SCF Iteration for Orthogonal Canonical Correlation Analysis

Ren-Cang Li

Department of Mathematics
University of Texas at Arlington

joint work with
Lei-hong Zhang Li Wang Zhaojun Bai

September 17, 2020

Outline

Outline

Single-Vector CCA

Canonical Correlation Analysis (CCA) is a two-view multivariate statistical method (H. Hotelling, 1936), where the variables of observations is partitioned into two sets, i.e., two views of the data.

Data matrices $S_1 \in \mathbb{R}^{n\times q}$ (view 1, n features), $S_2 \in \mathbb{R}^{m\times q}$ (view 2, m features), q is the number of samples.
Both centralized: $S_i \mathbf{1}_q = 0$; otherwise, $S_i \leftarrow S_i - \frac{1}{q}(S_i\mathbf{1}_q)\mathbf{1}_q^{\mathrm{T}}$.

Canonical Variates $\boldsymbol{z}_1 = S_1^{\mathrm{T}}\boldsymbol{x}_1$, $\boldsymbol{z}_2 = S_2^{\mathrm{T}}\boldsymbol{x}_2$ defined in terms of Canonical Weight Vectors: $\boldsymbol{x}_1 \in \mathbb{R}^n$, $\boldsymbol{x}_2 \in \mathbb{R}^m$.

Canonical Correlation: $\rho(\boldsymbol{x}_1, \boldsymbol{x}_2) := \dfrac{\boldsymbol{z}_1^{\mathrm{T}}\boldsymbol{z}_2}{||\boldsymbol{z}_1||_2||\boldsymbol{z}_2||_2} = \dfrac{\boldsymbol{x}_1^{\mathrm{T}}C_{1,2}\boldsymbol{x}_2}{\sqrt{\boldsymbol{x}_1^{\mathrm{T}}C_{1,1}\boldsymbol{x}_1}\sqrt{\boldsymbol{x}_2^{\mathrm{T}}C_{2,2}\boldsymbol{x}_2}}$,

where $C_{i,j} = S_i S_j^{\mathrm{T}}$, (Cross-)Covariance.

CCA aims to find the pair of canonical weight vectors to maximize canonical correlation:

$$\max_{\boldsymbol{x}_1,\boldsymbol{x}_2} \rho(\boldsymbol{x}_1, \boldsymbol{x}_2). \tag{1}$$

Single-Vector CCA

Canonical Correlation Analysis (CCA) is a two-view multivariate statistical method (H. Hotelling, 1936), where the variables of observations is partitioned into two sets, i.e., two views of the data.

Data matrices $S_1 \in \mathbb{R}^{n\times q}$ (view 1, n features), $S_2 \in \mathbb{R}^{m\times q}$ (view 2, m features), q is the number of samples.
Both centralized: $S_i \mathbf{1}_q = 0$; otherwise, $S_i \leftarrow S_i - \frac{1}{q}(S_i \mathbf{1}_q)\mathbf{1}_q^{\mathrm{T}}$.

Canonical Variates $\boldsymbol{z}_1 = S_1^{\mathrm{T}}\boldsymbol{x}_1$, $\boldsymbol{z}_2 = S_2^{\mathrm{T}}\boldsymbol{x}_2$ defined in terms of Canonical Weight Vectors: $\boldsymbol{x}_1 \in \mathbb{R}^n$, $\boldsymbol{x}_2 \in \mathbb{R}^m$.

Canonical Correlation: $\rho(\boldsymbol{x}_1, \boldsymbol{x}_2) := \dfrac{\boldsymbol{z}_1^{\mathrm{T}}\boldsymbol{z}_2}{||\boldsymbol{z}_1||_2||\boldsymbol{z}_2||_2} = \dfrac{\boldsymbol{x}_1^{\mathrm{T}} C_{1,2}\boldsymbol{x}_2}{\sqrt{\boldsymbol{x}_1^{\mathrm{T}} C_{1,1}\boldsymbol{x}_1}\sqrt{\boldsymbol{x}_2^{\mathrm{T}} C_{2,2}\boldsymbol{x}_2}}$,

where $C_{i,j} = S_i S_j^{\mathrm{T}}$, (Cross-)Covariance.

