



中国研究生创新实践系列大赛
“华为杯”第十八届中国研究生
数学建模竞赛

学 校 电子科技大学

参赛队号 21106140105

1.崔峻豪

队员姓名 2.王宇丰

3.王小颖

中国研究生创新实践系列大赛

“华为杯”第十八届中国研究生

数学建模竞赛

题 目 相关矩阵组低复杂度计算和存储建模

摘 要：

在数字信号处理领域，绝大多数信号均可表示为矩阵的形式。然而，随着矩阵规模的快速增长，矩阵处理的计算复杂度和存储复杂度急剧增加，这对计算时间及存储空间带来了巨大的挑战。幸运的是，矩阵之间通常存在一定的相关性，因此，充分挖掘矩阵间的相关性，能够大大降低矩阵处理算法的计算复杂度和存储复杂度。基于以上背景，本文针对给定的相关矩阵组，充分挖掘矩阵间的相关性，对具体的矩阵处理算法进行优化并进行存储建模。

对于问题 1 第一部分，首先基于 Pearson 线性相关系数，分析了矩阵内及矩阵间的线性相关性，并得出矩阵间存在明显相关性，而矩阵内部几乎不存在相关性的结论。因此可以利用左右相邻矩阵描述中间矩阵，也可以用中间矩阵描述其左右相邻矩阵。基于此结论，相关矩阵组的低复杂计算可以从两个方面进行优化：（1）利用计算复杂度更低的随机奇异值分解算法，用降维后小矩阵的奇异值分解代替大矩阵的奇异值分解。另一方面，仅生成一次全局共用的随机奇异值分解中需要的高斯分布随机矩阵，并用可变模型精度的迭代 QR 分解算法替代常规 QR 分解，仅得到需要的前 P 列标准正交矩阵 Q 。（2）充分利用矩阵间的线性相关性，对于矩阵组的每一行块，将每三个相邻矩阵视为一组，用中间矩阵计算得到的 Q 矩阵作为整组共用的 Q 矩阵。通过上述两方面的优化，能够大幅度降低相关矩阵组奇异值分解的计算复杂度。

对于问题 1 第二部分，其重点在于矩阵的求逆运算。通过分析可知，求逆对象为 Hermite 矩阵，因此可采用基于 Hermite 矩阵求逆引理的矩阵递推求逆方法进行矩阵求逆。递推求逆方法能够在得到与常规方法相同精确解的前提下，进一步降低计算复杂度。将该递推求逆方法与改进的奇异值分解算法结合，可实现从 H 矩阵组到 W 矩阵组的低复杂度求解过程。

对于问题 2，考虑到同一行块矩阵间线性相关性强的特点，利用线性插值的思想，设计出三种压缩与解压缩模型：

模型①：通过对矩阵组同一行块中矩阵做隔位舍弃的操作，能仅依靠矩阵组的拆解、组合操作，实现对矩阵组的有效压缩；在解压过程中，通过利用舍弃矩阵相邻两个矩阵的算术平均，实现矩阵的插值恢复。该模型计算复杂度低，但解压缩误差较大，仅适合相关性较强的矩阵组。

模型②：为了改善模型压缩误差较大的问题，使得模型能够满足给定的压缩误差，模型②通过对矩阵组同一行块中序号为 k 倍数的矩阵进行舍弃，实现压缩误差与压缩率的互换；在解压过程中，依旧利用算术平均的方式，实现舍弃矩阵的插值恢复。相比于模型①，尽管压缩效率有所损失，但压缩误差会有效降低，并且解压缩的计算复杂度也会有一定的降低。

模型③：为了充分发挥相关性对矩阵压缩的作用，结合 Pearson 线性相关系数对插值恢复过程进行进一步改进。压缩阶段中，在舍弃序列号为 k 倍数的矩阵同时，计算舍弃矩阵与相邻两个矩阵的 Pearson 线性相关系数。以此作为基础，在解压阶段，根据舍弃矩阵与相邻左右两个矩阵的相关系数，得到加权平均的权值，并对舍弃矩阵进行加权平均的插值恢复。相比模型②，该模型能够在不改变压缩率的情况下，充分利用矩阵间相关性，进一步降低压缩误差，解决了由于舍弃矩阵与两边相邻矩阵相关性差异较大导致的恢复效果较差的问题。但该模型会增加一定的计算复杂度与储存空间。

对于问题 3，考虑从矩阵输入信号 H 到近似矩阵输出信号 W 的端到端流程。对本文建立的模型分析可知，本文采用的是舍弃列块的压缩方法。因此可以充分利用需要舍弃的列块在原矩阵组中的位置信息对本文的模型进行优化。通过利用舍弃矩阵列块的位置信息，使得 V 矩阵的计算复杂度进一步降低，并能直接生成压缩矩阵 W ，无需生成压缩前的 W 矩阵，因此改进后的模型拥有更低的计算复杂度和存储空间。

关键词：Pearson 线性相关系数；矩阵相关性；随机奇异值分解；线性插值；Hermite 矩阵递推

目录

1 问题重述	4
1.1 问题背景	4
1.2 问题重述	4
2 符号说明	5
3 问题分析	6
3.1 问题 1 分析	6
3.1.1 问题 1.1 分析	6
3.1.2 问题 1.2 分析	7
3.2 问题 2 分析	7
4 模型建立与求解分析	8
4.1 问题 1	8
4.1.1 问题 1.1——低计算复杂度奇异值分解	8
4.1.2 问题 1.2——低计算复杂度矩阵求逆	15
4.2 问题 2	18
4.2.1 线性插值	18
4.2.2 基于矩阵相关性的压缩与解压缩模型	19
4.2.3 基于矩阵间相关性的 H 矩阵组压缩	21
4.2.4 基于矩阵间相关性的 W 矩阵组压缩	22
4.2.5 存储复杂度及计算复杂度分析	24
4.3 问题 3	25
5 模型评价与改进	26
5.1 模型优点	26
5.2 模型缺点及改进方向	26
参考文献	27
附录 A 计算复杂度分析	28

1 问题重述

1.1 问题背景

在图像视频处理、遥感探测、射电天文观测、通信网络等各个领域，其采用的信号通常可表示为矩阵的形式。而这一系列的矩阵之间通常存在一定的相关性，因此在数学上，通常可采用相关矩阵组进行信号表示。近些年来随着信息数据总量的爆炸性增长，相关矩阵组的规模急剧增大，这对相关矩阵组的处理及存储算法带来了巨大的挑战。因此，充分挖掘相关矩阵组中矩阵间和矩阵内部的相关性，对实现矩阵的快速处理及高效率存储具有重大的价值与意义。

1.2 问题重述

基于上述背景，本文需要对六组给定的复数相关矩阵组的处理算法进行优化并设计合理的数据压缩解压缩算法，具体的复数相关矩阵组处理算法如图 1.1 所示：

- 对给定的一组复数矩阵 $H = \{H_{j,k}\}$ 中的每一个矩阵做奇异值分解（singular value decomposition, SVD），取其右奇异向量的前两列得到右奇异向量矩阵组 $V = \{V_{j,k}\}$ 。将不同 j 下标、相同 k 下标的 $V_{j,k}$ 进行横向的拼接，得到 $V_k = [V_{1,k} \cdots V_{j,k} \cdots V_{J,k}]$ 。
- 利用公式 $W_k = V_k (V_k^H V_k + \sigma^2 I)^{-1}$ 计算，再拆解 $W_k = [W_{1,k} \cdots W_{j,k} \cdots W_{J,k}]$ ，并按对应行列进行组合得到输出结果矩阵组 $W = \{W_{j,k}\}$ 。

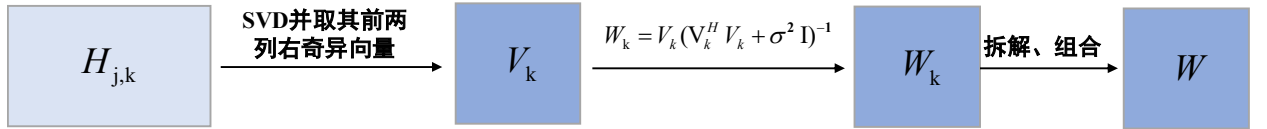


图 1.1 复数矩阵组的处理算法流程图

当矩阵规模增加时，该处理算法的时间复杂度及空间复杂度将急剧增加。因此，本文的任务是充分挖掘矩阵间的相关性，在保证满足给定最低建模精度的前提下，尽可能优化算法的计算复杂度和存储复杂度。并设计相应的压缩与解压缩算法，在保证压缩与解压缩方法低复杂度的前提下使得矩阵组占用的存储空间尽可能小。

2 符号说明

符号	含义
$\{H_{j,k}\}$	输入复数相关矩阵组
$\{V_{j,k}\}$	右奇异向量组
$\{W_{j,k}\}$	输出结果矩阵组
N, M	矩阵组的行，列，
J, K	矩阵组内部矩阵的行，列
$H_{j,k}$	输入复数相关矩阵组中第 j 行、第 k 列 矩阵
$V_{j,k}$	由 $H_{j,k}$ 生成的右奇异向量组
$Q_{j,k}$	由 $H_{j,k}$ 生成的标准正交矩阵
$B_{j,k}$	由 $H_{j,k}$ 生成的降维后矩阵
N_{Lost}	舍弃的列块矩阵组个数
η	压缩率

3 问题分析

3.1 问题 1 分析

3.1.1 问题 1.1 分析

本题主要涉及相关矩阵组的 SVD 分解运算，假如不对该部分做任何优化，则对于给定的 $M \times N$ 相关矩阵组需要做 $M \times N$ 次 $J \times K$ 大小的 SVD 分解，每次取前 2 个右奇异向量组成右奇异向量组 V ，因此该算法的计算复杂度极高。

因此，本题可以从两个方面进行优化：1)、挖掘矩阵组中的相关性。2) 采用低复杂度 SVD 分解方法。

关于矩阵组的相关性，通过分析可以得到：对于矩阵组的每一行而言，某一个矩阵与其左右相邻矩阵有很强的线性相关性。因此可以利用左右相邻矩阵描述中间的矩阵，也可以用中间的矩阵描述其左右相邻矩阵。

对于低复杂度 SVD 分解方法，本文采用随机 SVD 分解算法，并针对以下三个方面进行进一步改进：

1) 采用全局高斯分布随机矩阵 Ω ，代替每次随机 SVD 中的随机矩阵。

2) 采用可变精度的 QR 迭代分解法。在随机 SVD 的 QR 分解步骤中，由于并不需要得到完整的标准正交矩阵 Q ，因此采用 QR 迭代法，仅计算需要的前 P 列标准正交矩阵 Q 。

3) 利用矩阵间的相关性，将矩阵组按行分割，同时将同一行块中三个相邻矩阵划分为一组。对中间矩阵进行 QR 分解求得 Q 矩阵后，将其作为整个组三个矩阵共用的 Q 矩阵。因此每三个矩阵仅需要做一次 QR 迭代分解。

