文章编号：1673-3193(2025)05-0661-06　　J. North Univ. China, Nat. Sci. Ed., 2025, 46(5): 661-666.

求解大型稀疏矩阵方程组的SPIKE算法

秦芳芳[1]，左沐雨[2]，季一木[3]

（1. 南京邮电大学 理学院，江苏 南京 210023；2. 南京航空航天大学 数学学院，江苏 南京 210016；
3. 南京邮电大学 计算机学院，江苏 南京 210023）

摘　要：不同于传统的LU分解算法和QR分解算法，本文研究了一种新的基于DS矩阵分解的递归SPIKE算法。SPIKE算法采用了一种新颖的分解方法来平衡通信和算法开销，相比其他方法在现代并行架构上有更好的延展性。首先，从系数矩阵的分块、DS分解、简化系数矩阵方程组的提取和求解四方面介绍了递归SPIKE算法的工作原理。然后，首次将其应用到具体的系数矩阵规模不同的线性方程组中，并与LU分解算法与QR分解算法进行了比较。三组数值实验分别给出了各个求解算法的结果和运行时间。实验结果表明，递归SPIKE算法不仅能够求解得到准确结果，而且求解速度更快。数值案例表明，递归SPIKE算法所需的计算时间约为LU算法的40%，约为QR分解算法的8%。

关键词：一般带状矩阵；三对角矩阵；DS矩阵分解；递归SPIKE算法
中图分类号：O24　　**文献标识码：**A　　**doi:** 10.62756/jnuc.issn.1673-3193.2023.07.0004
引用格式：秦芳芳，左沐雨，季一木．求解大型稀疏矩阵方程组的SPIKE算法[J]．中北大学学报（自然科学版），2025，46(5)：661-666.
QIN Fangfang, ZUO Muyu, JI Yimu. The SPIKE algorithm for solving large sparse matrix equations[J]. Journal of North University of China(Natural Science Edition), 2025, 46(5): 661-666.

The SPIKE Algorithm for Solving Large Sparse Matrix Equations

QIN Fangfang[1], ZUO Muyu[2], JI Yimu[3]

（1. School of Science, Nanjing University of Posts and Telecommunications, Nanjing 210003, China;
2. School of Mathematics, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China;
3. School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing 210023, China）

Abstract: A recursive SPIKE algorithm was presented based on DS matrix decomposition, which was very different from LU decomposition algorithm and QR decomposition algorithm. The SPIKE algorithm used a novel decomposition method to balance communication overhead with arithmetic cost to achieve better scalability than other methods on modern parallel architectures. Firstly, the basic operating principle of a recursive SPIKE algorithm was introduced by four aspects: partitioning coefficient matrix, DS decomposition, extraction and solution of simplified coefficient matrix equations. The SPIKE algorithm was applied to specific linear equations with different coefficient matrix for the first time. Three sets of numerical experiments gave the results and running time of each solution algorithm. The experimental results show that the recursive SPIKE algorithm can not only obtain accurate results, but also has a faster solution speed. Experiments indicate the computing time of the proposed algorithms is about 40 percent of LU

收稿日期：2023-07-06
基金项目：国家自然科学基金资助项目(11801281)；江苏省博士后科研资助项目(2020Z380)
作者简介：秦芳芳(1988－)，女，讲师，博士，主要从事偏微分方程数值解、科学工程计算等的研究。E-mail：ffqin@njupt.edu.cn。
通信作者：季一木(1978－)，男，教授，博士，主要从事网格计算和云计算相关技术的研究。E-mail:jiym@njupt.edu.cn。

decomposition algorithm, and about 8 percent of QR decomposition algorithm.

Key words: general banded matrix; tridiagonal matrix; DS matrix decomposition; recursive SPIKE algorithm

0 引 言

偏微分方程在实际工程应用中起着至关重要的作用，它经常出现在计算机科学和工程等领域，如有限元分析[1]、计算力学(流体、结构和流体结构的相互作用)[2-6]、计算纳米电子学[7]、计算机视觉[8]。然而，对于偏微分方程，很难求得其解析解，于是人们转而求解其数值近似解。通常通过离散化方法将大型稀疏矩阵化为系数矩阵的线性方程组来求解。大型稀疏矩阵大多为带宽较窄的大型带状矩阵。带状矩阵是指一个矩阵中只有对角线附近的元素可以是非零的，而其他位置的元素均为零。这种矩阵结构的特点是它拥有较少的非零元素，因此储存和计算时占用的资源较少，这使得它在实际问题的求解中具有很大优势。虽然求解带状矩阵线性方程组所需的储存和计算量相比一般的矩阵线性方程组要小，但是在解带状矩阵线性方程组时，当前主流的直接求解方法，如LU分解算法[9]和QR分解算法[10]，由于要处理大量的零元素，效率往往较低。因此，如何高效快速准确地求解带状矩阵线性方程组是一个重要的研究课题。

