Advances in Applied Mathematics 应用数学进展, 2026, 15(2), 270-280
Published Online February 2026 in Hans. https://www.hanspub.org/journal/aam
https://doi.org/10.12677/aam.2026.152068

Hans汉斯

基于几何平滑动量的大规模线性方程组自适应确定性块Kaczmarz方法

马家靓[1*]，丁洁玉[2#]

[1]青岛大学数学与统计学院，山东 青岛
[2]青岛大学计算机科学技术学院，山东 青岛

收稿日期：2026年1月10日；录用日期：2026年2月4日；发布日期：2026年2月10日

摘　要

为高效求解大规模相容线性方程组，本文在已有自适应确定性块Kaczmarz (ADBK)方法的基础上，引入几何平滑动量加速机制，提出了基于几何平滑动量的自适应确定性块Kaczmarz方法。该方法继承了ADBK通过残差范数自适应选取块索引、避免显式计算子矩阵伪逆的计算优势，并在不增加额外计算复杂度的前提下进一步提升了收敛速度。数值实验结果表明，与原始ADBK方法及多种典型算法相比，所提出的方法在迭代次数、计算时间和收敛速度方面均具有明显优势。

关键词

线性系统，Kaczmarz迭代，块方法，动量加速

An Adaptive Deterministic Block Kaczmarz Algorithm Based on Geometrically Smoothed Momentum for Solving Large-Scale Linear Systems

Jialiang Ma[1*], Jieyu Ding[2#]

[1]School of Mathematics and Statistics, Qingdao University, Qingdao Shandong
[2]College of Computer Science and Technology, Qingdao University, Qingdao Shandong

Received: January 10, 2026; accepted: February 4, 2026; published: February 10, 2026

[*]第一作者。
[#]通讯作者。

文章引用: 马家靓, 丁洁玉. 基于几何平滑动量的大规模线性方程组自适应确定性块 Kaczmarz 方法[J]. 应用数学进展, 2026, 15(2): 270-280. DOI: 10.12677/aam.2026.152068

Abstract

To efficiently solve large-scale consistent systems of linear equations, this paper proposes an adaptive deterministic block Kaczmarz method based on geometric smooth momentum, building upon the existing Adaptive Deterministic Block Kaczmarz (ADBK) method. The proposed method inherits the computational advantages of ADBK in adaptively selecting block indices through residual norms and avoiding explicit computation of submatrix pseudoinverses. Furthermore, by introducing a geometric smooth momentum acceleration mechanism, the convergence speed is enhanced without increasing additional computational complexity. Numerical experimental results demonstrate that compared with the original ADBK method and various typical algorithms, the proposed method exhibits significant advantages in terms of iteration count, computational time, and convergence rate.

Keywords

System of Linear Equations, Kaczmarz Iteration, Block Method, Momentum Acceleration

1. 引言

线性方程组的高效求解是大数据分析、机器学习和图像处理等领域中高性能数值算法和软件开发的核心环节，常常受到问题规模、不相容等因素制约。因此，构造格式简单、内存耗费低和收敛速度快的迭代方法在大规模线性方程组、线性反问题和优化等问题中尤为重要。

波兰数学家 Stefan Kaczmarz 于 1937 年提出 Kaczmarz 方法[1]，用于求解线性方程组和线性不适定问题，如电子计算机断层扫描和信号处理[2]等。然而，Kaczmarz 方法的收敛性质依赖于行指标的选择顺序，收敛速度慢且很难做出收敛分析结果。在此方法基础上，2009 年 Strohmer 和 Vershynin [3]提出了基于工作行指标随机选取的随机 Kaczmarz (RK)方法，并证明了其依期望指数收敛，但是 RK 方法易受大型矩阵稀疏度的影响。

块 Kaczmarz (BK)方法最早可以追溯到 Elfving [4]的工作，是 Kaczmarz 方法的自然推广。理论分析和数值实验表明构造块 Kaczmarz 类方法是提升 Kaczmarz 方法和 RK 方法数值性能的有效途径。目前国内外学者主要从两方面来设计新型高效的块方法：一种是基于投影思想建立的 BK 方法[5]-[7]，这类方法在每一次迭代过程中都需要计算伪逆或求解一个最小二乘问题，且难以实施并行计算。另一种是基于免伪逆思想建立的 BK 方法，该类方法能避免投影类方法的主要缺陷。比较典型的方法有求解相容线性方程组的随机平均块 Kaczmarz (RaBK)方法[8]和快速确定块 Kaczmarz (FDBK)方法[9]。

本文研究基于免伪逆思想建立的 BK 方法，在自适应确定性块 Kaczmarz 方法的基础上将几何平滑动量运用到该方法中得到基于动量的块 Kaczmarz 方法。利用数值实验验证带几何平滑动量的自适应确定性块 Kaczmarz 方法的有效性和高效性。

