Advances in Applied Mathematics 应用数学进展, 2025, 14(2), 244-250
Published Online February 2025 in Hans. https://www.hanspub.org/journal/aam
https://doi.org/10.12677/aam.2025.142067

Hans汉斯

一种用于求解高度超定线性系统的计数草图随机平均块Kaczmarz方法

姜昊辰

广东工业大学数学与统计学院，广东 广州

收稿日期：2025年1月20日；录用日期：2025年2月14日；发布日期：2025年2月21日

摘　要

受计数草图对Kaczmarz方法加速效果的启发，文章将计数草图与随机平均块Kaczmarz方法相结合，得出了一种求解高度超定线性系统的新方法。为了验证新方法的可行性，进行了大量数值实验。数值实验表明，在相同精度下，新方法在计算时间上表现良好。

关键词

计数草图，随机平均块Kaczmarz方法，高度超定线性系统，计算时间

A Count Sketch Randomized Average Block Kaczmarz Method for Solving Highly Overdetermined Linear Systems

Haochen Jiang

School of Mathematics and Statistics, Guangdong University of Technology, Guangzhou Guangdong

Received: Jan. 20th, 2025; accepted: Feb. 14th, 2025; published: Feb. 21st, 2025

Abstract

Inspired by the acceleration effect of the count sketch on the Kaczmarz method, this paper combines the count sketch with the randomized average block Kaczmarz method to derive a new method for solving highly overdetermined linear systems. To verify the feasibility of the new method, a large number of numerical experiments were conducted. The numerical experiments show that, under the same precision, the new method performs well in terms of computing time.

文章引用: 姜昊辰. 一种用于求解高度超定线性系统的计数草图随机平均块 Kaczmarz 方法[J]. 应用数学进展, 2025, 14(2): 244-250. DOI: 10.12677/aam.2025.142067

Keywords

Count Sketch, Randomized Average Block Kaczmarz Method, Highly Overdetermined Linear System, Computing Time

1. 引言

在科学与工程计算领域，求解大规模线性方程组是一个极为常见且关键的任务。这类方程组广泛出现于诸如信号处理、医学成像、机器学习以及计算机图形学等众多重要应用场景之中。传统的直接求解方法，像高斯消元法，虽然在理论上能够精确地求解线性方程组，但当面对大规模甚至超大规模的方程组时，其计算复杂度会急剧增加，所需的计算时间和存储空间常常超出实际可承受的范围，导致求解过程变得极为耗时且在很多情况下难以实现。迭代法应运而生，成为解决大规模线性方程组的重要途径。

我们考虑大规模的线性方程组的解：

$$Ax = b \tag{1}$$

其中 $A \in R^{m\times n}\left(m \gg n\right)$，$b \in R^m$，$x \in R^n$ 为未知向量。Kaczmarz 算法[1]是一种非常流行的求解此类方程的迭代投影方法，它是作用方法的一种特例，并被广泛用于医学图像重构[2]、信号处理[3]和分布式计算[4]等问题中。如果我们用 $A^{(i)}$ 表示矩阵 A 的第 i 行，$b^{(i)}$ 表示向量 b 的第 i 项，那么给定一个初始估计 x_0，Kaczmarz 方法可以定义为

$$x_{k+1} = x_k + \frac{b^{(i_k)} - A^{(i_k)}x_k}{\left\|A^{(i_k)}\right\|_2^2}\left(A^{(i_k)}\right)^*, k = 0,1,2\cdots$$

这里的上标 $*$ 表示一个向量或矩阵的转置，$\|\ \|_2$ 表示欧式范数，根据 $i_k = \mathrm{mod}\left(k,m\right)+1$ 选择编号为 i_k 的目标行。

考虑到具体应用的特殊数据结构和算法实现的计算机体系结构，许多研究人员提出了块式迭代方法。其中块 Kaczmarz [5]方法是这些方法的典型代表。在块 Kaczmarz 方法中，解空间由系数矩阵 A 的多行生成。具体来说，在每次迭代 k 时，从矩阵 $A \in R^{m\times n}$ 的行指标的一个分区 $T = \left\{\tau_1, \tau_2, \cdots, \tau_p\right\}$ 中循环地选择一个子集 τ_{i_k}，则下一个近似解可通过以下方案得到：

$$x_{k+1} = x_k + A_{\tau_{i_k}}^{\dagger}\left(b_{\tau_{i_k}} - A_{\tau_{i_k}}x_k\right), k = 0,1,2\cdots,$$

