J Sci Comput (2016) 67:821–835
DOI 10.1007/s10915-015-0103-7

REVIEW PAPER

Numerical Methods for the Genvar Criterion of Multiple-Sets Canonical Analysis

Xinguo Liu[1] · **Jianping You**[1]

Received: 9 February 2015 / Revised: 9 September 2015 / Accepted: 11 September 2015 /
Published online: 18 September 2015

Abstract The Genvar criterion, proposed by Steel, is one of the important generalizations of canonical correlation analysis. This paper deals with iterative methods for the Genvar criterion. An alternating variable method is analysed and an inexact version of it is proposed. Two starting point strategies are suggested to enhance these iterative algorithms. Numerical results show that, these starting point strategies not only can improve the rate of convergence, but also boost up the probability of finding a global solution.

Keywords Multiple-sets canonical correlation analysis · Genvar criterion · Alternating variable method · Starting point strategy

Mathematics Subject Classification 62H20 · 65C60

1 Introduction

Since Hotelling [1] proposed canonical correlation analysis (CCA) as the method for describing the relation between the scores of a set of observation units on two groups of variables, CCA has become an important method in numerous applications, including cluster analysis, principal component analysis, multidimensional scaling, pattern recognition, artificial intelligence, and bioinformatics. Several generalizations of canonical correlation analysis for multiple-sets of variables have been proposed by Steel [2], Kettenring [3], Horst [4], Van de Geer [5], Hanafi and Kiers [6], Via et al. [7,8], and other scholars. Among the five CCA criteria in [3], the sum of correlation (SUMCOR) or its covariance-based version called

This work was partially supported by NSF of China, Grant 11371333.

✉ Jianping You
you_jian_ping@163.com

Xinguo Liu
liuxinguo@ouc.edu.cn

[1] School of Mathematical Sciences, Ocean University of China, Qingdao 266003, China

Maxbet in [5] is the most popular one in practical applications and has been extensively studied. In this paper, we shall concern ourselves with the Genvar criterion originally proposed by Steel [2], which can be introduced briefly as follows.

Let $y_i = \begin{pmatrix} y_{i,1} \\ \vdots \\ y_{i,n_i} \end{pmatrix}$, $i = 1, \ldots, m$ be m-sets of random variables. Suppose the covariance matrix A of $y = \begin{pmatrix} y_1 \\ \vdots \\ y_m \end{pmatrix}$ exists and is given by

$$A = \left(\Sigma_{ij}\right)_{m\times m}, \quad \Sigma_{ii} \in \mathbb{R}^{n_i \times n_i}.$$

Here Σ_{ij} is the covariance between y_i and y_j, and we suppose all of the covariance matrix Σ_{ii} of y_i are invertible. Then the correlation matrix of $y = \begin{pmatrix} y_1 \\ \vdots \\ y_m \end{pmatrix}$ is given by

$$R = \left(R_{ij}\right)_{m\times m}, \quad R_{ij} = \Sigma_{ii}^{-\frac{1}{2}} \Sigma_{ij} \Sigma_{jj}^{-\frac{1}{2}}. \tag{1.1}$$

Here $\Sigma_{ii}^{\frac{1}{2}}$ denotes the square root of Σ_{ii}.

Let

$$\Sigma_m = \left\{ t = \begin{pmatrix} t_1 \\ \vdots \\ t_m \end{pmatrix} \in \mathbb{R}^n : t_i \in \mathbb{R}^{n_i}, \quad \| t_i \|_2 = 1 \right\}, \tag{1.2}$$

$$R(t) = \left(t_i^T R_{ij} t_j\right)_{m\times m}. \tag{1.3}$$

The Genvar criterion proposed by Steel [2] is to find a minimizer of the following optimization problem

$$\min \det(R(t)), \quad s.t.\ t \in \Sigma_m. \tag{1.4}$$

Because Σ_m is a bounded and closed set in $\mathbb{R}^n$, $\det(R(t))$ is a continuous function, we see that the solution of Genvar always exists. One important feature of the Genvar criterion is that it reduces to Hotelling's CCA when the number of sets is only two. Another feature of Genvar is that it is related to the multi-normal negative log-likelihood [2,9]. Its covariance-based version is closely related to the Box–Cox approach in regression [9]. In [10], some interesting statistical properties of Genvar are proved. In the recent paper [11], the authors point out that the multi-feature fusion method based on SUMCOR is difficult to reveal integral relation among multi-set variables, so they suggest applying Genvar to extract multi-set integral canonical correlation features. In the present paper, we mainly focus on the computational issue of Genvar. The Genvar criterion is a constrained nonlinear optimization problem and can only be solved by means of iteration. General-purpose iterative methods (see, e.g., [12]) can be employed. However, these methods stop at a stationary point, which serves only as a necessary condition for the global minima of Genvar. For statistical applications on the other hand, a global solution is quite desirable, which has substantial importance for the reliability of the statistical prediction. In fact, finding a global solution of the general Genvar is a very hard problem because a particular Genvar where $m = n$ is equivalent to the so-called Boolean

quadratic program, which is long-known to be computationally difficult and belongs to the class of NP-hard problems (see e.g., [13]). To tackle Genvar, Steel [2] using a compound matrix approach developed a system of nonlinear equations. These equations seem to be quite difficult to solve except in special cases. Kettenring [3] outlined an alternating variable method (AVM, see Algorithm 2.2 in Sect. 2), which is a descent-type iterative procedure. To our best knowledge, this is the state-of-art algorithm for Genvar. The AVM algorithm has several disadvantages. Numerical experiment indicates that AVM depends closely on the starting point, and unless it is specially selected, the AVM algorithm has a high probability to stop at a local but non-global minima of Genvar. In addition, for some cases, the AVM algorithm with random starting point converges very slowly. In the present paper, we will numerically demonstrate the performance of AVM and present some theoretical results. An inexact version of AVM is presented. Two starting point strategies are suggested to improve these iterative methods in both reducing the number of iterations and boosting up the probability of finding a global solution of (1.4).