2. 自适应确定性块 Kaczmarz 法

考虑如下的大规模线性方程组：

$$\boldsymbol{A}\boldsymbol{x}=\boldsymbol{b}, \tag{1}$$

其中 $\boldsymbol{A}\in\mathbb{R}^{m\times n}$，未知向量 $\boldsymbol{x}\in\mathbb{R}^{n}$ 和右端项 $\boldsymbol{b}\in\mathbb{R}^{m}$。对于相容的线性方程组，若解不唯一，往往关注方程组(1)的最小欧式范数解

$$\boldsymbol{x}_{LN}=\arg\min_{\boldsymbol{x}\in\mathbb{R}^{n}}\|\boldsymbol{x}\|_{2}，\text{使得 } \boldsymbol{A}\boldsymbol{x}=\boldsymbol{b}，$$

其中 $\|\cdot\|_{2}$ 表示欧几里得范数。对于不相容的线性方程组，一般考虑最小二乘解

$$\boldsymbol{x}_{LS}=\arg\min_{\boldsymbol{x}\in\mathbb{R}^{n}}\|\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}\|_{2}，$$

重点研究的是最小二乘最小范数解：$\boldsymbol{x}_{*}=\boldsymbol{A}^{\dagger}\boldsymbol{b}$。对于相容线性方程组，其最小二乘最小范数解就是它的最小范数解[10]。

2.1. 块 Kaczmarz 型方法

设 $[m]$ 表示集合 $\{1,2,\cdots,m\}$，如果一组索引集 $J_1,J_2,\cdots,J_p$ 满足：对于任意 $i\neq j(i,j=1,2,\cdots,p)$，有 $J_i\cap J_j=\varnothing$，并且 $\bigcup_{i=1}^{p}J_i=[m]$，那么可以称这组集合 $J_1,J_2,\cdots,J_p$ 是对集合 $[m]$ 的一个划分。给定一个正整数 p，可以将矩阵 $\boldsymbol{A}$ 的行索引划分为 $J=\{J_1,J_2,\cdots,J_p\}$，即将 $\boldsymbol{A}$ 表示为 $\boldsymbol{A}=\left[\boldsymbol{A}_{J_1}^{\mathrm{T}},\boldsymbol{A}_{J_2}^{\mathrm{T}},\cdots,\boldsymbol{A}_{J_p}^{\mathrm{T}}\right]^{\mathrm{T}}$，同时将向量 $\boldsymbol{b}$ 表示为 $\boldsymbol{b}=\left[\boldsymbol{b}_{J_1}^{\mathrm{T}},\boldsymbol{b}_{J_2}^{\mathrm{T}},\cdots,\boldsymbol{b}_{J_p}^{\mathrm{T}}\right]^{\mathrm{T}}$。在第 $k+1$ 次迭代中，选择一个子矩阵 $\boldsymbol{A}_{J_{i_k}}$ 和对应的子向量 $\boldsymbol{b}_{J_{i_k}}$，其中 $i_k=\mathrm{mod}(k,p)+1$。在该步中，当前迭代点 $\boldsymbol{x}_k$ 被正交投影到解空间 $\left\{\boldsymbol{x}\in\mathbb{R}^{n}:\boldsymbol{A}_{J_{i_k}}\boldsymbol{x}=\boldsymbol{b}_{J_{i_k}}\right\}$ 上。因此从初始向量 $\boldsymbol{x}_0$ 出发，块 Kaczmarz 方法的迭代格式可以表示为：

$$\boldsymbol{x}_{k+1}=\boldsymbol{x}_k+\boldsymbol{A}_{J_{i_k}}^{\dagger}\left(\boldsymbol{b}_{J_{i_k}}-\boldsymbol{A}_{J_{i_k}}\boldsymbol{x}_k\right)，\quad k=0,1,2,\cdots \tag{2}$$ \

其中 $\boldsymbol{A}_{J_{i_k}}^{\dagger}$ 表示矩阵 $\boldsymbol{A}_{J_{i_k}}$ 的 Moore-Penrose 伪逆。

在给定集合 $[m]$ 的一个预定划分 J 之后，Needell 和 Tropp [5]提出随机块 Kaczmarz (RBK)方法。为降低计算复杂度，Necoara [8]提出了一个统一的随机平均块 Kaczmarz (RaBK)方法，该方法不需要计算伪逆，其迭代格式如下：

$$\boldsymbol{x}_{k+1}=\boldsymbol{x}_k+\alpha_k\left(\sum_{i\in J_{i_k}}\omega_i^{k}\frac{\boldsymbol{b}^{(i)}-\boldsymbol{A}^{(i)}\boldsymbol{x}_k}{\left\|\boldsymbol{A}^{(i)}\right\|_{2}^{2}}\left(\boldsymbol{A}^{(i)}\right)^{\mathrm{T}}\right), \tag{3}$$

其中 $\omega_i^{k}\in[0,1]$ 为权重，满足 $\sum_{i\in J_{i_k}}\omega_i^{k}=1$，$\alpha_k\in(0,2)$ 为步长。为了进一步提高收敛速度，Chen 和 Huang 提出了一个快速确定性块 Kaczmarz (FDBK)方法[9]，数值实验表明，FDBK 方法在性能上优于 RaBK 方法。

2.2. 自适应确定性块 Kaczmarz 方法

首先提出一种基于残差向量的欧几里得范数的新指标集序列

$$U_k=\left\{i_k\,\middle|\,\left|\boldsymbol{b}^{(i_k)}-\boldsymbol{A}^{(i_k)}\boldsymbol{x}_k\right|^{2}\geq\frac{\|\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}_k\|_{2}^{2}}{m}\right\}, \tag{4}$$