本文主要探究一种新的基于DS矩阵分解的求解大型带状矩阵线性方程组的递归SPIKE算法[11-15]的性能。2001年，基于DS矩阵分解的递归SPIKE算法问世，该算法通过对系数矩阵的分块形式进行简化分解来降低计算复杂度。2008年，递归SPIKE算法引入随机的DS矩阵分解来增加算法的随机性，从而进一步提高算法的计算速度。2020年，Braegan S. Spring突破了递归SPIKE算法求解大型带状矩阵线性方程组时系数矩阵阶数为2^n的限制，使其能够求解更为一般的大型带状矩阵线性方程组[11]。本文首次将递归SPIKE算法运用于具体的数值算例，并与当前主流的直接算法LU分解算法和QR分解算法进行比较，以此来检验基于DS矩阵分解的递归SPIKE算法的优越性，即递归SPIKE算法相比LU分解和QR分解在求解的准确性和速度上的优越性能。本文研究可以为国产化服务器的高性能计算提供更优的求解大型带状线性系统的工具。

1 递归SPIKE算法

递归SPIKE算法最早可以追溯到20世纪70年代，其首先应用于求解三对角矩阵线性方程组，后来扩展到处理带状矩阵线性方程组。一般带状矩阵可以经过一定的分块看成块状三对角矩阵，因此，递归SPIKE算法可以被看作一种求解块状三对角矩阵的区域分解算法。递归SPIKE算法的核心思想与传统的LU分解算法的思想不同，引入了一种新的DS分解算法，通过这样的矩阵分解方式，能够有效降低计算的复杂度。

1.1 系数矩阵的分块

求解$\boldsymbol{Ax}=\boldsymbol{b}$，其中矩阵$\boldsymbol{A}$是一个$n=2^m(m\in\mathbf{N}^+)$阶的带状矩阵。一些带宽较窄的$n$阶带状矩阵都可以划分为块状三对角矩阵。假定划分对角线上对角块的数量为p，那么每个带状对角块$\boldsymbol{A}_j(j=1,\cdots,p)$的阶数为$n_j$(或者大致为$n/p$)。对于已经划定好的分区，$\boldsymbol{B}_j(j=1,\cdots,p-1)$和$\boldsymbol{C}_j(j=2,\cdots,p)$是与对角线上对角块相耦合的次对角线上的块状矩阵，具体为

$$\boldsymbol{A}=\begin{bmatrix}\boldsymbol{A}_1 & \boldsymbol{B}_1 & & \\ \boldsymbol{C}_2 & \boldsymbol{A}_2 & \ddots & \\ & \ddots & \ddots & \boldsymbol{B}_{p-1}\\ & & \boldsymbol{C}_p & \boldsymbol{A}_p\end{bmatrix},\tag{1}$$

式中：$\boldsymbol{B}_j$和$\boldsymbol{C}_j$的阶数均为$n_j\times k$，$k=\dfrac{\text{矩阵的带宽数}-1}{2}$，且矩阵$\boldsymbol{B}_j$和$\boldsymbol{C}_j$包含着大量的零元素，具体为

$$\boldsymbol{B}_j=\begin{bmatrix}0 & 0\\ \bar{\boldsymbol{B}}_j & 0\end{bmatrix},\boldsymbol{C}_j=\begin{bmatrix}0 & \bar{\boldsymbol{C}}_j\\ 0 & 0\end{bmatrix},\tag{2}$$

式中：$\bar{\boldsymbol{B}}_j$和$\bar{\boldsymbol{C}}_j$是k阶矩阵。

1.2 系数矩阵的DS分解

基于以上的分块形式，接下来我们将对系数矩阵 进行DS分解，且具体分解形式为

$$\boldsymbol{A}=\boldsymbol{DS}=\begin{bmatrix}\boldsymbol{D}_1 & & & \\ & \boldsymbol{D}_2 & & \\ & & \ddots & \\ & & & \boldsymbol{D}_p\end{bmatrix}\begin{bmatrix}\boldsymbol{I}_1 & \boldsymbol{V}_1 & & \\ \boldsymbol{W}_2 & \boldsymbol{I}_2 & \ddots & \\ & \ddots & \ddots & \boldsymbol{V}_{p-1}\\ & & \boldsymbol{W}_p & \boldsymbol{I}_p\end{bmatrix},\tag{3}$$

