arXiv:2601.01342v1 [quant-ph] 4 Jan 2026

用于求解线性代数方程的量子 Kaczmarz 算法

Nhat A. Nghiem, [1], * Tuan K. Do, [2], † 和 Trung V. Phan[3], ‡

[1] *Department of Physics and Astronomy, State University of New York at Stony Brook, Stony Brook, NY 11794-3800, USA* [2] *Department of Mathematics, University of California, Santa Barbara, CA 93106, USA* [3] *Department of Natural Sciences, Scripps and Pitzer Colleges,Claremont Colleges Consortium, Claremont, CA 91711, USA*

我们介绍一种基于 Kaczmarz 方法的量子线性系统求解算法。Kaczmarz 方法广泛应用于大型线性方程组和最小二乘问题，通过每次强制满足一个方程来更新解。它简单且内存开销低，因此在数据回归、断层重建和优化等领域都是实用的选择。与许多现有的量子线性求解器不同，我们的方法不依赖于通过预言机访问来查询元素，从而消除了一个关键的实用性瓶颈。具体而言，当所研究系统的秩足够小，且相关矩阵的行具有适当结构时，我们实现了电路复杂度$\mathcal{O}\left(\frac{1}{\varepsilon}\log\ m\right)$，其中 m 是变量数，ε 是目标精度；该复杂度不依赖稀疏度 s，并且有望不显式依赖条件数 κ。这表明，相较于先前的量子线性求解器，我们取得了显著改进；先前方法对 κ, s的依赖至少是线性的。同时，当各行具有任意结构且至多包含s 个非零元素时，我们使用额外的 $\mathcal{O}(s)$个辅助量子比特，获得电路深度 $\mathcal{O}\left(\frac{1}{\varepsilon}\log\ s\right)$，因此深度随稀疏度 s仅呈对数增长。当稀疏度 s 按 $\mathcal{O}(\log\ m)$增长时，与现有量子算法相比，我们的方法在电路深度方面可实现指数级改进，同时使用渐近意义下相同数量的量子比特。

I. 引言

线性系统是科学计算的核心，例如数据回归/拟合 [1], 逆问题，如断层重建 [2, 3], 以及大规模现代优化 [4]。这些任务反复归结为求解大量线性代数方程——这些方程通常是超定的，且受到噪声污染——因此必须高效地大规模处理。在这种场景下，一种强大的经典方法是 *Kaczmarz* 方法 [5]，这是一种轻量级的“行作用”迭代方法，通过一次施加一个方程来逐步修正估计值；在大数据场景中，它以极低的内存开销实现了出色的实际性能。随着数据规模和模型复杂度不断增长 [6, 7], ，这些线性代数子程序日益成为端到端流程中的主要耗时环节，因此，探索能够加速多种应用中这些基础线性求解任务的量子方法正当其时。

用于求解线性代数方程的量子算法已成为最具前景的量子计算机应用之一。首个用于求解稀疏线性系统的量子算法在 [8], 中提出，展示了相较于已知最佳经典算法，在维数方面的指数级提升。具体而言， [8] 证明了矩阵求逆是 BQP 完全的，这凸显了经典计算机不太可能达到量子复杂度。此后，量子线性求解算法在一系列研究中得到了改进和完善。参考文献 [9] 提出了一种量子线性求解器，其对误差容限的依赖实现了指数级改进。参考文献 [10]提出了一种预处理量子线性求解器，能够比 [8, 9]更高效地处理病态线性系统。参考文献 [11] 概述了一种变分量子线性求解器，启发式结果表明其在实践中表现良好。参考文献 [12]构造了一种适用于稠密系统的量子线性求解器。参考文献 [13] 与我们的方法有一定相似之处，因为两种方法都依赖行迭代。正如我们稍后将更明确地展示的，与 [13]相比，我们的方法具有一定优势。

在本工作中，我们提出了经典 Kaczmarz 算法的量子版本。尽管已有研究探讨了 Kaczmarz 方法的量子适配版本以及相关的量子线性求解器 [8–13], 但我们在两个方面超越了以往方法：在相关情形下实现了更快的扩展速度，而且至关重要的是，我们采用了更轻量的输入访问模型，避免查询单个矩阵元素的量子随机存取存储器/预言机——因此，我们算法的性能不依赖于一种数据访问假设，而这种假设很可能是实践中最难实现的部分。本文组织如下。第二节II回顾经典 Kaczmarz 方法，包括其随机迭代和收敛行为。第三节III 介绍量子 Kaczmarz 算法：我们阐述输入访问假设和量子态制备流程（第三节IIIA），通过块编码构造量子行操作更新，随后证明主要的复杂度界（第三节IIIB），并讨论和比较以往的量子线性求解器（第三节IIIC）。附录汇集了块编码/量子奇异值变换工具所需的背景知识（附录A），并给出详细的复杂度分析

* nhatanh.nghiemvu@stonybrook.edu
† ktdo@ucsb.edu
‡ tphan@natsci.claremont.edu

主定理的依据（附录 C）。

II. 经典 KACZMARZ 方法

考虑一个线性系统

$$Ax = b \ , \tag{II.1}$$

其中， A 是一个 $n \times m$ 实值矩阵， b 是一个已知的 n 实值向量，而 x 是待估计的未知 m 实值向量。为方便记号，我们写作

$$b = \begin{pmatrix} b_1 \\ b_2 \\ \dots \\ b_n \end{pmatrix} \in \mathbb{R}^n \ , \ x = \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_m \end{pmatrix} \in \mathbb{R}^m \ ,$$

并将矩阵 A 按行表示为

$$A = \begin{pmatrix} a_1^\top \\ a_2^\top \\ \dots \\ a_n^\top \end{pmatrix} \in \mathbb{R}^{n\times m} \ , \ a_j = \begin{pmatrix} a_{j1} \\ a_{j2} \\ \dots \\ a_{jm} \end{pmatrix} \in \mathbb{R}^m \ ,$$

其中， $^\top$ 表示转置运算，因此，方程（II.1）等价于以下一组 n标量线性方程

$$a_j^\top x = a_{j1}x_1 + a_{j2}x_2 + \dots + a_{jm}x_m = b_j \ .$$

在许多应用中，矩阵 A 不必是方阵。特别值得关注的是矩形系统，其中方程的数量不少于未知数的数量，即 $n \geq m$。在这种情况下，由于测量噪声或建模误差，方程（II.1）可能没有精确解，因此通常需要求一个近似解。

大规模求解这一问题的一种经典方法是代数重建技术，即*Kaczmarz* 方法 [5], 这是一种迭代行作用方案，每次通过满足一个方程来更新当前估计值 [14]。其过程如下：

算法 1 (用于求解线性系统的经典 Kaczmarz 方法)。 设 $Ax = b$ 为所关注的线性系统，其中 $A \in \mathbb{R}^{n\times m}$， $b \in \mathbb{R}^n$。