上述指标序列 U_k 非空，具体内容见算法 1。

算法 1. ADBK 方法[10]

1：输入： $\boldsymbol{A},\boldsymbol{b},l,\boldsymbol{x}_0\in range\left(\boldsymbol{A}^{\mathrm{T}}\right)$；

2：输出： $\boldsymbol{x}_l$

3：**for** $k=0,1,\cdots,l-1$ **do**

4：确定指标集序列

$$U_k=\left\{i_k\,|\left|\boldsymbol{b}^{(i_k)}-\boldsymbol{A}^{(i_k)}\boldsymbol{x}_k\right|^2\geq\frac{\left\|\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}_k\right\|_2^2}{m}\right\};$$

5：计算

$$\eta_k=\sum_{i_k\in U_k}\left(\boldsymbol{b}^{(i_k)}-\boldsymbol{A}^{(i_k)}\boldsymbol{x}_k\right)\boldsymbol{e}_{i_k}; \tag{5}$$

6：计算

$$\boldsymbol{x}_{k+1}=\boldsymbol{x}_k+\frac{\eta_k^{\mathrm{T}}\left(\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}_k\right)}{\left\|\boldsymbol{A}^{\mathrm{T}}\eta_k\right\|_2^2}\boldsymbol{A}^{\mathrm{T}}\eta_k. \tag{6}$$

7：**end for**

定理 2.2.1. 对 $\boldsymbol{A}^{\mathrm{T}}$ 列空间中的任意初始猜测向量 $\boldsymbol{x}_0$，由 ADBK 方法生成的迭代解序列 $\left\{\boldsymbol{x}_k\right\}_{k=0}^{\infty}$ 收敛到相容线性方程组(1)的最小范数解 $\boldsymbol{x}_*=\boldsymbol{A}^{\dagger}\boldsymbol{b}$，且迭代解序列 $\left\{\boldsymbol{x}_k\right\}_{k=0}^{\infty}$ 的误差满足

$$\left\|\boldsymbol{x}_{k+1}-\boldsymbol{x}_*\right\|_2^2\leq\left(1-\frac{\mu\lambda_{\min}\left(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\right)}{m\lambda_{\max}^{block}}\right)\left\|\boldsymbol{x}_k-\boldsymbol{x}_*\right\|_2^2, \tag{7}$$

其中 $\mu=\inf_k\left\{N\left(U_k\right)\right\}$，$N\left(U_k\right)$ 表示指标集 U_k 中所含元素的个数以及 $\lambda_{\max}^{block}=\max_{U_k}\left\{\lambda_{\max}\left(\boldsymbol{A}_{U_k}\boldsymbol{A}_{U_k}^{\mathrm{T}}\right)\right\}$。

3. 基于几何平滑动量的自适应确定性块 Kaczmarz 方法

考虑使用几何平滑动量来加快 ADBK 方法的收敛速度。类似于文献[11]中的工作，对于参数 $\beta\in[0,1)$ 和 $M\in[0,1]$，定义几何平滑动量 Kaczmarz 方法的算法，其迭代形式如下：

$$\begin{cases}\boldsymbol{x}_{k+1}=\boldsymbol{x}_k+\dfrac{\boldsymbol{b}^{(i_k)}-\boldsymbol{A}^{(i_k)}\boldsymbol{x}_k}{\left\|\boldsymbol{A}^{(i_k)}\right\|_2^2}\left(\boldsymbol{A}^{(i_k)}\right)^{\mathrm{T}}+M\boldsymbol{y}_k\\ \boldsymbol{y}_{k+1}=\beta\boldsymbol{y}_k+(1-\beta)\left(\boldsymbol{x}_{k+1}-\boldsymbol{x}_k\right)\end{cases}, \tag{8}$$

下面给出几何平滑动量 Kaczmarz (KGSM)方法的收敛理论。

定理 3.1. 固定参数 $\beta\in[0,1)$，$M\in[0,1]$，以及 $l\in\{1,\cdots,n\}$。设 $\boldsymbol{v}_l$ 是矩阵 A 的第 l 大奇异值 σ_l 所对应的右奇异向量。假设 $\boldsymbol{x}_k$ 是由 KGSM 方法所定义的。那么对于所有 $k\geq0$, 有

$$\mathbb{E}\left\langle\boldsymbol{x}_{k+1}-\boldsymbol{x},\boldsymbol{v}_l\right\rangle=\begin{bmatrix}r\\ \varsigma\end{bmatrix}^{\mathrm{T}}\begin{bmatrix}r & \varsigma\\ -1 & \beta\end{bmatrix}^k\begin{bmatrix}1\\ -1/(1-\beta)\end{bmatrix}\left\langle\boldsymbol{x}_0-\boldsymbol{x},\boldsymbol{v}_l\right\rangle,$$

其中

$$r:=1-\frac{\sigma_l^2}{\left\|\boldsymbol{A}\right\|_F^2}+M(1-\beta),\ \varsigma:=M(1-\beta)^2。$$

