THE MULTI-DIMENSIONAL DECOMPOSITION WITH
CONSTRAINTS

Ilgis Ibragimov, Elena Ibragimova

Elegant Mathematics LLC, 82834 WY USA &
Elegant Mathematics Ltd, 66564 Germany

e-mail: ii@elegant-mathematics.com

ABSTRACT

We search for the best fit in the Frobenius norm of $A \in \mathbb{C}^{m \times n}$ by a matrix product $BC^*$, where $B \in \mathbb{C}^{m \times r}$ and $C \in \mathbb{C}^{n \times r}$, with $r \leq m$ so that $B = \{b_{ij}\}_{i=1, \ldots, m}^{j=1, \ldots, r}$ is defined by some unknown parameters $\sigma_1, \ldots, \sigma_k$, $k << mr$, and all partial derivatives of $\frac{\delta b_{ij}}{\delta \sigma_l}$ are definite, bounded, and can be computed analytically.

We show that this problem transforms to a new minimization problem with only $k$ unknowns by the analytical computation of the gradient of the minimized function over all $\sigma$. The complexity of computation of this gradient is only 4 times greater than the complexity of computation of the function, and this new algorithm needs only $3mr$ additional words in memory.

We apply this approach for the solution of the three-way decomposition problem and obtain good result of convergence for the Broyden algorithm.

INTRODUCTION

Suppose we have $A \in \mathbb{C}^{m \times n}$. The idea is to find $B \in \mathbb{C}^{m \times r}$ and $C \in \mathbb{C}^{n \times r}$, $r \leq m$ so

$$\min_{C,\sigma_1,\ldots,\sigma_k} ||A - B(\bar{\sigma})C^*||_F^2,$$

that $B = \{b_{ij}\}_{i=1, \ldots, m}^{j=1, \ldots, r}$ is defined by unknown parameters $\sigma_1, \ldots, \sigma_k$, $k << mr$, and all partial derivatives of $\frac{\delta b_{ij}}{\delta \sigma_l}$ are definite, bounded and can be computed analytically.

This problem occurs in statistics [1], nuclear magnetic resonance [2], spectroscopy and multi-dimensional decomposition [3]. Consider one popular application [4] — a low rank approximation of two and multidimensional data array with one factor matrix containing vectors formed as complex exponents:

$$\min_{B,\sigma} \sum_{j=1}^{J} \sum_{k=1}^{K} \left| \sum_{l=1}^{L} a_{jk} - \sum_{l=1}^{L} b_{jl} e^{i\sigma_k} \right|_2^2,$$
and

\[
\min_{B, \sigma} \sum_{j=1}^{J} \sum_{k=1}^{K} \left| a_{jk} - \sum_{l=1}^{L} b_{jl} e^{i\sigma_k l} \right|^2,
\]

Since the total amount of minimizing parameters \( \sigma \) usually is several orders less than the total amount of minimizing parameters in \( B \), it is highly desired to perform minimization over only \( \sigma \) to save computational complexity.

If we freeze \( B \), then this function is linear in \( C \), and \( C = A^* B (B^* B)^{-1} \). The problem (1) then turns into a new nonlinear problem with only \( k \) unknowns:

\[
\min_{\sigma_1, \ldots, \sigma_k} ||A - B(B^* B)^{-1} B^* A||_F = \min_{\sigma_1, \ldots, \sigma_k} \sqrt{||A||^2_F - ||A^* Q(B)||^2_F},
\]

where \( Q(B) \in \mathbb{C}^{m \times r} \) contains the orthonormal subspace from \( B \).

The main difficulty in applying minimization methods for (4) is the computation of the gradient of the function over all \( \sigma \). The finite difference method needs \( k^2 \) or \( 2k^2 \) computations of this function for one evaluation of the gradient and cannot be considered accurate. There is a good alternative for it, Baur-Strassen (BS) method [6], which allows computing the gradient of a function using only \( 5n \) operations if the original function can be computed by \( n \) simple arithmetical operations with no more than 2 operands. The big disadvantage of the BS method is its memory requirement: it needs \( \mathcal{O}(n) \) words in memory, which is too many for most applications.

We suggest a new approach for computing the gradient of a function. This approach contains Modified Gramm–Schmidt (MGS) orthogonalization with low memory requirements and is based on the BS method.

**ALGORITHM**

To compute (4), we perform the following steps:

1) create \( B \) from \( \sigma_1, \ldots, \sigma_k \);

2) compute orthonormal subspace \( Q \) in \( B \);

3) compute (4).

In this article, we discuss how to compute a gradient \( \hat{g} \in \mathbb{C}^{mr} \) of (4) over all entries of \( B \). We will use both \( G \in \mathbb{C}^{m \times r} \) and \( \hat{g} \) for the same data. Let the dependence of \( B \) on \( \sigma_1, \ldots, \sigma_k \) be so simple that one can compute the gradient of (4) by \( \sigma_1, \ldots, \sigma_k \) if \( G \) is known.