其中，$A_{\tau_{i_k}}^{\dagger}$ 表示以 τ_{i_k} 为索引的 A 行子矩阵，$i_k = \mathrm{mod}\left(k,p\right)+1$，上标†表示一个矩阵的 Moore-Penrose 逆。

块 Kaczmarz 方法的实现与表现很大程度上取决于由划分 T 中的块 τ_{i_k} 所索引的子矩阵 $A_{\tau_{i_k}}$ 的性质。在合适的块划分情况下，块 Kaczmarz 方法可能具有更快的收敛速度。因为每次投影利用了一个块中的多个方程的信息，相比于经典 Kaczmarz 方法每次只使用一个方程，它能够更全面地调整近似解，使其更快地接近真实解。例如，当方程组中存在一些方程之间相关性很强的情况，将这些相关方程划分为一个块，可以更有效地利用它们之间的信息来加速收敛。

计数草图[6] [7]是一种在数据处理和数值线性代数等领域广泛应用的随机线性变换技术。它的主要目

的是对高维数据进行降维处理，同时保留数据的一些关键特性。在数值线性代数中，对于大型线性方程组 $Ax=b$，其中 A 是一个大型矩阵，计数草图可以用于加速迭代求解算法。例如，通过对矩阵或向量等进行计数草图变换，在保证一定求解精度的前提下，减少计算量，加快收敛速度。

受文献[8]中使用计数草图加速最大加权残差 Kaczmarz 方法的启发，本文将计数草图与平均块 Kaczmarz 方法[9]相结合，提出了计数草图平均块 Kaczmarz 方法。

本文的结构安排如下：第一节分析了本文的研究背景；第二节具体讲述了计数草图平均块 Kaczmarz 方法的创新点和算法步骤；第三节进行数值实验分析。

2. 计数草图平均块 Kaczmarz 方法(CS-RaBK)

随机块 Kaczmarz 方法[10]需要求解子矩阵的摩尔－彭罗斯伪逆，这个计算耗时较长。因此，为了避免这一缺点，引入了平均块技术，Necoara [9]提出了随机平均块 Kaczmarz 方法。随机平均块 Kaczmarz 方法的迭代相当于随机 Kaczmarz 方法在不同方向上以一定步长进行更新的凸组合。随机平均块 Kaczmarz 的算法描述如下：

算法 1. 随机平均块 Kaczmarz 方法(RaBK)

输入：$A\in R^{m\times n}$，$b\in R^{m}$，初始值 x_0，步长序列 $\{\alpha_k\}_{k\geq 0}$，权重序列 $\{\omega_k\}_{k\geq 0}$

1) 令 $k=0,1,2\cdots$ 直到满足停止标准为止

2) 绘制样本 $J_k\sim P$ 并更新

3) $$x_{k+1}=x_k-\alpha_k\left(\sum_{i\in J_k}\omega_k^i\frac{A^{(i)}x_k-b^{(i)}}{\left\|A^{(i)}\right\|_2^2}\left(A^{(i)}\right)^*\right)$$

4) 结束

这里，$J_k=\left\{i_1^k,\cdots,i_{\tau_k}^k\right\}$ 是在第 k 次迭代时所选择的行对应的索引集合，P 表示在集合 $[m]$ 的索引子集的集合上的概率分布。

在该方法中，步长序列存在三种选择方式。其一为常数步长，即在整个迭代过程里，步长始终固定不变；其二是自适应步长，它会依据迭代过程中的相关信息，例如残差大小、迭代进展状况等动态地对步长加以调整；其三则是切比雪夫步长，此步长依据切比雪夫多项式的特性来确定，通过预估系数矩阵的特征值范围，将矩阵变换至适宜区间，进而构建步长序列。本文第三节主要对前两种步长的随机平均块 Kaczmarz 方法进行加速研究。