当 β 接近于 1 时，动量项会变得高度平滑。所以将几何平滑动量更新公式嵌入 ADBK 方法的更新迭代公式中有

$$\begin{cases} \boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \dfrac{\eta_k^{\mathrm{T}}\left(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k\right)}{\left\|\boldsymbol{A}^{\mathrm{T}}\eta_k\right\|_2^2}\boldsymbol{A}^{\mathrm{T}}\eta_k + M\boldsymbol{y}_k \\ \boldsymbol{y}_{k+1} = \beta\boldsymbol{y}_k + (1-\beta)\left(\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\right) \end{cases} \tag{9}$$

综上，可得带几何平滑动量的 ADBK (gsmADBK)方法，具体内容见如下算法 2。

算法 2. gsmADBK 方法

1：输入： $\boldsymbol{A}, \boldsymbol{b}, l, 0 \le \beta \le 1$，并且 $\boldsymbol{x}_0 \in range\left(\boldsymbol{A}^{\mathrm{T}}\right)$， $\boldsymbol{y}_0 = \boldsymbol{x}_0$；

2：输出： $\boldsymbol{x}_l$

3：**for** $k = 0,1,\cdots,l-1$ **do**

4：确定指标集序列

$$U_k = \left\{ i_k \,\middle|\, \left|\boldsymbol{b}^{(i_k)} - \boldsymbol{A}^{(i_k)}\boldsymbol{x}_k\right|^2 \ge \frac{\left\|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k\right\|_2^2}{m} \right\};$$

5：计算

$$\eta_k = \sum_{i_k \in U_k} \left(\boldsymbol{b}^{(i_k)} - \boldsymbol{A}^{(i_k)}\boldsymbol{x}_k\right)\boldsymbol{e}_{i_k};$$

6：计算

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \frac{\eta_k^{\mathrm{T}}\left(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k\right)}{\left\|\boldsymbol{A}^{\mathrm{T}}\eta_k\right\|_2^2}\boldsymbol{A}^{\mathrm{T}}\eta_k + M\boldsymbol{y}_k$$

7：计算

$$\boldsymbol{y}_{k+1} = \beta\boldsymbol{y}_k + (1-\beta)\left(\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\right).$$

8：**end for**

注记 3.1. 当参数 $M = 0$ 时，gsmADBK 方法退化为 ADBK 方法。

gsmADBK 方法的收敛性分析由于 η_k 的选择不确定，因此给出一个准确的收敛速度的上界是困难的，有待今后的进一步研究。

4. 数值实验

下面通过几组数值算例来验证 gsmADBK 方法求解大规模相容线性方程组的有效性以及高效性，实验均通过 MATLAB (版本 R2020b)编程实现，初始向量均设为 $\boldsymbol{x}_0 = zeros(n,1)$。停机准则是相对解误差满足

$$RSE = \frac{\left\|\boldsymbol{x}_k - \boldsymbol{x}_*\right\|_2}{\left\|\boldsymbol{x}_*\right\|_2} \le 10^{-3}$$

或者迭代次数超过 100,000 次，或者计算时间超过 3600 秒。在后两种情况下，相应的 IT 迭代次数和 CPU 时间记为“--”。IT 和 CPU 是 50 次重复计算所需迭代步数和 CPU 时间的平均值(单位：秒)。令右端向量 $\boldsymbol{b} = \boldsymbol{A}\boldsymbol{x}_*$, $\boldsymbol{x}_* \in \mathbb{R}^n$ 由 MATLAB 函数 randn 随机生成。

4.1. 超定和欠定线性方程组

首先考虑第一类系数矩阵高斯矩阵，由 MATLAB 函数 randn 随机生成，即 $\boldsymbol{A}=randn(m,n)$。使用 RaBK、FDBK、ADBK 和 gsmADBK 方法求解大规模相容线性方程组 $\boldsymbol{Ax}=\boldsymbol{b}$，其中 RaBK 方法的采样方式为分块采样，块大小为 $\left\lceil m/\|\boldsymbol{A}\|_2^2 \right\rceil$，步长 $\alpha_k=1.95L_k$，$L_k=\dfrac{\sum_{i\in J_k}\bar{\omega}_i^k\left(\boldsymbol{A}^{(i)}\boldsymbol{x}_k-b^{(i)}\right)^2}{\left\|\sum_{i\in J_k}\bar{\omega}_i^k\left(\boldsymbol{A}^{(i)}\boldsymbol{x}_k-b^{(i)}\right)^2\left(\boldsymbol{A}^{(i)}\right)^{\mathrm{T}}\right\|_2^2}$，$\bar{\omega}_i^k=\dfrac{\omega_i^k}{\left\|\boldsymbol{A}^{(i)}\right\|_2^2}$。数值结果由表 1 和表 2 给出，其中 gsmADBK 方法采用的参数是数值实验过程中确定的最优参数，即通过 CPU 值最小获取。为了显示收敛程度，gsmADBK 方法对其他方法的加速定义为

$$\text{speed-up\_ADBK}=\frac{\text{CPU of Method}}{\text{CPU of gsmADBK}}。$$

由表 1 和表 2 的数值结果可以看出，RaBK、FDBK、ADBK 和 gsmADBK 方法求解超定及欠定线性方程组(1)都是有效的。从迭代步数和计算时间来看，gsmADBK 方法优于 RaBK、FDBK 和 ADBK 方法，speed-up_RaBK、speed-up_FDBK 和 speed-up_ADBK 的取值范围分别为 895.56~5419.41、8.69~103.16、1.05~3.39，即增加动量项给 ADBK 方法的收敛特性带来有效的改进并且优于传统方法。

