Universität des Saarlandes

Fachrichtung 6.1 – Mathematik

Preprint

SVD-Like Decomposition With Constraints

Ilghiz Ibraghimov

Preprint No. 26
Saarbrücken 2001

Universität des Saarlandes

Fachrichtung 6.1 – Mathematik

SVD-Like Decomposition With Constraints

Ilghiz Ibraghimov

Saarland University
Department of Mathematics
Postfach 15 11 50
D–66041 Saarbrücken
Germany
E-Mail: ilgis@num.uni-sb.de

submitted: January 31, 2001

Preprint No. 26

Saarbrücken 2001

Edited by
FR 6.1 – Mathematik
Im Stadtwald
D–66041 Saarbrücken
Germany

Fax:	+ 49 681 302 4443
e-mail:	preprint@math.uni-sb.de
WWW:	http://www.math.uni-sb.de/

Abstract

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

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

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

1 Introduction

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

$$\min_{C,\sigma_1,\dots,\sigma_k} ||A - B(\bar{\sigma})C^*||_F^2, \tag{1}$$

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

This problem occurs in statistics [1], nuclear magnetic resonance [2] and three-way decomposition [3]. Usually, the number of parameters σ is small ($3-10$ in [2]) since m and n can be sufficiently large ($100-1000$ in [3]), then it is impossible to make gradient minimization with thousands variables.

If we freeze B then this functional is linear in C, and $C = A^*B(B^*B)^{-1}$, then the problem (1) turns to the new nonlinear problem with only k unknowns:

$$\min_{\sigma_1,\dots,\sigma_k} ||A - B(B^*B)^{-1}B^*A||_F =$$

$$\min_{\sigma_1,\dots,\sigma_k} \sqrt{||A||_F^2 - ||A^*Q(B)||_F^2}, \tag{2}$$

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

The main difficulty in applying minimization methods for (2) is a computation of gradient of functional by all σ. Finite difference method needs k or $2k$ computations of this functional for one evaluation of gradient and cannot be considered an accurate. There is a good alternative for it — Baur-Strassen (BS) method [5], which allows to compute a gradient of function for 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 a memory requirement: it needs $\mathcal{O}(n)$ words in memory, which is too much in the most applications.
We suggest a new approach to compute the gradient of function containing Modified Gramm–Schmidt (MGS) orthogonalization [6] with low memory requirements based on BS method.

2 Algorithm

To compute (2) we should make the following steps:

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

2) compute orthonormal subspace Q in B;

3) compute (2).

In this article we discuss how to compute a gradient $\hat{g} \in \mathbb{C}^{mr}$ of (2) by all entries of B. Further we will use both $G \in \mathbb{C}^{m \times r}$ and $\hat{g}$ for the same data. Let the dependence of B by $\sigma_1, \ldots, \sigma_k$ be so simple that one can compute the gradient of (2) 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 case that MGS algorithm is used for the step 2. BS algorithm can compute the gradient with the same order of arithmetical complexity but with $4mr(r+n)$ additional words in memory.
Let us consider a computation of (2) from B. Let $B = [b_1, \ldots, b_k]$ be an initial matrix and $Q = [q_1, \ldots, q_k]$ the orthonormal subspace, which we are going to compute. Then

$q_1 = \dfrac{b_1}{||b_1||_2}$
do $i = 2,\ r$
    $u = b_i,$
    do $j = 1,\ i-1$
        $u = u - q_j q_j^* u$
    enddo

$$q_i = \frac{u}{||u||_2}$$

enddo

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

Let's construct a gradient of f by B. We call $\mathbf{d}y_i \in \mathbb{C}^m$ the vector of derivatives — each k-th element of this vector contains the derivative of k-th element of vector y_i. Then there are the following formulas for the gradient:

$$\mathbf{d}q_1 = \frac{1}{||b_1||_2}(I - q_1 q_1^*)\mathbf{d}b_1$$

do $i = 2,\ r$

$u = b_i$

do $j = 1,\ i-1$

$$\mathbf{d}u_{new} = (I - q_j q_j^*)\mathbf{d}u_{old} - (q_j^* u_{old} I + u_{old} q_j^*)\mathbf{d}q_j$$

enddo

$$\mathbf{d}q_i = \frac{1}{||u||_2}(I - q_i q_i^*)\mathbf{d}u$$

enddo

$$\mathbf{d}f = -\frac{1}{f}\sum_{i=1}^{r} q_i^* A A^* \mathbf{d}q_i$$

We can write all these equations in matrix notations:

$$\begin{pmatrix} I_{mr\times mr} & 0 & 0 \\ F_{mr\times\frac{mr(r+1)}{2}} & L_{\frac{mr(r+1)}{2}\times\frac{mr(r+1)}{2}} & 0 \\ 0 & h^*_{\frac{mr(r+1)}{2}} & 1 \end{pmatrix} \begin{pmatrix} \frac{\delta}{\delta b_{ij}} \\ \hat{g}^* \end{pmatrix} = \begin{pmatrix} I_{mr\times mr} \\ 0 \end{pmatrix} \qquad \text{or}$$

$$\begin{pmatrix} I_{mr\times mr} & F^*_{mr\times\frac{mr(r+1)}{2}} & 0 \\ 0 & L^*_{\frac{mr(r+1)}{2}\times\frac{mr(r+1)}{2}} & h_{\frac{mr(r+1)}{2}} \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \hat{g} \\ * \\ \vdots \\ * \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \tag{3}$$

where $L_{\frac{mr(r+1)}{2}\times\frac{mr(r+1)}{2}}$ is a lower block triangular matrix with the 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 $*$ the elements that we are not interested in, $\hat{g}$ is vector of the gradient of original function by all b_{ij}. To compute $\hat{g}$ we should solve the linear system (3). If we create L and F matrices, then we need at least $mr(r+3)$ words to solve L.