定义 1 [6] 计数草图变换：定义一个计数草图变换为 $S=\Phi D\in R^{d\times m}$。这里，$D$ 是一个 $m\times m$ 随机对角矩阵，每个对角项独立选择为 $+1$ 或 -1 的概率相等，$\Phi\in\{0,1\}^{d\times m}$ 是一个 $d\times m$ 二进制矩阵，其中 $\Phi_{h(i),i}=1$，其余所有项为 0，其中 $h:[m]\to[d]$ 是一个随机映射，使得对于每个 $i\in[m]$，$h(i)=j$，每个 $j\in[d]$ 的概率为 $1/d$。

尽管进行了降维，计数草图能够在一定程度上保留数据的结构特性。例如，在对高维数据进行聚类分析时，计数草图得到的低维表示仍然可以大致反映原始数据的聚类结构。这是因为计数草图在构造过程中，通过随机的哈希和符号分配，使得数据中的重要信息得以以一种近似的方式保留下来。

在上面算法的基础上，本文将计数草图技术与随机平均块 Kaczmarz 方法结合，产生新的方法。计数草图随机平均块 Kaczmarz 的算法描述如下：

算法 2. 计数草图随机平均块 Kaczmarz 方法(CS-RaBK)

输入：$A\in R^{m\times n}$，$b\in R^{m}$，初始值 x_0，步长序列 $\{\alpha_k\}_{k\geq 0}$，权重序列 $\{\omega_k\}_{k\geq 0}$

1) 引入计数草图矩阵 $S\in R^{d\times m}$， $d<m$

2) 计算 $\tilde{A}=SA\in R^{d\times n}$， $\tilde{b}=Sb\in R^{d}$

3) 令 $k=0,1,2\cdots$ 直到满足停止标准为止

4) 绘制样本 $J_k\sim P$ 并更新

5) $x_{k+1}=x_k-\alpha_k\left(\sum_{i\in J_k}\omega_k^i\dfrac{\tilde{A}^{(i)}x_k-\tilde{b}^{(i)}}{\left\|\tilde{A}^{(i)}\right\|_2^2}\left(\tilde{A}^{(i)}\right)^*\right)$

6) 结束

引理 1 [8]当计数草图随机平均块 Kaczmarz 方法的步长为常数步长或者自适应步长时，有以下收敛速率：

$$E\left[\left\|x^k-x_k^*\right\|^2\right]\leq\left(1-\frac{r}{m}\frac{\lambda_{\min}^{nz}\left(A^*A\right)}{\lambda_{\max}^{block}}\right)^k\left\|x^0-x_0^*\right\|^2$$

其中 $\lambda_{\min}^{nz}$ 表示给定矩阵的最小非零特征值，$\lambda_{\max}^{block}=\max\limits_{J\sim P}\lambda_{\max}\left(A_J^TA_J\right)$。

定理 1 [9]设 $S\in R^{d\times m}$ 是一个计数草图矩阵，其中 $d=\left(n^2+n\right)\Big/\left(\delta\varepsilon^2\right)$，这里 $0<\delta,\varepsilon<1$，对于计数草图随机平均块 Kaczmarz 方法的序列 $\left\{x^k\right\}_{k=0}^{\infty}$，以概率 $1-\delta$，有

$$E\left[\left\|x^k-x_k^*\right\|^2\right]\leq\left(1-\frac{r}{m}\frac{\lambda_{\min}^{nz}\left(ASSA\right)}{\lambda_{\max}^{block}}\right)^k\left\|x^0-x_0^*\right\|^2$$

成立。

3. 数值实验

在本节中，我们考虑以下 RaBK 变体[7]：

1) 具有均匀采样和自适应外推步长的 RaBK：$\alpha_k=1.95L_k$ 称为自适应步长的 RaBK (用 RaBK-a 表示)。

2) 具有均匀采样和恒定步长的 RaBK： $\alpha_k=1.95$ 称为常数步长的 RaBK (用 RaBK-c 表示)。

本节的所有实验都是在个人计算机上使用 MATLAB R2022a 进行的，计算机中央处理器为 2.50 GHz (Intel(R) Core(TM) i5-12500H CPU)，内存为 16.00 GB，操作系统为 Windows 11。值得注意的是，当测试方法运行 50 次时，“CPU”是平均运行时间。为了便于对测试方法的运行时间进行比较，我们定义

$$\mathrm{CPUspeedup1}=\frac{\mathrm{CPU\cdot of\cdot RaB}K-a}{\mathrm{CPU\cdot of\cdot CS-RaBK}-a}$$

$$\mathrm{CPUspeedup2}=\frac{\mathrm{CPU\cdot of\cdot RaBK}-c}{\mathrm{CPU\cdot of\cdot CS-RaBK}-c}$$