- 步骤 *0*：从一个初始猜测 $x^{(0)}$开始，通常将其选为零向量 $(0, 0, \dots, 0)^\top$。
- 步骤 $k \geq 1$：选取一个索引 $j_k \in \{1, 2, \dots, n\}$，通常按循环顺序或随机选取 *[15]*, ，并通过以下方式迭代更新猜测值：

$$x^{(k+1)} = x^{(k)} + \lambda \left(\frac{b_{j_k} - a_{j_k}^\top x^{(k)}}{||a_{j_k}||_2} \right) a_{j_k} \ , \tag{II.2}$$

其中， $||a_{j_k}||_2 = a_{j_k}^\top a_{j_k}$是行 l_2-范数， λ是松弛参数，通常取值介于0 和2之间。对于 $\lambda = 1$，此更新可解释为一次精确投影[1]。

换言之，Kaczmarz 方法将每个行方程 $a_j^\top x = b_j$ 视为 $\mathbb{R}^m$中的一个超平面，然后通过反复投影到这些超平面上，生成序列 $\{x^{(k)}\}$ ，投影过程通过应用公式（II.2）实现。达到预定的步骤数后，或满足所选收敛准则时，迭代便会终止。已有严格证明表明，采用随机行选择（按行 l_2范数成比例的概率进行抽样）时，相对于最小二乘解的相对残差保证以指数速率 [15]收敛。具体而言，研究表明，如果随机选取行的概率为：

$$\mathbb{P}(j_k = j) = \frac{||a_j||_2^2}{||A||_F} \ , \tag{II.3}$$

那么，迭代 $k \gg 1$ 时的期望偏差会按几何级数递减：

$$\begin{aligned} &\mathbb{E}\left(||x^{(k)} - x_{\text{sol}}||^2 \right) \\ &\quad \leq \left(1 - \frac{\sigma_{\min}^2(A)}{||A||_F^2} \right)^k ||x^{(0)} - x_{\text{sol}}||_2^2 \ . \end{aligned} \tag{II.4}$$

其中， $\sigma_{\min}(A)$ 是 A 的最小奇异值（按绝对值计），$||A||_F$ 是 A 的 Frobenius 范数， x_{sol} 是原线性系统 $Ax = b$ 的真实解（如果存在）。从上述不等式可以清楚地推导出，对于给定的加性误差 ϵ，令右侧等于 ϵ 后，所需的迭代步数 T （即 $T = \max k$）为：

$$T = \mathcal{O}\left(\frac{1}{-\log\left(1 - \frac{\sigma_{\min}^2(A)}{||A||_F^2}\right)} \log \frac{1}{\epsilon} \right) \ . \tag{II.5}$$

与直接求逆等更传统的方法相比，Kaczmarz 方法的一个缺点是它属于迭代方法，通常只能渐近地逼近解，而无法在有限步内精确求得解。它的主要优点是简单且每次迭代的成本低，因此通常能很快得到一个合理的最小二乘意义下的近似解。

III. 量子 Kaczmarz 方法

在本节中，我们概述用于求解以下线性方程组的量子 Kaczmarz 方法的构造：

---

[1] 当 $\lambda = 1$时，更新是将 $x^{(k-1)}$正交投影到超平面 $a_{j_k}^\top x = b_{j_k}$上，因为它会产生一个满足 $a_{j_k}^\top x^{(k)} = b_{j_k}$ 的迭代点 $x^{(k)}$。

形式 $Ax = b$ 其中 A，b 分别是上述所讨论形式中的矩阵和向量。我们的量子算法建立在 Algo. 1中描述的经典版本之上。下面，我们首先介绍量子算法所需的一些输入假设。随后，我们构造一个量子过程，用于执行（经典）Kaczmarz 算法的迭代。

A. 输入假设

本文假设已知的信息如下：

- 矩阵 A 的算子范数为 $||A|| \leq 1$ （即其最大奇异值小于 1）。我们还假设向量 b 的 l_2-范数为 $||b||_2 \leq 1$。
- 向量b的元素 $\{b_j\}_{j=1}^n$ 的经典知识。
- 矩阵 A的元素 $\{a_{jl}\}^{m,\ n}_{j,\ l=1}$ 的经典知识。
- l_2矩阵 A各行的 $\{||a_j||_2\}_{j=1}^n$ -范数。

我们指出，关于 A 的算子范数以及 l_2的 b范数的第一项假设并不失一般性，因为我们总可以用某个常数因子重新缩放该系统。事实上，所有相关工作中也都采用了这一假设 [8, 9, 12]。接下来的三项假设，尤其是对矩阵 A 的元素具有 l_2经典知识，以及 A各行的范数，使我们能够利用近期量子态制备协议中的诸多进展，例如 [17–21], ，以（近似）制备量子态$\{|a_j\rangle\}_{j=1}^n$ （其中，对每个 j，$|a_j\rangle \equiv \frac{a_j}{||a_j||_2}$）。这些参考文献中给出了具体的态制备过程，感兴趣的读者可参阅。这里，为了本文的目的，我们在以下引理中重述其结果：

引理 III.1。若已知$\{a_{ij}\}^{m,\ n}_{i,\ j=1}$ 以及 l_2-范数 $\{||a_j||_2\}_{j=1}^n$，即 a_j 的各行 A的范数，则对每个 j，若 a_j 具有 *[18–21]*, 所述的结构，则可使用门复杂度为 $\mathcal{O}(\log m)$的量子电路制备量子态 $|a_j\rangle$，另需 $\mathcal{O}(1)$ 个辅助量子比特。特别地，对于 a_j的一般任意结构，以使用 $\mathcal{O}(s_j)$ 个辅助量子比特为代价，可使用深度为$\mathcal{O}(\log s_j)$ 的量子电路制备量子态$|a_j\rangle$ （见参考文献 *[17]*），其中 s_j 是 A的第 a_j 行中的非零元素个数。

根据这一方案，我们现在继续介绍用于求解线性系统的量子 Kaczmarz 方法。

B. 求解线性方程组的量子 Kaczmarz 算法

对于每次迭代 k，算法都会依次执行一组固定的五个步骤，下面将对此进行详细说明：

步骤1：首先要构造因子 $a_{j_k}^\top x^{(k)}$ 的块编码（这里指左上角元素，即第一行第一列的元素为 $a_{j_k}^\top x^{(k)}$ 的酉算子）。具体做法如下。根据引理 III.1，在第 k 次迭代步骤 j_k（对于 $j_k \in [1, 2, ..., n]$）中，我们可以构造一个记为 U_{j_k} 的酉算子，用它来制备量子态 $|a_{j_k}\rangle$。还可以看出，酉算子 U_{j_k} 的第一列为 $|a_{j_k}\rangle$。进一步假设，在第 k 次迭代步骤中，我们已得到一个酉算子 $U_{x^{(k)}}$，它是一个矩阵的块编码，且该矩阵的第一列包含时间解 $x^{(k)}$。然后，我们可以利用引理 A.2 构造 $U_{j_k}^\dagger U_{x^{(k)}}$ 的块编码。由于 U_{j_k} 的第一列为 $|a_{j_k}\rangle$，因此 $U_{j_k}^\dagger$ 的第一行为 $|a_{j_k}\rangle$。所以，$U_{j_k}^\dagger U_{x^{(k)}}$ 的左上角元素（即第一行第一列的元素）为 $\langle a_{j_k}| x^{(k)}$。

