p_{k,1:n_k} = 0 \text{ for all } k \neq q, \text{ and } p \in T_2$$

- 应生成其余各行，使得 $\text{rank}([A^B, A^R, A^{T_2}]) = m$。

12: 计算 $b = Ax^*$ 和 $c = A^\top y^* + s^*$13: 返回具有最大互补解（x^*, y^*, s^*）的二阶锥优化问题 (A, b, c)

与算法 13 相比，算法 14 对 A 的生成方式施加了更多限制。

定理 4.1. 对于算法 14 生成的任意二阶锥优化问题 (A, b, c)，所生成的最优解 (x^*, y^*, s^*) 是最大互补的。

证明。可以验证，$b_1 = 0$，而 A 的第一行则强制任意最优解 $\bar{x}$ 满足

$$\bar{x}^p = 0 \text{ for all } p \in T_1 \cup T_3.$$

从约束 2 到 $|T_2| + 1$，我们为划分 T_2中的每个锥 p 添加一个约束：除锥 p中的变量外，所有变量的系数均为零。由于相应的右端项为零，且系数是该点处锥 $|T_2| + 1$的法向量 p

对于锥 p，在点 x^*处，所有可行解都必须位于一条射线上；该射线位于锥 p 的边界上并经过点 x^*。因此，对于任意最优解 $\bar{x}$，我们有

$$\bar{x}_1^p = \|x_{2:n_p}^p\| \text{ for all } p \in T_2.$$

到目前为止，我们已证明 $x^* \in ri(\mathcal{P}^*)$，证明的最后一部分是确立对偶问题具有唯一的最优解 (y^*, s^*)。为证明这一点，假设它还有另一个最优解 $(\bar{y}, \bar{s})$，且 $X^* = \text{diag}(x^*)$。于是，我们有

$$X^* A^\top (\overline{y} - y^*) = -X^* (\overline{s} - s^*) = 0.$$

由于 A 的生成方式使得 $X^* A^\top$ 的秩为 m，因此我们有 $\overline{y} = y^*$。由此可得，(x^*, y^*, s^*) 是所生成问题的极大互补解。

□

推论 4.2. 对于算法 14 生成的任意二阶锥优化问题，其最优划分 $(\mathcal{B}, \mathcal{N}, \mathcal{R}, \mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3)$ 等于 (B, N, R, T_1, T_2, T_3)。

不出所料，生成具有预定义最优划分的二阶锥优化问题，会限制矩阵 A 的生成方式。不过，A 的某些分量不受限制，因此用户可以控制 A 的性质。下文将对此进行讨论。

4.2.4. 具有最优解和内点解的二阶锥优化问题

通过向矩阵 A 添加一行和一列，我们可以扩展算法 13，同时提供特定的内点解和最优解。我们的目标是以一般方式生成最优解和内点解，但需要满足正交条件：

$$(x^0 - x^*)^\top (s^0 - s^*) = 0. \tag{9}$$

请注意，这是一个自然的要求；与线性优化问题类似，我们有 $(x^* - x^0) \in \text{Lin}^\perp(A)$ 和 $(s^* - s^0) \in \text{Lin}(A)$。

算法 15 生成具有特定内点和最优解的二阶锥优化问题

1: 选择维数 $m < n$ （生成的二阶锥优化问题的维数： $m+1, n+1$） ˆ ˆ2: 通过算法 13 生成具有最优解 $(\hat{x}, \hat{y}, \hat{s})$ 的 $(A, b, \hat{c})$（维度： $p \times m$） 3: 生成 $x^0 \in \mathbb{L}^{n_1,\ldots,}_{+}{}^{n_p+1}$，并令 $\hat{x}^0 = ((x^0)^1; \ldots; (x^0)^p_{1:n_p})$ 4: 构造 $x^* \in \mathbb{L}^{n_1,\ldots,}_{+}{}^{n_p+1}$，使得

$$(x^*)^{1:p-1} = (\hat{x})^{1:p-1},\ (x^*)^p_{1:n_p} = (\hat{x})^p_{1:n_p}, \text{ and } (x^*)^p_{n_p+1} = 0$$