	b_1	b_2	b_3	b_4	q_1	u_{12}	q_2	u_{13}	u_{23}	q_3	u_{14}	u_{24}	u_{34}	q_4
q_1	S_1				I									
u_{12}		W_2			V_2	I								
q_2						S_2	I							
u_{13}			W_3		V_3			I						
u_{23}							V_3	W_3	I					
q_3									S_3	I				
u_{14}				W_4	V_4						I			
u_{24}							V_4				W_4	I		
u_{34}										V_4		W_4	I	
q_4													S_4	I

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 it 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, it needs $r-1$ 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 to compute G:

$G = -\frac{1}{f}AA^*Q$

do $i = r,\ 1,\ -1$

  $t_1 = b_i$

  do $j = 1,\ i-1$

    $z_j = q_j^* t_j$

    $t_{j+1} = t_j - z_j q_j$

  enddo

  $$g_i = \frac{g_i - q_i q_i^* g_i}{||t_i||_2}$$

  do $j = i-1,\ 1,\ -1$

    $\alpha = q_j^* g_i$

    $g_j = g_j - z_j^* g_i - \alpha^* t_j$

    $g_i = g_i - \alpha q_j$

  enddo

enddo

Table 1. Memory requirements (in words) for FD, BS, AGS, 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	10	19.5k	33.5m	39.3k	58.6k
1000	10	100	195k	337m	410k	586k

Here we use $Z = (z_1, \ldots, z_r) \in \mathbb{C}^r$ array with only r elements for better performance.
The total arithmetical complexity of computation of gradient is $4mr(2r+n)$ operations. If we compare it with MGS ($2mr(r+n)$) it is less than 4 times bigger.
We obtain similar results for Gramm-Schmidt (not MGS) orthogonalization: it needs $3mr + \dfrac{r(r+1)}{2}$ words in memory and works with $2mr(3r+2n)$ operations, but because of stability we do not recommend to use it.

3 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 and compute the derivatives for the different size of the problems by our new methods based on Gramm-Schmidt (AGS) and Modified Gramm-Schmidt (AMGS) algorithms and compare it with finite difference (FD) and Baur-Strassen (BS) methods (Tables 1, 2).
Further, we show how those algorithms work. We make the set of experiments and check the number of iterations for the convergence of Broyden method [7]. In this set of experiments the matrix B is real matrix with $b_i = p_i \otimes q_i \in \mathbb{R}^{n^2}$, $i = 1, \ldots, R$, where $\otimes$ is Kronecker product of vectors, $p_i, q_i \in \mathbb{R}^n$ 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, 4].
Hence, our new method (AMGS) is stable enough (like BS method) and faster than BS and FD methods up to thousand times and does not require much additional memory (only 4 times more than FD).

Table 2. Computational time of FD, BS, AGS, AMGS methods on `AMD Athlon 500`.

m	n	r	FD	BS	AGS	AMGS
10	2	2	0.0003s	0.0001s	0.0003s	0.0001s
100	10	10	2.09s	0.055s	0.0016s	0.0011s
1000	100	100	95h	1.07m	1.75s	2.08s
1000	100	10	9.2h	3.59s	0.092s	0.093s
1000	10	100	50h	38.8s	1.54s	2.05s

Table 3. The dependence of the total number of iterations in the Broyden method on the method of computation of the gradient and the size of problem for the first series of experiments.

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

N_u is the total number of unknowns.

References

[1] Harshman R., Ladefoged P. and Goldstein L., Factor analysis of tongue shapes, *J. Acoust. Soc. Am.*, 62:693, (1977).

[2] Schulze D., Stilbs P., Analysis of multicomponent FT-PGSE experiments by multivariate statistical methods applied to the complete bandshapes. *J. Magn. Res. Ser. A.* 105:54–58 (1993).

[3] Ibraghimov I., A new approach to solution of SVD-like approximation problem. *ENUMATH 99 - Proceedings of the 3rd European Conference on Numerical Mathematics and Advanced Applications,* Jyvaskyla, Finland, July 26–30, 1999, ed. by P. Neittaanmki, T. Tiihonen and P. Tarvainen, World Scientific, Singapore, 548–555, (2000).

[4] Ibraghimov I., Three-way decomposition. *In press in J. Appl. Math. Lett.*

[5] Baur W., Strassen V., The complexity of partial derivatives. *Theor. Comput. Sci.* 22:317–330 (1983).

[6] Tyrtyshnikov E.E., *A brief introduction to numerical analysis.* Birkhäuser, 1997.

[7] Dennis J.E., Schnabel R.B., *Numerical methods for unconstrained optimization and nonlinear equations.* Prentice-Hall, 1983.