接下来，我们可以使用引理 A.5以及缩放因子 $||a_{j_k}||_2$ 来获得.....的块编码

$$||a_{j_k}||_2 U_{j_k}^\dagger U_{x^{(k)}} \ ,$$

其第一行第一列的元素为乘积 $||a_{j_k}||_2 \langle a_{j_k}| x^{(k)} = a_{j_k}^\top x^{(k)}$。我们注意到，这里用到了以下性质：

$$||a_{j_k}||_2 \langle a_{j_k}| = ||a_{j_k}||_2 \frac{a_{j_k}}{||a_{j_k}||_2} = a_{j_k} \ .$$

步骤2：现在我们需要 b_{j_k} 的块编码，只需考虑一个旋转门即可得到：

$$R_x(\theta) = \begin{pmatrix} \cos(\theta/2) & -i\sin(\theta/2) \\ -i\sin(\theta/2) & \cos(\theta/2) \end{pmatrix} \ . \tag{III.1}$$

通过选择 θ，使得 $\cos(\theta/2) = b_{j_k}$ （这总是可行的，因为我们假设 l_2 范数 $||b||_2 \leq 1$，所以所有元素 $\{b_j\}_{j=1}^n$ 都小于1），我们得到酉算子 $R_x(\theta)$，其左上角元素为 b_{j_k}。

步骤3：接下来，鉴于单位矩阵 $\mathbb{I}$ 的任意维数都可以平凡地获得（并且也可以进行块编码，参见定义A.1），我们使用引理A.3 来构造 $R_x(\theta) \otimes \mathbb{I}$的块编码，其中 $\mathbb{I}$的维数选择为使得 $R_x(\theta) \otimes \mathbb{I}$ 的维数与 $||a_{j_k}||_2 U_{j_k}^\dagger U_{x^{(k)}}$的维数相匹配。

基于这些块编码，我们可以使用引理A.4来构造它们的减法运算的块编码，例如：

$$\frac{1}{2}\hat{\Xi} \quad \text{with} \quad \hat{\Xi} \equiv R_x(\theta) \otimes \mathbb{I} - ||a_{j_k}||_2 U_{j_k}^\dagger U_{x^{(k)}} \ .$$

该算子的左上角元素为$\frac{1}{2}\left(b_{j_k} - a_{j_k}^\top x^{(k)}\right)$。

基于上述块编码，我们随后再次使用引理A.3 来构造

$$\frac{1}{2}\hat{\Xi} \otimes U_{j_k} \ .$$

该算子的第一列可以表示为$\frac{1}{2}\left(b_{j_k} - a_{j_k}^\top x^{(k)}\right)|a_{j_k}\rangle$，因为 U_{j_k}的第一列是 $|a_{j_j}\rangle$。

然后我们可以使用引理A.1 将上述块编码算子乘以因子 $\frac{1}{||a_{j_k}||_2}$，从而得到以下对象的块编码：

$$\frac{1}{2||a_{j_k}||_2}\hat{\Xi}\otimes U_{j_k} \tag{III.2}$$

根据这一块编码，我们可以使用引理A.5 ，并选取适当的缩放因子 λ，从而得到以下对象的块编码：

$$\frac{\lambda}{2||a_{j_k}||_2}\hat{\Xi}\otimes U_{j_k} \tag{III.3}$$

步骤 4：根据这一块编码以及块编码$U_{x^{(k)}}$，引理A.4使我们能够得到以下对象的块编码：

$$\frac{1}{2}\left(U_{x^{(k)}}-\frac{\lambda}{2||a_{j_k}||_2}\hat{\Xi}\otimes U_{j_k}\right)\ . \tag{III.4}$$

回想一下， $U_{x^{(k)}}$ 是一个矩阵的块编码，该矩阵的第一列为 $x^{(k)}$ 。因此，上述算子实际上是一个矩阵的块编码，该矩阵的第一列为以下向量：

$$\frac{1}{2}\left[x^{(k)}-\frac{\lambda}{2||a_{j_k}||_2}\left(b_{j_k}-a_{j_k}^{\top}x^{(k)}\right)|a_{j_k}\rangle\right]\ . \tag{III.5}$$

将前置因子 $\lambda/2$ 视为对应于对松弛参数 λ 的合理选取（稍微滥用一下记号），并注意到 $|a_{j_k}\rangle=\frac{a_{j_k}}{||a_{j_k}||_2}$，上述向量可表示为：

$$\frac{1}{2}\left[x^{(k)}-\lambda\left(b_{j_k}-a_{j_k}^{\top}x^{(k)}\frac{a_{j_k}}{||a_{j_k}||_2^2}\right)\right]\ . \tag{III.6}$$

根据算法1的第二步，该向量等价于 $\frac{1}{2}x^{(k+1)}$。总之，从酉算子 $U_{x^{(k)}}$（其块编码的矩阵以 $x^{(k)}$ 为第一列）以及酉算子 U_{j_k} （其以 $|a_{j_k}\rangle$ 为第一列）出发，我们已展示了如何得到某个算子（式III.4）的块编码，该算子以 $\frac{1}{2}x^{(k+1)}$ 为第一列。

步骤 5：因子 $1/2$ 可通过应用引理 A.1 将方程III.4中的块编码算子乘以 2 来消去，从而得到酉算子 $U_{x^{(k+1)}}$，它对以下算子进行块编码：

$$U_{x^{(k)}}-\frac{1}{2}\hat{\Xi}\otimes U_{j_k}\ . \tag{III.7}$$

如上所示，该算子的第一列为 $x^{(k)}$ 。借助这个酉块编码 $U_{x^{(k+1)}}$，我们可以执行与上述类似的过程，只需将 $U_{x^{(k)}}$ 替换为 $U_{x^{(k+1)}}$。该过程的结果是酉算子 $U_{x^{(k+2)}}$ ，它对一个矩阵进行块编码，该矩阵的第一列为$x^{(k+2)}$ 。随后，使用这个新的酉算子 $U_{x^{(k+2)}}$ 替换上一步中的$U_{x^{(k+1)}}$ ，并重复该过程。