This paper is organised as follows. Section 2 focuses on the optimality conditions for (1.4), and suggests the Gauss–Seidel method. We present upper and lower bounds of the optimal objective function value in Sect. 3. In Sect. 4, we suggest two starting point strategies. We present numerical test results in Sect. 5 and followed by some remarks in Sect. 6.

2 Optimality Conditions

Let $R_i(t)$ denote the matrix obtained by deleting the i-th row and column of $R(t)$. We note that $R_i(t)$ is independent of t_i.

Let

$$G_i = \begin{pmatrix} t_1^T R_{1,i} \\ \vdots \\ t_{i-1}^T R_{i-1,i} \\ t_{i+1}^T R_{i+1,i} \\ \vdots \\ t_m^T R_{m,i} \end{pmatrix}$$

Theorem 2.1 *Suppose $t \in \Sigma_m$ is a solution of (1.4). Then there exist real numbers $u_1, \ldots, u_m$ such that*

$$\begin{pmatrix} \det(R_1(t)) G_1^T R_1(t)^{-1} G_1 & & \\ & \ddots & \\ & & \det(R_m(t)) G_m^T R_m(t)^{-1} G_m \end{pmatrix} t = \begin{pmatrix} u_1 I_{n_1} & & \\ & \ddots & \\ & & u_m I_{n_m} \end{pmatrix} t \tag{2.1}$$

Proof Consider the Lagrangian function of (1.4)

$$L(t, u) = \det(R(t)) + \sum_{j=1}^{m} u_j \left(t_j^T t_j - 1 \right).$$

We see that

$$\det(R(t)) = \det(R_i(t)) \left(1 - (G_i t_i)^T R_i(t)^{-1} G_i t_i\right). \tag{2.2}$$

Therefore,

$$\frac{\partial \det(R(t))}{\partial t_i} = -2 \det(R_i(t)) G_i^T R_i(t)^{-1} G_i t_i.$$

Consequently,

$$\frac{\partial L(t,\ u)}{\partial t} = -2diag\left(\det(R_1(t)) G_1^T R_1(t)^{-1} G_1, \ldots, \det(R_m(t)) G_m^T R_m(t)^{-1} G_m\right) t + 2\Omega t.$$

Here $\Omega \equiv diag\left(u_1 I_{n_1}, \ldots, u_m I_{n_m}\right)$. It follows that (2.1) holds. □

We note that (2.1) can be rewritten as

$$\det(R_i(t)) G_i^T R_i(t)^{-1} G_i t_i = u_i t_i, \quad \| t_i \|_2 = 1, \quad i = 1, 2, \ldots, m, \tag{2.3}$$

and G_i and $R_i(t)$ are independent of t_i, but depend on $t_j (j \neq i)$. Hence, naturally, we suggest the following Gauss–Seidel method to solve (2.3).

Algorithm 2.1 (Gauss–Seidel type method, G–S method)

(1) Take $t^{(0)} = \begin{pmatrix} t_1^{(0)} \\ \vdots \\ t_m^{(0)} \end{pmatrix} \in \Sigma_m$.

(2) Suppose that the current approximate point is $t^{(k)} = \begin{pmatrix} t_1^{(k)} \\ \vdots \\ t_m^{(k)} \end{pmatrix} \in \Sigma_m$. Then compute $t^{(k+1)}$ as follows.

$$\begin{aligned} y_i^{(k)} &= G_i \left(z^{(i)}\right)^T R_i \left(z^{(i)}\right)^{-1} G_i \left(z^{(i)}\right) t_i^{(k)}, \\ u_i^{(k)} &= \| y_i^{(k)} \|_2, \\ t_i^{(k+1)} &= \frac{1}{u_i^{(k)}} y_i^{(k)}. \end{aligned} \tag{2.4}$$

Here

$$z^{(i)} = \left(t_1^{(k+1)^T}, \ldots, t_{i-1}^{(k+1)^T}, t_i^{(k)^T}, \ldots, t_m^{(k)^T}\right)^T, \quad i = 1, 2, \ldots, m.$$

(3) If $|\det(R(t^{(k+1)})) - \det(R(t^{(k)}))| \leqslant \varepsilon$, then stop the algorithm with the approximate solution $t^{(k+1)}$.

The expression (2.2) of the objective function motivates us to employ an alternating variable method (AVM) to solve (1.4), which is a natural extension of the so-called coordinated descent method [12, p. 53] and is described in Algorithm 2.2.

Algorithm 2.2 (Alternating variable method [3], AVM)

(1) Take $t^{(0)} = \begin{pmatrix} t_1^{(0)} \\ \vdots \\ t_m^{(0)} \end{pmatrix} \in \Sigma_m$.