Table 1. Numerical results of four iterative methods when $\boldsymbol{A}=randn(m,n)$, $m>n$

表 1. $\boldsymbol{A}=randn(m,n)$，$m>n$ 时四种迭代方法的数值结果

m		1000	3000	5000	7000	9000	12,000	15,000
n		500	1000	2000	3500	5000	6500	8000
RaBK	IT	12,085	12,606	32,123	86,189	>105	>105	>105
	CPU	2.1848	16.6574	173.0970	1243.2086	--	--	--
FDBK	IT	278	128	230	446	689	636	650
	CPU	0.0289	0.1617	1.2585	5.2504	15.0544	23.5612	36.3798
ADBK	IT	70	32	42	70	100	87	88
	CPU	0.0062	0.0378	0.2238	0.8485	2.1926	3.2254	4.9407
gsmADBK	IT	23	15	17	23	29	28	28
	CPU	0.0022	0.0186	0.0915	0.2919	0.6470	1.0641	1.6170
	M_{opt}	0.5	0.3	0.4	0.5	0.6	0.6	0.5
	β_{opt}	0.2	0.2	0.1	0.2	0.1	0.1	0.2
speed-up_RaBK		993.09	895.56	1891.77	4259.02	--	--	--
speed-up_FDBK		13.14	8.69	13.75	17.99	23.27	22.14	22.50
speed-up_ADBK		2.82	2.03	2.45	2.91	3.39	3.03	3.06

Table 2. Numerical results of four iterative methods when $\boldsymbol{A}=randn(m,n)$, $m<n$

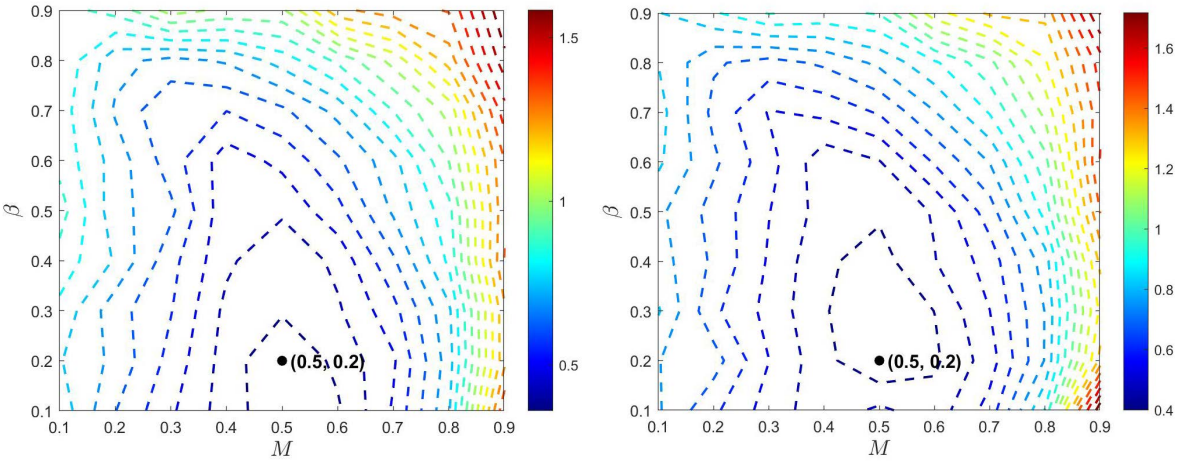
表 2. $\boldsymbol{A}=randn(m,n)$，$m<n$ 时四种迭代方法的数值结果

m		500	1000	2000	3500	5000	6500	8000
n		10,000	3000	5000	7000	9000	12,000	15,000
RaBK	IT	11,740	12,467	32,609	86,022	>105	>105	>105

续表

	CPU	4.3706	39.2750	319.8674	1764.5594	--	--	--
FDBK	IT	320	174	269	506	753	709	721
	CPU	0.2579	0.3220	1.7888	6.4280	17.2074	26.1081	40.1144
ADBK	IT	70	32	42	70	100	87	88
	CPU	0.0065	0.0372	0.2247	0.8590	2.1213	3.3347	4.7675
gsmADBK	IT	23	15	17	23	29	28	28
	CPU	0.0025	0.0226	0.1096	0.3256	0.7025	1.1920	1.7674
	M_{opt}	0.5	0.4	0.4	0.5	0.6	0.6	0.5
	β_{opt}	0.3	0.2	0.2	0.2	0.2	0.1	0.3
speed-up_RaBK		1748.24	1737.83	2918.50	5419.41	--	--	--
speed-up_FDBK		103.16	14.25	16.32	19.74	24.49	21.90	22.70
speed-up_ADBK		2.60	1.65	1.05	2.64	3.02	2.80	2.70