上述算法可描述为：首先对矩阵组中同一行块的矩阵分组，对每组做改进的随机 SVD，最终求解出右奇异向量矩阵组 $\{V_k\}$ 。主要流程图如图 3.1 所示：

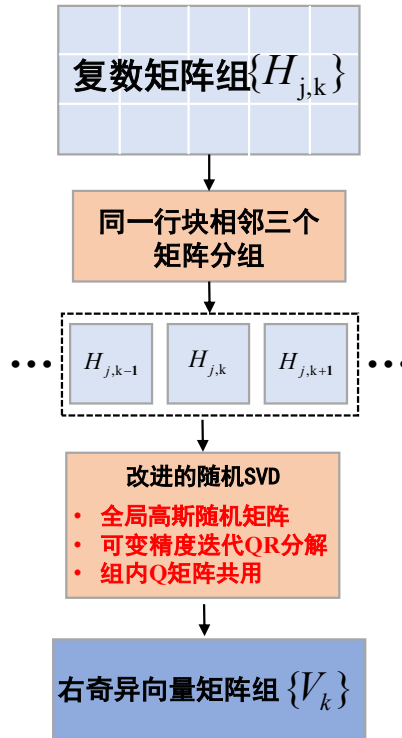


图 3.1 复数矩阵组 H 到 V_k 流程图

3.1.2 问题 1.2 分析

得到 $\{V_k\}$ 矩阵组后利用公式 $W_k = V_k(V_k^H V_k + \sigma^2 I)^{-1}$ 可得到 $\{W_k\}$ 矩阵组，再通过拆解和组合得到最终 $\{W_{j,k}\}$ 矩阵组，由于数据的拆分拼接不涉及计算复杂度，因此本题的重点为 $V_k^H V_k + \sigma^2 I$ 矩阵的求逆。

本题的核心在于应用低复杂度矩阵求逆算法。容易知道， $V_k^H V_k + \sigma^2 I$ 矩阵为正定 Hermite 矩阵，因此采用基于 Hermite 矩阵求逆引理的递推求逆算法。利用该求逆算法对 $V_k^H V_k + \sigma^2 I$ 矩阵进行快速递推求逆后，再通过拆分拼接即可得到 W 矩阵。

3.2 问题 2 分析

由之前的分析可知， H 、 W 矩阵组中，某一处矩阵与其左右矩阵的线性相关性很强，因此可以使用线性插值的思想进行数据压缩与解压缩。本文设计三种模型如图 3.2 所示：

当矩阵组中同一行块中矩阵间相关性较强时，在压缩时可以每隔一个矩阵就舍弃一个矩阵，舍弃掉的矩阵用其左右两边的矩阵进行算术平均预测。该方法可实现接近 50% 的压缩率及极低的计算复杂度。

为了能够改变压缩率及压缩误差，本文在上述方案的基础上进行改进，在压缩时对序列号为 k 的倍数的矩阵进行舍弃，在解压时则利用算术平均的方法预测出舍弃的矩阵，从而能够根据数据压缩效果改变压缩率及压缩误差。

进一步地，为了充分发挥相关性对矩阵压缩的作用，本文以相关系数作为权值，利用加权平均的思想设计压缩及解压缩模型。即：压缩阶段中，在舍弃序列号为 k 的倍数的矩阵时，计算舍弃矩阵与相邻两个矩阵的 Pearson 线性相关系数。以此作为基础，在解压阶段，根据舍弃矩阵与相邻左右两个矩阵的相关系数，求出加权平均的权值，并对舍弃矩阵进行预测。该方案虽然能够更大程度地减小压缩误差，但也会增加计算复杂度和存储复杂度。

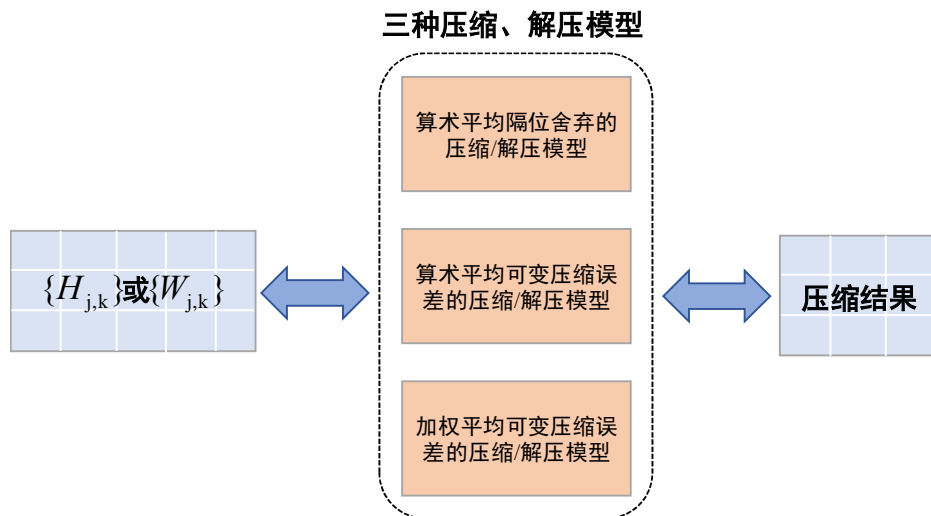


图 3.2 矩阵压缩与解压缩模型

4 模型建立与求解分析

4.1 问题 1

4.1.1 问题 1.1——低计算复杂度奇异值分解

1) 矩阵组相关性分析

由题目可知，矩阵内部及属于同一行块的矩阵间存在相关性。为了利用其相关性，以便进行低计算复杂度的奇异值分解，本文首先分析了矩阵内部和矩阵间的相关性。

为了计算矩阵之间的相关性，本文采用最常用的 Pearson 线性相关系数分析了矩阵组中某一行块 384 个矩阵之间的相关性。对于两个向量 X_a 和 Y_b ，Pearson 线性相关系数计算公式如下：

$$\rho(X_a, Y_b) = \frac{\sum_{i=1}^n (X_{a,i} - \bar{X}_a)(Y_{b,i} - \bar{Y}_b)}{\left\{ \sum_{i=1}^n (X_{a,i} - \bar{X}_a)^2 \sum_{j=1}^n (Y_{b,j} - \bar{Y}_b)^2 \right\}^{1/2}} \quad (4.1)$$

其中 $\bar{X}_a = \sum_{i=1}^n (X_{a,i}) / n$ ， $\bar{Y}_b = \sum_{j=1}^n (Y_{b,j}) / n$ ， n 为向量的长度。

Pearson 线性相关系数反映了两个向量之间的线性相关性，其值分布在 -1 至 1 之间，其值越接近于 1 或 -1，相关度越强，相关系数越接近于 0，相关度越弱。通常 Pearson 线性相关系数绝对值在 0.8 以上时，说明两个向量极强相关^[1]。

对于多个向量之间的 Pearson 线性相关系数，则采用相关系数矩阵进行表示。即计算某个向量与包含自身在内的所有向量之间的相关性，得到一个相关性数组；对所有向量如此操作，即可得到一个相关系数矩阵。

对于本题中矩阵间的 Pearson 线性相关系数，本文将矩阵向量化为一维向量后再进行相关系数计算。具体方法为：将每个 4x64 的矩阵向量化为 256x1 的列向量，得到 384 个列向量，计算列向量之间的相关系数，并取其模。本文以 Data1 中第一行块 384 个矩阵的 Pearson 线性相关系数为例，绘制其相关系数矩阵如图 4.1 所示：

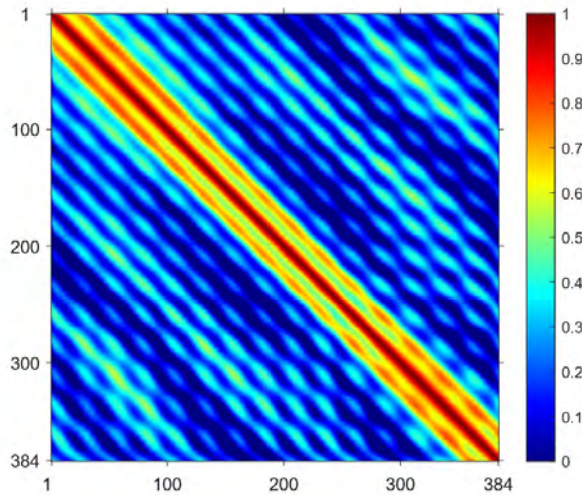


图 4.1 矩阵组中同一行块中各个矩阵间相关系数

可以观察到，相邻两个矩阵之间存在很强的线性相关性。通过对大量数据进行分析，本文发现相邻两个矩阵之间相关系数可达到 0.9 以上。且两个矩阵相距越远，其线性相关性越弱。例如图 4.2 展示了不同位置两个矩阵的相关性。

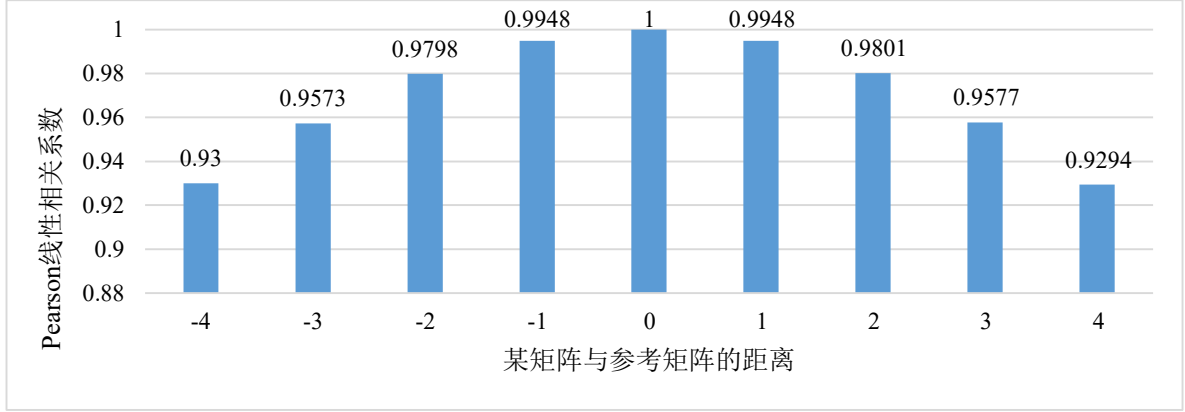


图 4.2 不同位置矩阵间的相关性

当多个矩阵之间呈现高度线性相关性时，某一个矩阵则可以表示为其他矩阵的线性组合，例如，当矩阵 $A_1, A_2, \dots, A_n$ 之间存在高度线性相关性时，则对于其中一个 A_i ，有：

$$A_i \approx k_1 A_1 + k_2 A_2 + \dots + k_{i-1} A_{i-1} + k_{i+1} A_{i+1} + \dots + k_n A_n \quad (4.2)$$

对于矩阵内部的相关性，本文同样采用 Pearson 线性相关系数分析其相关性。但通过对大量数据进行分析，发现不论矩阵行之间还是列之间，其相关性均较弱，如图 4.3 所示。因此，本文重点利用同一行块中矩阵间的相关性进行后续计算优化。