式中：$\boldsymbol{I}_j$表示一个矩阵阶数为n_j的单位矩阵且$\boldsymbol{D}_j \equiv \boldsymbol{A}_j$，而$\boldsymbol{V}_j$和$\boldsymbol{W}_j$的非零元素所在的列形成了大小为$n_j \times k$（又称尖峰）的高而窄的子矩阵。由上述DS分解的矩阵乘法可以得到$\boldsymbol{V}_j$和$\boldsymbol{W}_j$的表达式

$$\boldsymbol{V}_j = (\boldsymbol{A}_j)^{-1}\boldsymbol{B}_j,\ \boldsymbol{W}_j = (\boldsymbol{A}_j)^{-1}\boldsymbol{C}_j。$$

$\boldsymbol{V}_j$和$\boldsymbol{W}_j$可以通过求解矩阵方程(4)得到。

$$\boldsymbol{A}_j[\boldsymbol{V}_j \quad \boldsymbol{W}_j] = \begin{bmatrix} 0 & \dot{\boldsymbol{C}}_j \\ \vdots & 0 \\ 0 & \vdots \\ \dot{\boldsymbol{B}}_j & 0 \end{bmatrix}。 \tag{4}$$

1.3　简化系数矩阵方程组的提取

复杂矩阵线性方程组的求解经过前面的分解后缩减到两个步骤，即$\boldsymbol{DG} = \boldsymbol{b}$和$\boldsymbol{Sx} = \boldsymbol{G}$。

求解$\boldsymbol{DG} = \boldsymbol{b}$中的线性方程组所得向量$\boldsymbol{G}$将作为求解$\boldsymbol{Sx} = \boldsymbol{G}$的线性方程组的右端向量。在求解$\boldsymbol{DG} = \boldsymbol{b}$的线性方程组时，利用矩阵$\boldsymbol{D}$的特殊对角块形式，将线性方程组$\boldsymbol{DG} = \boldsymbol{b}$化为$p$个阶数较小的系数矩阵线性方程组来求解，这将在很大程度上降低计算的复杂度。如果将DS分解阶段和简化线性方程组求解阶段分离，则求解$\boldsymbol{DG} = \boldsymbol{b}$的过程可以与分解过程中的尖峰矩阵$\boldsymbol{V}_j$和$\boldsymbol{W}_j$的生成相结合。

将求解线性方程组$\boldsymbol{Sx} = \boldsymbol{G}$进一步化简为求解一个系数矩阵阶数更小的线性方程组

$$\hat{\boldsymbol{S}}\hat{\boldsymbol{x}} = \hat{\boldsymbol{G}}, \tag{5}$$

式中：$\hat{\boldsymbol{S}}$由矩阵$\boldsymbol{S}$每个分块的正上方和正下方的k行组成。实际上，尖峰矩阵$\boldsymbol{V}_j$和$\boldsymbol{W}_j$也可以划分为

$$\boldsymbol{V}_j = \begin{bmatrix} \boldsymbol{V}_j^{(t)} \\ \boldsymbol{V}_j' \\ \boldsymbol{V}_j^{(b)} \end{bmatrix},\ \boldsymbol{W}_j = \begin{bmatrix} \boldsymbol{W}_j^{(t)} \\ \boldsymbol{W}_j' \\ \boldsymbol{W}_j^{(b)} \end{bmatrix}, \tag{6}$$

式中：$\boldsymbol{V}_j^{(t)}$、$\boldsymbol{V}_j'$、$\boldsymbol{V}_j^{(b)}$和$\boldsymbol{W}_j^{(t)}$、$\boldsymbol{W}_j'$、$\boldsymbol{W}_j^{(b)}$是$\boldsymbol{V}_j$和$\boldsymbol{W}_j$的顶部k行、中间$n_j - 2k$和底部k行。这里，

$$\boldsymbol{V}_j^{(b)} = [0 \quad \boldsymbol{I}_k]\boldsymbol{V}_j,\ \boldsymbol{W}_j^{(t)} = [\boldsymbol{I}_k \quad 0]\boldsymbol{W}_j,$$

和　　$$\boldsymbol{V}_j^{(t)} = [\boldsymbol{I}_k \quad 0]\boldsymbol{V}_j,\ \boldsymbol{W}_j^{(b)} = [0 \quad \boldsymbol{I}_k]\boldsymbol{W}_j,$$

类似地，如果$\boldsymbol{x}_j$和$\boldsymbol{G}_j$是列向量$\boldsymbol{x}$和$\boldsymbol{G}$的第j个分区，有

$$\boldsymbol{x}_j = \begin{bmatrix} \boldsymbol{x}_j^{(t)} \\ \boldsymbol{x}_j' \\ \boldsymbol{x}_j^{(b)} \end{bmatrix},\ \boldsymbol{G}_j = \begin{bmatrix} \boldsymbol{G}_j^{(t)} \\ \boldsymbol{G}_j' \\ \boldsymbol{G}_j^{(b)} \end{bmatrix}。 \tag{7}$$