图 1 展示了两个不同参数 M 和 β 对 gsmADBK 方法计算效率的影响。确定 gsmADBK 方法的最优参数主要是通过计算不同参数对应的 CPU 值进行选择的。



(a) $m=7500, n=3000$ (b) $m=3500, n=7000$

Figure 1. CPU values of solving overdetermined and underdetermined linear systems using gsmADBK method with different (M,β)

图 1. 不同(M,β)的 gsmADBK 方法求解超定和欠定线性方程组的 CPU 值

在图 2 和图 3 中，对于超定和欠定线性方程组，有 RSE 与迭代步数(左)和计算时间(右)的关系曲线。从中反映出 gsmADBK 方法收敛最快，综上所述，gsmADBK 方法的数值性能表现优秀。

4.2. 稀疏线性方程组

考虑第二类系数矩阵稀疏矩阵，均是从 SuiteSparse 矩阵集合[12]中选取的。表 3 和表 4 分别列出了六个细长满秩矩阵和六个扁平满秩矩阵的相关信息，表 5 列出了六个秩亏稀疏矩阵的相关信息。使用 RaBK、FDBK、ADBK、和 gsmADBK 方法求解初始向量为 $\boldsymbol{x}_0$ 的大规模相容线性方程组 $\boldsymbol{Ax}=\boldsymbol{b}$。数值结果由表

6，表 7 以及表 8 给出，其中不同方法中采用的参数是数值实验过程中确定的最优参数，即通过 CPU 值最小获取。

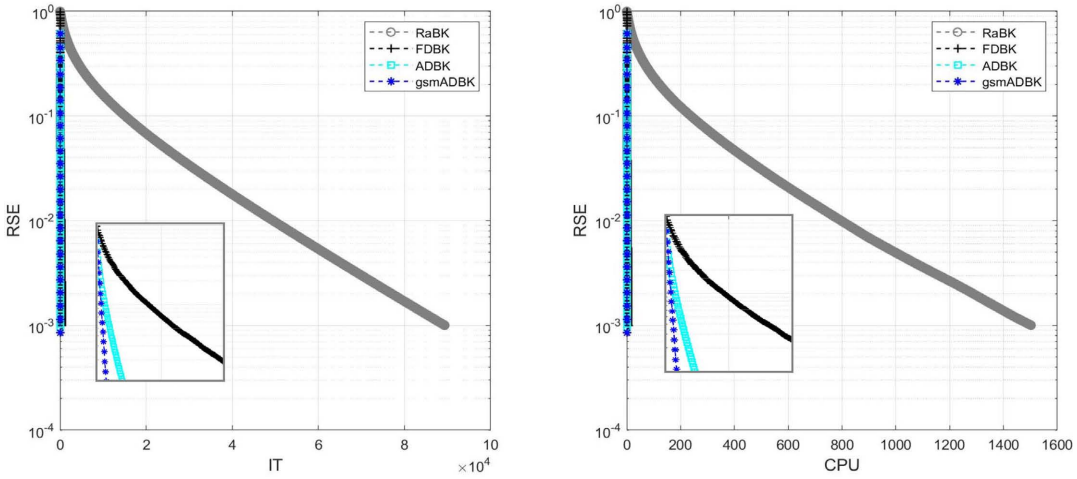


Figure 2. The relationship of RSE with IT (left) and CPU (right) for RaBK, FDBK, ADBK and gsmADBK methods of $\boldsymbol{A} = randn(7000,3500)$

图 2. $\boldsymbol{A} = randn(7000,3500)$ 的 RaBK、FDBK、ADBK 和 gsmADBK 方法的 RSE 与 IT (左)和 CPU (右)的关系

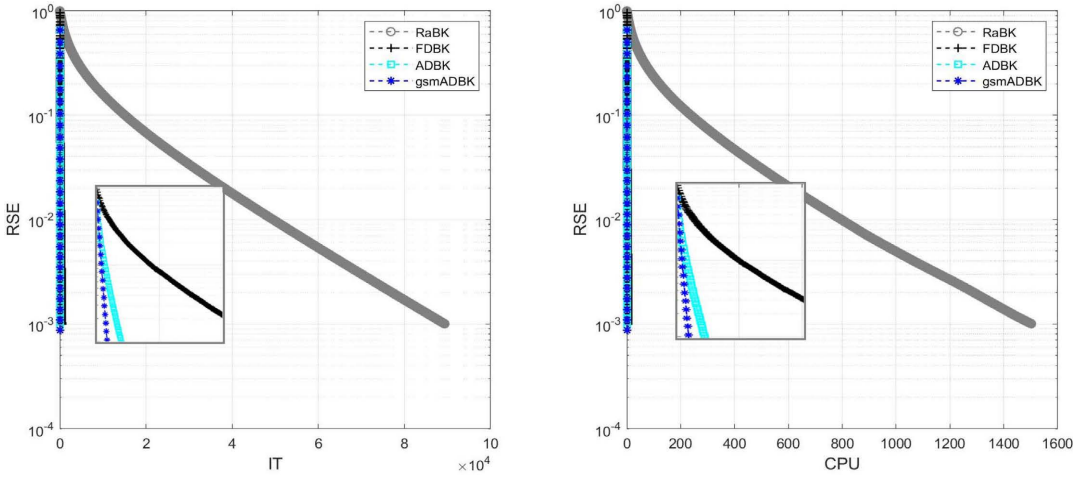


Figure 3. The relationship of RSE with IT (left) and CPU (right) for RaBK, FDBK, ADBK and gsmADBK methods of $\boldsymbol{A} = randn(3500,7000)$

