

基于稳健联合分块对角化的卷积盲分离

汤辉¹ 王殊¹

摘要 针对卷积盲分离问题, 提出一种新的矩阵联合分块对角化 (Joint block diagonalization, JBD) 算法. 现有的迭代非正交联合分块对角化算法都存在不收敛的情况, 本文利用分离矩阵的特殊结构确保其可逆性, 使得算法的迭代过程稳定. 在已知矩阵分块结构的条件下, 首先, 将卷积盲分离模型写成瞬时形式, 并说明其满足联合分块对角化结构; 然后, 提出联合分块对角化的代价函数, 依据代价函数的最小化等价于矩阵中每个分块的范数最小化, 将整个分离矩阵的迭代更新转化成每个分块的迭代更新; 最后, 利用最小化条件得到迭代算法. 实数和复数两种情况下的算法都进行了推导. 基本实验验证了新算法在不同条件下的性能; 仿真实验中对在时域和频域都重叠的信号的卷积混合进行盲分离, 实验结果验证了新算法具有更好的分离性能和更稳定的分离能力.

关键词 卷积盲分离, 联合分块对角化, 稳健迭代, 矩阵范数

引用格式 汤辉, 王殊. 基于稳健联合分块对角化的卷积盲分离. 自动化学报, 2013, 39(9): 1502–1510

DOI 10.3724/SP.J.1004.2013.01502

Robust Joint Block Diagonalization Based Convolutional Blind Source Separation

TANG Hui¹ WANG Shu¹

Abstract A new joint block diagonalization (JBD) algorithm is proposed to solve the convolutional blind source separation problem. The existing iterative algorithms may not converge to the correct solution. We use a special structured separation matrix which is always invertible to avoid the divergence of the algorithm. First, the convolutional mixture is rewritten as an instantaneous one which satisfies the joint block diagonalization model. Second, the cost function is built with the priori information of block structure. Then the whole matrix iteration is transformed into update of the every block sub-matrix in the sense that the minimization of the cost function is equivalent to the minimization of Frobenius norm of each block. The iterative algorithm is deduced both in the situations of real and complex models. Sanity check experiment has verified the good performance of the new algorithm in different conditions. In the application of blind separation of signals which are both overlapping in the time and frequency domains, simulation results demonstrate the better performance and more robust ability of the proposed algorithm, as compared to others.

Key words Convolutional blind source separation, joint block diagonalization (JBD), robust iteration, matrix Frobenius norm

Citation Tang Hui, Wang Shu. Robust joint block diagonalization based convolutional blind source separation. *Acta Automatica Sinica*, 2013, 39(9): 1502–1510

卷积盲信号分离在语音信号处理、生物信号检测和通信频谱监察等领域中有着广泛的应用^[1]. 如何从未知类型的卷积混合信号中提取或分离出有用的信号是卷积盲分离的主要任务. Castella 等利用调制信号的常模特性, 在时频域构建代价函数获得源信号的盲解卷积^[2]. He 等利用稀疏表示理论在频域实现了卷积混合的盲分离^[3]. 作为一种处理卷积盲分离的有效方法, 联合分块对角化 (Joint block diagonalization, JBD) 算法在近年来引起了许多学者的关注和研究. 根据分离矩阵是否是酉 (正交) 矩阵, JBD 算法可以分为酉 (正交) JBD 算法和非

酉 (正交) JBD 算法. Abed-Meraim 等首先利用 Jacobi 旋转推导出了一种正交 JBD 算法, 但这种算法只能处理分块数目相等的问题^[4]; Maehara 等利用矩阵代数的概念, 通过对角化一个简单代数矩阵的方法成功地解决了 JBD 问题^[5]. 这两种算法只能处理正交 JBD 问题, 在卷积盲分离应用中, 这就需要利用白化方法将非正交混合矩阵转化成正交混合矩阵, 但是这会带来分离性能上的损失, 并且这种损失是后续处理所无法弥补的^[6]. 因而非正交 JBD 算法应运而生, Ghennioui 等提出了基于随机梯度和相对梯度的两组迭代算法, 并且利用步长最优化得到了快速算法^[7]; Xu 等根据矩阵的分块 Toeplitz 结构导出了一种三·二次迭代算法, 在卷积盲分离中得到了成功的应用^[8]. 而 Nion 利用张量分解的思想也提出了一种迭代的最优步长算法^[9]. Tichavský

收稿日期 2012-04-25 录用日期 2012-06-29
Manuscript received April 25, 2012; accepted June 29, 2012
本文责任编辑 刘成林
Recommended by Associate Editor LIU Cheng-Lin

1. 华中科技大学电信系 武汉 430074
1. Department of Electronic and Information, Huazhong University of Science and Technology, Wuhan 430074

等^[10] 和 Lahat 等^[11] 研究了 JBD 方法中的一些复杂度和适用场景问题. 但这些非正交算法都存在迭代不收敛的情况, 尤其在初始化条件不佳的情况下, 算法发散的可能性很大. 因此, 本文提出了一种新的稳健的非正交 JBD 算法, 通过强制分离矩阵满足某种特殊结构保证其可逆性, 进而使得算法收敛性大为提高.

本文分为 4 个部分, 首先, 将卷积盲分离问题写成瞬时形式并说明其满足联合分块对角化模型; 然后, 构造代价函数, 并提出可逆分离矩阵以及稳健迭代算法; 最后, 是仿真实验和结论总结.

1 卷积盲分离的分块对角化模型

卷积盲分离的目的是从未知卷积混合的信号中将源信号恢复出来, 假定有 m 个混合信号和 n 个观测信号, 典型的卷积盲分离模型可以写为^[2]

$$x_i(t) = \sum_{j=1}^n \sum_{l=1}^{L_j} H_{ij}(l) s_j(t-l) + n_i(t), \quad i = 1, \dots, m \quad (1)$$

其中, $x_i(t)$ 表示第 i 个混合信号在 t 时刻的瞬时值, 类似地可以定义 $s_j(t)$, $n_i(t)$ 表示附加的噪声, 一般认为是高斯的. $H_{ij}(\cdot)$ 表示第 i 个混合信号和第 j 个源信号之间的卷积混合过程, 通常是一个线性有限冲激响应滤波器. 为了利用分块对角化方法处理卷积盲分离模型, 首先, 需要将卷积模型转化成线性模型, 转化方法可以分为多时延和多传感器两种. 当每个滤波器长度相同时, 两种方法都可以适用; 而当滤波器长度不同或者其系数具有稀疏性时, 多时延法将不再适用, 只能用多传感器法. 下面分别介绍两种方法.