从$\boldsymbol{Sx} = \boldsymbol{G}$中提取出简化的线性方程组(5)，该线性方程组仅包含$\boldsymbol{V}_j$、$\boldsymbol{W}_j$、$\boldsymbol{x}_j$和$\boldsymbol{G}_j$顶部和底部k行。这个简化的线性方程组的系数矩阵$\hat{\boldsymbol{S}}$包含$p-1$个对角线对角块，其第i个对角块表示为

$$\begin{bmatrix} \boldsymbol{I}_k & \boldsymbol{V}_i^{(b)} \\ \boldsymbol{W}_{i+1}^{(t)} & \boldsymbol{I}_k \end{bmatrix}, \tag{8}$$

相应的次对角块表示为

$$\begin{bmatrix} \boldsymbol{W}_i^{(b)} & 0 \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 \\ 0 & \boldsymbol{V}_{i+1}^{(t)} \end{bmatrix}, \tag{9}$$

该缩减线性方程组中的解向量块和右端向量块表示为

$$\begin{bmatrix} \boldsymbol{x}_i^{(b)} \\ \boldsymbol{x}_{i+1}^{(t)} \end{bmatrix}, \begin{bmatrix} \boldsymbol{G}_i^{(b)} \\ \boldsymbol{G}_{i+1}^{(t)} \end{bmatrix}。 \tag{10}$$

如果已经求解出缩减线性方程组(5)，得到原线性方程组的部分解，就可以快速检索获得原线性方程组的全局解，具体求解方程为

$$\begin{cases} \boldsymbol{x}_1' = \boldsymbol{G}_1' - \boldsymbol{V}_1'\boldsymbol{x}_2^{(t)}, \\ \boldsymbol{x}_j' = \boldsymbol{G}_j' - \boldsymbol{V}_j'\boldsymbol{x}_{j+1}^{(t)} - \boldsymbol{W}_j'\boldsymbol{x}_{j-1}^{(b)}, j = 2, \cdots, p-1, \\ \boldsymbol{x}_p' = \boldsymbol{G}_p' - \boldsymbol{W}_p'\boldsymbol{x}_{p-1}^{(b)}。 \end{cases} \tag{11}$$

1.4　简化系数矩阵方程组的求解

并行求解简化系数矩阵方程组(5)的一种自然方法是Krylov子空间迭代法[16]。然而对于系数矩阵为非对角占优矩阵的线性方程组，该处理方法对大量的分块矩阵无效。因此，为了实现较小的相对残差，需要进行大量的外部迭代，这反过来又会导致计算复杂度的增加。此外，如果简化的线性方程组的系数矩阵阶数较大，那么这种方案可能因内存限制而无法实现。为此，一种新的求解简化线性方程组的直接递归方法被提出，这种递归方案涉及SPIKE算法的连续迭代，能够有效降低算法的计算复杂度。

如果简化线性方程组(5)的系数矩阵只有两个分块，即$p = 2$，那么，只由一个对角块组成的系数矩阵简化线性方程组就可以直接求解，其线性方程组表示为

$$\begin{bmatrix} \boldsymbol{I}_k & \boldsymbol{V}_1^{(b)} \\ \boldsymbol{W}_2^{(t)} & \boldsymbol{I}_k \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_1^{(b)} \\ \boldsymbol{x}_2^{(t)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{G}_1^{(b)} \\ \boldsymbol{G}_2^{(t)} \end{bmatrix}。 \tag{12}$$

具体求解步骤：首先计算$\boldsymbol{E} = \boldsymbol{I}_k - \boldsymbol{W}_2^{(t)}\boldsymbol{V}_1^{(b)}$，然后通过$\boldsymbol{E}\boldsymbol{x}_2^{(t)} = \boldsymbol{G}_2^{(t)} - \boldsymbol{W}_2^{(t)}\boldsymbol{G}_1^{(b)}$来获取$\boldsymbol{x}_2^{(t)}$，最后计算$\boldsymbol{x}_1^{(b)} = \boldsymbol{G}_1^{(b)} - \boldsymbol{V}_1^{(b)}\boldsymbol{x}_2^{(t)}$。

$\boldsymbol{x}_1$和$\boldsymbol{x}_2$其余部分的解可以通过式(11)来获得。

下面讨论系数矩阵的多分块情况，假设分块数量$p = 2^d$，经过分解后形成SPIKE矩阵$\boldsymbol{S}$，新

缩减线性方程组的分区数为2的倍数，并且可以利用另一个级别的SPIKE算法。这个过程被递归地重复执行，直到最新的矩阵$\boldsymbol{S}$只有两个分区，由此得到的简化线性方程组的形式如式(12)。