(2) Suppose that the current approximate point is $t^{(k)} = \begin{pmatrix} t_1^{(k)} \\ \vdots \\ t_m^{(k)} \end{pmatrix} \in \Sigma_m$. Then compute $t^{(k+1)}$ by

$$t_i^{(k+1)} = \arg \min_{\|t_i\|_2=1} \det\left(R(t_1^{(k+1)}, \ldots, t_{i-1}^{(k+1)}, t_i, t_{i+1}^{(k)}, \ldots, t_m^{(k)})\right), \quad i = 1, 2, \ldots, m. \tag{2.5}$$

(3) If $|\det(R(t^{(k+1)})) - \det(R(t^{(k)}))| \leqslant \varepsilon$, then stop the algorithm, and $t^{(k+1)}$ is an approximate solution.

This algorithm was outlined by Kettenring [3]. We will present some analysis below and numerically demonstrate its performance in Sect. 5.

Let

$$\rho(t) = \det(R(t)), \quad \hat{z}^{(i)} = \left(t_1^{(k+1)T}, \ldots, t_{i-1}^{(k+1)T}, t_i^{(k)T}, \ldots, t_m^{(k)T}\right)^T.$$

We see that

$$\rho\left(t^{(k)}\right) \geq \rho\left(\hat{z}^{(2)}\right) \geq \cdots \geq \rho\left(\hat{z}^{(m+1)}\right) = \rho\left(t^{(k+1)}\right). \tag{2.6}$$

Thus it is very clear that the sequence $\{\rho(t^{(k)})\}$ is monotonically non-increasing and bounded. This directly implies the convergence of $\{\rho(t^{(k)})\}$.

Let

$$G_i^{(k)} = \begin{pmatrix} t_1^{(k+1)T} R_{1,i} \\ \vdots \\ t_{i-1}^{(k+1)T} R_{i-1,i} \\ t_{i+1}^{(k)T} R_{i+1,i} \\ \vdots \\ t_m^{(k)T} R_{m,i} \end{pmatrix}.$$

Then (2.2) implies that (2.5) is equivalent to the following optimization problem:

$$\max_{\|t_i\|_2=1} t_i^T G_i^{(k)T} R_i (z^{(i)})^{-1} G_i^{(k)} t_i, \tag{2.7}$$

and its solution $t_i^{(k+1)}$ is the unit eigenvector according to the largest eigenvalue of $G_i^{(k)T} R_i (z^{(i)})^{-1} G_i^{(k)}$. There are several efficient algorithms to compute $t_i^{(k+1)}$, see e.g., [14].

Next, we will show Algorithm 2.1 is closely relating to Algorithm 2.2. Indeed, we will apply iterative majorization principle(see, e.g., [15]) to (2.7) to demonstrate that Algorithm 2.1 is an inexact version of Algorithm 2.2. To this end, let

$$M_{ik} = G_i^{(k)T} R_i (z^{(i)})^{-1} G_i^{(k)},$$
$$g(t_i) = t_i^T M_{ik} t_i.$$

Then, M_{ik} is a symmetric and positive semi-definite matrix, $g(t_i)$ is a convex function. We note that

$$\begin{aligned} g(t_i) &= g\left(t_i^{(k)}\right) + 2\left(M_{ik} t_i^{(k)}\right)^T \left(t_i - t_i^{(k)}\right) + \left(t_i - t_i^{(k)}\right)^T M_{ik} \left(t_i - t_i^{(k)}\right) \\ &\geq h_i(t_i) \equiv g_i\left(t_i^{(k)}\right) + 2\left(M_{ik} t_i^{(k)}\right)^T \left(t_i - t_i^{(k)}\right). \end{aligned} \tag{2.8}$$

Noticing that $g_i(t_i^{(k)}) = h_i(t_i^{(k)})$, (2.7) can be approximately solved by

$$\begin{aligned} t_i^{(k+1)} &= \arg \max_{\|t_i\|_2 = 1} h_i(t_i) \\ &= y_i^{(k)} / \| y_i^{(k)} \|_2 . \end{aligned} \tag{2.9}$$

Here $y_i^{(k)} = M_{ik} t_i^{(k)}$.

We see that (2.9) is same as (2.4). Therefore, Algorithm 2.1 can be seen as an inexact version of the AVM algorithm. From (2.8), we see that the sequence $\{t^{(k)}\}$ generated by Algorithm 2.1 has the following property:

$$\rho\left(t^{(k)}\right) \geq \rho\left(z^{(2)}\right) \geq \cdots \geq \rho\left(z^{(m)}\right) \geq \rho\left(t^{(k+1)}\right). \tag{2.10}$$

That is, Algorithm 2.1 is also a descent algorithm. Furthermore, we see that $\rho(t^{(k)}) = \rho(t^{(k+1)})$ if and only if there exists $u_i^{(k)} \geq 0$ to satisfy

$$M_{ik} t_i^{(k)} = u_i^{(k)} t_i^{(k)}.$$

In other words, for Algorithm 2.1, $\rho(t^{(k)}) = \rho(t^{(k+1)})$ if and only if $t^{(k)}$ is a stationary point of the Genvar criterion.