1.1 卷积盲分离线性化的多时延法

当 $H_{ij}(\cdot)$ 的长度都相等, 即 $L_j = L, j = 1, \dots, n$ 时, 并且 l 是已知的, 那么选取某个 L' 满足 $mL' \geq n(L + L')$, 令 $P = mL', Q = n(L + L') = nQ_0$, 定义如下的矢量:

$$\begin{aligned} \mathbf{s}_i(t) &= [s_i(t), s_i(t-1), \dots, s_i(t-Q_0+1)]^T, \\ & \quad i = 1, \dots, n \\ \mathbf{x}_i(t) &= [x_i(t), x_i(t-1), \dots, x_i(t-L'+1)]^T, \\ & \quad i = 1, \dots, m \\ \mathbf{n}_i(t) &= [n_i(t), n_i(t-1), \dots, n_i(t-L'+1)]^T, \\ & \quad i = 1, \dots, m \end{aligned}$$

$$\begin{aligned} \mathbf{s}(t) &= [\mathbf{s}_1^T(t), \mathbf{s}_2^T(t), \dots, \mathbf{s}_n^T(t)]^T \\ \mathbf{x}(t) &= [\mathbf{x}_1^T(t), \mathbf{x}_2^T(t), \dots, \mathbf{x}_m^T(t)]^T \end{aligned}$$

$$\mathbf{n}(t) = [\mathbf{n}_1^T(t), \mathbf{n}_2^T(t), \dots, \mathbf{n}_m^T(t)]^T$$

式 (1) 可以重新写成:

$$\mathbf{x}(t) = A\mathbf{s}(t) + \mathbf{n}(t) \quad (2)$$

其中, 矩阵 A 是一个分块矩阵, 定义为 $A = (A_{ij}), i = 1, \dots, m, j = 1, \dots, n$, 每个分块 A_{ij} 都是 $L' \times Q_0$ 的 Toeplitz 矩阵:

$$A_{ij} = \begin{bmatrix} H_{ij}(0) & \dots & H_{ij}(L) & 0 & \dots & 0 \\ 0 & H_{ij}(0) & \dots & H_{ij}(L) & 0 & \dots \\ 0 & 0 & \ddots & \vdots & \ddots & 0 \\ 0 & \dots & 0 & H_{ij}(0) & \dots & H_{ij}(L) \end{bmatrix} \quad (3)$$

这样就将卷积混合模型 (1) 变成了线性混合模型 (2), 值得注意的是, 这种方法只能在每个滤波器长度相同并且已知的条件下才成立. 并且当滤波器的系数具有稀疏性时, 由式 (3) 看出构造矩阵 A 的空间成本非常巨大, 这时下述的多传感器法将具有更好的性能.

1.2 卷积盲分离线性化的多传感器法

假定接收机上共有 P 个传感器, 并且令 $Q = \sum_{j=1}^n L_j$ 表示所有路径数目之和, 而且满足 $P \geq Q$. 定义 $\mathbf{x}(t) = [x_1(t), x_2(t), \dots, x_P(t)]^T$, $\mathbf{n}(t) = [n_1(t), n_2(t), \dots, n_P(t)]^T$, $\mathbf{s}(t) = [s_1(t), \dots, s_1(t-L_1), \dots, s_n(t), \dots, s_n(t-L_n)]^T$ 分别表示传感器矢量、噪声矢量和源信号等效矢量. 将式 (1) 写成矩阵-矢量形式有:

$$\mathbf{x}(t) = A\mathbf{s}(t) + \mathbf{n}(t) \quad (4)$$

其中, $A = [\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_P]^T$ 和 $\mathbf{a}_p = [h_{p1}, \dots, h_{pL_1}, \dots, h_{pQ}]^T$ 分别表示等效传感器矩阵及其行向量. 这个方法不需要每个滤波器长度相同, 但是付出了更多传感器的代价. 多时延法所需的传感器最小个数为 $n(L + L')/L'$, 而多传感器法所需的最小个数为 $\sum_{j=1}^n L_j$, 特别地, 当每个滤波器长度都等于 L 时, 其值为 nL . 在传感器数目允许的条件下, 用多传感器法会获得更为准确的结果.

对式 (2) 或式 (4) 计算等式两边的自相关函数, 容易得到:

$$R_{\mathbf{x}}(\tau) = AR_{\mathbf{s}}(\tau)A^T + R_{\mathbf{n}}(\tau) \quad (5)$$

由于源信号矢量 $\mathbf{s}(t)$ 中不同信号是相互独立的, 而属于同一个源信号的不同时延的分量是相关的, 那么 $R_{\mathbf{x}}(\tau)$ 是分块对角矩阵. 当忽略传感器

噪声, 即 $R_n(\tau) = \mathbf{0}$ 时, 通过求解矩阵 W , 使得 $WR_x(\tau)W^T = R_s(\tau)$, 就可以分离得到信号:

$$\hat{s}(t) = Wx(t) \quad (6)$$

为了获得较好的分离效果, 一般需要选取多个时延 τ_1, \dots, τ_K , 进而有多个需要同时分块对角化的自相关矩阵 $R_x(\tau_1), \dots, R_x(\tau_K)$, 考虑到噪声的影响和自相关矩阵估计所引入的误差, 精确的 JBD 是不可能实现的, 但是依然可以用联合分块对角化算法近似估计分离矩阵和源信号.

2 JBD 代价函数和可逆分离矩阵

定义 $X = X^1, X^2, \dots, X^K, X^k \in \mathbf{R}^{M \times M}$ 为可以同时分块对角化的矩阵集合, 即 X^k 满足:

$$X^k = AD^kA^T = A \begin{bmatrix} D_1^k & \cdots & \cdots & 0 \\ \vdots & D_2^k & \vdots & 0 \\ 0 & \cdots & \cdots & D_R^k \end{bmatrix} A^T$$

其中, $D^k \in \mathbf{R}^{N \times N}$ 是分块对角矩阵, 而 R 是分块的个数, 矩阵 $A \in \mathbf{R}^{M \times N}$ 称为分块成型矩阵. 考虑正定的情况, 即 $M = N$. 典型的 JBD 代价函数可以分为正交型和非正交型两种, 下面考虑非正交型代价函数, 如下式:

$$\min_B \phi_{\text{off}} = \min_B \sum_{k=1}^K \|\text{off}(BX^k B^T)\|_F^2 \quad (7)$$

其中, $B \in \mathbf{R}^{M \times M}$ 表示要求解的可逆块对角化矩阵, $\|\cdot\|_F$ 为矩阵的 Frobenius 范数. 在已知矩阵对角分块结构的条件下, $\text{off}(A)$ 表示矩阵 A 的不属于对角块的部分, 如下所示:

$$\text{off}(A) = \begin{bmatrix} 0_{11} & A_{12} & \cdots & A_{1r} \\ A_{21} & 0_{22} & \cdots & \cdots \\ \vdots & A_{mn} & 0_{nn} & \vdots \\ A_{r1} & \cdots & \cdots & 0_{rr} \end{bmatrix}$$

根据式 (7) 的代价函数可以推出关于 B 的迭代更新算法, 但是这种算法往往存在着不收敛的情况^[7], 即矩阵 B 在迭代过程中会变得不可逆, 为了保证 B 的可逆性, 提出具有如下结构的块对角化矩阵:

$$B = \prod_{p=1}^P (I + W_{(p)})B_0 \quad (8)$$

其中, B_0 为初始化矩阵, I 表示与 B 相同维数的单位矩阵, $W_{(p)}$ 是第 p 步迭代的更新矩阵, 而且与 $\text{off}(\cdot)$ 具有相同的结构, 即仅在不属于块对角的部分