为了进一步比较收敛性能，在表 1 和表 2 中，分别列出了 CS-RaBK-a 方法、RaBK-a 方法、CS-RaBK-c 以及 RaBK-c 方法针对不同规模随机矩阵所耗费的 CPU 时间。在表 3 和表 4 中，分别列出了 CS-RaBK-a 方法、RaBK-a 方法、CS-RaBK-c 以及 RaBK-c 方法针对不同规模随机矩阵所需要的 IT 迭代次数。我们可以看到 CS-RaBK-a 方法和 CS-RaBK-c 方法对比 RaBK-a 和 RaBK-c 方法在运行时间上有显著提升，并且在我们的实验中，其 CPUspeedup1 和 CPUspeedup2 分别可高达 3.4315 和 9.5128。这主要因为计数草图

将矩阵 $A \in R^{m \times n}$ 降维成 $\tilde{A} \in R^{d \times n}$，大大减小了迭代过程的计算量，从而加快了运行时间。

Table 1. CPU numerical results of CS-RaBK-a, RaBK-a, CS-RaBK-c and RaBK-c for different sized matrices A
表 1. CS-RaBK-a、RaBK-a、CS-RaBK-c 和 RaBK-c 对不同规模矩阵 A 的 CPU 数值结果

$m \times n$	d	CS-RaBK-a	RaBK-a	CS-RaBK-c	RaBK-c
50000×50	20n	0.0438	0.0750	0.0563	0.0828
	30n	0.0594	0.0828	0.0359	0.0672
	40n	0.0453	0.0734	0.0313	0.1016
	50n	0.0625	0.0875	0.0422	0.0703
50000×100	10n	0.0875	0.1422	0.0156	0.1484
	20n	0.1094	0.1547	0.0766	0.1844
	40n	0.0609	0.1172	0.0313	0.1203
	50n	0.0750	0.1328	0.0828	0.1906
50000×150	10n	0.0906	0.2844	0.0922	0.1953
	20n	0.0953	0.2438	0.1266	0.2000
	30n	0.1063	0.2266	0.1438	0.2031

Table 2. CPU numerical results of CS-RaBK-a, RaBK-a, CS-RaBK-c and RaBK-c for different sized matrices A
表 2. CS-RaBK-a、RaBK-a、CS-RaBK-c 和 RaBK-c 对不同规模矩阵 A 的 CPU 数值结果

$m \times n$	d	CS-RaBK-a	RaBK-a	CS-RaBK-c	RaBK-c
500000×50	20n	0.1891	0.6000	0.2531	0.5766
	30n	0.2281	0.5531	0.2422	0.5844
	40n	0.2203	0.5250	0.1938	0.5750
	50n	0.2391	0.5984	0.2547	0.5719
500000×100	10n	0.3469	1.1904	0.3922	1.2594
	20n	0.5344	1.1547	0.5078	1.1938
	40n	0.5297	1.2422	0.5469	1.2047
	50n	0.4297	1.1813	0.4797	1.1719
500000×150	10n	0.6953	2.0109	0.7578	2.0703
	20n	0.7109	1.9313	0.7328	1.9578
	30n	0.8141	2.0266	0.8250	2.0516

Table 3. IT numerical results of CS-RaBK-a, RaBK-a, CS-RaBK-c and RaBK-c for different sized matrices A
表 3. CS-RaBK-a、RaBK-a、CS-RaBK-c 和 RaBK-c 对不同规模矩阵 A 的 IT 数值结果

$m \times n$	d	CS-RaBK-a	RaBK-a	CS-RaBK-c	RaBK-c
50000×50	20n	733.9800	606.1700	226.1800	182.7800
	30n	513.5600	453.7800	211.3800	180.4800
	40n	433.2700	388.9700	205.7900	179.4700
	50n	376.8900	358.4500	198.6800	178.2400

续表

50000×100	10*n*	2978.7600	2167.5000	564.3800	376.6700
	20*n*	1456.3300	1256.0700	458.9800	368.2400
	40*n*	736.2700	712.5600	389.7900	359.7400
	50*n*	627.5600	609.0400	376.3400	357.3600
50000×150	10*n*	1124.7000	1073.7800	587.3400	545.8000
	20*n*	808.7400	788.6000	564.0000	540.7000
	30*n*	715.7600	734.5800	559.6800	534.6200