Algorithms 2.1 and 2.2 mainly deal with the first-stage of Genvar. That is, a first-stage canonical vector, $s^{(1)}$, is obtained. We can expand these algorithms successively to later stages as follows. Suppose we have obtained the first $p-1$ canonical vectors $s^{(i)} = \begin{pmatrix} s_1^{(i)} \\ \vdots \\ s_m^{(i)} \end{pmatrix} \in \Sigma_m, \quad i = 1, 2, \ldots, p-1; \quad p \leq \min_{1 \leq i \leq m} n_i$. To compute the p-stage canonical vector $s^{(p)} = \begin{pmatrix} s_1^{(p)} \\ \vdots \\ s_m^{(p)} \end{pmatrix}$, we consider the following optimization problem

$$\min \det\left(R(t)\right), \quad s.t.\ t \in \Sigma_m, t_i^T s_i^{(l)} = 0, \quad \text{for} \quad l = 1, \ldots, p-1;\ i = 1, \ldots, m. \tag{2.11}$$

Let $Q_i^{(1)} = [s_i^{(1)}, \ldots, s_i^{(p-1)}]$ and let $Q_i = [Q_i^{(1)}, X_i]$ be an $n_i \times n_i$ orthogonal matrix. Let

$$\tilde{R} = \left(\tilde{R}_{ij}\right)_{m\times m}, \quad \tilde{R}_{ij} \equiv X_i^T R_{ij} X_j,$$

$$\tilde{\Sigma}_m = \left\{ \tilde{t} = \begin{pmatrix} \tilde{t}_1 \\ \vdots \\ \tilde{t}_m \end{pmatrix} : \tilde{t}_i \in \mathbb{R}^{n_i - p + 1}, \quad \| \tilde{t}_i \|_2 = 1 \right\},$$

$$\tilde{R}(\tilde{t}) = \left(\tilde{t}_i^T \tilde{R}_{ij} \tilde{t}_j\right)_{m\times m}.$$

The constraints in (2.11) mean that t_i can be expressed as $t_i = X_i \tilde{t}_i$ and so we consider the following optimization problem

$$\min \det\left(\tilde{R}(\tilde{t})\right), \quad s.t.\ \tilde{t} \in \tilde{\Sigma}_m. \tag{2.12}$$

Suppose $\tilde{t} = \begin{pmatrix} \tilde{t}_1^{(p)} \\ \vdots \\ \tilde{t}_m^{(p)} \end{pmatrix}$ is one solution of (2.12), then we see that $s^{(p)} = \begin{pmatrix} s_1^{(p)} \\ \vdots \\ s_m^{(p)} \end{pmatrix}$ with $s_i^{(p)} = X_i \tilde{t}_i^{(p)} (i = 1, 2, \ldots, m)$ is the p-stage canonical vector of Genvar. Obviously, we can apply Algorithms 2.1 and 2.2 to solve problem (2.12).

3 Upper and lower bounds of the optimal objective function value

In this section, we present some upper and lower bounds of the optimal objective function value of (1.4). Throughout this section we suppose R is an invertible matrix (then symmetric and positive definite matrix), and therefore can be factorized as

$$R = P^T P, \quad P = [P_1, \ldots, P_m], \quad P_j \in \mathbb{R}^{n\times n_j}. \tag{3.1}$$

Clearly, $P_j^T P_j = I_{n_j} \quad (j = 1, 2, \ldots, m)$.
Let

$$R_m(t) = (t_i^T R_{ij} t_j)_{1\le i,j\le m-1},$$

$$a_m(t) = \begin{pmatrix} t_1^T R_{1,m} t_m \\ \vdots \\ t_{m-1}^T R_{m-1,m} t_m \end{pmatrix}.$$

Then, we see that

$$\det(R(t)) = \det(R_m(t))(1 - a_m^T R_m(t)^{-1} a_m). \tag{3.2}$$

Let

$$L(t) = [P_1 t_1, \ldots, P_m t_m], \quad L_{m-1}(t) = [P_1 t_1, \ldots, P_{m-1} t_{m-1}].$$

The following auxiliary result presents a least squares expression of $1 - a_m^T R_m(t)^{-1} a_m$.

Lemma 3.1

$$1 - a_m^T R_m(t)^{-1} a_m = \min_{y\in\mathbb{R}^{m-1}} \| P_m t_m - L_{m-1}(t) y \|_2^2. \tag{3.3}$$

Proof The solution of the least squares problem (3.3) is

$$y = L_{m-1}(t)^{\dagger} P_m t_m.$$

Consequently,

$$\begin{aligned}
\gamma_m &\equiv \min_{y\in\mathbb{R}^{m-1}} \parallel P_m t_m - L_{m-1}(t)y \parallel_2^2 \\
&= \parallel P_m t_m - L_{m-1}L_{m-1}^{\dagger} P_m t_m \parallel_2^2 \\
&= \parallel P_m t_m \parallel_2^2 - \parallel L_{m-1}L_{m-1}^{\dagger} P_m t_m \parallel_2^2 \\
&= 1 - \left(L_{m-1}^T P_m t_m\right)^T R_m(t)^{-1} \left(L_{m-1}^T P_m t_m\right) \\
&= 1 - a_m^T R_m(t)^{-1} a_m.
\end{aligned}$$

□

Now, we present a lower bound of $\min\limits_{t\in\Sigma_m} det(R(t))$. Let

$$H_{m-1} = \left[P_1, \ldots, P_{m-1}\right]. \tag{3.4}$$

Then,

$$\begin{aligned}
\gamma_m &= \min_{y\in\mathbb{R}^{m-1}} \parallel P_m t_m - H_{m-1} \begin{pmatrix} y_1 t_1 \\ \vdots \\ y_{m-1} t_{m-1} \end{pmatrix} \parallel_2^2 \\
&\geq \min_{z\in\mathbb{R}^{n-n_m}} \parallel P_m t_m - H_{m-1} z \parallel_2^2 \\
&= 1 - \parallel H_{m-1} H_{m-1}^{\dagger} P_m t_m \parallel_2^2 .
\end{aligned} \tag{3.5}$$