回顾算法 1 ，经典 Kaczmarz 方法从一个初始猜测 $x^{(0)}$开始。不失一般性，我们选择一个任意量子态 $|x^{(0)}\rangle$ ，并已知其酉态制备 $U_{x^{(0)}}$，例如：

$$U_{x^{(0)}}\,|0\rangle^{\otimes\log m}=|x^{(0)}\rangle\ .$$

可以看出，$U_{x^{(0)}}$ 的第一列为 $|x^{(0)}\rangle$。选择 T 和$U_{x^{(0)}}$ 作为初始酉算子后，我们可以将前几段中概述的过程迭代 T 次，得到

$$U_{x^{(0)}},U_{x^{(1)}},U_{x^{(2)}},....,U_{x^{(T)}}$$

它对具有时间解 $\{x^k\}_{k=1}^T$ 的矩阵进行块编码，这些解是矩阵的第一列。为了得到对应于线性系统（近似）解的量子态 $|x^{(T)}\rangle=\frac{x^{(T)}}{||x^{(T)}||_2}$，我们可以执行以下步骤。取酉算子 $U_{x^{(T)}}$ 并将其作用于量子态 $|\mathbf{0}\rangle\,|0\rangle^{\otimes\log m}$（其中 $|\mathbf{0}\rangle$ 对应于对 $U_{x^{(T)}}$进行块编码所需的辅助量子比特）。根据定义 A.1，可得以下量子态：

$$U_{x^{(T)}}\,|\mathbf{0}\rangle\,|0\rangle^{\otimes\log m}=|\mathbf{0}\rangle\,x^{(T)}+|\text{Garbage}\rangle\ , \tag{III.8}$$

其中， $|$垃圾态$\rangle$ 是与 $|\mathbf{0}\rangle\,x^{(T)}$正交的冗余态。通过测量辅助量子比特并后选择测得 $|\mathbf{0}\rangle$，即可得到量子态 $|x^{(T)}\rangle=\frac{x^{(T)}}{||x^{(T)}||_2}$。至此，我们完成了用于求解线性方程组的量子 Kaczmarz 算法。

为完整起见，我们将上述过程总结如下：

算法 2 （求解线性系统的量子 Kaczmarz 方法）。 设 $Ax=b$ 为所关注的线性系统，其中 A，b 满足前文提到的四个假设（参见第 *IIIA*节）。设 $U_{x^{(0)}}$ 为态制备酉算子，其第一列为 $x^{(0)}$ ，并令 T 为总迭代步数。假设在第 k次迭代步骤中，我们已得到酉算子 $U_{x^{(k)}}$ ，它对一个矩阵进行块编码，该矩阵的时间解 $x^{(k)}$ 位于第一列。接着，将以下五步过程迭代 T次：

1. 获得一个酉块编码，其所对应算子的左上角元素为 $a_{j_k}^T x^{(k)}$。*2.* 获得一个酉块编码，其所对应算子的左上角元素为 b_{j_k}。*3.* 获得一个酉块编码，其所对应算子的第一列为 $\frac{1}{2}\left(b_{j_k}-a_{j_k}^{\top}x^{(k)}\right)|a_{j_k}\rangle$。*4.* 获得一个酉块编码，其所对应算子的第一列为 $\frac{1}{2}x^{(k+1)}$。*5.* 获得一个酉块编码，其所对应算子的第一列为 $x^{(k+1)}$。

作为上述过程经过 T 次迭代后的最后一步，我们将所得酉算子 $U_{x^{(T)}}$ 作用于量子态 $|\mathbf{0}\rangle|0\rangle^{\otimes \log m}$，对辅助系统进行测量，并后选择测量结果为 $|\mathbf{0}\rangle$ 的情况。

输出： 量子态 $|x^{(T)}\rangle$ 对应于 $x^{(T)}$，后者是线性系统 $Ax = b$ 解的一个近似。

在第 II 节中，我们指出，对于加性精度 ϵ，即 $||x_T - x_{\text{solution}}|| \leq \epsilon$，所需的迭代次数 T 由式（II.5）给出。上述量子算法的深入分析将在附录 C 中提供。这里为简洁起见，我们在以下定理中概述主要结果：

定理 III.1。 设所研究的线性系统为 $Ax = b$，其中 $A \in \mathbb{R}^{n\times m}$，$b \in \mathbb{R}^n$ 满足前文所述的四个假设，并且 r_A 表示 A 的秩。对于加性精度 ϵ，算法 2 可以输出 $|x_T\rangle \equiv \frac{x_T}{||x_T||_2}$，使得 $||x_T - x_{\text{solution}}|| \leq \epsilon$。

- 如果对于每个 j，行 a_j 具有 *[18–21]* 所示的结构，那么算法 2 的电路复杂度为 $\mathcal{O}\left(2^{r_A}||x_T||_2 \frac{1}{\epsilon}\log m\right)$，额外需要 $\mathcal{O}(1)$ 个辅助量子比特。

- 对于 a_j 的一般结构，令 s_j 表示 a_j 的非零元素数量。定义 $s = \max\{s_j\}_{j=1}^n$。那么算法 2 所需的量子电路深度为 $\mathcal{O}\left(2^{r_A}||x_T||_2 \frac{1}{\epsilon}\log s\right)$，代价是额外需要 $\mathcal{O}(s)$ 个辅助量子比特。

C. 讨论

本节将从更广泛的视角讨论我们的量子算法，考察其效率最高的适用范围，以及相较于现有算法的潜在优势。

处理矩形线性系统。我们的量子算法建立在经典 Kaczmarz 方法之上，该方法天然适用于处理矩形线性系统。在这种情况下，唯一解可能不存在，因此，[8–10, 12] 中的量子线性求解器无法求得解，因为此时 A 的逆矩阵未定义。与此同时，我们的量子算法仍然可以返回“解”；在这种情况下，应从最小二乘的角度理解该解，例如，求出 x 使 $||Ax - b||_2$ 最小。我们指出，现有的其他量子线性求解算法也可以处理矩形情形[11, 13]。

复杂度中的另一个重要因素（定理III.1）是 r_A 上的指数级增长，其中 A 是其秩。由于这一点，当 r_A 足够小时，我们的算法最为高效；或者说，当 A 的秩足够小时也是如此。我们指出，正如 [8, 9]，这正是他们的算法无效的情形。因此，这表明，在矩形系统这一场景下，我们的量子算法可以很好地补充现有的量子线性求解器；在该场景中，方程的数量 n 超过未知数的数量。

对稀疏度参数 s 和条件数的改进 κ。 对于方形线性系统，我们将 $n = m$ 设为主维度。为便于比较，我们提供了总结现有量子线性求解器复杂度的表格。

	复杂度	量子随机存取存储器/预言机
参考文献 [8]	$\mathcal{O}\left(\frac{1}{\epsilon}s^2\kappa \log\ m\right)$	YES
参考文献 [9]	$\mathcal{O}\left(s\kappa^2 \text{ 对数 } \frac{m}{\epsilon}\right)$	YES
参考文献 [12]	$\mathcal{O}\left(\kappa^2\sqrt{m} \text{ 多重对数} \frac{m}{\epsilon}\right)$	YES
参考文献 [10]	$\mathcal{O}\left(s^7 \frac{1}{\epsilon} \text{对数 } m\right)$	YES
参考文献 [11]	启发式方法	NO
参考文献 [13]	$\mathcal{O}\left(\kappa^2 \text{ 对数 } \frac{1}{\epsilon} \text{ 对数 } m\right)$	YES
我们的工作	$\mathcal{O}\left(\frac{1}{\epsilon} \text{对数 } m\right)$ 或 $\mathcal{O}\left(\frac{1}{\epsilon} \text{对数 } s\right)$	NO