有值, 且 B 满足对角占优性, 即主对角线上的元素远大于其余位置上的元素, 根据 Levi-Desplamques 定理^[12], $M \times M$ 对角占优矩阵一定是可逆的. 对角占优性可以在初始化的时候得到保证, 但是在每步迭代过程中可能会丧失. 为了确保 B 的可逆性, 在第 p 步迭代后, 计算 $W_{(p)}$ 的无穷范数, 若其大于预设的某个门限值 θ , 则进行归一化操作:

$$W_{(p)} = \frac{\theta}{\|W_{(p)}\|_\infty} W_{(p)} \quad (9)$$

事实上, 式 (8) 的乘法迭代与普通的加法迭代算法是一致的. 因为由式 (8) 可以得到第 $p+1$ 步分离矩阵和第 p 步分离矩阵有如下关系:

$$B_{(p+1)} = (I + W_{(p)})B_{(p)} = B_{(p)} + W_{(p)}B_{(p)} \quad (10)$$

在文献 [7] 的加法迭代算法中, 有如下迭代公式成立:

$$B_{(p+1)} = B_{(p)} - \Delta_{(p)}B_{(p)} \quad (11)$$

比较式 (10) 和式 (11), 发现 W_p 和 $\Delta_{(p)}$ 都是与 $B_{(p)}$ 存在非线性关系的迭代矩阵, 只是具体迭代形式不同, 那么式 (10) 和式 (11) 是相似的, 式 (8) 的乘法迭代与普通的加法迭代也是一致的.

在第 $p+1$ 步迭代后, 第 k 个矩阵 $X_{(p+1)}^k$ 的更新可以写成:

$$X_{(p+1)}^k = (I + W_{(p)})X_{(p)}^k(I + W_{(p)})^T \quad (12)$$

在矩阵分块结构已知的条件下, 令 $D_{(p)}^k$ 和 $E_{(p)}^k$ 分别表示矩阵 $X_{(p)}^k$ 的块对角部分和非块对角部分, 式 (12) 可以写成:

$$\begin{aligned} X_{(p+1)}^k &= (I + W_{(p)})(D_{(p)}^k + E_{(p)}^k)(I + W_{(p)}^T) = \\ &D_{(p)}^k + W_{(p)}D_{(p)}^k + E_{(p)}^k + W_{(p)}E_{(p)}^k + \\ &D_{(p)}^k W_{(p)}^T + W_{(p)}D_{(p)}^k W_{(p)}^T + E_{(p)}^k W_{(p)}^T + \\ &W_{(p)}E_{(p)}^k W_{(p)}^T \approx D_{(p)}^k + W_{(p)}D_{(p)}^k + \\ &D_{(p)}^k W_{(p)}^T + E_{(p)}^k \end{aligned} \quad (13)$$

上式的第三个等式是因为 $W_{(p)}$ 和 $E_{(p)}^k$ 在迭代中都是较小的值, 它们对应的乘积和高于一次的项都可以忽略.

利用式 (13), 由于 $D_{(p)}^k$ 是分块对角矩阵, 并且 $W_{(p)}D_{(p)}^k$ 和 $D_{(p)}^k W_{(p)}^T$ 都是只在非分块对角部分上有值, 忽略迭代的下标 p , 则代价函数 (7) 可以写成:

$$\phi_{\text{off}} = \min_W \sum_{k=1}^K \|WD^k + D^k W^T + E^k\|_F^2 \quad (14)$$

下一部分将考虑实数和复数两种情况, 分别提出相应的算法, 两种算法在本质上是类似的, 都是将

矩阵的范数最小化问题转化为每个分块矩阵最小化的问题, 进而得到矩阵 W 中每个分块的具体迭代算法, 再遍历所有不同的分块, 就得到整个矩阵的更新. 复数条件下的算法比实数情况下的算法要复杂一些, 需要计算的矩阵数目也更多.

3 迭代算法

为了更好地说明迭代算法, 在矩阵 X^k 的分块结构已知的条件下, 将矩阵 W 表示成分块形式. 当 X^k 中每个分块的大小相等时, W 的每个分块的大小也是相等的, 而当 X^k 中分块大小不同时, W 的分块大小也是不一样的, 不失一般性, 假设 X^k 中的分块满足 $[2, 2, 3]$ 的结构, 相应的 W 的分块结构如下所示:

$$W = \begin{bmatrix} 0_{(11)2 \times 2} & W_{(12)2 \times 2} & W_{(13)2 \times 3} \\ W_{(21)2 \times 2} & 0_{(22)2 \times 2} & W_{(23)2 \times 3} \\ W_{(31)3 \times 2} & W_{(32)3 \times 2} & 0_{(33)3 \times 3} \end{bmatrix}$$

其中, $W_{(ij)}$ 代表每个小分块, 其余分块情况下的矩阵 W 的结构可以类似地给出. 而矩阵 E^k 和 W 具有一致的结构, 可以类似地构造. 下面首先推导实数情况下的更新算法, 然后, 将算法推广至复数的情况.

3.1 BDIAG (Block diagonalization) 算法的推导

首先, 令 $U^k = WD^k + D^k W^T + E^k$, 则有:

$$\min \phi_{\text{off}} = \min \sum_{k=1}^K \|U^k\|_F^2 \quad (15)$$

进一步注意到, U^k 与 E^k 具有相同的结构, 也可以将其写成每个分块的形式, 并且有下式成立:

$$\|U^k\|_F^2 = \sum_{i,j,i \neq j} \|U_{ij}^k\|_F^2 \quad (16)$$

即矩阵的 Frobenius 范数平方等于各个分块的 Frobenius 范数平方之和. 那么当每个 $\|U_{ij}^k\|_F^2$ 最小时, 即 $\|U^k\|_F^2$ 也取最小值, 数学形式表示为

$$\phi_{\text{off}} = \sum_{k=1}^K \sum_{i,j,i \neq j}^R \|U_{ij}^k\|_F^2 = \sum_{i,j,i \neq j}^R \sum_{k=1}^K \|U_{ij}^k\|_F^2 \quad (17)$$

式 (17) 最后一个等式成立是因为需要分块对角化的矩阵个数 K 和矩阵分块数 R 是不相关的, 所以将累加算子进行了互换. 将 U_{ij}^k 和 U_{ji}^k 中的乘积项展开成块矩阵相乘的形式, 有:

$$U_{ij}^k = \sum_p W_{ip} D_{pj}^k + \sum_q D_{iq}^k W_{jq}^T + E_{ij}^k = W_{ij} D_{jj}^k + D_{ii}^k W_{ji}^T + E_{ij}^k \quad (18)$$

$$U_{ji}^k = \sum_p W_{jp} D_{pi}^k + \sum_q D_{jq}^k W_{iq}^T + E_{ji}^k = W_{ji} D_{ii}^k + D_{jj}^k W_{ij}^T + E_{ji}^k \quad (19)$$

其中, D_{ii}^k 和 D_{jj}^k 分别表示矩阵 D^k 的第 i 和 j 个对角块. 那么有:

$$\phi_{\text{off}} = \sum_{i=1:R-1, j=i:R} \sum_{k=1}^K (\|U_{ij}^k\|_F^2 + \|U_{ji}^k\|_F^2) \quad (20)$$

式 (20) 将矩阵范数表示成两个成对分块的范数相加再求和, 注意到共有 $R(R-1)/2$ 个这样的成对分块, 这样做是因为 U_{ij}^k 和 U_{ji}^k 都是只和 W_{ij} 与 W_{ji} 有关. 这样就可以将整个矩阵 W 转化为其中不同分块对 (W_{ij}, W_{ji}) 的更新.