CCA aims to find the pair of canonical weight vectors to maximize canonical correlation:

$$\max_{\boldsymbol{x}_1,\boldsymbol{x}_2} \rho(\boldsymbol{x}_1, \boldsymbol{x}_2). \tag{1}$$

Single-Vector CCA

Canonical Correlation Analysis (CCA) is a two-view multivariate statistical method (H. Hotelling, 1936), where the variables of observations is partitioned into two sets, i.e., two views of the data.

Data matrices $S_1 \in \mathbb{R}^{n\times q}$ (view 1, n features), $S_2 \in \mathbb{R}^{m\times q}$ (view 2, m features), q is the number of samples.
Both centralized: $S_i \mathbf{1}_q = 0$; otherwise, $S_i \leftarrow S_i - \frac{1}{q}(S_i \mathbf{1}_q)\mathbf{1}_q^{\mathrm{T}}$.

Canonical Variates $\boldsymbol{z}_1 = S_1^{\mathrm{T}}\boldsymbol{x}_1$, $\boldsymbol{z}_2 = S_2^{\mathrm{T}}\boldsymbol{x}_2$ defined in terms of Canonical Weight Vectors: $\boldsymbol{x}_1 \in \mathbb{R}^n$, $\boldsymbol{x}_2 \in \mathbb{R}^m$.

Canonical Correlation: $\rho(\boldsymbol{x}_1, \boldsymbol{x}_2) := \dfrac{\boldsymbol{z}_1^{\mathrm{T}}\boldsymbol{z}_2}{||\boldsymbol{z}_1||_2||\boldsymbol{z}_2||_2} = \dfrac{\boldsymbol{x}_1^{\mathrm{T}}C_{1,2}\boldsymbol{x}_2}{\sqrt{\boldsymbol{x}_1^{\mathrm{T}}C_{1,1}\boldsymbol{x}_1}\sqrt{\boldsymbol{x}_2^{\mathrm{T}}C_{2,2}\boldsymbol{x}_2}}$,
where $C_{i,j} = S_i S_j^{\mathrm{T}}$, (Cross-)Covariance.

CCA aims to find the pair of canonical weight vectors to maximize canonical correlation:

$$\max_{\boldsymbol{x}_1,\boldsymbol{x}_2} \rho(\boldsymbol{x}_1, \boldsymbol{x}_2). \tag{1}$$

Single-Vector CCA

Canonical Correlation Analysis (CCA) is a two-view multivariate statistical method (H. Hotelling, 1936), where the variables of observations is partitioned into two sets, i.e., two views of the data.

Data matrices $S_1 \in \mathbb{R}^{n\times q}$ (view 1, n features), $S_2 \in \mathbb{R}^{m\times q}$ (view 2, m features), q is the number of samples.
Both centralized: $S_i \mathbf{1}_q = 0$; otherwise, $S_i \leftarrow S_i - \frac{1}{q}(S_i \mathbf{1}_q)\mathbf{1}_q^{\mathrm{T}}$.

Canonical Variates $\boldsymbol{z}_1 = S_1^{\mathrm{T}}\boldsymbol{x}_1$, $\boldsymbol{z}_2 = S_2^{\mathrm{T}}\boldsymbol{x}_2$ defined in terms of Canonical Weight Vectors: $\boldsymbol{x}_1 \in \mathbb{R}^n$, $\boldsymbol{x}_2 \in \mathbb{R}^m$.

Canonical Correlation: $\rho(\boldsymbol{x}_1, \boldsymbol{x}_2) := \dfrac{\boldsymbol{z}_1^{\mathrm{T}}\boldsymbol{z}_2}{||\boldsymbol{z}_1||_2||\boldsymbol{z}_2||_2} = \dfrac{\boldsymbol{x}_1^{\mathrm{T}}C_{1,2}\boldsymbol{x}_2}{\sqrt{\boldsymbol{x}_1^{\mathrm{T}}C_{1,1}\boldsymbol{x}_1}\sqrt{\boldsymbol{x}_2^{\mathrm{T}}C_{2,2}\boldsymbol{x}_2}}$,
where $C_{i,j} = S_i S_j^{\mathrm{T}}$, (Cross-)Covariance.

CCA aims to find the pair of canonical weight vectors to maximize canonical correlation:

$$\max_{\boldsymbol{x}_1,\boldsymbol{x}_2} \rho(\boldsymbol{x}_1, \boldsymbol{x}_2). \tag{1}$$

CCA in General

Single-vector CCA (1) has been extended to Canonical Weight Matrices.