表一。 总结现有量子线性求解算法电路复杂度的表格。在上表中，κ 表示 A 的条件数，s 表示 A 的稀疏度（每行或每列中非零元素的最大数量）。

从表一中可以看出，我们的方法在条件数 κ 和稀疏度参数 s 方面具有一定优势。更具体地说，如定理 III.1 所示，当 A 的所有行都具有某些结构时，我们的复杂度将为 $\mathcal{O}\left(||x_T||_2 \frac{1}{\epsilon} \text{对数 } m\right)$，且与 s 无关。我们指出，如果 $||x_T||_2 = \mathcal{O}(1)$，该复杂度也可以与 κ 无关。在附录 C 中，将证明（利用 [8] 的结果） $||x_T||_2$ 由 $\mathcal{O}(\kappa)$ 给出上界。因此，在最坏情况下，我们的复杂度关于 κ 呈线性标度。尽管如此，与现有结果相比，这仍然是重大改进；现有结果对 κ、s 的依赖呈线性至多项式增长。与此同时，如果 A 的行具有任意结构，那么我们的量子算法可以实现关于稀疏度参数 s 呈对数级深度的电路，但需要额外使用 $\mathcal{O}(s)$ 个辅助量子比特。因此，在实践中，只有当 s 关于维数 m 呈多对数级标度时，该方法才具有量子比特效率。在这种情况下，就电路深度而言，我们的算法相对于现有工作在稀疏度 s 方面实现了指数级改进。所需的辅助量子比特数量为 $\mathcal{O}(s) = \mathcal{O}(\text{对数 } n)$，因此量子比特总数为 $\mathcal{O}(\text{对数 } n)$，与现有工作相近。

放宽强输入假设。正如表I所示，现有的大多数量子线性求解算法都假设可以访问一个能够高效查询 A 各元素的预言机。已有少数方案尝试实现这一预言机，例如量子随机存取存储器（量子随机存取存储器） [22, 23]。然而，这种预言机假设一直被视为一种

相当强的输入假设。一方面，目前尚无大规模且容错的量子随机存取存储器，这使得依赖量子随机存取存储器的量子算法难以通过实验实现。另一方面，关于去量子化算法的研究进展 [24–26] 表明，若没有预言机假设，量子算法就无法实现指数加速，至少在一般情形下如此。在同一系列研究中 [24–26]，作者特别指出，如果经典算法能够访问某种特定输入（类似于预言机假设），那么经典计算机就能以多对数级复杂度解决许多任务。

误差容限倒数方面的权衡。 从表 I可以看出，我们的方法在$\frac{1}{\epsilon}$ 方面呈线性标度，其效率比大多数现有工作低了指数级（ [10]除外）。这是因为我们的算法的复杂度随 T 呈指数级增长；由于我们的方法是迭代式的，每一步都需要多次使用前一步的结果。我们认为，这是为了更好地依赖于s 以及可能的 κ （如果 $||x_T||_2$ 表现为 $\mathcal{O}(1)$）而做出的权衡，同时我们的方法也不依赖预言机假设。鉴于此，我们认为，实际上，我们的方法可以很好地补充现有的量子线性求解器。对于条件数和稀疏度都较大的线性系统，或者预言机访问难以高效实现的情况，我们的算法可能更有优势。

IV. 结论

本文概述了一种用于求解线性代数方程组的量子 Kaczmarz 算法。我们的算法直接基于经典 Kaczmarz 算法，该算法通过基于随机列选择迭代更新解来求解线性系统。在对 A的 b结构以及包含时间解 $x^{(k)}$的块编码算子作出适当输入假设后，我们展示了如何构造$a_{j_k}^T x^{(k)}$、 b_{j_k}， 12($b_{j_k} - a_{j_k}^T x^{(k)}$)的块编码，并最终构造出具有所需更新后解的算子 $x^{(k+1)}$。随后，将这一过程迭代总计T 次，再将其作用于已知量子态并进行测量。后选择测量的结果，就是给定线性系统解的所需近似值。接着，我们进行了讨论，指出在某些方面，我们的算法相比以往的量子线性求解器更具优势。尽管在 κ方面， s有了重大改进，但我们的方法在 T上呈指数级标度，因而导致 $\frac{1}{\epsilon}$ 呈线性标度。这比 [8, 9]的效率低了指数级。我们认为，为了改进 κ方面， s，这是合理的权衡。不过，我们尚不清楚这种权衡是否不可避免。因此，如何改善 T 上的指数级标度（进而改善 $\frac{1}{\epsilon}$ ）是一个值得探索的方向。

致谢

本研究得到美国能源部科学办公室国家量子信息科学研究中心量子优势协同设计中心（C2QA）的支持，合同编号为 DE-SC0012704。N.A.N. 还感谢石溪大学分布式量子处理中心提供的支持。N.A.N. 感谢哈佛大学在本项目完成期间给予的接待，并曾于此进行学术访问。

[1]Gilbert Strang。 *Introduction to linear algebra*。SIAM, 2022。[2] Richard Gordon、Robert Bender 和 Gabor T Herman。用于三维电子显微镜和 X 射线摄影的代数重建技术（ART）。*Journal of theoretical Biology*，29(3):471–481, 1970。[3] Gabor T Herman, *Fundamentals of computerized tomography: image reconstruction from projections* 。施普林格科学与商业媒体，2009。[4] Stephen Wright、Jorge Nocedal 等。数值优化。 *Springer Science*，35(67-68):7, 1999。[5] Stefan Kaczmarz。线性方程组的近似解。 *International Journal of Control*，57 6 1269 1271，1993 ( ): – 。[6] David Reinsel-John Gantz-John Rydning，John Reinsel 和 John Gantz。从边缘到核心的世界数字化。 *Framingham: International Data Corporation*，16 1 28, 2018: – 。[7] Jared Kaplan、Sam McCandlish、Tom Henighan、Tom B Brown、Benjamin Chess、Rewon Child、Scott Gray、Alec Radford、Jeffrey Wu 和 Dario Amodei。缩放神经语言模型中的规律。 *arXiv preprint arXiv:2001.08361* 2020 , .[8] Aram W Harrow、Avinatan Hassidim 和 Seth Lloyd。用于求解线性方程组的量子算法。*Physical Review Letters*, 103(15):150502, 2009。[9] Andrew M Childs、Robin Kothari 和 Rolando D Somma。求解线性方程组的量子算法：精度依赖性的指数级改进。 *SIAM Journal on Computing* 46 6 1920 1950, , ( ): – .[10] 2017 B David Clader、Bryan C Jacobs 和 Chad R Sprouse。预条件量子线性系统算法。 *Physical Review Letters* 110 25 250504, 2013 , ( ): .[11] Hsin-Yuan Huang、Kishor Bharti 和 Patrick Rebentrost。具有回归损失函数的线性方程组近期量子算法。 *New Journal of Physics* 23 11 113021, 2021 , ( ): .[12] Leonard Wossnig、Zhikuan Zhao 和 Anupam Prakash。稠密矩阵的量子线性系统算法。*Physical review letters*, 120(5):050502, 2018。