为了使推导过程的表示简单, 做如下的替换:

$$W_1 = W_{ij}, \quad W_2 = W_{ji}$$

$$D_1^k = D_{ii}^k, \quad D_2^k = D_{jj}^k$$

$$E_1^k = E_{ij}^k, \quad E_2^k = E_{ji}^k$$

利用矩阵矢量化算子 $\text{vec}(\cdot)$ 及其相关性质^[13]:

$$\text{vec}(ABC) = (C^T \otimes A)\text{vec}(B)$$

$$\text{vec}(A^T) = K_A \text{vec}(A)$$

其中, K_A 是一个与 A 有关的排序矩阵, 它仅和矩阵 A 的结构有关, 而与其具体元素无关. 将 U_{ij}^k 和 U_{ji}^k 都表示成矢量的形式, 如下式:

$$\begin{aligned} \mathbf{u}_{ij}^k &= \text{vec}(U_{ij}^k) = \\ &= \text{vec}(W_1 D_2^k + D_1^k W_2^T + E_1^k) = \\ &= \text{vec}(W_1 D_2^k) + \text{vec}(D_1^k W_2^T) + \text{vec}(E_1^k) = \\ &= ((D_2^k)^T \otimes I_{r1})\text{vec}(W_1) + \\ &= (I_{r2} \otimes D_1^k)\text{vec}(W_2^T) + \text{vec}(E_1^k) = \\ &= A_1^k \mathbf{w}_1 + A_2^k K_2 \mathbf{w}_2 + \mathbf{e}_1^k = \\ &= [A_1^k, A_2^k K_2][\mathbf{w}_1, \mathbf{w}_2]^T + \mathbf{e}_1^k = \\ &= F_1^k \mathbf{w} + \mathbf{e}_1^k \end{aligned} \quad (21)$$

$$\begin{aligned} \mathbf{u}_{ji}^k &= \text{vec}(U_{ji}^k) = \\ &= ((D_1^k)^T \otimes I_{r2})\text{vec}(W_2) + \\ &= (I_{r1} \otimes D_2^k)\text{vec}(W_1^T) + \text{vec}(E_2^k) = \\ &= A_3^k \mathbf{w}_2 + A_4^k K_1 \mathbf{w}_1 + \mathbf{e}_2^k = \\ &= [A_4^k K_1, A_3^k][\mathbf{w}_1, \mathbf{w}_2]^T + \mathbf{e}_2^k = \\ &= F_2^k \mathbf{w} + \mathbf{e}_2^k \end{aligned} \quad (22)$$

其中, “ \otimes ” 表示 Kronecker 乘积, I_{r_1} 和 I_{r_2} 分别是和矩阵 D_1^k 和 D_2^k 维数相同的单位矩阵, K_1 和 K_2 则满足:

$$\text{vec}(W_1^T) = K_1 \text{vec}(W_1), \text{vec}(W_2^T) = K_2 \text{vec}(W_2)$$

而 $A_1^k, A_2^k, A_3^k, A_4^k, \mathbf{w}_1, \mathbf{w}_2, \mathbf{e}_1^k, \mathbf{e}_2^k$ 分别为对应的矩阵和矢量, $F_1^k = [A_1^k, A_2^k K_2]$, $F_2^k = [A_4^k K_1, A_3^k]$, $\mathbf{w} = [\mathbf{w}_1, \mathbf{w}_2]^T$, 这样替换的好处是可以直接计算 $\mathbf{w} = [\mathbf{w}_1, \mathbf{w}_2]^T$.

由矩阵范数和对应矢量化表示的特殊关系, 有:

$$\begin{aligned} \|U_{ij}^k\|_F^2 + \|U_{ji}^k\|_F^2 = & (F_1^k \mathbf{w} + \mathbf{e}_1^k)^T (F_1^k \mathbf{w} + \mathbf{e}_1^k) + \\ & (F_2^k \mathbf{w} + \mathbf{e}_2^k)^T (F_2^k \mathbf{w} + \mathbf{e}_2^k) \end{aligned} \quad (23)$$

考虑到共有 K 个矩阵, 将其表示成求和的形式. 最后, 为了求取范数的最小值, 计算式 (23) 对 \mathbf{w} 的导数, 并令其等于 0, 则有:

$$F \mathbf{w} = \mathbf{g} \quad (24)$$

其中, $F = \sum_{k=1}^K ((F_1^k)^T F_1^k + (F_2^k)^T F_2^k)$, $\mathbf{g} = -\sum_{k=1}^K ((F_1^k)^T \mathbf{e}_1^k + (F_2^k)^T \mathbf{e}_2^k)$, 求解出最优的 \mathbf{w}^{opt} :

$$\mathbf{w}^{\text{opt}} = F^{-1} \mathbf{g} \quad (25)$$

由于 $\mathbf{w} = [\mathbf{w}_1, \mathbf{w}_2]^T$, 可以利用矢量转矩阵算子 $\text{vec2mat}(\cdot)$, 将 \mathbf{w}_1 和 \mathbf{w}_2 转化成 W_{ij} 和 W_{ji} , 再遍历所有的成对分块进行同样的计算, 最后, 还要用式 (9) 保证矩阵 W 的小范数性, 就完成了了一步完整的迭代更新, 定义新算法为 BDIAG 算法.

3.2 复数情况下的算法

在复数情况下, 将矩阵表示成对应的实部和虚部之和的形式, 有:

$$W_{ij} = W_{ij}^{(\text{Re})} + jW_{ij}^{(\text{Im})}$$

$$E_{ij}^k = E_{ij}^{k(\text{Re})} + jE_{ij}^{k(\text{Im})}$$

$$D_{ii}^k = D_{ii}^{k(\text{Re})} + jD_{ii}^{k(\text{Im})}$$

令 $W_1 = W_{ij}, W_2 = W_{ji}, D_1^k = D_{ii}^k, D_2^k = D_{jj}^k, E_1^k = E_{ij}^k, E_2^k = E_{ji}^k$, 计算复数矩阵 U_{ij}^k, U_{ji}^k 的实部和虚部得到:

$$\begin{aligned} U_{ij}^{k(\text{Re})} = & W_1^{(\text{Re})} D_2^{k(\text{Re})} - W_1^{(\text{Im})} D_2^{k(\text{Im})} + \\ & D_1^{k(\text{Re})} (W_2^{(\text{Re})})^T + D_1^{k(\text{Im})} (W_2^{(\text{Im})})^T + E_1^{k(\text{Re})} \end{aligned}$$

$$\begin{aligned} U_{ij}^{k(\text{Im})} = & W_1^{(\text{Im})} D_2^{k(\text{Re})} + W_1^{(\text{Re})} D_2^{k(\text{Im})} - \\ & D_1^{k(\text{Re})} (W_2^{(\text{Im})})^T + D_1^{k(\text{Im})} (W_2^{(\text{Re})})^T + E_1^{k(\text{Im})} \end{aligned}$$

$$\begin{aligned} U_{ji}^{k(\text{Re})} = & W_2^{(\text{Re})} D_1^{k(\text{Re})} - W_2^{(\text{Im})} D_1^{k(\text{Im})} + \\ & D_2^{k(\text{Re})} (W_1^{(\text{Re})})^T + D_2^{k(\text{Im})} (W_1^{(\text{Im})})^T + E_2^{k(\text{Re})} \end{aligned}$$

$$\begin{aligned} U_{ji}^{k(\text{Im})} = & W_2^{(\text{Re})} D_1^{k(\text{Im})} + W_2^{(\text{Im})} D_1^{k(\text{Re})} - \\ & D_2^{k(\text{Re})} (W_1^{(\text{Im})})^T + D_2^{k(\text{Im})} (W_1^{(\text{Re})})^T + E_2^{k(\text{Im})} \end{aligned}$$