Theorem 3.1 *Let* $H_k = [P_1, \ldots, P_k]$, *then,*

$$\det(R(t)) \geq \prod_{k=1}^{m-1} \left(1 - \parallel H_k H_k^{\dagger} P_{k+1} \parallel_2^2\right), \quad \textit{for all} \quad t \in \Sigma_m. \tag{3.6}$$

Proof From (3.5), we see that

$$\begin{aligned}
\det(R(t)) &= \det(R_m(t))(1 - a_m^T R_m(t)^{-1} a_m) \\
&\geq \det(R_m(t))(1 - \parallel H_{m-1} H_{m-1}^{\dagger} P_m \parallel_2^2).
\end{aligned}$$

Recursively, it follows that (3.6) holds. □

An upper bound of $\min\limits_{t\in\Sigma_m} \det(R(t))$ is given below.Let

$$\begin{aligned}
\rho^{(1)} &= \parallel R_{i_1,j_1} \parallel_2 = \max_{i\neq j} \parallel R_{i,j} \parallel_2, \\
\rho^{(2)} &= \parallel R_{i_2,j_2} \parallel_2 = \max_{\substack{i\neq i_1 \\ j\neq j_1}} \parallel R_{i,j} \parallel_2, \\
&\vdots
\end{aligned}$$

$$\rho^{(k)} = \| R_{i_k,j_k} \|_2 = \max_{\substack{i \neq i_1,\ldots,i_{k-1} \\ j \neq j_1,\ldots,j_{k-1}}} \| R_{i,j} \|_2,$$

$$k \equiv \left[\frac{m}{2}\right].$$

Theorem 3.2

$$\min_{t \in \Sigma_m} \det(R(t)) \leq \prod_{j=1}^{k} (1 - \rho^{(j)^2}). \tag{3.7}$$

Proof Without loss of generality, we suppose that

$$(i_1, j_1) = (1, 2), \quad (i_s, j_s) = (2s-1, 2s), \quad s = 2, 3, \ldots, k.$$

Let t_{2s-1} and t_{2s} be respectively the unit right and left singular vectors corresponding to the largest singular value of $R_{2s-1,2s}$. Then

$$R_{2s-1,2s} t_{2s-1} = \| R_{2s-1,2s} \|_2 \, t_{2s}, \quad s = 1, 2, \ldots, k.$$

The Hadamard inequality implies that

$$\begin{aligned} \min_{t \in \Sigma_m} \det(R(t)) &\leq \prod_{s=1}^{k} \det \begin{pmatrix} 1 & t_{2s-1}^T R_{2s-1,2s} t_{2s} \\ t_{2s-1}^T R_{2s-1,2s} t_{2s} & 1 \end{pmatrix} \\ &= \prod_{s=1}^{k} \left(1 - \rho^{(s)^2}\right). \end{aligned}$$

□

From the view point of practical application, the covariance matrix A usually is unknown, we should replace A by a sample covariance matrix, and consequently a perturbed correlation matrix R is applied in Genvar.

If $\rho_{min} \equiv \min_{t \in \Sigma_m} \det(R(t))$ is very small, then the perturbation of R may result in a large relative perturbation of ρ_{min}. The inequality (3.7) means that if any among $\rho^{(s)}$ is close to 1 then ρ_{min} will be small. We note that $\rho^{(s)}$ is the canonical correlation between y_{2s-1} and y_{2s}[1]. Furthermore, by applying the interlacing property of eigenvalues for real symmetric matrix, it is easy to see that $1 - \rho^{(s)}$ is an upper bound of the smallest eigenvalue of R.

4 Starting point strategies

Although enjoying the monotone convergence, neither of Algorithms 2.1 and 2.2 can guarantee convergence to a global solution of (1.4). So in this section we consider a preconditioning technique to improve the performance of Algorithms 2.1 and 2.2. The idea of the preconditioning technique is to choose a good approximate solution $t^{(0)}$ of (1.4) as a starting point for both algorithms. Since Algorithms 2.1 and 2.2 are monotone, and

$$\rho(t^{(0)}) - \rho(t^{(k)}) = \sum_{i=1}^{k} (\rho(t^{(i-1)}) - \rho(t^{(i)})), \tag{4.1}$$

we see that if $\rho(t^{(0)})$ has been close to the optimal objective function value of Genvar, then (4.1) means that the sequence $\{\rho(t^{(i-1)}) - \rho(t^{(i)})\}$ will quickly converges to zero, and

therefore the special selection of $t^{(0)}$ increases the rate of convergence. On the other hand, if t^* is an accumulation point of $\{t^{(k)}\}$, then

$$\rho(t^*) \leq \rho(t^{(0)}).$$

This result implies that some stationary points of (1.4) can be ruled out from the limiting points of Algorithms 2.1 and 2.2. In this sense, a good starting point $t^{(0)}$ can boost up the probability of finding a global solution.

Let the eigenvalues of $R(t)$ be $\lambda_1(t) \geq \ldots \geq \lambda_m(t)$. Then

$$\lambda_1(t) + \cdots + \lambda_m(t) = m, \tag{4.2}$$

$$\det(R(t)) = \prod_{j=1}^{m} \lambda_j(t), \tag{4.3}$$

$$\lambda_{min}(R) \leq \lambda_j(t) \leq \lambda_{max}(R). \tag{4.4}$$

Noticing that the solution of the maximization problem $\max\limits_{s.t.\, s_1+\cdots+s_m=m} \prod\limits_{j=1}^{m} s_j$ is $s_1 = \cdots = s_m = 1$, we consider two starting point strategies as following, in which the aim is to make $\lambda_1(t)$ as large as possible (Strategy 4.1) or to make $\lambda_m(t)$ as small as possible (Strategy 4.2).