5: Calculate vector $\alpha = \frac{1}{(x^0)^p_{n_p+1}} \hat{A}(\hat{x} - \hat{x}^0)$
6: Build $\tilde{A} = (\hat{A}, \alpha)$ (concatenation of a column to matrix)
7: Generate $y^0 \in \mathbb{R}^{m+1}$ such $y^0_{m+1} \neq 0$, and build $y^* = (\hat{y}, 0)^\top \in \mathbb{R}^{m+1}$
8: Generate $\hat{s}^0 \in \mathbb{L}_{+}^{n_1,\ldots,n_p}$
9: Let $\delta = (\hat{x}^0 - \hat{x})^\top (\hat{s}^0 - \hat{s})$, and generate $(s^0)^p_{n_p+1} > (\frac{-\delta}{(x^0)^p_{n_p+1}})^+$
10: Calculate $\hat{s}^p_{n_p+1} = \frac{\delta}{(x^0)^p_{n_p+1}} + (s^0)^p_{n_p+1}$
11: Build $s^* = (\hat{s}, \hat{s}^p_{n_p+1})^\top$, $s^0 = (\hat{s}^0, (s^0)^p_{n_p+1})^\top$;
12: Calculate vector $\beta = \frac{1}{y^0_{m+1}} (\tilde{A}^\top (\hat{y} - (y^0)_{1:m}) + s^* - s^0)$
13: Build $A = \begin{pmatrix} \tilde{A} \\ \beta^\top \end{pmatrix}$ (concatenation of a row to matrix)
14: Calculate $b = Ax^*$ and $c = A^\top y^* + s^*$
15: **Return** SOCOP (A, b, c) with interior solution (x^0, y^0, s^0) and optimal solution (x^*, y^*, s^*)

定理4.4表明，(x^0, y^0, s^0) 和 (x^*, y^*, s^*) 所声称的性质确实成立。在给出定理4.4之前，我们需要验证所生成解的正交性。

Lemma 4.3. 对于算法 15 生成的任意 (x^0, y^0, s^0) 和 (x^*, y^*, s^*)，有

$$(x^0 - x^*)^\top (s^0 - s^*) = 0.$$

证明。与引理 2.1 的证明类似，算法 15 的步骤 9 和 11 确保正交条件成立。

□

利用引理4.3，下面的定理表明，生成问题满足所需的性质。

The定理 4.4. 设 (x^0, y^0, s^0) 和 (x^*, y^*, s^*) 由算法 15 生成。则 ,

$$\begin{aligned} x^* \circ s^* &= 0, \\ Ax^* &= b, \\ A^\top y^* + s^* &= c, \\ Ax^0 &= b, \\ A^\top y^0 + s^0 &= c, \\ x^* \in \mathbb{L}^{n+1},\ s^* \in \mathbb{L}^{n+1},\ x^0 \in \mathbb{L}^{n+1}_{+},\ s^0 &\in \mathbb{L}^{n+1}_{+}. \end{aligned}$$

也就是说，对于生成的二阶锥优化问题 (A, b, c)，(x^0, y^0, s^0) 和 (x^*, y^*, s^*) 分别是内点解和最优解。

证明。该证明与定理 2.2 的证明类似。 □

与半定优化问题生成器相比，二阶锥优化问题生成器在计算上更简单，因为它们不需要生成随机正交归一矩阵或半正定矩阵。设 t_R 为随机生成一个数所需的算术运算次数。为了生成正交归一矩阵或半正定矩阵，我们需要使用分解方法；如附录所述，该方法需要 $\mathcal{O}(n^3)$ 次算术运算。一般情况下，线性优化问题和二阶锥优化问题生成器需要 $\mathcal{O}(n^2 t_r)$ 次算术运算，而半定优化生成器需要 $\mathcal{O}(n^3 t_r)$ 次算术运算。需要指出的是，如果我们想在线性优化问题和二阶锥优化问题中生成具有特定条件数的随机矩阵 A，则需要使用分解方法，线性优化问题和二阶锥优化问题生成器的复杂度将增加到 $\mathcal{O}(n^3 t_r)$ 次算术运算。