Steps 2 and 3 need \( mr \) additional words in memory and compute within \( 2mr(r + n) \) arithmetical operations in the event that the MGS algorithm is used for step 2. The BS
algorithm can compute the gradient with the same order of arithmetical complexity but needs $4mr(r + n)$ additional words in memory.

Let us consider a computation of (4) from $B$. Let $B = [b_1, \ldots, b_k]$ be the initial matrix and $Q = [q_1, \ldots, q_k]$ the orthonormal subspace, which we are going to compute. Then

\[ q_1 = \frac{b_1}{||b_1||_2}, \]

\[ \text{do } i = 2, r \]
\[ u = b_i, \]
\[ \text{do } j = 1, i - 1 \]
\[ u = u - q_jq_j^*u \]
\[ \text{endo} \]
\[ q_i = \frac{u}{||u||_2} \]
\[ \text{endo} \]

\[ f = \sqrt{||A||_F^2 - \sum_{i=1}^{r} ||A^*q_i||_2^2} \]

Let’s construct a gradient of $f$ by $B$. We will call $dy_i \in \mathbb{C}^m$ the vector of derivatives — each $k$-th element of this vector contains the derivative of the $k$-th element of vector $y_i$.

Then there are the following formulas for the gradient:

\[ dq_1 = \frac{1}{||b_1||_2}(I - q_1q_1^*)db_1 \]

\[ \text{do } i = 2, r \]
\[ u = b_i \]
\[ \text{do } j = 1, i - 1 \]
\[ du_{\text{new}} = (I - q_jq_j^*)du_{\text{old}} - (q_j^*u_{\text{old}}I + u_{\text{old}}q_j^*)dq_j \]
\[ \text{endo} \]
\[ dq_i = \frac{1}{||u||_2}(I - q_iq_i^*)du \]
\[ \text{endo} \]

\[ df = -\frac{1}{f} \sum_{i=1}^{r} q_i^*AA^*dq_i \]

We can write all these equations in matrix notation:

\[
\begin{pmatrix}
I_{mr \times mr} \\
F_{mr \times \frac{mr(r+1)}{2}} \\
0 \\
\frac{L_{mr(r+1) \times \frac{mr(r+1)}{2}}}{h_{mr(r+1) \times \frac{mr(r+1)}{2}}} \\
0 \\
0 \\
0
\end{pmatrix}
\begin{pmatrix}
\frac{\delta}{\delta b_{ij}} \\
\frac{\delta}{\delta g^*}
\end{pmatrix}
= \begin{pmatrix} I_{mr \times mr} \\
0 \end{pmatrix}
\] or
\[
\begin{pmatrix}
I_{mr \times mr} & F^*_{mr \times \frac{mr(r+1)}{2}} & 0 \\
0 & L^*_{\frac{mr(r+1)}{2} \times \frac{mr(r+1)}{2}} & h_{\frac{mr(r+1)}{2}} \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
\hat{g} \\
\ast \\
\ast
\end{pmatrix}
= \begin{pmatrix}
1 \\
0 \\
\vdots \\
0
\end{pmatrix},
\]

(5)

where \( L_{\frac{mr(r+1)}{2} \times \frac{mr(r+1)}{2}} \) is a lower block triangular matrix with block size \( m \times m \). This matrix has \( I_{m \times m} \) blocks on the diagonal. The block matrix \([F, L]\) contains no more than 3 nonzero blocks in each row (see Fig. 1 for one example with 4 vectors). We marked with \( \ast \) the elements that we are not interested in; \( \hat{g} \) is the vector of the gradient of the original function over all \( b_{ij} \). To compute \( \hat{g} \), we solve the linear system (5). If we create \( L \) and \( F \) matrices, then we need at least \( mr(r + 3) \) words to solve \( L \).

Figure 1 Shows the matrix \([F, L]\) when \( B \) has 4 vectors, here \( S = \frac{I - uu^*}{||u||_2}, V = q^*uI + uq^*, W = qq^* - I \).

We suggest an improvement where we need only \( 4mr \) words to store some parts of \( L \) and \( F \) but still compute the solution. Let’s remark that in the loop for the variable \( j \), we update \((i - 1) \) times vector \( u \). If we store matrices \( B \) and \( Q \), we can recompute all updates of \( u \) from this loop for particular \( i \) with \( 2(i - 1)M \) additional arithmetical operations and store them in one additional array \( T \in \mathbb{C}^{m \times r} \). Then, during backward substitution we recompute all updates of \( u \) only when we need it. Obviously, this occurs \( r - 1 \) times for all \( i = r, \ldots, 2 \). All multiplications to matrices \( S, V, \) and \( W \) need \( \mathcal{O}(m) \) arithmetical operations. Thus, we need only \( B, Q, T, \) and \( G \) arrays with size \( m \times r \) for this computation. Here is an algorithm:
\[ G = -\frac{1}{f}AA^*Q \]

\[ \text{do } i = r, 1, -1 \]
\[ t_1 = b_i \]
\[ \text{do } j = 1, i - 1 \]
\[ z_j = q_j^*t_j \]
\[ t_{j+1} = t_j - z_j q_j \]
\[ \text{endo} \]
\[ g_i = \frac{g_i - g_i q_i^* g_i}{\|t_i\|_2} \]
\[ \text{do } j = i - 1, 1, -1 \]
\[ \alpha = q_j^* g_i \]
\[ g_j = g_j - z_j^* g_i - \alpha^* t_j \]
\[ g_i = g_i - \alpha q_j \]
\[ \text{endo} \]

Here we use the \( Z = (z_1, \ldots, z_r) \in \mathbb{C}^r \) array with only \( r \) elements for better performance.