利用 $\text{vec}(U_{ij}^k) = \text{vec}(U_{ij}^{k(\text{Re})}) + j\text{vec}(U_{ij}^{k(\text{Im})}) = \mathbf{u}_{ij}^{k(\text{Re})} + j\mathbf{u}_{ij}^{k(\text{Im})}$ 和 $\|U_{ij}^k\|_F^2 = (\mathbf{u}_{ij}^{k(\text{Re})})^T \mathbf{u}_{ij}^{k(\text{Re})} + (\mathbf{u}_{ij}^{k(\text{Im})})^T \mathbf{u}_{ij}^{k(\text{Im})}$, 分别计算 $\mathbf{u}_{ij}^{k(\text{Re})}, \mathbf{u}_{ij}^{k(\text{Im})}$ 可以得到:

$$\mathbf{u}_{ij}^{k(\text{Re})} = F_1^k \mathbf{w} + \mathbf{e}_1^{k(\text{Re})}$$

$$\mathbf{u}_{ij}^{k(\text{Im})} = F_2^k \mathbf{w} + \mathbf{e}_1^{k(\text{Im})}$$

式中的矩阵和变量满足:

$$F_1^k = [A_1^k, -A_3^k, A_2^k K_2^{(\text{Re})}, A_4^k K_2^{(\text{Im})}]$$

$$F_2^k = [A_3^k, A_1^k, A_4^k K_2^{(\text{Re})}, -A_2^k K_2^{(\text{Im})}]$$

$$\mathbf{w} = [\mathbf{w}_1^{(\text{Re})}, \mathbf{w}_1^{(\text{Im})}, \mathbf{w}_2^{(\text{Re})}, \mathbf{w}_2^{(\text{Im})}]$$

$$A_1^k = (D_2^{k(\text{Re})})^T \otimes I_{r_1}, A_2^k = I_{r_2} \otimes D_1^{k(\text{Re})}$$

$$A_3^k = (D_2^{k(\text{Im})})^T \otimes I_{r_1}, A_4^k = I_{r_2} \otimes D_1^{k(\text{Im})}$$

$$\mathbf{w}_1^{(\text{Re})} = \text{vec}(W_1^{(\text{Re})}), \mathbf{w}_1^{(\text{Im})} = \text{vec}(W_1^{(\text{Im})})$$

$$\mathbf{w}_2^{(\text{Re})} = \text{vec}(W_2^{(\text{Re})}), \mathbf{w}_2^{(\text{Im})} = \text{vec}(W_2^{(\text{Im})})$$

$$\mathbf{e}_1^{k(\text{Re})} = \text{vec}(E_1^{k(\text{Re})}), \mathbf{e}_1^{k(\text{Im})} = \text{vec}(E_1^{k(\text{Im})})$$

类似地, 可以计算出对应于 $\mathbf{u}_{ji}^{k(\text{Re})}, \mathbf{u}_{ji}^{k(\text{Im})}$ 的矩阵和向量如下:

$$F_3^k = [A_5^k K_1^{(\text{Re})}, A_6^k K_1^{(\text{Im})}, A_7^k, -A_8^k]$$

$$F_4^k = [A_6^k K_1^{(\text{Im})}, A_5^k K_1^{(\text{Re})}, A_8^k, -A_7^k]$$

$$A_5^k = I_{r_1} \otimes D_2^{k(\text{Re})}, A_6^k = I_{r_1} \otimes D_2^{k(\text{Im})}$$

$$A_7^k = (D_1^{k(\text{Re})})^T \otimes I_{r_2}, A_8^k = (D_1^{k(\text{Im})})^T \otimes I_{r_2}$$

$$\mathbf{e}_2^{k(\text{Re})} = \text{vec}(E_2^{k(\text{Re})}), \mathbf{e}_2^{k(\text{Im})} = \text{vec}(E_2^{k(\text{Im})})$$

而 $\mathbf{u}_{ji}^{k(\text{Re})}, \mathbf{u}_{ji}^{k(\text{Im})}$ 可以表示为

$$\mathbf{u}_{ji}^{k(\text{Re})} = F_3^k \mathbf{w} + \mathbf{e}_2^{k(\text{Re})}$$

$$\mathbf{u}_{ji}^{k(\text{Im})} = F_4^k \mathbf{w} + \mathbf{e}_2^{k(\text{Im})}$$

计算相应的范数, 有:

$$\begin{aligned} \|U_{ij}^k\|_F^2 + \|U_{ji}^k\|_F^2 = & (F_1^k \mathbf{w} + \mathbf{e}_1^{k(\text{Re})})^T (F_1^k \mathbf{w} + \mathbf{e}_1^{k(\text{Re})}) + \\ & (F_2^k \mathbf{w} + \mathbf{e}_1^{k(\text{Im})})^T (F_2^k \mathbf{w} + \mathbf{e}_1^{k(\text{Im})}) + \\ & (F_3^k \mathbf{w} + \mathbf{e}_2^{k(\text{Re})})^T (F_3^k \mathbf{w} + \mathbf{e}_2^{k(\text{Re})}) + \\ & (F_4^k \mathbf{w} + \mathbf{e}_2^{k(\text{Im})})^T (F_4^k \mathbf{w} + \mathbf{e}_2^{k(\text{Im})}) \end{aligned} \quad (26)$$

类似于实数的情况, 累积 K 个矩阵, 计算式 (26) 对于 \mathbf{w} 的导数并使其等于 0, 容易得到:

$$\mathbf{w}^{\text{opt}} = F^{-1} \mathbf{g} \quad (27)$$

其中, $F = \sum_{k=1}^K \sum_{i=1}^4 (F_i^k)^T F_i^k$ 和 $\mathbf{g} = -\sum_{k=1}^K ((F_1^k)^T \mathbf{e}_1^{k(\text{Re})} + (F_2^k)^T \mathbf{e}_1^{k(\text{Im})} + (F_3^k)^T \mathbf{e}_2^{k(\text{Re})} + (F_4^k)^T \mathbf{e}_2^{k(\text{Im})})$.

最后, 利用向量矩阵化算子分别得到矩阵 W_{ij} 和 W_{ji} , 迭代计算不同的分块, 得到全矩阵 W 的更新, 并保证其具有小值性, 复数情况下的算法定义为 CVBIDIAG (Complex value block diagonalization) 算法.