Canonical Weight Matrices: $X_1 \in \mathbb{R}^{n\times k}$, $X_2 \in \mathbb{R}^{m\times k}$.

Canonical Correlation: $f(X_1, X_2) = \dfrac{\operatorname{tr}(X_1^{\mathrm{T}} C_{1,2} X_2)}{\sqrt{\operatorname{tr}(X_1^{\mathrm{T}} C_{1,1} X_1)}\sqrt{\operatorname{tr}(X_2^{\mathrm{T}} C_{2,2} X_2)}}$,

CCA in general seeks to maximize canonical correlation:

$$\max_{X_1, X_2} f(X_1, X_2), \text{ s.t. } X_i^{\mathrm{T}} C_{i,i} X_i = I_k,\ i = 1, 2, \tag{2}$$

Closed form solution in terms of SVD for $C_{1,1}^{-1/2} C_{1,2} C_{2,2}^{-1/2}$. Collectively, traditional CCA or, simply, CCA is referred to either (1) or (2).

CCA is not suitable for: orthogonal projections are required such as for data visualization in an orthogonal coordinate system, because optimal X_1 and X_2 in (2) usually do not have orthonormal columns.

One can orthogonalize the columns of X_1 and X_2 as a post-processing step, but outcome is often suboptimal – less discriminatory.

CCA in General

Single-vector CCA (1) has been extended to Canonical Weight Matrices.

Canonical Weight Matrices: $X_1 \in \mathbb{R}^{n\times k}$, $X_2 \in \mathbb{R}^{m\times k}$.

Canonical Correlation: $f(X_1, X_2) = \frac{\operatorname{tr}(X_1^{\mathrm{T}} C_{1,2} X_2)}{\sqrt{\operatorname{tr}(X_1^{\mathrm{T}} C_{1,1} X_1)}\sqrt{\operatorname{tr}(X_2^{\mathrm{T}} C_{2,2} X_2)}}$,

CCA in general seeks to maximize canonical correlation:

$$\max_{X_1, X_2} f(X_1, X_2), \text{ s.t. } X_i^{\mathrm{T}} C_{i,i} X_i = I_k,\ i = 1, 2, \tag{2}$$

Closed form solution in terms of SVD for $C_{1,1}^{-1/2} C_{1,2} C_{2,2}^{-1/2}$. Collectively, traditional CCA or, simply, CCA is referred to either (1) or (2).

CCA is not suitable for: orthogonal projections are required such as for data visualization in an orthogonal coordinate system, because optimal X_1 and X_2 in (2) usually do not have orthonormal columns.

One can orthogonalize the columns of X_1 and X_2 as a post-processing step, but outcome is often suboptimal – less discriminatory.

CCA in General

Single-vector CCA (1) has been extended to Canonical Weight Matrices.

Canonical Weight Matrices: $X_1 \in \mathbb{R}^{n\times k}$, $X_2 \in \mathbb{R}^{m\times k}$.

Canonical Correlation: $f(X_1, X_2) = \dfrac{\operatorname{tr}(X_1^{\mathrm{T}} C_{1,2} X_2)}{\sqrt{\operatorname{tr}(X_1^{\mathrm{T}} C_{1,1} X_1)}\sqrt{\operatorname{tr}(X_2^{\mathrm{T}} C_{2,2} X_2)}}$,

CCA in general seeks to maximize canonical correlation:

$$\max_{X_1, X_2} f(X_1, X_2), \text{ s.t. } X_i^{\mathrm{T}} C_{i,i} X_i = I_k,\ i = 1, 2, \tag{2}$$

Closed form solution in terms of SVD for $C_{1,1}^{-1/2} C_{1,2} C_{2,2}^{-1/2}$. Collectively, traditional CCA or, simply, CCA is referred to either (1) or (2).

CCA is not suitable for: orthogonal projections are required such as for data visualization in an orthogonal coordinate system, because optimal X_1 and X_2 in (2) usually do not have orthonormal columns.

One can orthogonalize the columns of X_1 and X_2 as a post-processing step, but outcome is often suboptimal – less discriminatory.

CCA in General

Single-vector CCA (1) has been extended to Canonical Weight Matrices.

Canonical Weight Matrices: $X_1 \in \mathbb{R}^{n\times k}$, $X_2 \in \mathbb{R}^{m\times k}$.