Table 4. IT numerical results of CS-RaBK-a, RaBK-a, CS-RaBK-c and RaBK-c for different sized matrices A
表 4. CS-RaBK-a、RaBK-a、CS-RaBK-c 和 RaBK-c 对不同规模矩阵 A 的 IT 数值结果

m×*n*	*d*	CS-RaBK-a	RaBK-a	CS-RaBK-c	RaBK-c
500000×50	20*n*	716.8900	612.7000	227.5600	184.2200
	30*n*	423.6700	397.5600	203.8900	178.4700
	40*n*	349.3500	312.4600	189.6500	166.5800
	50*n*	314.6800	297.3900	156.4600	136.5400
500000×100	10*n*	1476.4600	1245.8900	455.7600	356.4500
	20*n*	768.3600	704.3600	399.7500	351.6400
	40*n*	568.5100	534.7700	380.2600	347.4500
	50*n*	526.5700	511.1600	374.2600	332.5400
500000×150	10*n*	1158.3800	1055.7600	598.5000	531.6200
	20*n*	859.2600	786.4600	565.0000	528.3700
	30*n*	706.6000	698.4700	552.2100	518.4900

4. 结束语

本文提出一种用于求解高度超定线性系统的计数草图随机平均块 Kaczmarz 方法，并通过数值实验验证了方法的可行性，从数值实验中发现 CS-RaBK-a 和 CS-RaBK-c 方法在 CPU 方面优于 RaBK-a 和 RaBK-c 方法，CS-RaBK-a 方法在相同精度下所需的运行时间最少。

参考文献

[1] Karczmarz, S. (1937) Angenaherte Auflosung von Systemen Linearer Gleichungen. *Bulletin International de l'Académie Polonaise des Sciences et des Lettres*, **35**, 355-357.

[2] Popa, C. and Zdunek, R. (2004) Kaczmarz Extended Algorithm for Tomographic Image Reconstruction from Limited-Data. *Mathematics and Computers in Simulation*, **65**, 579-598. https://doi.org/10.1016/j.matcom.2004.01.021

[3] Abdelazeem, M. and Gobashy, M.M. (2016) A Solution to Unexploded Ordnance Detection Problem from Its Magnetic Anomaly Using Kaczmarz Regularization. *Interpretation*, **4**, SH61-SH69. https://doi.org/10.1190/int-2016-0001.1

[4] Pasqualetti, F., Carli, R. and Bullo, F. (2012) Distributed Estimation via Iterative Projections with Application to Power Network Monitoring. *Automatica*, **48**, 747-758. https://doi.org/10.1016/j.automatica.2012.02.025

[5] Elfving, T. (1980) Block-Iterative Methods for Consistent and Inconsistent Linear Equations. *Numerische Mathematik*, **35**, 1-12. https://doi.org/10.1007/bf01396365

[6] Woodruff, D.P. (2014) Sketching as a Tool for Numerical Linear Algebra. *Foundations and Trends in Theoretical*

Computer Science, **10**, 1-157.

[7] Meng, X. and Mahoney, M.W. (2013) Low-Distortion Subspace Embeddings in Input-Sparsity Time and Applications to Robust Linear Regression. *Proceedings of the Forty-Fifth annual ACM symposium on Theory of Computing*, Palo Alto, 2-4 June 2013, 91-100. https://doi.org/10.1145/2488608.2488621

[8] Zhang, Y. and Li, H. (2021) A Count Sketch Maximal Weighted Residual Kaczmarz Method for Solving Highly Over-determined Linear Systems. *Applied Mathematics and Computation*, **410**, Article ID: 126486. https://doi.org/10.1016/j.amc.2021.126486

[9] Necoara, I. (2019) Faster Randomized Block Kaczmarz Algorithms. *SIAM Journal on Matrix Analysis and Applications*, **40**, 1425-1452. https://doi.org/10.1137/19m1251643

[10] Needell, D. and Tropp, J.A. (2014) Paved with Good Intentions: Analysis of a Randomized Block Kaczmarz Method. *Linear Algebra and its Applications*, **441**, 199-221. https://doi.org/10.1016/j.laa.2012.12.022