[13] Changpeng Shao 和 Hua Xiang。在量子计算机上求解线性系统的行列迭代方法。 *Physical Review A*, 101(2):022322, 2020。[14] Yair Censor。用于超大规模稀疏系统及其应用的行作用方法。 *SIAM review*, 23(4):444– 466, 1981。[15] Thomas Strohmer 和 Roman Vershynin。具有指数收敛速度的随机 Kaczmarz 算法。 *Jour-nal of Fourier Analysis and Applications* 15 2 262 278, , 2009 ( ): – .[16] Gene H Golub 和 Charles F Van Loan。 *Matrix computations* 2013 。J HU 出版社，。[17] Xiao-Ming Zhang、Tongyang Li 和 Xiao Yuan。具有最优电路深度的量子态制备：实现与应用。 *Physical Review Letters*, 129 23( ):230504, 2022。[18] Sam McArdle、Andr´as Gily´en 和 Mario Berta。无需相干算术的量子态制备。*arXiv preprint arXiv:2210.14892*, 2022。[19] Gabriel Marin-Sanchez、Javier Gonzalez-Conde 和 Mikel Sanz。近似函数加载的量子算法。 *Physical Review Research* 5 3 033114, , ( ): . 2023 [20]Kouhei Nakaji、Shumpei Uno、Yohichi Suzuki、Rudy Raymond、Tamiya Onodera、Tomoki Tanaka、Hiroyuki Tezuka、Naoki Mitsuda 和 Naoki Yamamoto。在浅层参数化量子电路中进行近似振幅编码及其在金融市场指标中的应用。 *Physical Review Research*, 4(2):023136, 2022。[21] Christa Zoufal、Aur´elien Lucchi 和 Stefan Woerner。用于学习和加载随机分布的量子生成对抗网络。 *npj Quantum Information* 5 1 103, 2019 , ( ): .

[22] Vittorio Giovannetti、Seth Lloyd 和 Lorenzo Maccone。量子随机存取存储器的架构。 *Physical Review A—Atomic, Molecular, and Optical Physics*，78(5):052310，2008。[23] Vittorio Giovannetti、Seth Lloyd 和 Lorenzo Maccone。量子随机存取存储器。 *Physical review letters*，100 16 160501，2008（）：。[24] Ewin Tang。用于推荐系统的量子启发式经典算法。载于 *Proceedings of the 51st annual ACM SIGACT symposium on theory of computing*，第 217–228 页，2019。[25] Andr´as Gily´en、Seth Lloyd 和 Ewin Tang。具有对数维数依赖性的量子启发式低秩随机回归。 *arXiv preprint arXiv:1811.04909* 2018 ，。

[26] Changpeng Shao 和 Ashley Montanaro。用于求解线性方程组的更快速量子启发式算法。 *ACM Transactions on Quantum Computing* 3 4 1 23, , （）：–。2022 [27] Andr´as Gily´en、Yuan Su、Guang Hao Low 和 Nathan Wiebe。量子奇异值变换及其扩展：量子矩阵运算的指数级改进。载于 *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*，第 193–204 页，2019。[28] Junseo Lee 和 Nhat A Nghiem。量子拓扑数据分析的新视角：Betti 数估计，以及同调类与上同调类的测试和跟踪。 *arXiv preprint arXiv:2506.01432*，2025。[29] Guang Hao Low 和 Isaac L Chuang。通过量子信号处理实现最优哈密顿量模拟。 *Physical Review Letters* 118 1 010501, 2017 , （）：。

[30] Guang Hao Low 和 Isaac L Chuang。通过量子比特化实现哈密顿量模拟。 *Quantum*，3:163，2019。[31] Daan Camps 和 Roel Van Beeumen。使用块编码进行近似量子电路综合。 *Physical Review A* 102 5 052411, 2020 , （）：。

附录 A：块编码与量子奇异值变换

我们简要总结算法中使用的核心量子工具。为求简洁，我们仅突出主要结果，省略技术细节；这些细节在 [27] 中有详尽介绍。类似的总结也见于 [28]。

定义 A.1 (块编码酉矩阵，参见例如 [27, 29, 30])。 设 A 是一个大小为 $N \times N$ 、算子范数为 $\|A\| < 1$的厄米矩阵。若酉矩阵 U 满足以下条件，则称其为 A 的精确块编码

$$U = \begin{pmatrix} A & * \\ * & * \end{pmatrix}, \tag{A.1}$$

其中， U 的左上块对应于 A。等价地，可以写作

$$U = |\mathbf{0}\rangle\langle\mathbf{0}| \otimes A + (\cdots), \tag{A.2}$$

其中， $|\mathbf{0}\rangle$ 表示用于块编码的辅助态，而$(\cdots)$ 表示与 $|\mathbf{0}\rangle\langle\mathbf{0}| \otimes A$正交的其余分量。若 U 满足

$$U = |\mathbf{0}\rangle\langle\mathbf{0}| \otimes \tilde{A} + (\cdots), \tag{A.3}$$

˜ ˜对于某个 A ，使得 $\|A - \tilde{A}\| \leq \epsilon$，则称 U 是 A的 ϵ近似块编码。此外， U 作用于量子态 $|\mathbf{0}\rangle|\phi\rangle$ 上的结果为

$$U|\mathbf{0}\rangle|\phi\rangle = |\mathbf{0}\rangle A|\phi\rangle + |\text{Garbage}\rangle, \tag{A.4}$$

其中 $|$垃圾态$\rangle$ 是一个与 $|\mathbf{0}\rangle A|\phi\rangle$正交的量子态。 U 的电路复杂度（例如深度）称为块编码 A的复杂度。

~~引理 A.1 (放大, [27]的定理 30)。设 $U, \Pi, \Pi \in \mathrm{End}(\mathcal{H}_U)$ 是 $\mathcal{H}_U$上的线性算子，且 U是酉算子， $\Pi, \widetilde{\Pi}$是正交投影算子。设 $\gamma > 1$且 $\delta, \epsilon \in (0, 12)$。假设 $\widetilde{\Pi} U \Pi = W \Sigma V^\dagger = \sum_i \varsigma_i |w_i\rangle \langle v_i|$是奇异值分解。那么存在一个 $m = \mathcal{O}\left(\frac{\gamma}{\delta} \log\left(\frac{\gamma}{\epsilon}\right)\right)$ 以及一个可高效计算的 $\Phi \in \mathbb{R}^m$，使得 – –

$$\left(\langle +| \otimes \widetilde{\Pi}_{\leq \frac{1-\delta}{\gamma}}\right) U_\Phi \left(|+\rangle \otimes \Pi_{\leq \frac{1-\delta}{\gamma}}\right) = \sum_{i:\, \varsigma_i \leq \frac{1-\delta}{\gamma}} \tilde{\varsigma}_i |w_i\rangle \langle v_i|, \ \textit{where} \left\| \frac{\tilde{\varsigma}_i}{\gamma \varsigma_i} - 1 \right\| \leq \epsilon. \tag{A.5}$$