在实际应用中，递归方案与整体矩阵$\boldsymbol{S}$无关，而是与简化线性方程组(5)中的系数矩阵$\hat{\boldsymbol{S}}$有关。这样能够简化执行SPIKE算法，减少内存占有量，同时保存所有不同的级别的尖峰矩阵($\boldsymbol{V}_j$和$\boldsymbol{W}_j$)。值得注意的是，在简化线性方程组(5)中，系数矩阵$\hat{\boldsymbol{S}}$是块状三对角矩阵，对角块由式(8)给出，次对角块由式(9)给出。

如果将矩阵$\boldsymbol{S}$中的第一个分区和最后一个分区的顶部k行和底部k行提取到简化线性方程组系数矩阵$\hat{\boldsymbol{S}}$中，那么，矩阵$\hat{\boldsymbol{S}}$的阶数为$2kp$，而不是$2k(p-1)$，并且阶数为$2kp$的矩阵$\hat{\boldsymbol{S}}$的结构仍然保持块状三对角的形式。在这种情况下，每个对角线上的对角块都是阶数为$2k$的单位矩阵，与第i个对角块相关联的次对角块为

$$\begin{bmatrix}0 & \boldsymbol{W}_i^{(t)}\\ 0 & \boldsymbol{W}_i^{(b)}\end{bmatrix}(i=2,\cdots,p),$$
$$\begin{bmatrix}\boldsymbol{V}_i^{(t)} & 0\\ \boldsymbol{V}_i^{(b)} & 0\end{bmatrix}(i=1,\cdots,p-1)。\tag{13}$$

将$\boldsymbol{V}_i^{[1]}$和$\boldsymbol{W}_i^{[1]}$作为执行第一次递归的简化线性方程组的尖峰矩阵，其中，

$$\boldsymbol{V}_i^{[1]}=\begin{bmatrix}\boldsymbol{V}_i^{(t)}\\ \boldsymbol{V}_i^{(b)}\end{bmatrix},\ \boldsymbol{W}_i^{[1]}=\begin{bmatrix}\boldsymbol{W}_i^{(t)}\\ \boldsymbol{W}_i^{(b)}\end{bmatrix}。\tag{14}$$

当$p=4$时，新的简化线性方程组的系数矩阵$\tilde{\boldsymbol{S}}_1$表示为

$$\tilde{\boldsymbol{S}}_1=\begin{bmatrix}\boldsymbol{I} & \boldsymbol{V}_1^{[1]} & & \\ \boldsymbol{W}_2^{[1]} & \boldsymbol{I} & \boldsymbol{V}_2^{[1]} & \\ & \boldsymbol{W}_3^{[1]} & \boldsymbol{I} & \boldsymbol{V}_3^{[1]}\\ & & \boldsymbol{W}_4^{[1]} & \boldsymbol{I}\end{bmatrix}。\tag{15}$$

在进行SPIKE算法的第二次递归时，先将$\tilde{\boldsymbol{S}}_1$进行分块，每个分块的大小为$4k$，一共有$p/2$个对角块。然后对矩阵$\tilde{\boldsymbol{S}}_1$进行DS分解

$$\tilde{\boldsymbol{S}}_1=\boldsymbol{D}_1\tilde{\boldsymbol{S}}_2,\tag{16}$$

式中：$\boldsymbol{D}_1$由矩阵$\tilde{\boldsymbol{S}}_1$的对角线上对角块组成，每个块的阶数为$4k$。因此，$\tilde{\boldsymbol{S}}_2$是由尖峰矩阵$\boldsymbol{V}_i^{[2]}$和$\boldsymbol{W}_i^{[2]}$组成。当$p=4$时，这些矩阵表示为

$$\boldsymbol{D}_1=\begin{bmatrix}\begin{bmatrix}\boldsymbol{I}_{2k} & \boldsymbol{V}_1^{[1]}\\ \boldsymbol{W}_2^{[1]} & \boldsymbol{I}_{2k}\end{bmatrix} & & \\ & \begin{bmatrix}\boldsymbol{I}_{2k} & \boldsymbol{V}_3^{[1]}\\ \boldsymbol{W}_4^{[1]} & \boldsymbol{I}_{2k}\end{bmatrix}\end{bmatrix}$$

和

$$\tilde{\boldsymbol{S}}_2=\begin{bmatrix}\boldsymbol{I}_{4k} & \boldsymbol{V}_1^{[2]}\\ \boldsymbol{W}_2^{[2]} & \boldsymbol{I}_{4k}\end{bmatrix}。$$