Canonical Correlation: $f(X_1, X_2) = \dfrac{\mathrm{tr}(X_1^{\mathrm{T}} C_{1,2} X_2)}{\sqrt{\mathrm{tr}(X_1^{\mathrm{T}} C_{1,1} X_1)}\sqrt{\mathrm{tr}(X_2^{\mathrm{T}} C_{2,2} X_2)}}$,

CCA in general seeks to maximize canonical correlation:

$$\max_{X_1, X_2} f(X_1, X_2), \text{ s.t. } X_i^{\mathrm{T}} C_{i,i} X_i = I_k,\ i = 1, 2, \tag{2}$$

Closed form solution in terms of SVD for $C_{1,1}^{-1/2} C_{1,2} C_{2,2}^{-1/2}$. Collectively, traditional CCA or, simply, CCA is referred to either (1) or (2).

CCA is not suitable for: orthogonal projections are required such as for data visualization in an orthogonal coordinate system, because optimal X_1 and X_2 in (2) usually do not have orthonormal columns.

One can orthogonalize the columns of X_1 and X_2 as a post-processing step, but outcome is often suboptimal – less discriminatory.

Outline

1. Introduction to CCA
2. Orthogonal CCA (OCCA)
3. Submaximization Problem
4. Orthogonal Multiset CCA (OMCCA)
5. Applications
6. Summary

Orthogonal CCA (OCCA)

Orthogonal CCA (OCCA) seeks to maximize the correlation:

$$\max_{X_1 \in \mathbb{O}^{n\times k}, X_2 \in \mathbb{O}^{m\times k}} f(X_1, X_2) \tag{3}$$

directly over orthonormal matrices in $\mathbb{O}^{n\times k} = \{X \in \mathbb{R}^{n\times k} : X^{\mathrm{T}}X = I_k\}$.

Different from CCA, OCCA preserves the covariance of the original data S_1 and S_2 while correlation is maximized.

Can use generic optimization methods over the product of the Stiefel manifolds, and indeed applied.

Essentially, they are classical steepest ascent, trust-region, nonlinear CG methods over the Euclidean space extended to Riemannian manifolds. These methods don't recognize any special structure in f: less efficient, low accuracy, ...

P. Cunningham and Z. Ghahramani, Linear dimensionality reduction: Survey, insights, and generalizations, *J. Mach. Learning Res.*, 16(2015), 2859–2900.

P.-A. Absil, R. Mahony, and R. Sepulchre, Optimization Algorithms On Matrix Manifolds. Princeton University Press, 2008.

Orthogonal CCA (OCCA)

Orthogonal CCA (OCCA) seeks to maximize the correlation:

$$\max_{X_1 \in \mathbb{O}^{n\times k}, X_2 \in \mathbb{O}^{m\times k}} f(X_1, X_2) \tag{3}$$

directly over orthonormal matrices in $\mathbb{O}^{n\times k} = \{X \in \mathbb{R}^{n\times k} : X^{\mathrm{T}} X = I_k\}$.

Different from CCA, OCCA preserves the covariance of the original data S_1 and S_2 while correlation is maximized.

Can use generic optimization methods over the product of the Stiefel manifolds, and indeed applied.
Essentially, they are classical steepest ascent, trust-region, nonlinear CG methods over the Euclidean space extended to Riemannian manifolds. These methods don't recognize any special structure in f: less efficient, low accuracy, ...

P. Cunningham and Z. Ghahramani, Linear dimensionality reduction: Survey, insights, and generalizations, *J. Mach. Learning Res.*, 16(2015), 2859–2900.

P.-A. Absil, R. Mahony, and R. Sepulchre, Optimization Algorithms On Matrix Manifolds. Princeton University Press, 2008.

OCCA model (abstraction)

Data of both views centralized in advance: $S_1\mathbf{1}_q = 0$ and $S_2\mathbf{1}_q$. Define

$$A = S_1S_1^{\mathrm{T}} \in \mathbb{R}^{n\times n},\ B = S_2S_2^{\mathrm{T}} \in \mathbb{R}^{m\times m},\ C = S_1S_2^{\mathrm{T}} \in \mathbb{R}^{n\times m}.$$

OCCA: given an integer $1 \le k < \min\{m,n\}$ (usually $k \ll \min\{m,n\}$), solve

$$\max_{X\in\mathbb{O}^{n\times k}, Y\in\mathbb{O}^{m\times k}} f(X,Y) := \frac{\mathrm{tr}(X^{\mathrm{T}}CY)}{\sqrt{\mathrm{tr}(X^{\mathrm{T}}AX)}\sqrt{\mathrm{tr}(Y^{\mathrm{T}}BY)}}. \tag{4}$$

Propose to maximize $f(X,Y)$ alternatingly with respective to X and Y.