图 3. $\boldsymbol{A} = randn(3500,7000)$ 的 RaBK、FDBK、ADBK 和 gsmADBK 方法的 RSE 与 IT (左)和 CPU (右)的关系

Table 3. Related properties of full-rank sparse matrices, $m \geq n$
表 3. 满秩稀疏矩阵的相关性质，$m \geq n$

Name	Cities	ash219	ash331	ash608	cage9	cage10
$m \times n$	55 × 46	219 × 85	331 × 104	608 × 188	3534 × 3534	11,397 × 11,397
density	53.04%	2.35%	1.92%	1.06%	0.33%	0.16%
cond(A)	207.15	3.02	3.10	3.37	12.60	11.02

Table 4. Related properties of full-rank sparse matrices, $m < n$
表 4. 满秩稀疏矩阵的相关性质，$m < n$

Name	Trec8	crew1	bibd_17_8	model1	nemsafm	flower_5_4
$m \times n$	23 × 84	135 × 6469	136 × 24,310	362 × 798	334 × 2348	5226 × 14,721
density	39.44%	5.38%	20.59%	1.05%	0.36%	0.057%
cond(A)	26.89	18.20	9.04	17.57	4.77	14.9

Table 5. Relevant properties of low-rank sparse matrices
表 5. 秩亏稀疏矩阵的相关性质

Name	relat5	relat6	rel5	rel6	Sandi_sandi	flower_4_4
$m \times n$	340 × 35	2340 × 157	340 × 35	2340 × 157	314 × 360	1837 × 5529
density	8.89%	2.21%	5.51%	1.39%	0.54%	0.20%
cond(A)	Inf	Inf	Inf	Inf	1.47e+17	1.05e+19

Table 6. Numerical results for full-rank sparse matrices $\boldsymbol{A}$, $m \geq n$
表 6. 满秩稀疏矩阵 $\boldsymbol{A}$，$m \geq n$ 时的数值结果

Name		Cities	ash219	ash331	ash608	cage9	cage10
RaBK	IT	112488	846	814	1970	31,359	>105
	CPU	0.1753	0.0111	0.0138	0.0810	438.8697	--
FDBK	IT	45593	46	30	50	356	290
	CPU	0.4229	4.7216e−04	3.6335e−04	7.6797e−04	0.0698	0.6507
ADBK	IT	67770	21	19	23	252	107
	CPU	0.1366	5.5380e−05	7.2638e−05	8.5800e−05	0.0140	0.0187
gsmADBK	IT	2084	12	11	13	52	37
	CPU	0.0045	3.2180e−05	3.3107e−05	5.5590e−05	0.0032	0.0063
	M_{opt}	0.9	0.2	0.2	0.3	0.7	0.7
	β_{opt}	0.6	0.1	0.1	0.2	0.4	0.4
speed-up_RaBK		38.96	344.93	416.83	1457.10	137146.78	--
speed-up_FDBK		93.98	14.67	10.98	13.81	21.81	103.29
speed-up_ADBK		30.36	1.72	2.19	1.54	4.38	2.97

Table 7. Numerical results for full-rank sparse matrices $\boldsymbol{A}$, $m < n$
表 7. 满秩稀疏矩阵 $\boldsymbol{A}$，$m < n$ 时的数值结果

Name		Trec8	crew1	bibd_17_8	model1	nemsafm	flower_5_4
RaBK	IT	2303	7478	1293	11,357	2614	>105
	CPU	0.0046	2.0872	2.4240	1.5826	1.1650	--
FDBK	IT	1333	808	257	646	136	3501
	CPU	0.0111	0.1113	2.5492	0.0136	0.0036	0.9751
ADBK	IT	1240	319	119	375	56	463
	CPU	0.0019	0.0182	0.0492	0.0021	0.0007	0.0515

续表

gsmADBK	IT	134	92	38	69	24	57
	CPU	0.0004	0.0055	0.0174	0.0004	0.0003	0.0063
	M_{opt}	0.8	0.8	0.6	0.8	0.5	0.8
	β_{opt}	0.5	0.2	0.1	0.2	0.1	0.2
speed-up_RaBK		11.50	379.49	139.31	3956.50	3883.33	--
speed-up_FDBK		27.75	20.24	146.51	34.00	12.00	154.78
speed-up_ADBK		4.75	3.31	2.83	5.25	2.33	8.17

由表 6 和表 7 的数值结果可以得出以下结论：

1) RaBK、FDBK、ADBK 和 gsmADBK 方法求解满秩稀疏线性方程组 $\boldsymbol{Ax}=\boldsymbol{b}$ 都是有效的。