此外， U_Φ 可以使用一个辅助量子比特实现，需调用 U m 次、$C_\Pi NOT$ 门 m 次、$C\tilde{\Pi} NOT$ 门 m次，以及 m 个单量子比特门。这里，

- $C_\Pi NOT$: $= X \otimes \Pi + I \otimes$ （$I - \Pi$）以及 $C\tilde{\Pi} NOT$ 的类似定义；参见定义 *2*，见 *[27]* ，
- U_Φ：交替相位调制序列；参见 *[27]*,~中的定义 *15*
- $\Pi_{\leq \delta}$、 $\Pi_{\leq \delta}$：奇异值阈值投影算子；参见 *[27]* 中的定义 *24*。

基于 A.1，以下列出若干虽显而易见但尤为重要的性质。

注 A.1 （块编码酉矩阵的性质）。 块编码框架直接带来以下结论：

(i) 任意酉算子 U 都是其自身的精确块编码。

(ii) 如果 U 是 A 的块编码，那么对于任意 $m \geq 1$， $\mathbb{I}_m \otimes U$ 也同样是 A 的块编码。

(iii) 单位矩阵 $\mathbb{I}_m$ 可以通过例如 $\sigma_z \otimes \mathbb{I}_m$ 进行平凡的块编码。

给定一组块编码算子，可以对它们执行各种算术运算。这里，我们仅介绍一些与我们的算法尤其相关的关键操作，重点说明它们的实现方式和时间复杂度，而不展开证明。有关更详细的解释，请参见 [27, 31]。

引理 A.2 （非正式表述，块编码算子的乘积，参见例如 [27]）。 给定两个矩阵 A_1 和 A_2 的酉块编码，其相应的实现复杂度分别为 T_1 和 T_2，则存在一种高效过程，可以构造出其乘积 $A_1 A_2$ 的酉块编码，复杂度为 $T_1 + T_2$。

引理 A.3 （非正式表述，块编码算子的张量积，参见例如 [31, 定理 1]）。 给定多个算子 $\{M_i\}_{i=1}^m$ 的酉块编码 $\{U_i\}_{i=1}^m$ （假设它们是精确的），存在一个过程，只需对每个 U_i 和 $\mathcal{O}$ 应用一次，并使用（1）个 *SWAP* 门，即可构造 $\bigotimes_{i=1}^m M_i$ 的酉块编码。

引理 A.4 （非正式表述，块编码算子的线性组合，参见例如 [27, 定理 52]）。 给定多个算子 $\{A_i\}_{i=1}^m$ 的酉块编码。则存在一个过程，可在 $\mathcal{O}(m)$ 的时间复杂度内，生成算子 $\sum_{i=1}^m \pm$ 的酉块编码（A_i/m），例如，每个算子 A_i 的块编码只需使用一次。

引理 A.5 （非正式表述，块编码算子的缩放乘法）。 给定某个矩阵的块编码A，如*A.1*所述， A/p 的块编码，其中 $p > 1$，可通过额外的 $\mathcal{O}$（1）开销来制备。

引理 A.6 （矩阵求逆，参见例如 [9, 27]）。 给定某个矩阵的块编码 A，其算子范数为$||A|| \leq 1$，块编码复杂度为 T_A，则存在一个量子电路，可生成一个 ϵ-近似块编码 A^{-1}/κ，其中 κ 是 A的条件数。该量子电路的复杂度为 $\mathcal{O}$（$\kappa T_A \log$（$1/\epsilon$）。

附录 B： T 的上界和下界

在第 II节中，我们指出，对于加性精度 ϵ，迭代次数需要为（为匹配本节所用的符号，我们将k 替换为 T）：

$$T = \mathcal{O}\left(\frac{1}{-\log\left(1 - \frac{\sigma_{\min}^2(A)}{||A||_F^2}\right)} \log\frac{1}{\epsilon}\right)$$

众所周知，Frobenius 范数 $||A||_F^2$ 是 A 的奇异值平方和。因此有：

$$||A||_F^2 \leq r_A \sigma_{\max}^2(A) \tag{B.1}$$

其中 r_A 是 A的秩。因此，有：

$$\frac{\sigma_{\min}^2(A)}{||A||_F^2} \geq \frac{\sigma_{\min}^2(A)}{r_A \sigma_{\max}^2(A)} \tag{B.2}$$

由此可得：

$$1 - \frac{\sigma_{\min}^2(A)}{||A||_F^2} \leq 1 - \frac{1}{r_A \kappa_A^2} \tag{B.3}$$

其中 $\kappa_A \equiv \frac{\sigma_{\max}(A)}{\sigma_{\min}(A)}$ 是 A的条件数。因此：

$$-\log\left(1 - \frac{\sigma_{\min}^2(A)}{||A||_F^2}\right) \geq \log \frac{r_A \kappa_A^2}{r_A \kappa_A^2 - 1} \tag{B.4}$$

$$\longrightarrow \frac{1}{-\log\left(1 - \frac{\sigma_{\min}^2(A)}{||A||_F^2}\right)} \leq \frac{1}{\log \frac{r_A \kappa_A^2}{r_A \kappa_A^2 - 1}} \tag{B.5}$$

因此，迭代次数 T 有如下界：

$$T = \mathcal{O}\left(\frac{1}{\log \frac{r_A \kappa_A^2}{r_A \kappa_A^2 - 1}} \log \frac{1}{\epsilon}\right) \tag{B.6}$$

可以看出，如果 $r_A \kappa_A^2 \gg 1$，那么比值 $\frac{r_A \kappa_A^2}{r_A \kappa_A^2 - 1}$ 将趋近于 1，因此其对数可近似为：

$$\log \frac{r_A \kappa_A^2}{r_A \kappa_A^2 - 1} = \log\left(1 + \frac{1}{r_A \kappa_A^2 - 1}\right) \approx \frac{1}{r_A \kappa_A^2} \tag{B.7}$$

因此， T的上界为 $T = \mathcal{O}\left(r_A \kappa_A^2 \log \frac{1}{\epsilon}\right)$。需要指出的是，这是一个上界；实际上，$\log \frac{r_A \kappa_A^2}{r_A \kappa_A^2 - 1}$ 的值可能更小，例如为 $\mathcal{O}(1)$。在这种情况下， T 的值的渐近阶实际上为$\mathcal{O}\left(\log \frac{1}{\epsilon}\right)$。

同时，由于 $||A||_F^2 \geq r_A \sigma_{\min}^2$，按照上述相同的推理思路，也可以类似地证明：

$$\frac{1}{-\log\left(1 - \frac{\sigma_{\min}^2(A)}{||A||_F^2}\right)} \geq \frac{1}{\log \frac{r_A}{r_A - 1}} \approx r_A \tag{B.8}$$

因此， T 的下界为 $\Omega(r_A)$，从而实际上有 $T \in \mathcal{O}\left(r_A \log \frac{1}{\epsilon}\right)$。