3.3 算法复杂度分析

设分块对角矩阵 D^k 的第 i 和第 j 个分块的大小分别是 L_i 和 L_j , 以一次实数乘法运算作为衡量算法复杂度的单位, 则在一个迭代中 BDIAG 和 CVBIDIAG 算法的复杂度如表 1 所示.

为了容易与文献 [7] JBD-OG (Joint block diagonalization-optimal gradizent) 的算法复杂度进行比较, 假定矩阵个数 K 足够大, 并且矩阵中每个分块的大小相同, 都等于 L_0 , 那么有如下结论近似成立:

结论 1. 当分块矩阵中分块的数目大于每个分块的维数时, BDIAG (CVBIDIAG) 算法的复杂度要小于 JBD-OG 算法; 而当分块矩阵中分块的数目小于每个分块的维数时, BDIAG (CVBIDIAG) 算法的复杂度要大于 JBD-OG 算法; 当分块矩阵中分块的数目等于每个分块的维数时, 二者复杂度近似相同.

证明. 由于 K 足够大, 并且 $L_i = L_j = \dots = L_0$, 则 BDIAG 算法的复杂度为

$$T = (10KL_0^6 + 8KL_0^4) \frac{R(R-1)}{2} \approx 5KL_0^6 R^2 + 4KL_0^4 R^2 \approx O(5KN^2L_0^4) \quad (28)$$

类似地, 推出 CVBIDIAG 算法的复杂度约为 $O(18KN^2L_0^4)$, 由于 JBD-OG 算法在实数和复数条件下的复杂度^[7] 分别为 $O(9KN^4)$ 和 $O(27KN^4)$, 忽略系数的差异, 并且注意到 $N = L_0R$, 那么当

$L_0 > R$ 时, $KN^2L_0^4 > KN^4$; 而当 $L_0 < R$ 时, $KN^2L_0^4 < KN^4$. \square

由结论 1 可知, 当分块矩阵具有分块数目多而维数小的特征时, BDIAG (CVBIDIAG) 算法具有更小的计算复杂度, 特别的, 对于 $L_0 = 1$ 的联合 (零) 对角化^[14-15] 问题, 算法的计算优势更加明显.

3.4 算法收敛性说明

迭代算法收敛性的严格数学证明是很困难的, 下面给出一个定性的说明. 由于代价函数 (14) 存在下界 0, 并且式 (25) 和式 (27) 的最小二乘算法保证了每一个非对角分块上的矩阵的 Frobenius 范数的递减, 这等价于整个矩阵的非对角分块部分的 Frobenius 范数递减, 即 $\|\phi_{\text{off}(p+1)} - \phi_{\text{off}(p)}\|_F^2 < 0$, 所以迭代算法能够保证当迭代次数 $p \rightarrow \infty$ 时, 代价函数 $\phi_{\text{off}} \rightarrow 0$.

4 仿真实验

仿真实验分为基本测试和仿真测试两个部分, 基本测试分析了算法在实数和复数条件下的收敛性, 以及在不同信噪比和不同矩阵个数情况下的性能; 仿真测试则给出了算法在卷积盲信号分离时的应用效果.

4.1 基本测试

构造 K 个分块对角矩阵组成的集合, 其中每个矩阵 D_k 的元素都是服从零均值和单位方差的高斯分布变量, D_k 对角分块上的子矩阵大小分别为 $2 \times 2, 2 \times 2, 3 \times 3$, 则需要进行对角分块测试的矩阵集合定义为:

$$X_k = AD_kA^T + \sigma^2 N_k, \quad k = 1, \dots, K \quad (29)$$

其中, A 为随机混合矩阵, 其元素满足均匀分布; N_k 是噪声矩阵, σ^2 是度量噪声大小的常数, 算法工作的信噪比环境定义为 $\text{SNR} = 10 \lg(1/\sigma^2)$. 如果是复数情况, 则 D_k, A, N_k 是相应的复数矩阵. 式 (29) 中转置符号 “T” 变成共轭转置符号 “H”.

为了衡量算法的性能, 定义如下的性能指标:

表 1 BDIAG 和 CVBIDIAG 算法的复杂度
Table 1 Complexity of BDIAG and CVBIDIAG

子步骤	BDIAG 复杂度	CVBIDIAG 复杂度
$A_1^k \sim A_4^k (A_1^k \sim A_8^k)$	$4KL_i^2L_j^2$	$8KL_i^2L_j^2$
$F_1^k \sim F_2^k (F_1^k \sim F_4^k)$	$2KL_i^3L_j^3$	$4KL_i^3L_j^3$
F	$8KL_i^3L_j^3$	$32KL_i^3L_j^3$
\mathbf{g}	$4KL_i^2L_j^2$	$16KL_i^2L_j^2$
\mathbf{w}_{opt}	$8L_i^3L_j^3 + 4L_i^2L_j^2$	$64L_i^3L_j^3 + 16L_i^2L_j^2$
总计	$\sum_{i \neq j}^R (10K + 8)L_i^3L_j^3 + (8K + 4)L_i^2L_j^2$	$\sum_{i \neq j}^R (36K + 64)L_i^3L_j^3 + (24K + 16)L_i^2L_j^2$

$$I_{\text{conv}} = 10 \lg \left[\frac{1}{R(R-1)} \left(\sum_{i=1}^R \left(\sum_{j=1}^R \frac{\|C_{ij}\|_F^2}{\max_p \|C_{ip}\|_F^2} \right) + \sum_{j=1}^R \left(\sum_{i=1}^R \frac{\|C_{ij}\|_F^2}{\max_q \|C_{qi}\|_F^2} \right) \right) \right] \quad (30)$$

其中, $C = BA$ 为全局分离混合矩阵, 矩阵 C 共有 $R \times R$ 个子块矩阵. $\|C_{ij}\|_F^2$ 表示第 (i, j) 个子矩阵的 Frobenius 范数, I_{conv} 描述了矩阵 C 与广义块对角矩阵^[16] 的相似程度, 当 I_{conv} 越小时, 表示分离的效果越好. 图 1 和图 2 分别给出了在 20 次蒙特卡洛实验中, 算法在实数和复数情况下随着 SNR 和 K 变化时的性能. 可以看出, 随着信噪比的增加, 性能会有很大的改善, 而增加矩阵数目只能在很小的范围内改变算法的性能. 从图中还可以看出算法在实数和复数条件下的性能非常接近, 验证了 BDIAG 和 CVBDIAG 算法是一致的.