2) RaBK、FDBK、ADBK 和 gsmADBK 方法呈现出与表 1 和表 2 中类似的数值结果。在所有测试算例求解中，gsmADBK 方法消耗的迭代步数和计算时间优于 RaBK、FDBK 和 ADBK 方法，speed-up_RaBK、speed-up_FDBK、speed-up_ADBK 的取值范围分别为 11.50~137146.78、10.98~154.78、1.54~30.96，表明添加动量后加速效果显著且优于传统方法。但搜索不同稀疏矩阵的最优参数是较为耗时的。

Table 8. Numerical results of low-rank sparse matrices
表 8. 秩亏稀疏矩阵的数值结果

Name		relat5	relat6	rel5	rel6	Sandi_sandi	flower_4_4
FDBK	IT	75	252	79	322	505	561
	CPU	0.0009	0.0122	0.0008	0.0117	0.0059	0.0481
ADBK	IT	58	84	70	192	353	147
	CPU	0.0002	0.0010	0.0002	0.0019	0.0011	0.0048
gsmADBK	IT	22	28	34	38	88	39
	CPU	6.6070e−05	0.0004	8.4310e−05	0.0004	0.0003	0.0014
	M_{opt}	0.4	0.6	0.4	0.7	0.8	0.6
	β_{opt}	0.3	0.2	0.2	0.1	0.3	0.3
speed-up_FDBK		13.62	30.50	9.49	29.25	19.67	34.36
speed-up_ADBK		3.03	2.50	2.37	4.75	3.67	3.43

由表 8 的数值结果可以得出以下结论：

1) RaBK 方法无法有效求解病态秩亏稀疏线性方程组，问题在于秩亏系数矩阵使 RaBK 方法步长的选择失效。其他 FDBK、ADBK 和 gsmADBK 方法都能够有效求解病态秩亏稀疏线性方程组。

2) 在所有测试算例求解中，gsmADBK 方法消耗的迭代步数和计算时间优于 FDBK 和 ADBK 方法，speed-up_FDBK、speed-up_ADBK 的取值范围分别为 9.49~34.36 和 2.37~4.75，表明添加动量加速效果显著且优于传统方法。但搜索不同稀疏矩阵的最优参数是较为耗时的。

5. 结论

本文基于数值代数与随机优化等领域的理论知识，在自适应确定性块 Kaczmarz (ADBK)方法的基础

上结合动量思想提出了基于几何平滑动量的自适应确定性块 Kaczmarz (gsmADBK)方法。将该方法应用于超定、欠定及稀疏线性方程组的求解问题，数值实验结果表明该方法能够有效求解大规模相容线性方程组并且引入动量项能够显著改善 ADBK 方法的收敛特性，在迭代次数、计算时间以及收敛速度方面均取得提升。

基金项目

国家自然科学基金(12172186: 11772166)。

参考文献

[1] Kaczmarz, S. (1937) Angenaherte auflosung von systemen linearer glei-chungen. Bulletin International de l'Académie des Sciences de Cracovie, Classe des Sciences Mathématiques et Naturelles, 355-357.

[2] Herman, G.T. and Davidi, R. (2008) Image Reconstruction from a Small Number of Projections. *Inverse Problems*, **24**, Article ID: 045011. https://doi.org/10.1088/0266-5611/24/4/045011

[3] Strohmer, T. and Vershynin, R. (2008) A Randomized Kaczmarz Algorithm with Exponential Convergence. *Journal of Fourier Analysis and Applications*, **15**, 262-278. https://doi.org/10.1007/s00041-008-9030-4

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

[5] 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

[6] Niu, Y. and Zheng, B. (2020) A Greedy Block Kaczmarz Algorithm for Solving Large-Scale Linear Systems. *Applied Mathematics Letters*, **104**, Article ID: 106294. https://doi.org/10.1016/j.aml.2020.106294

[7] Rebrova, E. and Needell, D. (2020) On Block Gaussian Sketching for the Kaczmarz Method. *Numerical Algorithms*, **86**, 443-473. https://doi.org/10.1007/s11075-020-00895-9

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

[9] Chen, J. and Huang, Z. (2021) On a Fast Deterministic Block Kaczmarz Method for Solving Large-Scale Linear Systems. *Numerical Algorithms*, **89**, 1007-1029. https://doi.org/10.1007/s11075-021-01143-4

[10] Tan, L., Guo, X., Deng, M. and Chen, J. (2025) On the Adaptive Deterministic Block Kaczmarz Method with Momentum for Solving Large-Scale Consistent Linear Systems. *Journal of Computational and Applied Mathematics*, **457**, Article ID: 116328. https://doi.org/10.1016/j.cam.2024.116328

[11] Alderman, S.J., Luikart, R.W. and Marshall, N.F. (2024) Randomized Kaczmarz with Geometrically Smoothed Momentum. *SIAM Journal on Matrix Analysis and Applications*, **45**, 2287-2313. https://doi.org/10.1137/24m1633820

[12] Davis, T.A. and Hu, Y. (2011) The University of Florida Sparse Matrix Collection. *ACM Transactions on Mathematical Software*, **38**, 1-25. https://doi.org/10.1145/2049662.2049663