4.2.5. 具有预定义内部解和最大互补解的二阶锥优化问题

若要生成同时具有内点解和最大互补解的二阶锥优化问题，可以使用算法 15，并在其第一步中使用算法 14，该算法可提供具有最大互补解的二阶锥优化问题。唯一的区别是，我们应选择划分，使最后一个锥属于划分 N。经过这一修改，算法 15 第 6 步中添加的列将属于划分 N，从而满足使 (x^*, y^*, s^*) 保持最大互补所需的所有限制。这样，我们就可以生成具有内点解和预定最优划分的二阶锥优化问题。

5. 实现

所有提到的生成器均已在一个 Python 软件包中实现，该软件包可在 https://github.com/qcol-lu/qipm 开源获取。此软件包支持指定向量的范数、条件数以及矩阵的稀疏性。此外，还实现了多种内点法版本，例如可行/不可行、精确/非精确以及长步/短步/预测校正方法，可供实验使用。此外，还可以选择牛顿系统的求解器。用户可以选择经典或量子线性系统算法。

6. 结论

我们开发并实现了若干适用于线性优化、半定优化和二阶锥优化的随机实例生成器，可生成具有特定最优解和／或内点解的实例。由于这些生成器具有很强的可控性，用户可以利用它们分析问题的不同特征，例如稀疏性和条件数，从而更有针对性地研究不同算法的性能。此外，我们还提出了具有预定义最优划分的半定优化问题和二阶锥优化问题生成器，可用于生成计算上具有挑战性的实例。

所提出的生成器还可用于研究求解线性优化、半定优化和二阶锥优化问题的算法在输入数据、最优解和内点解服从不同概率分布时的平均性能。未来研究方向包括

将这些生成器的构造扩展到其他类别的锥优化、多项式优化和非线性优化问题。

扩展所提出的生成器的一个有益方向，是生成对算法和求解器构成挑战的困难问题，例如原始或对偶无界的线性优化问题。对于半定优化和二阶锥优化，值得探索开发能够生成具有零对偶间隙但无法达到最优解的实例，或具有非零对偶间隙的实例的生成器。

7. 致谢

本研究由美国国防高级研究计划局资助，作为项目 W911NF2010022“量子计算革命与优化：挑战与机遇”的一部分。

参考文献

[1] J.L. Arthur 和 J.O. Frendewey，GENGUB：一种用于生成带有一般上界约束的线性规划的生成器，《计算机与运筹研究》20（1993），第565–573页。[2] S. Bowly、K. Smith-Miles、D. Baatar 和 H. Mittelmann，具有可控性质的线性规划实例生成技术，《数学规划计算》12（2020），第389–415页。[3] P.H. Calamai、L.N. Vicente 和 J.J. J´udice，一种生成二次规划测试问题的新技术，《数学规划》61（1993），第215–231页。[4] E. Castillo、R.E. Pruneda 和 M. Esquivel IV，自动生成用于计算机辅助教学的线性规划问题，《科学与技术中的数学教育国际期刊》32（2001），第209–232页。[5] S. Chakraborty 和 P.P. Choudhury，算法复杂度的统计分析，《应用数学快报》13（2000），第121–126页。[6] A. Charnes、W.M. Raike、J.D. Stutz 和 A.S. Walters，关于线性规划代码测试问题的生成，《美国计算机学会通讯》17（1974），第583–586页。[7] A. Coja-Oghlan、P. Gao、M. Hahn-Klimroth、J. Lee、N. M¨uller 和 M. Rolvien，稀疏随机矩阵的满秩条件，arXiv预印本 arXiv:2112.14090（2021）。[8] C. Cotta 和 P. Moscato，算法复杂度的混合进化统计分析，《应用数学快报》16（2003），第41–47页。[9] E. De Klerk，《半定规划：内点算法与若干应用》，第65卷，施普林格科学与商业媒体，2006年。[10] D.M. Gay，线性规划测试问题的电子邮件分发，《数学规划学会 COAL通讯》13（1985），第10–12页。[11] J.N. Hooker，需要一门算法实证科学，《运筹学》42（1994），第201–212页。[12] D. Klingman、A. Napier 和 J. Stutz，NETGEN：用于生成大规模容量约束指派、运输和最小费用流网络问题的程序，《管理科学》20（1974），第814–821页。[13] F. Mezzadri，如何从经典紧群生成随机矩阵，arXiv预印本 math-ph/0609050（2006）。[14] A. Mohammad-Nezhad，《锥优化：最优划分、参数分析与稳定性分析》，博士论文，利哈伊大学，2019年。[15] A. Mohammad-Nezhad 和 T. Terlaky，关于半定优化最优划分的识别，《INFOR：信息系统与运筹研究》58（2020），第225–263页。