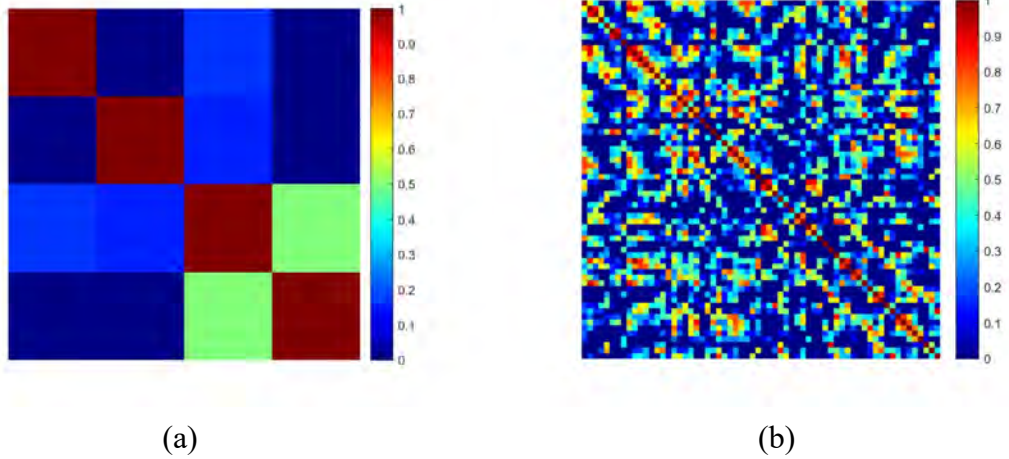


图 4.3 矩阵各列间的相关系数。(a)行之间相关系数。(b)列之间相关系数。

2) 矩阵的奇异值分解

奇异值分解属于矩阵分解的一种方式^[2]，可以用于数据降维及特征分解。其分解过程如图 4.4 所示：

对于任意一个矩阵， $X \in \mathbb{C}^{m \times n}$ ，则一定存在一种分解使得：

$$X = U \Sigma V^H \quad (4.3)$$

其中 U 是 $m \times m$ 的酉矩阵； Σ 是 $m \times n$ 阶非负实数对角矩阵； V^H 是 $n \times n$ 的酉矩阵， V^H

是 V 的共轭转置。这样的分解即为奇异值分解^[3]。

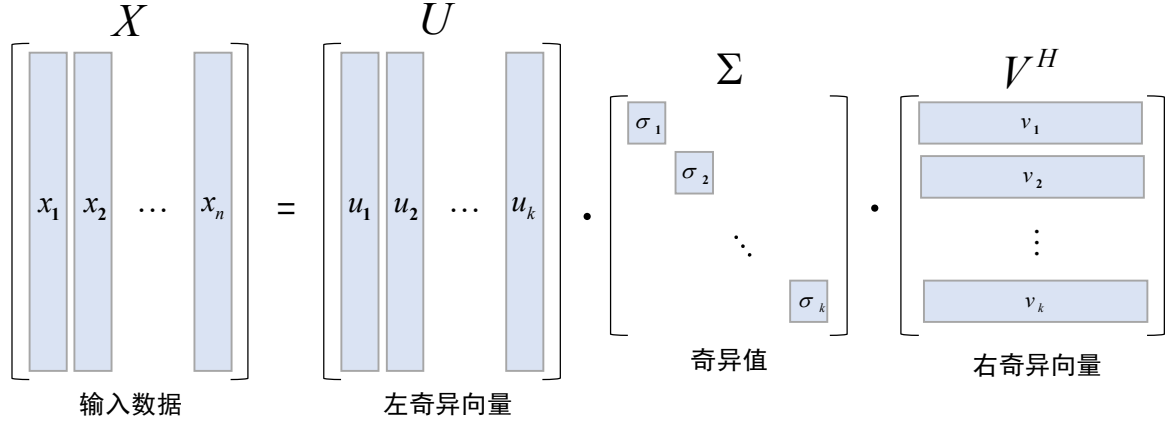


图 4.4 奇异值分解过程

3) 可变模型精度的随机奇异值分解

为了提高奇异值分解的计算速度，本文对随机奇异值分解算法做了改进，利用改进的随机奇异值分解算法能大大降低计算复杂度。常规的随机奇异值分解算法具体过程如图 4.5 所示：

①全局高斯随机矩阵

对于题目中某一个 $M \times N$ 的矩阵 $H_{j,k}$ ，本文首先生成一个全局高斯分布随机矩阵 $\Omega \in \mathbb{C}^{N \times k}$ 。其中 k 为 $H_{j,k}$ 的秩。由于 $k \leq \min\{M, N\}$ ，因此不妨设 $k=N$ 。需要注意的是，标准的随机奇异值分解法中， Ω 需要针对每个 $H_{j,k}$ 单独生成。考虑到 Ω 的随机性及每个 Ω 的尺寸均相同，本文仅选取固定的 Ω ，从而降低计算复杂度。

②可变模型精度的迭代 QR 分解

由于随即奇异值分解的输入矩阵需要行数大于列数，为了方便起见，本文下面描述的 $H_{j,k}$ 为原 H 矩阵的共轭转置。因此本文需要的右奇异向量对应于随机奇异值分解得到的左奇异向量。

根据 Ω 和 $H_{j,k}$ ，作 $Y_{j,k} = H_{j,k} \Omega$ ，得到新矩阵 $Y_{j,k}$ 。随后对 $Y_{j,k}$ 进行 QR 分解。即 $Y_{j,k} = Q_{j,k} R_{j,k}$ ，并仅取单位正交矩阵（Q 矩阵）的前 k 列，得到最终 $Q_{j,k}$ 。常规的 QR 分解计算复杂度相当于奇异值分解，且计算出的 $Q_{j,k}$ 为 64×64 。然而，本题通常只需 64×4 的矩阵。因此，为了降低计算复杂度，本文采用可变精度的迭代 QR 分解方法，快速计算出 $Q_{j,k}$ 。具体实现方法如下：

- 抽取 1 个维度为 n 的高斯随机向量 $\omega^{(i)}$ ，令 $y_{j,k}^{(i)} = Y_{j,k} \omega^{(i)}$
- 构造 $\tilde{q}_{j,k}^{(i)} = (I - Q_{j,k}^{(i-1)} (Q_{j,k}^{(i-1)})^H) y_{j,k}^{(i)}$ ；

- 令 $q_{j,k}^{(i)} = \tilde{q}_{j,k}^{(i)} / \left\| \tilde{q}_{j,k}^{(i)} \right\|_2$;
- 构造 $Q_{j,k}^{(i)} = [Q_{j,k}^{(i-1)} q_{j,k}^{(i)}]$ 。

其中 $Q^{(0)}$ 为空矩阵， $Q_{j,k}^{(i)}$ 为 $m \times i$ 的迭代标准正交矩阵（即后文提到的 Q 矩阵）。 i 值越大，保留的信息越多，即 $Y_{j,k}$ 与 $Q_{j,k} Q_{j,k}^H Y_{j,k}$ 越接近。

通常情况下，只需 64×4 的标准正交矩阵便可达到建模精度要求，此时只需迭代 4 次即可求出 $Q_{j,k}$ 。计算复杂度明显低于常规 QR 分解。当建模精度不满足要求时，可以选取适当的迭代次数，以保证精度。

③低规模输入奇异值分解

利用 $Q_{j,k}$ 矩阵和原矩阵 $H_{j,k}$ 做乘法 $B_{j,k} = Q_{j,k}^H H_{j,k}$ 可以得到降维矩阵 $B_{j,k}$ 。可以分析出，此处 $B_{j,k} \in \mathbb{C}^{k \times N}$ ，尺寸明显小于 $H_{j,k}$ ，再对 $B_{j,k}$ 采用常规奇异值分解算法，得到 $B_{j,k}$ 的左奇异向量 $\tilde{U}_{j,k}$ ，则 $H_{j,k}$ 的右奇异向量为：

$$V_{j,k} = Q_{j,k} \tilde{U}_{j,k} \quad (4.4)$$

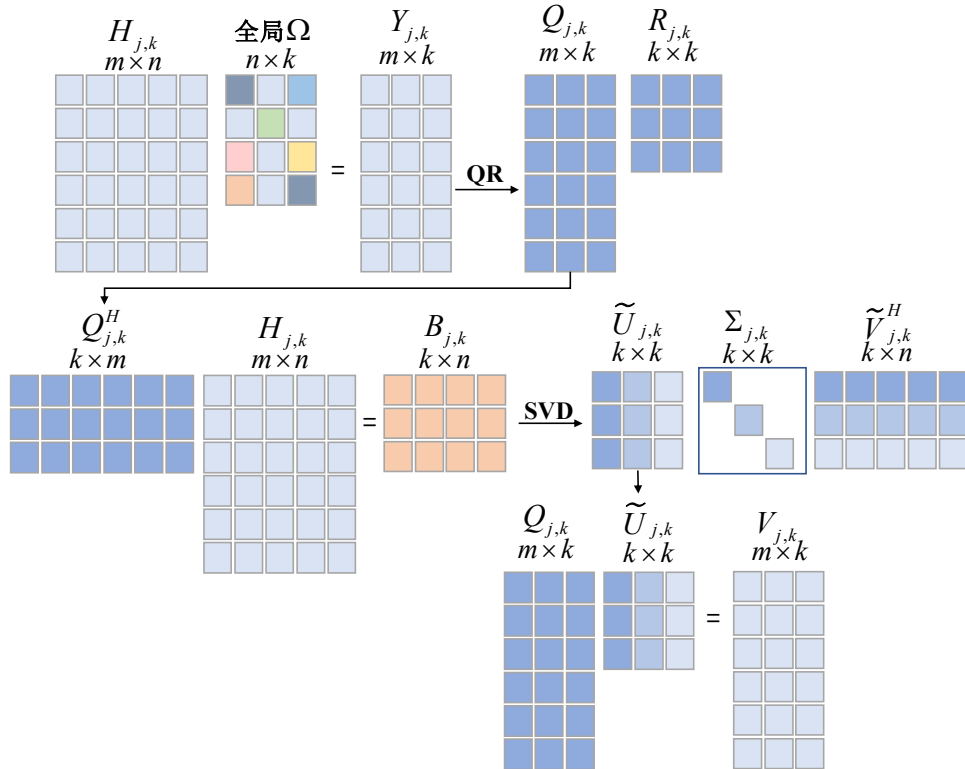


图 4.5 随机奇异值分解步骤示意图

4) 基于矩阵间相关性的改进随机奇异值分解

由于矩阵间的相关性，本文通过计算发现，相邻两个矩阵 $H_{j,k}$ 和 $H_{j,k+1}$ 计算出的 $Q_{j,k}$ 具有高度一致性。基于此结论，本文提出了一种共用标准正交矩阵的方法，从而改进随机奇异值分解算法以降低计算复杂度，最终得到本问题最终模型如图 4.6 所示。具体方法为：将同一行块中相邻 3 个矩阵划分为一组，用三个矩阵中中间矩阵的 Q 矩阵代替另外两个矩阵计算出的 Q 矩阵。即：把 $H_{j,k-1}$ 、 $H_{j,k}$ 和 $H_{j,k+1}$ 划分为一组，计算 $H_{j,k}$ 的 Q 矩阵 $Q_{j,k}$ 作为整个组共用的 Q 矩阵。通过该方法，可以省去每一组中两个矩阵的 QR 分解过程，从而大大降低计算复杂度。