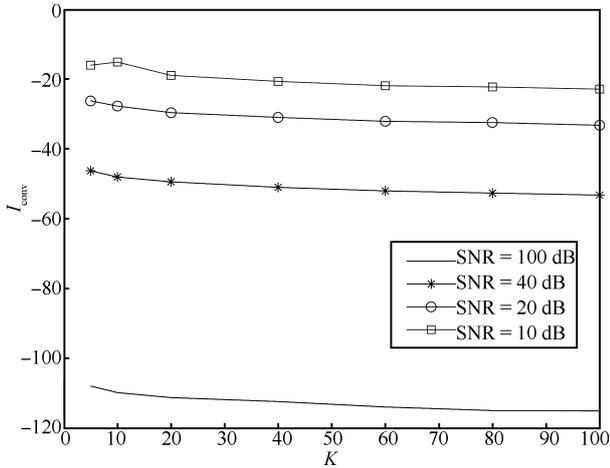


图 1 实数条件下的 BDIAG 算法随 K 和 SNR 的变化情况
Fig. 1 Performance of BDIAG with K and SNR in real case

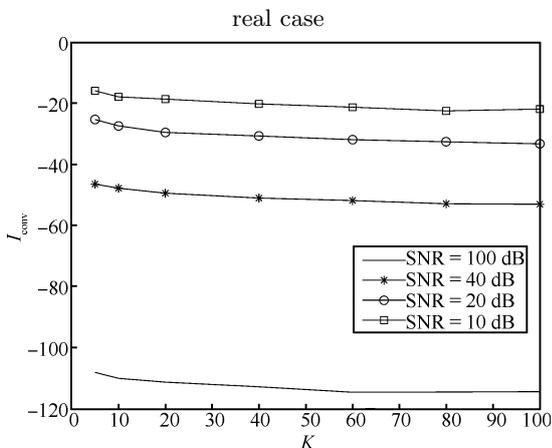


图 2 复数条件下的 CVBDIAG 算法随 K 和 SNR 的变化情况

Fig. 2 Performance of CVBDIAG with K and SNR in complex case

图 3 比较了本文算法和文献 [7] 的 JBD-OG 算法在 $K = 100$ 时, 不同信噪比条件下的分离性能, 试验结果是 50 次蒙特卡洛仿真的平均值. 由图 3 可以看出, 本文算法相比于 JBD-OG 算法有大约 3 dB 左右的性能提升.

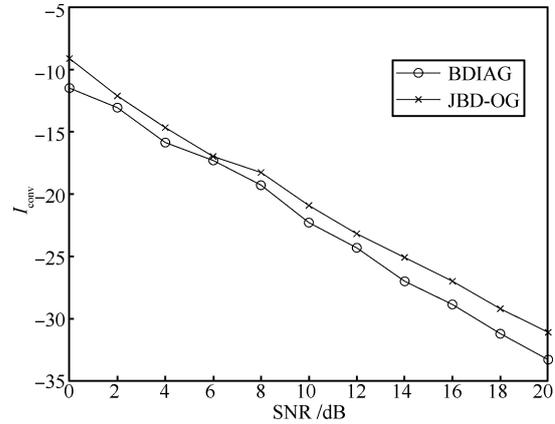


图 3 不同信噪比条件下本文算法和 JBD-OG 算法的性能比较

Fig. 3 Performances of BDIAG and JBD-OG in different SNR condition

4.2 仿真测试

给定 3 个在时域和频域都不可区分的源信号, 定义如下: 第 1 个为线性调频信号, 频率从 100 kHz 到 400 kHz; 第 2 个也为线性调频信号, 频率从 350 kHz 递减到 150 kHz; 第 3 个为频率 250 kHz 的正弦信号, 采样频率等于 1 MHz.

接收端卷积混合的信号个数 $m = 4$, 假定每个信号的滤波器长度相同 $L = 2$, 滤波器系数是服从正态分布的随机数. 利用多时延模型, 令 $L' = 6$, 取 1000 个采样点, 计算 $K = 20$ 个不同时延下的自相关矩阵, 然后, 分别用 BDIAG 算法和 JBD-OG 算法得到卷积分离矩阵, 进而获得源信号的估计. 理论上来说, 分离出的信号与源信号存在尺度和排序的不确定性, 而且分离信号是对应源信号的滤波形式^[2]. 为了衡量算法的性能, 计算源信号和估计信号的相关系数来判断是否达到了分离信号的效果. 在信噪比为 100 dB 的条件下源信号、混合信号、BDIAG 算法分离信号和 JBD-OG 算法分离信号分别如图 4 (a)~(d) 所示.

图 5 显示了在不同信噪比条件下的混合/分离信号与源信号的相关系数的分布图, 其中相关系数定义为每个混合/分离信号与三个源信号的相关系数的最大值. 仿真结果是 10 次蒙特卡洛仿真的平均值. 由图 5 可以看出, 在信噪比较大时, 两种算法都能取得很好的分离效果, 而当信噪比较低时, 本文算法具有更好的性能, 平均相关系数提升了约 20%.

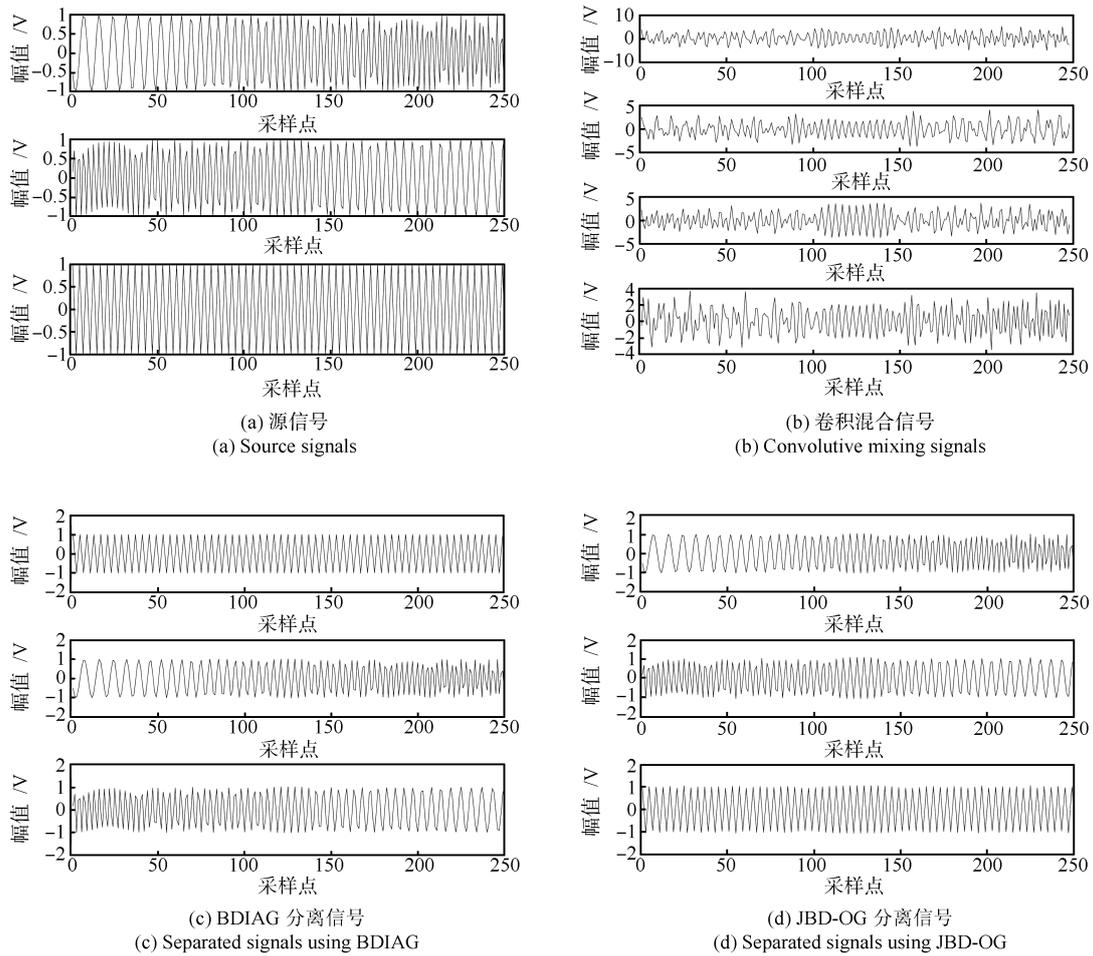


图 4 无噪声下一次试验中源信号、混合信号、BDIAG 分离信号和 JBD-OG 分离信号波形
 Fig. 4 Waves of sources, mixtures, BDIAG separated signals and JBD-OG separated signals in one experiment without noisy