The total arithmetical complexity of the computation of the gradient is \( 4mr(2r + n) \) operations. If we compare this with MGS (\( 2mr(r + n) \)), it is less than 4 times greater.

We obtain similar results for the Gramm-Schmidt (not MGS) orthogonalization: it needs \( 3mr + \frac{r(r+1)}{2} \) words in memory and works with \( 2mr(3r + 2n) \) operations, but because of stability issues we do not recommend using it.

**NUMERICAL EXPERIMENTS**

First we compare the general characteristics of our new approach with those of well-known approaches. We create the complex matrices \( A \) and \( B \) with random numbers, compute derivatives for different sizes of the problems by our new methods based on Gramm-Schmidt (AGS) and Modified Gramm-Schmidt (AMGS) algorithms, and compare our methods with the finite difference (FD) and Baur-Strassen (BS) methods (Tables 1, 2).

Furthermore, we show how those algorithms work. We perform a set of experiments and check the number of iterations for convergence of the Broyden method \[7\]. In this set of experiments, the matrix \( B \) is a real matrix with \( b_i = p_i \otimes q_i \in \mathbb{R}^{n^2} \), \( i = 1, \ldots, R \), where \( \otimes \) is the Kronecker product of vectors and \( p_i, q_i \in \mathbb{R}^n \) are unknown vectors. We change \( n \in [2, 20] \) and \( r \in [2, 20] \) (Table 3). This problem occurs in the three-way decomposition \[3, 5\].

Hence, our new method (AMGS) is stable enough (like the BS method), yet also up to a thousand times faster than BS and FD methods and does not require much additional memory (only 4 times more than FD).
Table 1. Memory requirements (in words) for FD, BS, AGS, and AMGS methods.

| m  | n  | r  | FD  | BS  | AGS | AMGS |
|----|----|----|-----|-----|-----|------|
| 2  | 1  | 1  | 4   | 152 | 12  | 14   |
| 10 | 2  | 2  | 40  | 2.5k| 92  | 124  |
| 100| 10 | 10 | 1.9k| 619k| 4.1k| 5.9k |
| 1000| 100| 100| 195k| 610m| 410k| 586k |
| 1000| 100| 100| 195k| 33.5m| 410k| 586k |

Table 2. Computational time of FD, BS, AGS, and AMGS methods.

| m  | n  | r  | FD  | BS  | AGS  | AMGS |
|----|----|----|-----|-----|------|------|
| 10 | 2  | 2  | 30us| 10us| 30us | 10us |
| 100| 10 | 10 | 209ms| 5.5us| 160us| 110us|
| 1000| 100| 100| 9.5h | 6.4s | 175ms| 208us|
| 1000| 100| 100| .92h | 359us| 9.2ms| 9.3ms|
| 1000| 10 | 100| .5h  | 3.9s | 154ms| 205ms|

Table 3. The dependence of the total number of iterations in the Broyden method on the method of gradient computation and problem size for the first series of experiments ($N_u$ is the total number of unknowns).

| m  | n  | r  | $N_u$ | FD  | BS  | AGS  | AMGS |
|----|----|----|-------|-----|-----|------|------|
| 2  | 2  | 2  | 8     | 4   | 4   | 4    | 4    |
| 5  | 5  | 5  | 125   | 244 | 238 | 521  | 238  |
| 10 | 10 | 10 | 1000  | 3061| 1581| >5000| 1581 |
| 20 | 20 | 20 | 8000  | >5000| 2464| >5000| 2464 |

References

[1] Harshman R., Ladefoged P. and Goldstein L., Factor analysis of tongue shapes, *J. Acoust. Soc. Am.*, 1977, 62:693.
[2] Jaravine V, Ibraghimov I, Orekhov V. Removal of a time barrier for high-resolution multidimensional NMR spectroscopy. *Nature*, 2006, 3:605–607.
[3] Ibraghimov I., A new approach to solution of SVD-like approximation problem. *ENU-MATH 99*, 2000, 548–555.
[4] Ibragimova, Ibragimov. The ELEGANT NMR Spectrometer. [arXiv:1706.00237], 2017.
[5] Ibragimov I. Application of the three-way decomposition for matrix compression. *Numer. Lin. Alg. Appl.*, 2002, 9:551–565.
[6] Baur W., Strassen V., The complexity of partial derivatives. *Theor. Comput. Sci.*, 1983, 22:317–330.
[7] Dennis J.E., Schnabel R.B., Numerical methods for unconstrained optimization and nonlinear equations. *Prentice-Hall*, 1983.