总的来说，在第j次递归的过程中，尖峰矩阵$\boldsymbol{V}_i^{[j]}$和$\boldsymbol{W}_i^{[j]}$($i=1\sim p/2^j$)的阶数为$2^jk\times k$。因此，如果原始矩阵的分区数为$p=2^d$，则需要进行递归操作的次数为$d-1$，且矩阵$\tilde{\boldsymbol{S}}_1$的分解形式为

$$\tilde{\boldsymbol{S}}_1=\boldsymbol{D}_1\boldsymbol{D}_2\cdots\boldsymbol{D}_{d-1}\tilde{\boldsymbol{S}}_d,\tag{17}$$

式中：矩阵$\tilde{\boldsymbol{S}}_d$仅含有两个尖峰矩阵$\boldsymbol{V}_1^{[d]}$和$\boldsymbol{W}_2^{[d]}$。

简化的线性矩阵方程组可以写成

$$\tilde{\boldsymbol{S}}_d\tilde{\boldsymbol{x}}=\boldsymbol{B},\tag{18}$$

式中：$\boldsymbol{B}$为修正的右端向量，具体表示为

$$\boldsymbol{B}=\boldsymbol{D}_{d-1}^{-1}\cdots\boldsymbol{D}_2^{-1}\boldsymbol{D}_1^{-1}\tilde{\boldsymbol{G}}。$$

如果假设矩阵$\tilde{\boldsymbol{S}}_j$的尖峰块状矩阵$\boldsymbol{V}_i^{[j]}$和$\boldsymbol{W}_i^{[j]}$($\forall i$)，在给定j的情况下，可以计算第$j+1$次递归的尖峰矩阵$\boldsymbol{V}_i^{[j+1]}$和$\boldsymbol{W}_i^{[j+1]}$。第j次递归的尖峰矩阵的顶部和底部k行分别表示为

$$\boldsymbol{V}_i^{[j](b)}=\begin{bmatrix}0 & \boldsymbol{I}_k\end{bmatrix}\boldsymbol{V}_i^{[j]},\tag{19}$$
$$\boldsymbol{W}_i^{[j](t)}=\begin{bmatrix}\boldsymbol{I}_k & 0\end{bmatrix}\boldsymbol{W}_i^{[j]}。\tag{20}$$

第$j+1$次递归的尖峰矩阵的中间$2k$行可表示为

$$\begin{bmatrix}\dot{\boldsymbol{V}}_i^{[j+1]}\\ \ddot{\boldsymbol{V}}_i^{[j+1]}\end{bmatrix}=\begin{bmatrix}0 & \boldsymbol{I}_{2k} & 0\end{bmatrix}\boldsymbol{V}_i^{[j+1]},\tag{21}$$
$$\begin{bmatrix}\dot{\boldsymbol{W}}_i^{[j+1]}\\ \ddot{\boldsymbol{W}}_i^{[j+1]}\end{bmatrix}=\begin{bmatrix}0 & \boldsymbol{I}_{2k} & 0\end{bmatrix}\boldsymbol{W}_i^{[j+1]},\tag{22}$$

则可以形成简化的矩阵方程

$$\begin{bmatrix}\boldsymbol{I}_k & \boldsymbol{V}_{2i-1}^{[j](b)}\\ \boldsymbol{W}_{2i}^{[j](t)} & \boldsymbol{I}_k\end{bmatrix}\begin{bmatrix}\dot{\boldsymbol{V}}_k^{[j+1]}\\ \ddot{\boldsymbol{V}}_k^{[j+1]}\end{bmatrix}=\begin{bmatrix}0\\ \boldsymbol{V}_{2i}^{[j](t)}\end{bmatrix},$$
$$i=1,2,\cdots,\frac{p}{2^{i-1}}-1$$

和

$$\begin{bmatrix}\boldsymbol{I}_k & \boldsymbol{V}_{2i-1}^{[j](b)}\\ \boldsymbol{W}_{2i}^{[j](t)} & \boldsymbol{I}_k\end{bmatrix}\begin{bmatrix}\dot{\boldsymbol{W}}_i^{[j+1]}\\ \ddot{\boldsymbol{W}}_i^{[j+1]}\end{bmatrix}=\begin{bmatrix}\boldsymbol{W}_{2i-1}^{[j](b)}\\ 0\end{bmatrix},$$
$$i=2,3,\cdots,\frac{p}{2^{i-1}}。$$

与线性方程组式(12)类似，可以利用求解式(12)的方法来获得$j+1$次递归的尖峰矩阵的中间$2k$行。然后检索获得第$i+1$次递归的全部尖峰矩阵

$$\begin{bmatrix}\boldsymbol{I}_{2^jk} & 0\end{bmatrix}\boldsymbol{V}_i^{[j+1]}=-\boldsymbol{V}_{2i-1}^{[j]}\ddot{\boldsymbol{V}}_i^{[j+1]},$$
$$\begin{bmatrix}0 & \boldsymbol{I}_{2^jk}\end{bmatrix}\boldsymbol{V}_i^{[j+1]}=\boldsymbol{V}_{2i}^{[j]}-\boldsymbol{W}_{2i}^{[j]}\dot{\boldsymbol{V}}_i^{[j+1]},$$