Strategy 4.1 Compute the eigenvector $y = \begin{pmatrix} y_1 \\ \vdots \\ y_m \end{pmatrix}$ corresponding to the largest eigenvalue $\lambda_{max}(R)$ of R, then

$$t_i^{(0)} = \begin{cases} y_i / \parallel y_i \parallel_2, & if \parallel y_i \parallel_2 > 0, \\ u_i, & otherwise. \end{cases} \tag{4.5}$$

Here, $u_i \in \mathbb{R}^{n_i}$, $\parallel u_i \parallel_2 = 1$.

Strategy 4.2 Compute the eigenvector $z = \begin{pmatrix} z_1 \\ \vdots \\ z_m \end{pmatrix}$ corresponding to the smallest eigenvalue $\lambda_{min}(R)$ of R, then

$$t_i^{(0)} = \begin{cases} z_i / \parallel z_i \parallel_2, & if \parallel z_i \parallel_2 > 0, \\ v_i, & otherwise. \end{cases} \tag{4.6}$$

Here, $v_i \in \mathbb{R}^{n_i}$, $\parallel v_i \parallel_2 = 1$.

We note that the following results have been proved by Kettenring [3].

$$\begin{aligned} \min_{t \in \Sigma_m} \lambda_m(t) &= \lambda_{min}(R), \\ \max_{t \in \Sigma_m} \lambda_1(t) &= \lambda_{max}(R). \end{aligned} \tag{4.7}$$

These results imply that $t^{(0)}$ generated by Strategy 4.1 is a solution of the so-called Maxvar, and $t^{(0)}$ generated by Strategy 4.2 is a solution of the so-called Minvar. Since both Maxvar and Minvar are generalizations of CCA, and Genvar also is a generalization of CCA, intuitively, $t^{(0)}$ should be a good approximate solution of the Genvar criterion.

If $\lambda_{min}(R)$ is small, then employ Strategy 4.2, however, if $\lambda_{max}(R)$ is large, then employ Strategy 4.1. we now present some estimates of $\lambda_{max}(R)$ and $\lambda_{min}(R)$.

Proposition 4.1 *Let* $Rx = \lambda_{max}(R)x$, $\| x \|_2 = 1$, $x = \begin{pmatrix} x_1 \\ \vdots \\ x_m \end{pmatrix}$, $x_j \in \mathbb{R}^{n_j}$. *Then*

$$\lambda_{max}(R) \leq \left(\sum_{j=1}^{m} \| x_j \|_2 \right)^2.$$

Proof Consider the decomposition of R (block Cholesky decomposition)

$$R = L^T L, \quad L = [L_1, \ldots, L_m], \quad L_i \in \mathbb{R}^{n \times n_i}.$$

Then,

$$\begin{aligned} \lambda_{max}(R) &= x^T R x \\ &= \| Lx \|_2^2 \\ &= \left(\| \sum_{j=1}^{m} L_j x_j \|_2 \right)^2 \\ &\leq \left(\sum_{j=1}^{m} \| x_j \|_2 \right)^2. \end{aligned}$$

□

The proof of Proposition 4.1 shows that, unless $L_j x_j \approx L_1 x_1, \ j = 2, \ldots, m, \lambda_{max}(R)$ is not very close to m. This seems to imply that usually Strategy 4.2 is better than Strategy 4.1. The following result is a consequence of interlacing property of eigenvalues of real symmetric matrix.

Proposition 4.2

$$\lambda_{min}(R) \leq 1 - \rho_{max}^2.$$

Here, $\rho_{max} = \max_{i \neq j} \| R_{ij} \|_2$.

5 Numerical Examples

In this section, we present our numerical experiment of Algorithms 2.1 and 2.2 with Strategies 4.1 and 4.2 to show their efficiency, and most importantly, the effectiveness of Strategies 4.1 and 4.2 both in reducing the number of iterations and in seeking the global minima of Genvar. All of our tests were conducted in MATLAB on a PC with Intel(R) Pentium(R)4 processor of 3.20 GHZ. The defaults value of ε is 10^{-5}.

For our comparison, we first create two small examples, Examples 5.1 and 5.2. Then, in Examples 5.3 and 5.4, the matrices are of size 100×100, which were randomly generated, and we ran both algorithms for each matrix starting from 100 random initial points. The test results are documented in Tables 1, 2 and 3, respectively. Under columns "$Avg.Iter\sharp$" are the average numbers of iterations needed to meet the stopping criterion. Under columns "$Avg.Time$" are the average CPU times (in seconds, measured by the MATLAB function **cputime**) including

the cost of computing initial point. Under columns "$\bar{\rho}_{Genvar}$" are the average objective function values. Under columns "$Total\ Global\%$" are the sample probabilities, out of all converged objective function values for fixed A, of hitting the best we've seen.

First, we list the related algorithms below:

Gauss-R: Algorithm 2.1 with random starting point.
Gauss-4.1: Algorithm 2.1 with Strategy 4.1.
Gauss-4.2: Algorithm 2.1 with Strategy 4.2.
AVM-R: Algorithm 2.2 with random starting point.
AVM-4.1: Algorithm 2.2 with Strategy 4.1.
AVM-4.2: Algorithm 2.2 with Strategy 4.2.