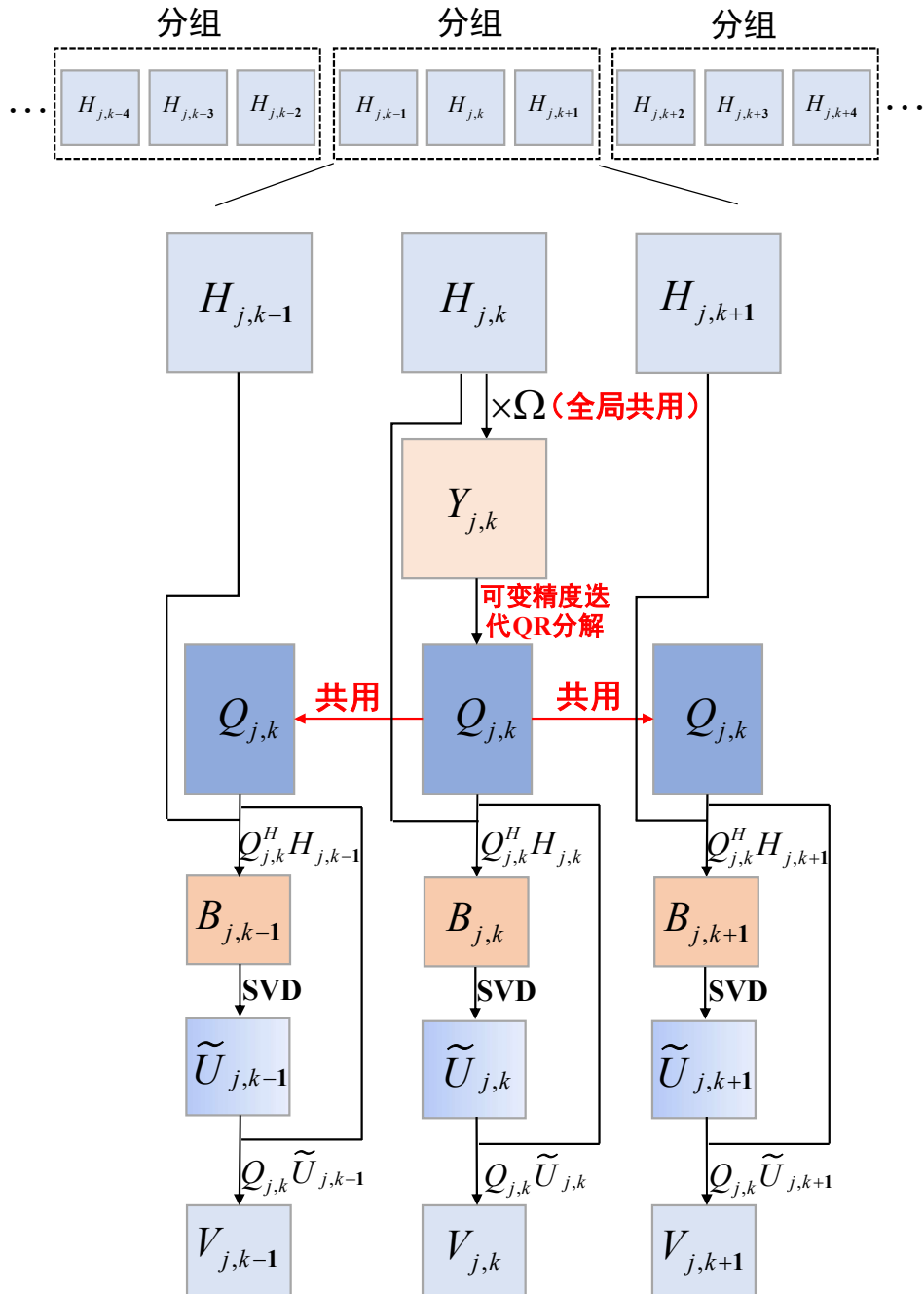


图 4.6 基于标准正交矩阵的改进随机奇异值分解流程图

5) 模型精度分析

本文按照本模型进行 6 组数据的奇异值分解，并进行建模精度分析，如图 4.7 所示。可以观察到，在迭代次数选择为 4 的情况下 Data1、Data2 和 Data3 不满足建模精度要求。

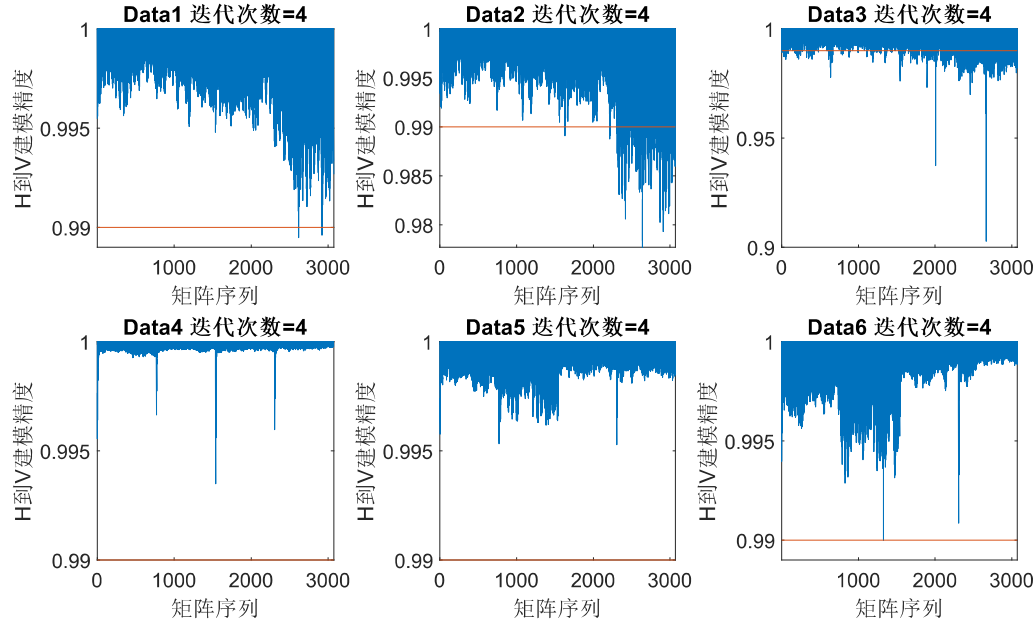


图 4.7 建模迭代次数均为 4 时建模精度 $\rho_{l,j,k}(V)$

因此，需要对 QR 分解过程中的迭代次数进行调整。最终所有数据建模精度均在 0.99 以上，满足建模要求，迭代次数及建模精度如图 4.8 所示。

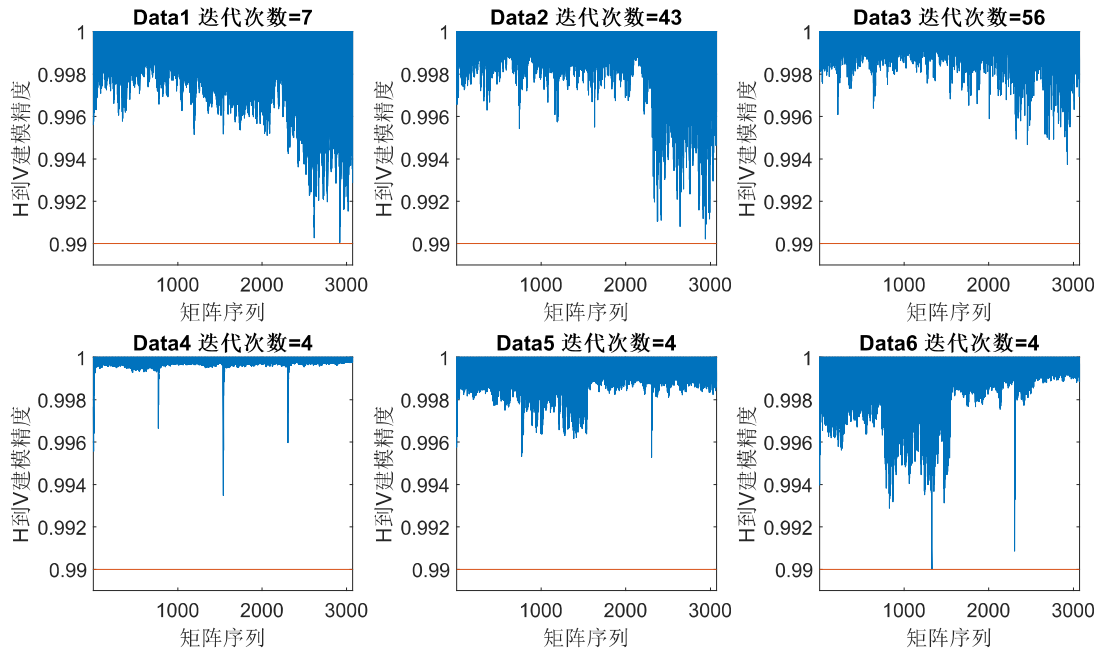


图 4.8 调整迭代次数后建模精度 $\rho_{l,j,k}(V)$

6) 计算复杂度分析

本文涉及到的所有计算复杂度详细计算过程见附录 A。

输入为 $J \times K$ 大小的相关复数矩阵组 $H = \{H_{j,k}\}$ ，其中每个矩阵大小为 $M \times N$ ，其中 $M = J$ ，对上述算法按步骤进行拆分，分析其计算复杂度：

(1) 生成一个高斯分布随机矩阵 $\Omega \in \mathbb{C}^{J \times J}$ 。这里将此运算归于其他运算类型中，其计算复杂度为 100。

将矩阵组 H 按行块进行考虑，下面的描述仅针对矩阵组中的某一行块。

(2) 对于某一个行块，每三个相邻矩阵 $H_{j,k-1}$ ， $H_{j,k}$ ， $H_{j,k+1}$ 组合为一组，该操作不涉及计算复杂度。

(3) 用三个矩阵中的中间矩阵 $H_{j,k} \in \mathbb{C}^{M \times N}$ 的共轭转置与 $\Omega \in \mathbb{C}^{J \times J}$ 矩阵相乘，得到临时矩阵 $Y \in \mathbb{C}^{N \times J}$ ，该操作计算复杂度为： **$16MNJ$** 。