以及

$$\begin{bmatrix}\boldsymbol{I}_{2^jk} & 0\end{bmatrix}\boldsymbol{W}_i^{[j+1]}=\boldsymbol{W}_{2i-1}^{[j]}-\boldsymbol{V}_{2i-1}^{[j]}\ddot{\boldsymbol{W}}_i^{[j+1]},$$
$$\begin{bmatrix}0 & \boldsymbol{I}_{2^jk}\end{bmatrix}\boldsymbol{W}_i^{[j+1]}=-\boldsymbol{W}_{2i}^{[j]}\dot{\boldsymbol{W}}_i^{[i+1]}。$$

接着，通过求解式（12）的方法来求解式（18）的解向量$\tilde{\boldsymbol{x}}$。最后，通过式（11）检索获得整个解向量$\boldsymbol{x}$。

2　数值实验

对两组带宽相同阶数不同的带状矩阵线性方程组进行数值实验，以证明递归SPIKE算法不仅能够得到预期的结果，而且相比传统的LU分解算法和QR分解算法，能够在求解速度上更具优越性。

数值实验环境配置：设备为LAPTOP-R6HU69BR；处理器为AMD Ryzen 7 6800H with Radeon Graphics，3.20 GHz；机带RAM为16.0 GB；系统为基于x64处理器的64位操作系统；平台为matlab R2022a。

LU分解算法和QR分解算法使用matlab内置函数来求解，以便将递归SPIKE算法与最优化的LU分解算法和QR分解算法进行比较。

例 1[17] 带宽为3的带状矩阵线性方程组，对应系数矩阵和右端向量分别为

$$\boldsymbol{A}=\begin{bmatrix}4 & 2 & & & & \\ 1 & 4 & 1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & 1 & 4 & 1 & \\ & & & 2 & 4 & \end{bmatrix}_{n\times n},\quad \boldsymbol{b}=\begin{bmatrix}6\\6\\\vdots\\6\\6\end{bmatrix}_{n\times 1}。$$

利用递归SPIKE算法、LU分解算法和QR分解算法分别求解系数矩阵阶数$n=4\,096, 8\,192, 16\,384$时的带状矩阵线性方程组，且3种算法都能求得准确的结果$\boldsymbol{x}=(1,1,\cdots,1)^{\mathrm{T}}_{1\times n}$。3种算法求解运行的时间如表1所示。

表1　3种算法求解带宽为3且矩阵阶数不同的带状矩阵线性方程组的求解时间比较

Tab. 1　Comparison of solution times of three algorithms for banded matrix linear equations with bandwidth of 3 and different matrix orders

矩阵阶数	求解时间t/s		
	LU分解算法	QR分解算法	SPIKE算法
4 096	0.474 6	2.099 9	0.453 5
8 192	3.834 8	12.427 9	1.365 7
16 384	18.227 3	91.822 8	7.929 2

例 2 带宽为5的带状矩阵线性方程组，对应系数矩阵和右端向量分别为

$$\boldsymbol{A}=\begin{bmatrix}35 & 5 & 1 & & & \\ 5 & \ddots & \ddots & \ddots & & \\ 1 & \ddots & \ddots & \ddots & 1 & \\ & \ddots & \ddots & \ddots & 5 & \\ & & 1 & 5 & 35 & \end{bmatrix}_{n\times n},\quad \boldsymbol{b}=\begin{bmatrix}41\\46\\47\\\vdots\\47\\46\\41\end{bmatrix}_{n\times 1}。$$

利用递归SPIKE算法、LU分解算法和QR分解算法分别求解系数矩阵阶数$n=4\,096, 8\,192, 16\,384$时的带状矩阵线性方程组，且3种算法都能求得准确的结果$\boldsymbol{x}=(1,1,\cdots,1)^{\mathrm{T}}_{1\times n}$。3种算法求解运行的时间如表2所示。

表2　3种算法求解带宽为5且矩阵阶数不同的带状矩阵线性方程组的求解时间比较

Tab. 2　Comparison of solution times of three algorithms for banded matrix linear equations with bandwidth of 5 and different matrix orders

矩阵阶数	求解时间t/s		
	LU分解算法	QR分解算法	SPIKE算法
4 096	0.331 9	1.320 7	0.384 1
8 192	2.613 1	11.283 1	1.413 0
16 384	17.968 5	89.124 5	7.138 0