Example 5.1 The matrix X is given by:

$$X = \begin{pmatrix} 7 & 0 & 5 & 8 & 4 & 0 & 5 & 7 & 4 & 2 & 6 & 9 & 7 & 9 \\ 5 & 6 & 6 & 3 & 3 & 6 & 5 & 2 & 0 & 6 & 6 & 8 & 6 & 7 \\ 7 & 2 & 5 & 5 & 3 & 3 & 7 & 6 & 3 & 4 & 7 & 10 & 6 & 7 \\ 5 & 7 & 7 & 4 & 1 & 6 & 3 & 2 & 0 & 6 & 4 & 7 & 7 & 8 \\ 5 & 7 & 6 & 4 & 2 & 5 & 5 & 5 & 1 & 5 & 6 & 8 & 7 & 8 \\ 6 & 8 & 6 & 1 & 1 & 5 & 4 & 3 & 0 & 5 & 5 & 8 & 8 & 10 \\ 5 & 4 & 10 & 3 & 0 & 4 & 2 & 2 & 0 & 4 & 3 & 6 & 5 & 10 \\ 6 & 6 & 6 & 5 & 2 & 6 & 4 & 4 & 0 & 4 & 5 & 8 & 9 & 10 \end{pmatrix}.$$

We centered each column of X and got a matrix $\hat{X}$. Then, the matrix A was formed by $A = \hat{X}^T \hat{X}$. Here, $m = 4, n_1 = 4, n_2 = 3, n_3 = 4, n_4 = 3$. This example, which is based on data from sensory evaluation of port wines, had been used by Hanafi and Kiers [6].

Example 5.2 The matrix A is given by

$$A = \begin{pmatrix} 7.4470 & -5.3387 & -4.8906 & 0.0903 & 2.2446 & 3.6421 & -0.1752 & -0.0246 & 1.7437 & 5.8028 \\ -5.3387 & 13.8905 & 4.5704 & -1.4263 & 0.5230 & 1.8020 & -3.7392 & 3.5696 & -1.9936 & -8.1650 \\ -4.8906 & 4.4704 & 8.8486 & -0.1523 & -0.3143 & -1.6288 & -1.3250 & 0.4012 & -2.7730 & -2.6152 \\ -0.0903 & -1.4263 & -0.1523 & 4.4653 & -4.2442 & -5.9271 & -3.8049 & -3.9631 & 5.1445 & -0.4010 \\ 2.2446 & 0.5230 & -0.3143 & -4.2442 & 25.9532 & 6.8747 & -4.8952 & -0.8508 & 3.9045 & -4.8526 \\ 3.6421 & 1.8020 & -1.6228 & -5.9271 & 6.8747 & 12.3494 & 2.8848 & 3.0664 & -3.8247 & 0.7834 \\ -0.1752 & -3.7392 & -1.3250 & -3.8049 & -4.8952 & 2.8848 & 11.7609 & 5.7035 & -7.7411 & 6.0963 \\ -0.0246 & 3.5696 & 0.4012 & -3.9631 & -0.8508 & 3.0664 & 5.7035 & 12.4512 & -11.9284 & 4.7289 \\ 1.7437 & -1.9936 & -2.7730 & 5.1445 & 3.9045 & -3.8247 & -7.7411 & -11.9284 & 16.8287 & -5.8446 \\ 5.8028 & -8.1650 & -2.6152 & -0.4010 & -4.8526 & 0.7834 & 6.0963 & 4.7289 & -5.8446 & 11.6403 \end{pmatrix}$$

with $m = 3, n_1 = 3, n_2 = 3, n_3 = 4$.

The numerical results of Examples 5.1 and 5.2 are documented in Table 1. These results show that for these examples Strategies 4.1 and 4.2 improved the performance of Algorithms 2.1 and 2.2 in terms of iteration numbers and CPU time.

Example 5.3 In this example, we randomly generated 100 matrices of size 100×100, with $m = 4,\ n_1 = 10,\ n_i = n_1 + (i-1)10,\quad i = 2, 3, 4$. These matrices were constructed by the following Matlab code.

$$\begin{aligned} C &= rand(100, 100); \\ [Q, R] &= qr(C); \\ \Lambda &= diag(rand(1, 100)); \\ A &= Q * \Lambda * Q'. \end{aligned}$$

Table 1 Numerical results for Examples 5.1 and 5.2

Example	Algorithms	Avg.Iter	Avg.Time	$\bar{\rho}_{Genvar}$
Exa. 5.1	AVM-4.1	1	0.014664094	9.81451E-09
	AVM-4.2	1	0.016068103	2.22703E-16
	AVM-R	2.95	0.013572087	6.79982E-08
	Gauss-4.1	1	0.014664094	1.61662E-07
	Gauss-4.2	1	0.014196091	0.1.8971E-16
	Gauss-R	7.83	0.034008218	2.64304E-06
Exa. 5.2	AVM-4.1	9	0.0312002	0.026311657
	AVM-4.2	7	0.0312002	0.026756604
	AVM-R	27.66	0.074256476	0.035880097
	Gauss-4.1	30	0.1092007	0.028023752
	Gauss-4.2	14	0.1092007	0.027663713
	Gauss-R	43.52	0.109668703	0.035243719

Table 2 Numerical results for Example 5.3

	Avg.Iter.♯	Avg.Time	Total Global%
AVM-4.1	85.26	1.638634504	10
AVM-4.2	2.11	0.157405009	93
AVM-R	82.8807	1.603339278	10.73
Gauss-4.1	90.44	1.116187155	8
Gauss-4.2	2.11	0.145236931	93
Gauss-R	87.5964	1.066034394	10.29

For each matrix A, we ran Algorithms 2.1 and 2.2 starting from 100 randomly generated initial points. Thus for each A, we obtained 204 objective function values. The numerical results are documented in Table 2.