[16] G. Pataki，糟糕的半定规划：它们看起来都一样，SIAM 优化学报27（2017），第146–172页。[17] M.G. Pilcher 和 R.L. Rardin，已知最优值的离散优化问题的部分多面体描述与生成，《海军研究物流》（NRL）39（1992），第839–858页。[18] C. Roos、T. Terlaky 和 J.P. Vial，《线性优化理论与算法：内点法》，Wiley Chichester，1997年。[19] P. Sampourmahani、M. Mohammadisiahroudi 和 T. Terlaky，关于二阶锥优化问题的半定表示，arXiv预印本 arXiv:2301.12007（2023）。[20] K. Smith-Miles 和 S. Bowly，通过在实例空间中演化生成新的测试实例，《计算机与运筹研究》63（2015），第102–113页。[21] L.B. Sokolinsky 和 I.M. Sokolinskaya，FRaGenLP：用于集群计算系统的随机线性规划问题生成器，载于《并行计算技术国际会议》。Springer，2021年，第164–177页。[22] S. Sremac、H.J. Woerdeman 和 H. Wolkowicz，半定规划中的误差界与奇异度，SIAM 优化学报31（2021），第812–836页。[23] T. Terlaky 和 Z. Wang，关于二阶锥优化问题最优划分的识别，SIAM 优化学报24（2014），第385–414页。[24] M.J. Todd，线性规划的概率模型，《运筹学数学》16（1991），第671–693页。[25] H. Wei 和 H. Wolkowicz，生成和度量困难的半定规划实例，《数学规划》125（2010），第31–45页。

8. 附录

这里，我们回顾一些生成随机半正定矩阵和正交矩阵的基本方法，这些矩阵可用于所提出的半定优化问题和二阶锥优化问题生成器。

附录 A. 生成随机半正定矩阵

T这里有几种生成半正定矩阵的方法 $P \in \mathcal{S}$ $^{n}_{+}$.

(1) 生成一个随机矩阵 $A \in \mathbb{R}^{n\times n}$，并计算目标矩阵 $P = AA^{\top}$。如果 A 以概率为 1 满秩，则矩阵 P 以概率为 1 半正定。(2) 更高效的方法是生成一个随机下三角矩阵 $L \in \mathbb{R}^{n\times n}$，并计算目标矩阵 $P = LL^{\top}$。如果 L 的对角元素均非零，则 P 为正定。如果 L 的部分对角元素为零，则 P 为半正定。(3) 生成一个正交归一矩阵 $Q \in \mathbb{R}^{n\times n}$ 和一个正对角矩阵 Λ。然后计算 $P = Q\Lambda Q^{\top}$。如果 Λ 的对角元素均大于零，则 P 为正定。如果 Λ 的对角元素均大于或等于零，则 P 为半正定。(4) 由于生成随机正交归一矩阵的计算成本高，我们可以改为生成一个随机下三角矩阵 $L \in \mathbb{R}^{n\times n}$，其所有对角元素均为 1。然后，我们可以将目标矩阵计算为 $P = LDL^{\top}$，其中 D 是一个元素非负的对角矩阵。

在上述四种方法中，第二种所需的算术运算最少。不过，第三种方法可以确定矩阵 P 的特征值范围，这有助于预先确定矩阵 P的条件数。

附录 B. 生成随机正交矩阵

一种通用方法是生成一个随机矩阵 $A \in \mathbb{R}^{n \times n}$，然后通过格拉姆–施密特过程或 QR 分解中的其他方法（例如改进的格拉姆–施密特方法和豪斯霍尔德方法）将其正交化。随机正交（或酉）矩阵的生成是一个活跃的研究领域，已有许多高效的生成方法，例如参见 [13]。