(4) 对矩阵 $Y \in \mathbb{C}^{N \times J}$ 做迭代 QR 分解，迭代次数为 P ，第 p 次 QR 迭代分解涉及：

- $Y \in \mathbb{C}^{N \times J}$ 矩阵乘高斯分布随机向量（取 Ω 矩阵中的一列 $\omega \in \mathbb{C}^{J \times 1}$ ），该操作计算复杂度为： **$16NJ$** 。
- $N \times P$ 矩阵乘 $P \times N$ 矩阵，两个 $N \times N$ 矩阵相乘，以及 $N \times N$ 矩阵乘 $N \times 1$ 向量。该操作计算复杂度为 **$16N^2P + 16N^3 + 16N^2$** 。
- 求 $N \times 1$ 向量的 2 范数的倒数，将此运算归于其他运算类型中，其计算复杂度为 100。
- $N \times 1$ 向量乘以上述 2 范数的倒数，计算复杂度为 **$14N$** 。
- 拼接，此操作不涉及计算复杂度。

因此， P 次迭代 QR 分解总的计算复杂度为 $\sum_{p=1}^P (16NJ + 16N^2P + 16N^3 + 16N^2 + 100 + 14N)$ 。

(5) QR 分解得到了一个共用的 $N \times P$ 大小的 Q 矩阵，用这个 Q 矩阵的共轭转置与矩阵 $H_{j,k-1}$ ， $H_{j,k}$ ， $H_{j,k+1}$ 做三次乘法，得到 3 个 $P \times M$ 大小的降维矩阵，该操作计算复杂度为 **$48NPM$** 。

(6) 对 3 个 $P \times M$ 大小矩阵做 SVD，这里使用基于双对角化（bi-diagonalization）结合 QR 分解的操作，该操作计算复杂度为 **$3 \times (14PM^2 + 14M^3)$** 。

因为上述操作的是针对于三个相邻矩阵，并且一共由 J 个行块，因此本算法总的计算复杂度为：

$$100 + \frac{K}{3} J \left[16MNJ + \sum_{p=1}^P (16NJ + 16N^2P + 16N^3 + 16N^2 + 100 + 14N) + 48NPM + 42PM^2 + 42M^3 \right]$$

在满足建模精度的前提下，代入 $M=4$ ， $N=64$ ， $J=4$ ， $K=384$ ， $L=2$ ；得到六组数据对应的计算复杂度如表 4.1 所示：

表 4.1 六组数据对应的计算复杂度

Data1	1.62812540E+10
Data2	1.25934328E+11
Data3	1.76218358E+11
Data4	9.10643210E+09
Data5	9.10643210E+09
Data6	9.10643210E+09

4.1.2 问题 1.2——低计算复杂度矩阵求逆

在由 $\hat{V} = f_1(H)$ 得到 $V_{j,k}$ 之后，为了得到最终输出结果 $W = \{w_{j,k}\}$ ，将不同 j 下标、相同 k 下标的 $V_{j,k}$ 进行横向的拼接，得到维度为 $N \times LJ$ 的 $V_k = [V_{1,k} \cdots V_{j,k} \cdots V_{J,k}]$ ，然后根据式(4.5)获取 W_k ：

$$W_k = V_k (V_k^H V_k + \sigma^2 I)^{-1} \quad (4.5)$$

其中， σ^2 为固定常数； W_k 同 V_k ； I 为单位矩阵，维度为 $LJ \times LJ$ 。最后，将各 W_k 按式(4.6)进行拆解：

$$W_k = [W_{1,k} \cdots W_{j,k} \cdots W_{J,k}] \quad (4.6)$$

其中， $W_{j,k}$ 是 W_k 中顺序排列的子矩阵，维度为 $N \times L$ 。

令 $R = V_k^H V_k + \sigma^2 I$ ，因此上述步骤中，关键的步骤为对 R 求逆。并且容易知道该矩阵为正定 Hermite 矩阵，于是我们参考文献^[4]给出如下求逆方法：

1) 基于 Hermite 矩阵求逆引理的矩阵递推求逆方法

引理^[5] 有 Hermite 矩阵 H_i ，可设其分块形式为：

$$H_i = \begin{bmatrix} H_{i-1} & g_i \\ g_i^H & \rho_i \end{bmatrix} \quad (4.7)$$

其中 H_i ， H_{i-1} 的逆矩阵 H_i^{-1} ， H_{i-1}^{-1} 存在。若令 $b = -H_{i-1}^{-1} g_i$ ， $\beta = \rho_i - g_i^H H_{i-1}^{-1} g_i = \rho_i + g_i^H b$ ，则有：

$$H_i^{-1} = \begin{bmatrix} H_{i-1}^{-1} & 0_i \\ 0_i^H & 0 \end{bmatrix} + \frac{1}{\beta} \begin{bmatrix} b b^H & b \\ b^H & 1 \end{bmatrix} \quad (4.8)$$

又根据 R 矩阵为正定 Hermite 矩阵，因此易知其各阶主子矩阵也均为非奇异正定 Hermite 矩阵，因此可利用上述 Hermite 矩阵求逆引理实现基于递推的矩阵求逆算法，求逆算法如下：

令矩阵 R 的维数为 $K = N \times M$ ， ρ_i 表示 R 的第 i 个对角元素： $\rho_i = R(i, i)$ ， g_i 表示 R 第 i 列的前 $i-1$ 个元素组成的列向量： $g_i = R(1:i-1, i)$ ， R_i 是 R 的第 i 顺序主子矩阵：

$$R_i = R(1:i, 1:i)。$$

①初始化：

$$R_1^{-1} = \frac{1}{\rho_1} \quad (4.9)$$

②迭代计算：

$$b = -R_{i-1}^{-1} g_i \in \mathbb{C}^{(i-1) \times 1} \quad (4.10)$$

$$\beta = \text{real}(\rho_i + g_i^H b) \in \mathbb{R}^{1 \times 1} \quad (4.11)$$

$$q = \frac{b}{\beta} \in \mathbb{C}^{(i-1) \times 1} \quad (4.12)$$

$$R_i^{-1} = \begin{bmatrix} R_{i-1}^{-1} + qb^H & q \\ q^H & \frac{1}{\beta} \end{bmatrix} \in \mathbb{C}^{i \times i} \quad (4.13)$$

其中 $i = 2, 3, \dots, K$ 。

③递推结束得到 $R^{-1} = R_K^{-1}$ 。

在该算法中，容易证明 $\rho_i + g_i^H b$ 应为一个实数，但由于在迭代计算过程中由于计算误差的原因， $\rho_i + g_i^H b$ 将出现理论上不存在的虚部，该虚部误差能向下传递，导致计算误差急剧积累，造成算法失效。于是对其取实部 $\beta = \text{real}(\rho_i + g_i^H b)$ 能有效避免这种误差的影响。

2) 求逆误差分析

基于 Hermite 矩阵求逆引理的矩阵递推求逆算法为精确求逆算法。为了对此进行验证，本文按照该方法计算了给定的六组数据由矩阵数据 V 到 W 的建模精度，如图 4.9 所示，可以看出，对于矩阵组中每个矩阵，其建模精度均为 1，因此能说明该求逆算法精度等效于普通求逆算法。

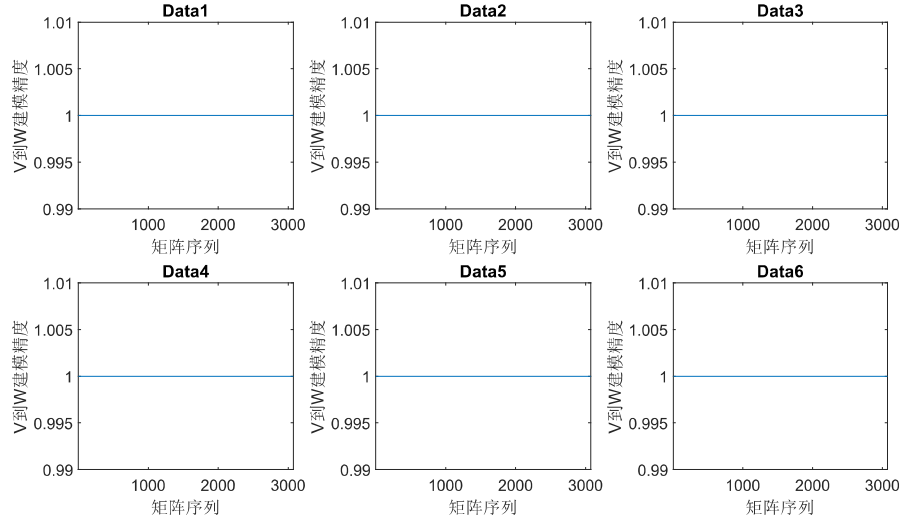


图 4.9 矩阵数据 V 到 W 的建模精度

3) 矩阵数据 H 到 W 建模精度分析

本文结合从矩阵数据 H 到 V 及 V 到 W 两部分算法，对从 H 到 W 整个过程的建模精度进行分析，得到结果如图 4.10 所示，发现整体误差增大，且部分数据的建模精度低于 0.99。分析其原因，虽然从矩阵数据及 V 到 W 为精确求逆，但由于从 H 到 V 部分具有误差，所以误差会在 V 到 W 部分进行放大，此外，还会受到计算舍入误差的影响。