The results in Table 2 show that Strategy 4.2 significantly improved the performance of Algorithms 2.1 and 2.2 in terms of iterative steps, CPU time and "Total Global". Strategy 4.2 is much better than Strategy 4.1. In addition, for this example, Algorithm 2.1 is faster than Algorithm 2.2.

Example 5.4 In this example, we randomly generated 100 matrices, with $m = 4, n_1 = 10, n_i = n_1 + (i-1)10, i = 2, 3, 4$. These matrices were constructed by the following Matlab code.

$$\begin{aligned} C &= rand(100, 100); \\ [Q, R] &= qr(C); \\ \Lambda &= diag(100 * rand(1, 20), rand(1, 80)); \\ A &= Q * \Lambda * Q' \end{aligned}$$

The numerical results are documented in Table 3.

The results in Table 3 show that Strategy 4.1 significantly improved the performance of Algorithms 2.1 and 2.2 in terms of iterative steps, CPU time and "Total Global". Strategy 4.1 is much better than Strategy 4.2.

Table 3 Numerical results for Example 5.4

	Avg.Iter.♯	Avg.Time	Total Global%
AVM-4.1	1.03	0.119652767	94
AVM-4.2	11.85	0.312938006	6
AVM-R	18.9307	0.39156095	0.09
Gauss-4.1	1.02	0.108732697	93
Gauss-4.2	17.14	0.325262085	6
Gauss-R	41.7432	0.51479238	0.023

Summarizing the results of Examples 5.1-5.4, we can make several observations as follows.

(i) Generally, both AVM and Gauss–Seidel methods converge to the global minima with a low probability if the starting point is randomly selected.
(ii) Strategies 4.1 and 4.2 can significantly improve these iterative methods in terms of iterative steps and CPU time.
(iii) Strategy 4.1 and/or Strategy 4.2 can significantly boost up the probability of getting a "better" solution of Genvar.
(iv) The Gauss–Seidel method frequently converges faster than the AVM in term of CPU time.

6 Final Remarks

In this paper, the AVM method proposed by Kettenring for the Genvar criterion is analysed. An inexact version of AVM, the Gauss–Seidel method, is presented. We suggest two starting point strategies to improve these iterative methods in terms of rate of convergence and the probability of finding a global solution of Genvar.

We ran Examples 5.3 and 5.4 for large m. The results show that, as m increases, the efficiency of Strategies 4.1 and 4.2 descend. Hence, how to construct other starting point strategy is a future topic. We hope the results in Sect. 3 can enlighten us on this topic.

Acknowledgments We are very grateful to both referees for their constructive comments and suggestions.

References

1. Hotelling, H.: Relations between two sets of variables. Biometrika **28**, 321–377 (1936)
2. Steel, R.G.D.: Minimum generalized variance for a set of linear functions. Ann. Math. Statist. **22**, 456–460 (1951)
3. Kettenring, J.R.: Canonical analysis of several sets of variables. Biometrika **58**, 433–451 (1971)
4. Horst, P.: Relations among m sets of measures. Psychometrika **26**, 129–149 (1961)
5. Van De Geer, J.P.: Linear relations among k sets of variables. Psychometrika **49**(1), 70–94 (1984)
6. Hanafi, M., Kiers, H.A.L.: Analysis of K sets of data, with differential emphasis on agreement between and within sets. Comput. Statist. Data Anal. **51**, 1491–1508 (2006)
7. Vía, J., Santamaría, I., Pérez, J.: Canonical correlation analysis (CCA) algorithms for multiple data sets: an application to blind SIMO equalization. European Signal Processing Conference, EUPSIPCO, Antalya, Turkey (2005)
8. Vía, J., Santamaría, I., Pérez, J.: A learning algorithm for adaptive canonical correlation analysis of several data sets. Neural Netw. **20**, 139–152 (2007)

9. De leeuw, J.: Multivariate analysis with optimal scaling. In: Das Gupta, S., Ghosh, J.K.(eds.) Proceedings of the International Conference on Advances in Multivariate Statistical Analysis, Calcutta, pp. 127–160 (1988)
10. Zhang, Y.T., Zhu, X.D.: Correlation of K groups of random variables. Chin. J. Appl. Probab. Statist. **4**(1), 27–34 (1988)
11. Yuan, Y.H., et al.: A novel multiset integrated canonical correlation analysis framework and its application in feature fusion. Pattern Recognit. **44**, 1031–1040 (2011)
12. Nocedal, J., Wright, S.J.: Numerical Optimization. Springer, New York (1999)
13. Geomans, M.Z., Williamson, D.P.: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. J. ACM **42**, 1115–1145 (1995)
14. Golub, G.H., Van Loan, C.F.: Matrix Computations, 3rd edn. Johns Hopkins University Press, Baltimore (1996)
15. Kiers, H.A.L.: Setting up alternating least squares and iterative majorization algorithms for solving various matrix optimization problems. Comput. Statist. Data Anal. **41**, 157–170 (2002)