附录 C：复杂度分析

为了分析复杂度，我们根据算法逐步讨论复杂度。2。令 $\mathcal{C}_k$ 表示实现酉算子 $U_{x^{(k)}}$的电路复杂度。此外，令 $\mathcal{C}(a_{j_k})$ 表示酉算子 U_{j_k}的电路复杂度，该酉算子用于制备行态 $|a_{j_k}\rangle$ (其中 $j_k \in 1、2、\ldots\ldots、n$)。采用此记号，算法 2 中每一步的开销如下：

- 步骤1：我们需要分别使用 $U_{x^{(k)}}$ U_{j_k} 一次。因此，总电路复杂度为 $\mathcal{O}(\mathcal{C}_k + \mathcal{C}(a_{j_k}))$。
- 步骤2：我们使用 x-旋转门 $R_x(\theta)$一次，因此复杂度为 $\mathcal{O}(1)$。
- 步骤3：我们各使用一次前两个步骤中的酉块编码，并另外使用一次酉算子 U_{j_k}。因此，总复杂度为：

$$\mathcal{O}\left(\mathcal{C}_k + 2\mathcal{C}(a_{j_k})\right) \tag{C.1}$$

- 步骤 4：我们使用上一步中的酉块编码，并另外使用一次酉算子$U_{x^{(k)}}$。因此，总电路复杂度为：

$$\mathcal{O}\left(2\mathcal{C}_k + 2\mathcal{C}(a_{j_k})\right) \tag{C.2}$$

- 步骤 5：此步骤使用上一步中的酉块编码 $\mathcal{O}$（1）（结合引理 A.1），因此总电路复杂度为：

$$\mathcal{O}\left(2\mathcal{C}_k + 2\mathcal{C}(a_{j_k})\right) \tag{C.3}$$

我们提醒一下，步骤 5（见算法 2）的结果是算子 $U_{x^{(k+1)}}$ 的酉块编码，该算子的 $x^{(k+1)}$ 为第一列。如上所述，$U_{x^{(k+1)}}$的电路复杂度为：

$$\mathcal{C}_{k+1} = \mathcal{O}\left(2\mathcal{C}_k + 2\mathcal{C}(a_{j_k})\right) \tag{C.4}$$

由此，我们可以使用归纳法继续进行。在类似的推理下，可以证明 $\mathcal{C}_k = \mathcal{O}\left(2\mathcal{C}_{k-1} + 2\mathcal{C}(a_{j_{k-1}})\right)$。为方便后续使用，我们定义 $\mathcal{C} \equiv \max\{\mathcal{C}(a_{j_k})\}_{k=1}^{T}$。于是有：

$$\begin{aligned}\mathcal{C}_{k+1} &= \mathcal{O}\left(2\mathcal{C}_k + 2\mathcal{C}(a_{j_k})\right) \\ &= \mathcal{O}\left(2(2\mathcal{C}_{k-1} + 2\mathcal{C}(a_{k-1})) + 2\mathcal{C}(a_{j_k})\right) \\ &= \mathcal{O}\left(4\mathcal{C}_{k-1} + (2+4)\mathcal{C}\right)\end{aligned} \tag{C.5}$$

类似地，$\mathcal{C}_{k-1} = \mathcal{O}\left(2\mathcal{C}_{k-2} + 2\mathcal{C}(a_{j_{k-2}})\right)$，因此：

$$\begin{aligned}\mathcal{C}_{k+1} &= \mathcal{O}\left(4\left(2\mathcal{C}_{k-2} + 2\mathcal{C}(a_{j_{k-2}})\right) + (2+4)\mathcal{C}\right) \\ &= \mathcal{O}\left(2^3\mathcal{C}_{k-2} + (2+2^2+2^3)\mathcal{C}\right)\end{aligned} \tag{C.6}$$

以类似的方式继续推导，可以得到：

$$\mathcal{C}_{k+1} = \mathcal{O}\left(2^{k+1}\mathcal{C}_0 + \sum_{i=1}^{k+1} 2^i\ \mathcal{C}\right) \tag{C.7}$$

其中，$\mathcal{C}_0$ 是$U_{x^{(0)}}$的电路复杂度。对于总共 T 次迭代，其电路复杂度 $\mathcal{C}_T$ 为$U_{x^{(T)}}$：

$$\mathcal{O}\left(2^T\mathcal{C}_0 + \sum_{i=1}^{T} 2^i\ \mathcal{C}\right) \tag{C.8}$$

众所周知，$\sum_{i=1}^{T} 2^i = 2^{T+1} - 2 = \mathcal{O}(2^T)$，因此上述总复杂度为 $\mathcal{O}\left(2^T\left(\mathcal{C}_0 + \mathcal{C}\right)\right)$。最后，我们测量式III.8中量子态的辅助量子比特，并后选择观测到 $|\mathbf{0}\rangle$。该测量的成功概率为 $||x_T||_2^2$，通过振幅放大法可以将其二次提升，但会引入额外的复杂度 $\mathcal{O}\left(\frac{1}{||x_T||_2}\right)$。计入这一放大和测量步骤后，我们得到最终总复杂度 $\mathcal{O}\left(\frac{1}{||x_T||_2}2^T\left(\mathcal{C}_0 + \mathcal{C}\right)\right)$。我们指出，通过适当选择 T，x_T是线性系统 $Ax = b$的真实解 x_{solution} 的良好近似。如 [8], 中所分析的，x_{solution} 的范数的逆矩阵的上界为 $\mathcal{O}(\kappa)$。因此，在最坏情况下，$\frac{1}{||x_T||_2} = \mathcal{O}(\kappa)$。

由于 $U_{x^{(0)}}$ 可以是任意的，因此可以安全地假设其电路复杂度为 $\mathcal{O}(1)$。对于 $\mathcal{C}$，其定义为 $\max\{\mathcal{C}(a_{j_k})\}_{k=1}^{T}$，根据引理 III.1，如果对于所有 j，量子态 $|a_j\rangle$ 都可以通过 [18–21], 中的方法制备，则最大电路复杂度（就门复杂度而言）为 $\mathcal{C} \in \mathcal{O}(\log m)$（使用常数数量的辅助量子比特）。同一个引理 III.1 还指出，如果改为通过 [17] 制备 $|a_j\rangle$（该方法适用于$|a_j\rangle$的任意结构），则深度复杂度可以达到 $\mathcal{O}\left(\max\{\log\ s_{j_k}\}_{k=1}^{T}\right)$（其中 s_{j_k}是 a_{j_k}的非零元素数量，至多为 m），代价是额外使用 $\mathcal{O}(\max\{s_{j_k}\}_{k=1}^{T})$ 个辅助量子比特。因此，我们得到最终的电路复杂度为 $\mathcal{O}\left(2^T\log\ m\right)$（使用额外的 $\mathcal{O}(1)$ 个辅助量子比特）。通过额外使用 $\mathcal{O}(\max\{s_{j_k}\}_{k=1}^{T})$ 个辅助量子比特，电路深度可以改进为 $\mathcal{O}\left(\max\{\log\ s_{j_k}\}_{k=1}^{T}\right)$。

根据前一附录中推导出的 T 界，将其代入最终复杂度中，即可得到定理III.1所述的结果。