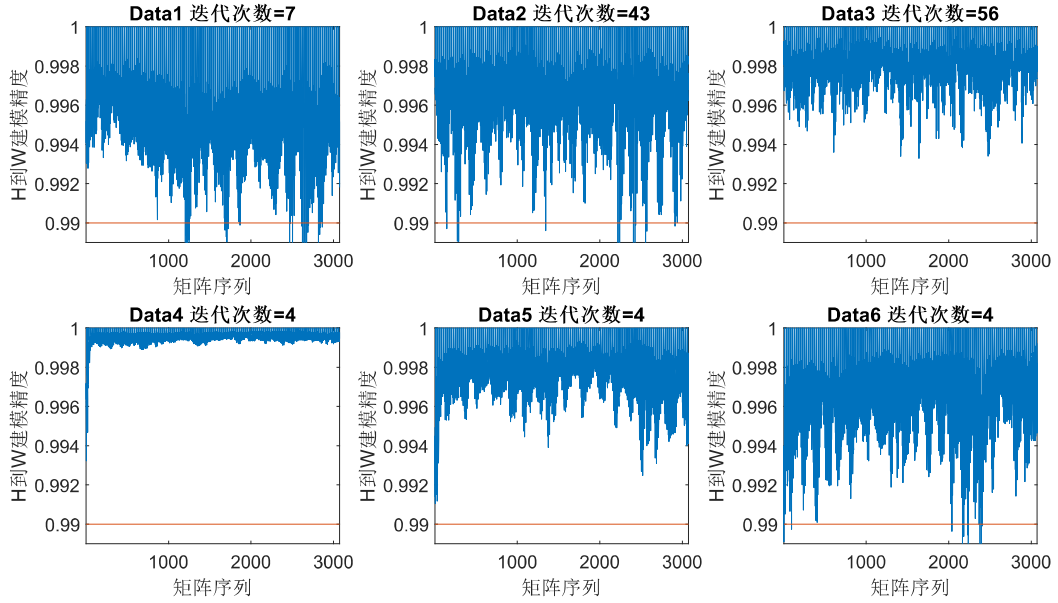


图 4.10 采用 3.1.1 对应迭代次数，矩阵数据 H 到 W 的建模精度

为了使所有结果满足建模精度要求，本文重新调整从 H 到 V 部分 QR 分解中的迭代次数，最终得到的建模精度和各自的迭代次数如图 4.11 所示。

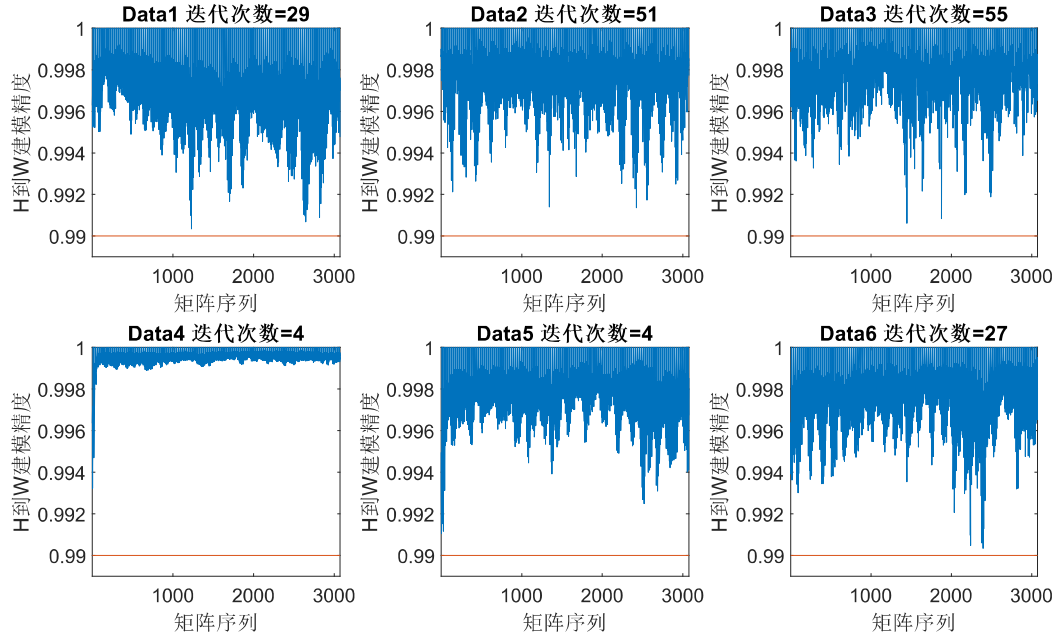


图 4.11 满足建模精度前提下调整迭代次数后，矩阵数据 H 到 W 的建模精度

4) 整体计算复杂度分析

首先分析基于 Hermite 矩阵递推求逆算法的计算复杂度：其输入为 $LJ \times LJ$ 的 V_k 矩阵。

在求逆过程中需要对于 $LJ \times LJ$ 的 V_k 矩阵迭代 $LJ - 1$ 次。

- (1) 初始化：求实数的倒数，计算复杂度为 25。
对于第 i 次迭代，有：
- (2) $i \times i$ 大小矩阵乘 $i \times 1$ 向量，并取实部，再取倒数，计算复杂度为 $16i^2 + 25$ 。
- (3) $i \times 1$ 向量乘以实数，计算复杂度为 $14i$ 。
- (4) $i \times 1$ 向量乘 $1 \times i$ 向量，两个 $i \times i$ 矩阵相加，拼接，计算复杂度为 $14i^2 + 2i^2 = 16i^2$

因此，求逆过程的总计算复杂度为：
$$\sum_{i=1}^{LJ-1} (25 + 16i^2 + 25 + 14i + 16i^2) = \sum_{i=1}^{LJ-1} (50 + 32i^2 + 14i)$$

- (5) $M \times LJ$ 矩阵乘以 LJ 阶矩阵，计算复杂度为 $16M(LJ)^2$ 。

因为上述操作的是针对于矩阵组的一整列而言，本算法总的计算复杂度为：

$$K \left[16M(LJ)^2 + \sum_{i=1}^{LJ-1} (50 + 32i^2 + 14i) \right]$$

在满足建模精度的前提下，代入 $M=4$, $N=64$, $J=4$, $K=384$, $L=2$ 求得 $LJ \times LJ$ 矩阵求逆复杂度为：3578112。

4.2 问题 2

4.2.1 线性插值

由 4.1.1 的线性相关分析可知，同一行块中相邻两个矩阵间存在明显的线性相关性。

因此可以采用线性插值的方法，通过周围矩阵的线性组合表示中间矩阵。对于本题中的 $H_{j,k}$ ，可采用其左右两个矩阵对其进行表示，即：

$$H_{j,k} = pH_{j,k-1} + qH_{j,k+1} \quad (4.14)$$

其中 p 和 q 为左右两个矩阵的权值，且 $p + q = 1$ 。

①加权平均

由式(4.14)的计算方法可知， $H_{j,k}$ 即为 $H_{j,k-1}$ 与 $H_{j,k+1}$ 的加权平均。 p 和 q 的选择可根据 4.1.1 节中的 Pearson 线性相关系数进行确定。设 R_1 、 R_2 分别为 $H_{j,k}$ 与 $H_{j,k-1}$ 、 $H_{j,k}$ 与 $H_{j,k+1}$ 的 Pearson 线性相关系数，则：

$$p = \frac{R_1}{R_1 + R_2}, q = \frac{R_2}{R_1 + R_2} \quad (4.15)$$

该式的数学意义为：对于相关性大的矩阵，在求加权平均时应当具有更大的权值，且权值与相关系数成正比，反之亦然。

②算数平均

采用加权平均尽管能很好地对中间矩阵 $H_{j,k}$ 进行预测，但计算相关系数的过程增大了计算复杂度。为此，本文令 $p = q = \frac{1}{2}$ ，则将式(4.15)转化为下式：

$$H_{j,k} = \frac{H_{j,k-1} + H_{j,k+1}}{2} \quad (4.16)$$

由式(4.16)的计算方法可知， $H_{j,k}$ 为 $H_{j,k-1}$ 与 $H_{j,k+1}$ 的算数平均。该平均方法在 $H_{j,k}$ 与左右两个矩阵相关性相似的情况下具有良好的预测能力。

4.2.2 基于矩阵相关性的压缩与解压缩模型

①基于算术平均预测的压缩与解压缩模型

本文基于计算复杂度较低的算数平均方法，首先提出一种基于算术平均预测的压缩与解压缩模型，如图 4.12 所示。对于同一行块中的矩阵，在压缩时，舍弃第 1、第 3、第 5 个等第奇数个矩阵，只存储第偶数个矩阵。解压时，读取第偶数个矩阵，对于之前舍弃的第奇数个矩阵，则采用公式(4.16)所示的方法进行预测，从而还原出整个矩阵。其他行块同理。

本文采用压缩率评价数据压缩性能。压缩率 η 计算公式为：

$$\eta = \frac{B_1}{B_0} \quad (4.17)$$

其中 B_1 为数据压缩后占用空间， B_0 为数据压缩前占用空间。由该式可知， η 越小，说明数据压缩程度越大。

对于本题中的数据，容易计算出该方法压缩率为 49.74%。

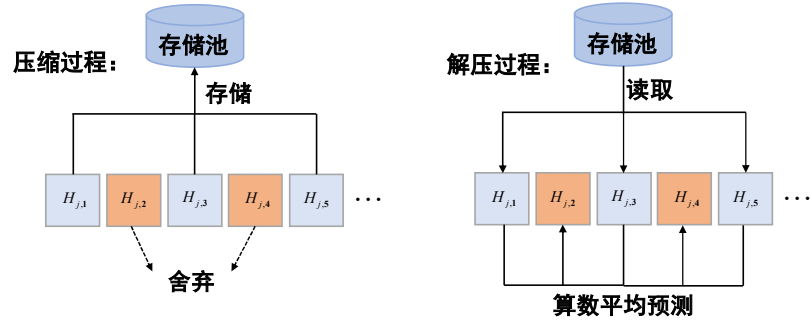


图 4.12 基于算术预测的压缩与解压缩模型

②基于可变压缩误差算术平均预测的压缩与解压缩模型

①模型具有较高的压缩率和 $O(n)$ 的时间复杂度。然而，由于其压缩方式固定，所以无法改变压缩率及压缩误差。因此，本文对①模型进行改进，得到基于可变压缩误差算术平均预测的压缩与解压缩模型。如图 4.13 所示。该模型的改进在于：压缩过程中，舍弃同一行块中序号为 k 的倍数（ k 为整数且 $2 \leq k < 384$ ）的矩阵，其余矩阵存储。解压过程则读取保存的矩阵，对于压缩过程中舍弃的矩阵，则利用舍弃矩阵左右两边矩阵的算术平均进行预测。其他行块同理。可见，当 k 值增大时，舍弃的矩阵变少，压缩率降低，但压缩误差减少。当 $k=2$ 时，该模型和①中的模型相同。

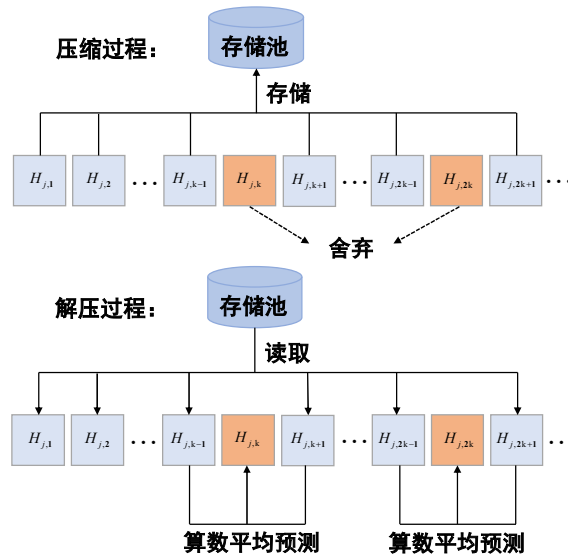


图 4.13 基于可变压缩误差算术平均预测的压缩与解压缩模型

③基于可变压缩误差加权平均预测的压缩与解压缩模型

为了在②的基础上，同时获得较高的压缩率和较低的压缩误差，本文进一步提出基于可变压缩误差加权平均预测的压缩与解压缩模型。如图 4.14 所示。相比于②模型，此处模型的改进在于：将原本的算术平均替换成了加权平均，从而获得更好的预测性能。因此在压缩阶段，相比于②，此处模型还计算了舍弃矩阵与其左右两边矩阵的 Pearson 线性相关系数的计算及相关系数的存储。在解压阶段，相比于②，此处模型还需要读取相关系数，并在预测时采用加权平均。在性能上，相比于前一种模型，此处模型具有更好的压缩率和更低的压缩误差。但由于需要计算相关系数，计算复杂度明显高于前两种模型。