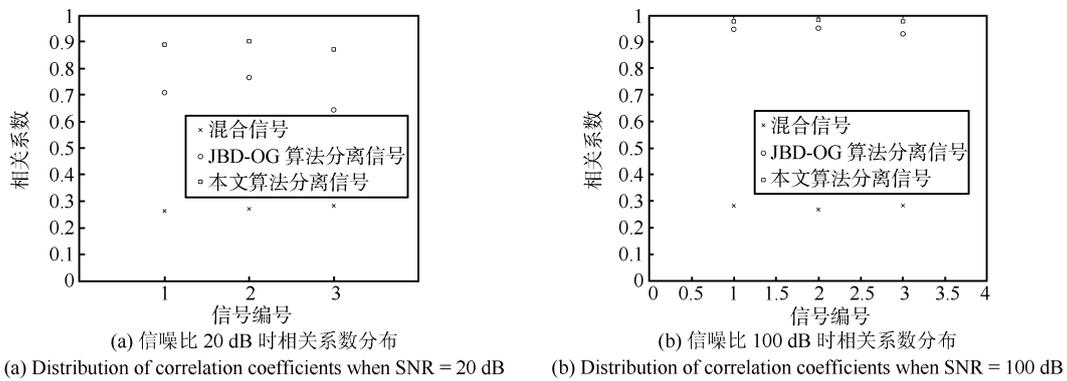


图 5 不同信噪比条件下混合/分离信号与源信号最大相关系数分布
 Fig. 5 Distribution of correlation coefficients mixed/separated signals with sources in different SNR conditions

5 结论

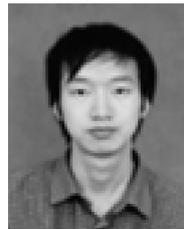
本文针对卷积盲分离问题, 提出了一种新的稳健联合分块对角化算法. 算法利用矩阵范数的特性, 将全矩阵的范数最小化首先转为各个分块子矩阵范

数的最小化, 再将矩阵范数的最小化转为对应矢量的范数最小化问题, 最后, 通过构造相应的矩阵得到一组最优解. 算法的迭代不需要考虑步长的影响, 而且能够保证分离矩阵的可逆性. 复数情况下的算法和实数情况本质上是一致的. 仿真实验验证和比较

了新算法在不同条件下的性能, 在线性调频信号和正弦信号卷积盲分离实验中, 算法也体现了更好的分离性能。

References

- 1 Yang Zhu-Qing, Li Yong, Hu De-Wen. Independent component analysis: a survey. *Acta Automation Sinica*, 2002, **28**(5): 762–772
(杨竹青, 李勇, 胡德文. 独立成分分析方法综述. 自动化学报, 2002, **28**(5): 762–772)
- 2 Castella M, Bianchi P, Chevereuil A, Prequet J C. A blind source separation framework for detecting CPM sources mixed by a convolutive MIMO filter. *Signal Processing*, 2006, **86**(8): 1950–1967
- 3 He Z S, Xie S L, Ding S X, Cichocki A. Convolutive blind source separation in the frequency domain based on sparse representation. *IEEE Transactions on Audio, Speech, and Language Processing*, 2007, **15**(5): 1551–1563
- 4 Abed-Meraim K, Belouchrani A. Algorithms for joint block diagonalization. In: Proceedings of the 12th European Signal Processing Conference. Vienna, Austria: IEEE, 2004. 209–212
- 5 Maehara T, Murota K. Algorithm for error-controlled simultaneous block-diagonalization of matrices. *SIAM Journal on Matrix Analysis and Applications*, 2011, **32**(2): 605–620
- 6 Ghennioui H, Fadaili E M, Thirion-Moreau N, Adib A, Moreau E. A nonunitary joint block diagonalization algorithm for blind separation of convolutive mixtures of sources. *IEEE Signal Processing Letters*, 2007, **14**(11): 860–863
- 7 Ghennioui H, Thirion-Moreau N, Moreau E, Aboutajdine D. Gradient-based joint block diagonalization algorithms: application to blind separation of fir convolutive mixtures. *Signal Processing*, 2010, **90**(6): 1836–1849
- 8 Xu X F, Feng D Z, Zheng W X, Zhang H. Convolutive blind source separation based on joint block Toeplitzization and block-inner diagonalization. *Signal Processing*, 2010, **90**(1): 119–133
- 9 Nion D. A tensor framework for nonunitary joint block diagonalization. *IEEE Transactions on Signal Processing*, 2011, **59**(10): 4585–4594
- 10 Tichavský P, Yeredor A, Koldovský Z. On computation of approximate joint block-diagonalization using ordinary AJD. In: Proceedings of the 10th International Conference on Latent Variable Analysis and Signal Separation. Tel Aviv, Israel: Springer, 2012. 163–171
- 11 Lahat D, Cardoso J F, Messer H. Joint block diagonalization algorithms for optimal separation of multidimensional components. In: Proceedings of the 10th International Conference on Latent Variable Analysis and Signal Separation. Tel Aviv, Israel: Springer, 2012. 155–162
- 12 Ziehe A, Lascov P, Nolte G, Müller K R. A fast algorithm for joint diagonalization with nonorthogonal transformation and its application to blind source separation. *Journal of Machine Learning Research*, 2004, **5**: 777–800
- 13 Zhang Xian-Da. Matrix Analysis and Applications. Beijing: Tsinghua University Press, 2004. 100–118
(张贤达. 矩阵分析与应用. 北京: 清华大学出版社, 2004. 100–118)
- 14 Theis F J. Uniqueness of complex and multidimensional independent component analysis. *Signal Processing*, 2004, **84**(5): 951–956



汤 辉 华中科技大学电信系博士研究生。2006 年获得华中科技大学电信系学士学位。主要研究方向为扩频信号检测和盲信号处理。本文通信作者。

E-mail: balatanghui@163.com

(TANG Hui Ph. D. candidate in the Department of Electronic and Information

and Technology. He received his bachelor degree from Huazhong University of Science and Technology in 2006. His research interest covers detection of spread spectrum signals and blind signals processing. Corresponding author of this paper.)



王 殊 华中科技大学教授。主要研究方向为智能信号处理, 无线传感器网络。

E-mail: shuwang@hust.edu.cn

(WANG Shu Professor at Huazhong University of Science and Technology. His research interest covers intelligent signal processing and wireless sensor network.)