从上面两组实验的结果可以看出，对带宽分别为3和5，矩阵阶数为4 096的带状矩阵线性方程组求解时，递归SPIKE算法的求解速度相比LU分解算法并没有很大优势，但是明显优于QR分解算法。当带状矩阵阶数变大时，即矩阵阶数为8 192和16 384时，SPIKE算法的求解速度远快于LU分解算法和QR分解算法。因此，对于求解实际工程问题中经常碰到的大型带状矩阵线性方程组，递归SPIKE算法优于传统的LU分解算法和QR分解算法。

3　结束语

与传统的矩阵分解方式不同，本文提出的递归SPIKE算法是一种基于新的DS矩阵分解方式的算法，该算法通过提取规模更小的简化线性方程组来进行求解，降低了算法的计算复杂度。数值实验结果表明，对于大型带状矩阵线性方程组的求解，递归SPIKE算法不仅能够取得良好的结果，而且求解速度更快。在后续工作中，将在SPIKE的处理细节方面进行进一步优化，以期得到更好的效果。

参考文献:

[1] FREYTAG M, SHAPIRO V, TSUKANOV I. Finite element analysis in situ[J]. Finite Elements in Analysis and Design, 2011, 47(9): 957-972.

[2] OISHI A, YAGAWA G. Computational mechanics enhanced by deep learning[J]. Computer Methods in Applied Mechanics and Engineering, 2017, 327: 327-351.

[3] HERFF S, NIEMÖLLER A, MEINKE M, et al. LES of a turbulent swirl flame using a mesh adaptive level-set method with dynamic load balancing[J]. Computers & Fluids, 2021, 221: 104900.

[4] SHAKEEL M R, MOKHEIMER E M A. Swirl flow in annular geometry with varying cross-section[J]. Engineering Applications of Computational Fluid Mechanics, 2022, 16(1): 1154-1172.

[5] ZEIDAN D, BÄHR P, FARBER P, et al. Numerical investigation of a mixture two-phase flow model in two-dimensional space[J]. Computers & Fluids, 2019, 181: 90-106.

[6] 董建伟, 娄光谱. 量子流体动力学等温模型的拟中性极限[J]. 中北大学学报(自然科学版), 2012, 33(2): 102-106.
DONG Jianwei, LOU Guangpu. Quasineutral limit of isothermalk quantum hydrodynamic model[J]. Journal of North University of China (Natural Science Edition), 2012, 33(2): 102-106. (in Chinese)

[7] POLIZZI E, ABDALLAH N B. Subband decomposition approach for the simulation of quantum electron transport in nanostructures[J]. Journal of Computational Physics, 2004, 202: 150-180.

[8] 杨彦利, 苗长云, 亢伉, 等. 输送带跑偏故障的机器视觉检测技术[J]. 中北大学学报(自然科学版), 2012, 33(6): 667-671.
YANG Yanli, MIAO Changyun, KANG Kang, et al. Machine vision inspection technique for conveyor belt deviation[J]. Journal of North University of China (Natural Science Edition), 2012, 33(6): 667-671. (in Chinese)

[9] BARTELS R H, GOLUB G H. The simplex method of linear programming using LU decomposition[J]. Communications of the ACM, 1969, 12(5): 266-268.

[10] SHARMA A, PALIWAL K K, IMOTO S, et al. Principal component analysis using QR decomposition[J]. International Journal of Machine Learning and Cybernetics, 2013, 4(6): 679-683.

[11] SPRING B S, POLIZZI E, SAMEH A H. A feature-complete spike dense banded solver[J]. ACM Transactions on Mathematical Software, 2020, 46(4): 1-35.

[12] SPRING B S. Enhanced capabilities of the spike algorithm and a new spike-openMP solver[D]. Amherst: University of Massachusetts Amherst, 2014.

[13] SAMEH A H, POLIZZI E. A parallel hybrid banded system solver: the SPIKE algorithm[J]. Parallel Computing, 2006, 32(2): 177-194.

[14] SPRING B S, POLIZZI E, SAMEH A H. A feature complete spike banded algorithm and solver[DB/OL]. (2018-11-08) [2023-07-06]. https://arxiv.org/abs/1811.03559.

[15] POLIZZI E, SAMEH A H. SPIKE: A parallel environment for solving banded linear systems[J]. Computers & Fluids, 2007, 36(1): 113-120.

[16] STYKEL T, SIMONCINI V. Krylov subspace methods for projected Lyapunov equations[J]. Applied Numerical Mathematics, 2012, 62(1): 35-50.

[17] 冉瑞生. 一些矩阵计算问题及其在图像识别中的应用研究[D]. 成都: 电子科技大学, 2006.