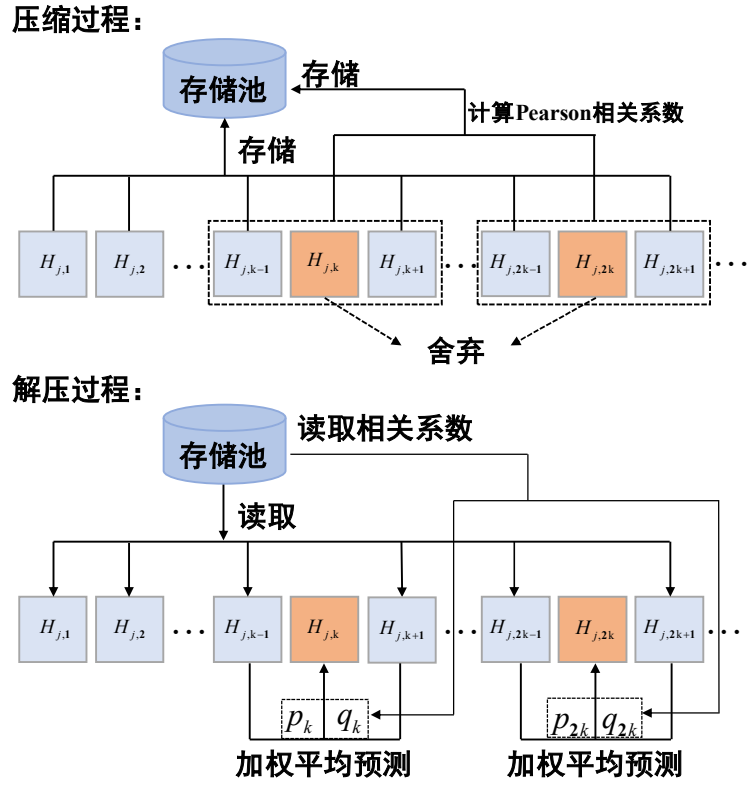


图 4.14 基于可变压缩误差加权平均预测的压缩与解压缩模型

4.2.3 基于矩阵间相关性的 H 矩阵组压缩

本文优先考虑计算复杂度和压缩率，所以按照①②③的顺序对 6 组数据进行尝试。首先采用①模型对 6 组数据进行压缩，得到 6 个矩阵组的 err_H 结果如图 4.15 所示：

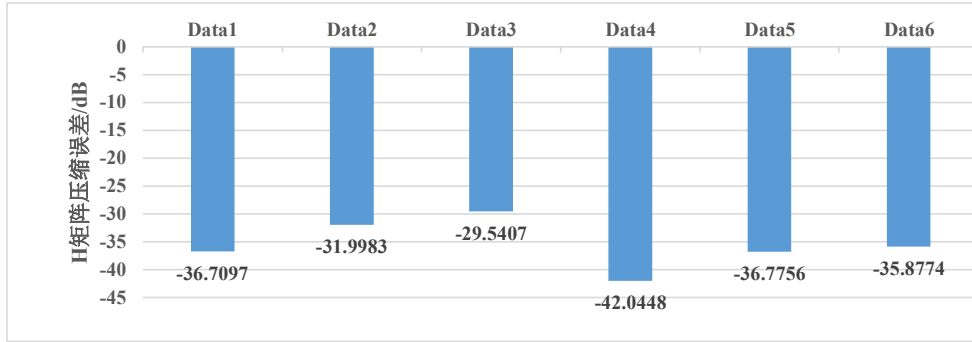


图 4.15 采用①模型的 err_H

可以观察到，只有 Data3 未达到条件。因此针对 Data3，采用②模型进一步处理，取 $k=3$ ，即每隔 2 个矩阵舍弃一个，得到压缩误差为 -31.3264，达到题目要求。因此，对于矩阵组 H 的压缩，最终方案为：除 Data3 以外，采用①模型，Data3 则采用 $k=3$ 的②模型。6 组数据最终的压缩误差如图 4.16 所示，6 个模型压缩率如表 4.2 所示。

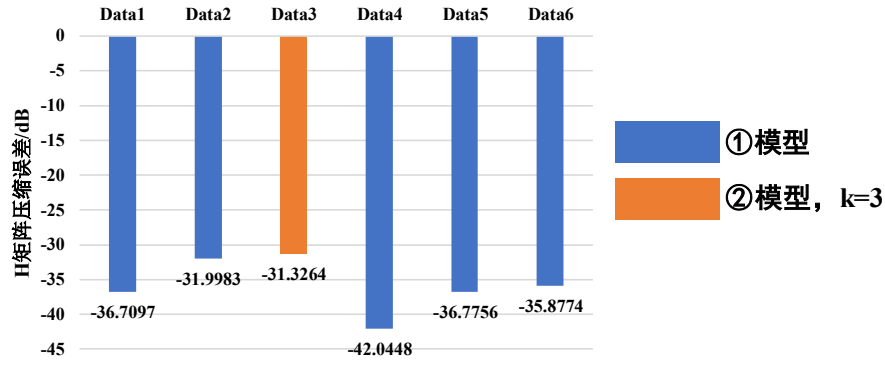


图 4.16 Data3 采用②模型，其余为①模型的 err_H

表 4.2 H 矩阵组压缩的 k 值选取及压缩率

数据	Data1	Data2	Data3	Data4	Data5	Data6
k 值	2	2	3	2	2	2
压缩率	50.26%	50.26%	66.67%	50.26%	50.26%	50.26%

4.2.4 基于矩阵间相关性的 W 矩阵组压缩

1) 压缩方案及压缩误差

本文同样先采用①模型进行压缩，压缩误差如图 4.18 所示，可以看出，除 Data4 外均未达到要求，且与 -30dB 相差较远。为了分析原因，本文绘制了矩阵压缩误差最大的 Data3 矩阵组 W 第一行块各矩阵间的相关系数矩阵，如图 4.17 所示。可以观察到，整体上 W 矩阵组的相关性明显弱于 H 矩阵，且观察局部放大图可知，个别矩阵至于其某一侧的矩阵有较强相关性，而与另一侧相关性较弱，这导致算数平均方法无法实现很好的预测性能，而加权平均的方法则能够很好地考虑到左右两侧矩阵与中间矩阵的相关性差异，从而实现更好的预测性能。

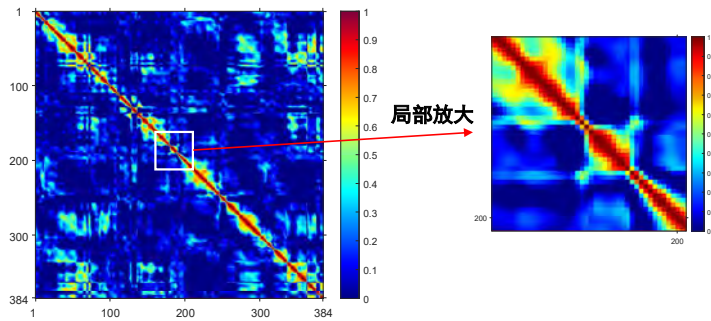


图 4.17 W 矩阵组相关系数矩阵

因此，本文对于其他除 Data4 外的其他数据，采用③模型。并根据具体数据，选择合适的 k 值，最终所有数据全部达到要求。最终结果如图所示，k 值选取及压缩率如表 4.3 所示。

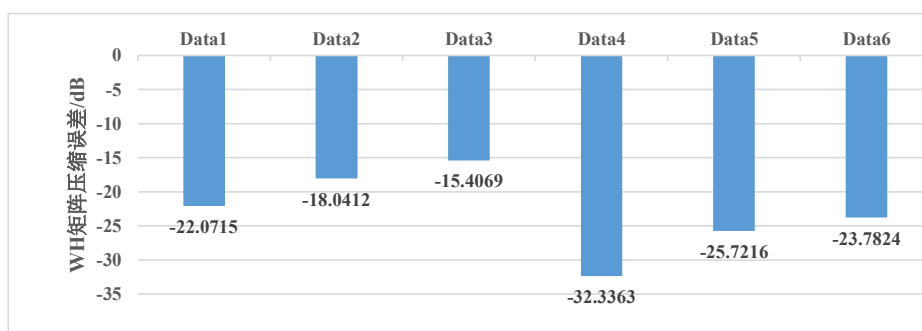


图 4.18 采用①模型的 err_H

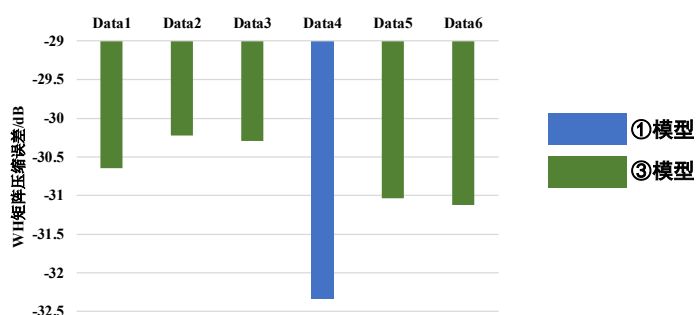


图 4.19 Data4 采用①模型，其余采用③模型的 err_H

表 4.3 W 矩阵组压缩的 k 值选取及压缩率

数据	Data1	Data2	Data3	Data4	Data5	Data6
k 值	7	18	29	2	3	5
压缩率	85.94%	94.53%	96.61%	50.26%	66.93%	80.21%

2) 不同压缩方案压缩性能对比

为了进一步分析不同压缩方案的压缩误差,本文使用三种模型对 W 矩阵 6 组数据进行处理,得到压缩误差如图 4.20 和表 4.4 所示。其中②模型采用的 k 值与③模型相同。可以观察到对比①模型和②模型,通过舍弃更少的矩阵可以使压缩误差得到明显降低。但压缩率会有所损失。对比②模型和③模型,可以看出采用加权平均的方法后,压缩误差得到了进一步降低,且不影响压缩率。但会增加计算复杂度和存储空间。

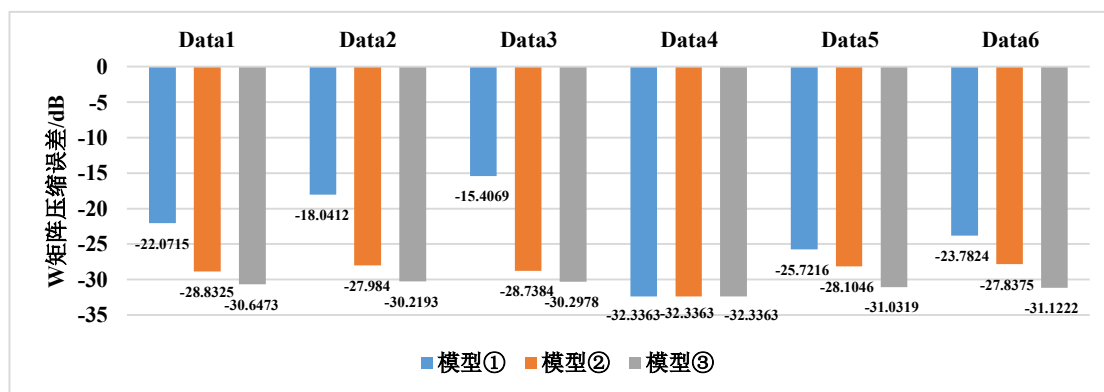


图 4.20 采用三种模型的 err_H

表 4.4 W 矩阵组压缩的 k 值选取及压缩率

数据	Data1	Data2	Data3	Data4	Data5	Data6
k 值	7	18	29	2	3	5
压缩率	85.94%	94.53%	96.61%	50.26%	66.93%	80.21%

4.2.5 存储复杂度及计算复杂度分析

在满足给定的建模精度下，计算存储复杂度与压缩解压缩复杂度。

对于模型①和②，对于某一行块，由于最后一个矩阵始终保留，可以分析出舍弃的矩阵数量 N_{Lost} 为：

$$N_{Lost} = \left\lfloor \frac{(K-1)}{k} \right\rfloor \quad (4.18)$$

因此，模型①和②算法的存储复杂度为：

$$64MNJK \frac{\left(K - \left\lfloor \frac{(K-1)}{k} \right\rfloor \right)}{K} = 64MNJ \left(K - \left\lfloor \frac{(K-1)}{k} \right\rfloor \right) \quad (4.19)$$

由于模型①和②，由于压缩过程只涉及数据丢弃，因此压缩过程计算复杂度为 0。

对于解压过程，涉及数据的算术平均，涉及到的操作为：对所有舍弃矩阵做算术平均插值，其计算复杂度为：

$$8MNJN_{lost} = 8MNJ \left\lfloor \frac{(K-1)}{k} \right\rfloor \quad (4.20)$$

对于模型③，其存储复杂度为模型①或②的基础上增加了相关系数，且增加的相关系数数量为：

$$2J \left\lfloor \frac{(K-1)}{k} \right\rfloor \quad (4.21)$$

每个系数为一个 32bit 单精度浮点数，所以模型③总存储复杂度为：

$$64J \left\lfloor \frac{(K-1)}{k} \right\rfloor + 64MNJ \left(K - \left\lfloor \frac{(K-1)}{k} \right\rfloor \right) \quad (4.22)$$

对于压缩过程，需要计算相关系数，对于一个矩阵组，总共需要计算的相关系数数量为： $2N_{Lost}J$ ，根据式(4.1)，对于一次相关系数的计算，其计算复杂度为 $82MN + 78$ ，因此压缩过程计算复杂度为：

$$2J \left\lfloor \frac{(K-1)}{k} \right\rfloor (82MN + 78) \quad (4.23)$$

对于解压过程，需要利用加权平均对丢弃的数据进行预测。因此解压过程计算复杂度为：

$$12MNJN_{lost} = 12MNJ \left\lfloor \frac{(K-1)}{k} \right\rfloor \quad (4.24)$$

4.3 问题 3

考虑从矩阵输入信号 H 到近似矩阵输出信号 $\hat{W}$ 的端到端流程, 利用本文建立的模型求解本问题流程图如图 4.21 所示: 由 H 矩阵到压缩后的 W 矩阵需要经过三次矩阵间的变换。

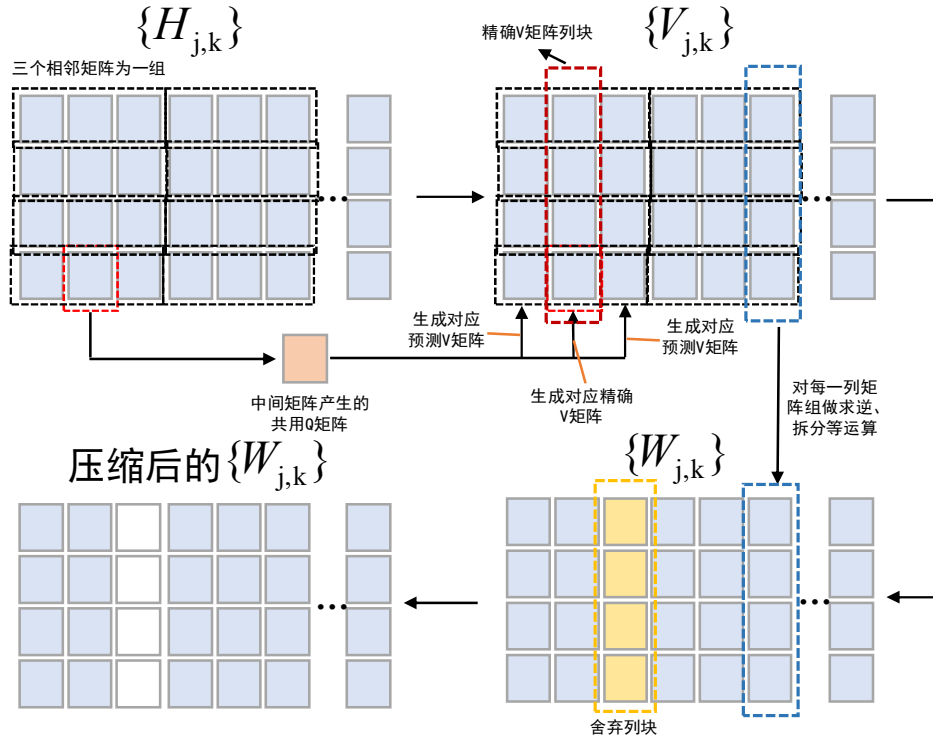


图 4.21 本文建立的模型流程图

通过分析, 由于压缩方法是通过舍弃列块的方式实现的, 因此可以充分利用需要舍弃的列块在原矩阵组中的位置信息。因此可以对本文的模型进行优化:

由 H 矩阵生成 V 矩阵时, 可以仅对需要的列块进行恢复。其中需要的列块可由问题 2 分析得到。再由得到的 V 矩阵通过求逆、拆分等运算, 可以直接得到压缩后的矩阵。

优化后的模型如图 4.22 所示。容易得出 V 矩阵相比于优化前的模型, V 矩阵的计算复杂度更小, 且不用生成压缩前的 W 矩阵, 而是直接得到压缩后的矩阵, 因此其计算复杂度和空间复杂度均得到较大的降低。

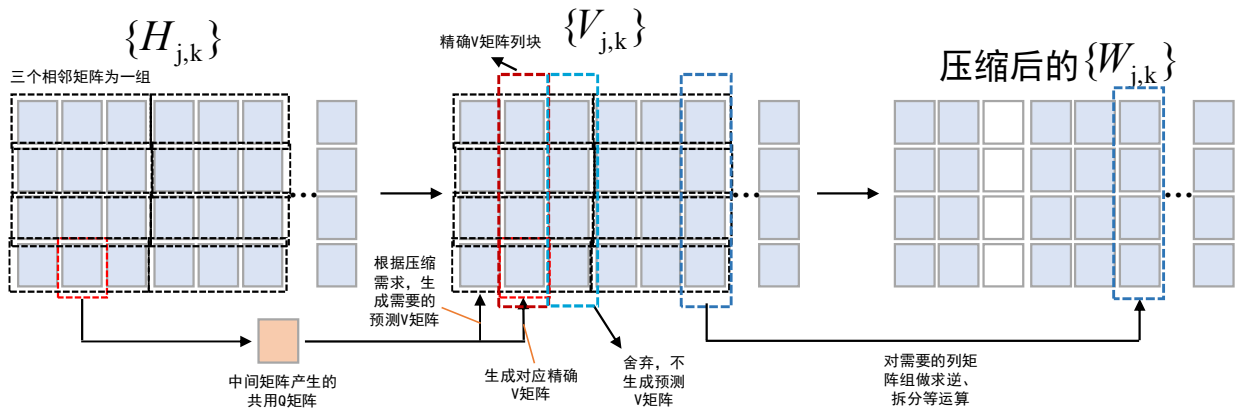


图 4.22 本文建立的模型优化后流程图

5 模型评价与改进

5.1 模型优点

- 1) 利用矩阵间相关性, 提出了全局高斯随机矩阵、相邻矩阵间 Q 矩阵共用的方法。基于以上方法对随机奇异值分解进行了改进。相比于常规奇异值分解和常规随机奇异值分解而言, 其计算复杂度与存储复杂度均有所降低。
- 2) 利用 Hermite 矩阵求逆引理得到了递推矩阵求逆算法。相比于常规矩阵求逆方法, 其计算复杂度更低。
- 3) 利用矩阵间相关性, 仅依靠矩阵的拆解与组合操作, 就能实现矩阵的压缩, 其计算复杂度接近于 0。
- 4) 本文提出的压缩与解压缩方法能实现压缩效率和计算复杂度的互换。

5.2 模型缺点及改进方向

- 1) 随机奇异值分解中, 对降维后矩阵做奇异值分解时可采用复杂度更低的方法。如基于双对角化 (bi-diagonalization) 结合 QR 分解的操作。
- 2) 对矩阵乘法能进一步做改进。如使用 Strassen's 方法或 Coppersmith–Winograd 方法。
- 3) 当矩阵组的相关性较弱时, 本模型的压缩效率较低。可以尝试充分利用相关系数矩阵进行更合理的插值预测方法。

参考文献

- [1] 曾小波, 魏祖宽, 金在弘. 协同过滤系统的矩阵稀疏性问题的研究[J]. 计算机应用, 2010, 30(4):1079-1082.
- [2] 郭文彬, 魏木生. 奇异值分解及其在广义逆理论中的应用[M]. 科学出版社, 2008.
- [3] 维基百科, 奇异值分解, <https://zh.wikipedia.org/wiki/奇异值分解>, 2021.10.17.
- [4] 曹建蜀. 机载相控阵雷达 STAP 算法研究[D]. 电子科技大学, 2007.
- [5] 格罗布.G.H.等. 矩阵计算[M]. 上海科学技术出版社, 1988.

附录 A 计算复杂度分析

本文采用复数向量内积和外积的计算复杂度为基础，分析复数矩阵运算的计算复杂度。

对于 $X \in \mathbb{C}^{N \times 1}$ 和 $Y \in \mathbb{C}^{N \times 1}$ 的向量，两个向量内积涉及 N 次复数乘法和 N 次复数加法，参考题目附录可知，其总计算复杂度为 $16N$ ；两个向量外积涉及 N^2 次复数乘法，计算复杂度为 $14N^2$ 。

对于两个复数矩阵 $A \in \mathbb{C}^{M \times N}$ 和 $B \in \mathbb{C}^{N \times K}$ ，相乘，涉及 $M \times K$ 次 N 维向量内积，因此总计算复杂度为 $16MNK$ 。

对于一个复数矩阵 $A \in \mathbb{C}^{M \times N}$ 和 $Y \in \mathbb{C}^{N \times 1}$ 的向量乘积，涉及 M 次 N 维向量内积，计算复杂度为 $16MN$ 。