Although the framework of the proposed numerical scheme is rather natural, novelty lies in the way how its core sub-maximization problems are solved.

Algorithm framework for 2-view OCCA

Algorithm 1. Alternating optimization scheme for (4)

Input: $\{X^{(0)}, Y^{(0)}\}$ with $X^{(0)} \in \mathbb{O}^{n\times k}, Y^{(0)} \in \mathbb{O}^{m\times k}$.

Output: a solution $\{X^{(\nu)}, Y^{(\nu)}\}$ to (4).

1: **for** $\nu = 1, 2, \ldots$ until convergence **do**

2: solve $X^{(\nu)} \in \arg\max_{X\in\mathbb{O}^{n\times k}} f(X, Y^{(\nu-1)})$;

3: solve $Y^{(\nu)} \in \arg\max_{Y\in\mathbb{O}^{m\times k}} f(X^{(\nu)}, Y)$;

4: compute SVD: $(X^{(\nu)})^{\mathrm{T}} C Y^{(\nu)} = \widetilde{U}\widetilde{\Sigma}\widetilde{V}^{\mathrm{T}}$, and set $X^{(\nu)} \leftarrow X^{(\nu)}\widetilde{U}$ and $Y^{(\nu)} \leftarrow Y^{(\nu)}\widetilde{V}$;

5: **end for**

6: **return** $\{X^{(\nu)}, Y^{(\nu)}\}$ as a numerical solution to (4).

The role of line 4 in Algorithm 1 is to make sure $X^{(\nu)}$ and $Y^{(\nu)}$ are best-aligned. In particular, $\mathrm{tr}({X^{(\nu)}}^{\mathrm{T}} C Y^{(\nu)}) > 0$ and maximized within the column spaces of $X^{(\nu)}$ and $Y^{(\nu)}$ at lines 2 & 3.

Algorithm framework for 2-view OCCA

Algorithm 1. Alternating optimization scheme for (4)

Input: $\{X^{(0)}, Y^{(0)}\}$ with $X^{(0)} \in \mathbb{O}^{n\times k}, Y^{(0)} \in \mathbb{O}^{m\times k}$.
Output: a solution $\{X^{(\nu)}, Y^{(\nu)}\}$ to (4).

1: **for** $\nu = 1, 2, \ldots$ until convergence **do**
2: solve $X^{(\nu)} \in \arg\max_{X\in\mathbb{O}^{n\times k}} f(X, Y^{(\nu-1)})$;
3: solve $Y^{(\nu)} \in \arg\max_{Y\in\mathbb{O}^{m\times k}} f(X^{(\nu)}, Y)$;
4: compute SVD: $(X^{(\nu)})^{\mathrm{T}} C Y^{(\nu)} = \widetilde{U}\widetilde{\Sigma}\widetilde{V}^{\mathrm{T}}$, and set $X^{(\nu)} \leftarrow X^{(\nu)}\widetilde{U}$ and $Y^{(\nu)} \leftarrow Y^{(\nu)}\widetilde{V}$;
5: **end for**
6: **return** $\{X^{(\nu)}, Y^{(\nu)}\}$ as a numerical solution to (4).

The role of line 4 in Algorithm 1 is to make sure $X^{(\nu)}$ and $Y^{(\nu)}$ are best-aligned. In particular, $\mathrm{tr}({X^{(\nu)}}^{\mathrm{T}} C Y^{(\nu)}) > 0$ and maximized within the column spaces of $X^{(\nu)}$ and $Y^{(\nu)}$ at lines 2 & 3.

Convergence

Convergence Theorem

Let $\{X_{\rm opt}, Y_{\rm opt}\}$ be the optimal solution to (4) and $\{X^{(\nu)}, Y^{(\nu)}\}$ be the νth approximation of Algorithm 1. Then

(i) $X_{\rm opt}^{\rm T} C Y_{\rm opt}$ is symmetric and positive semidefinite.

(ii) $(X^{(\nu)})^{\rm T} C Y^{(\nu)}$ is symmetric and positive semidefinite for $\nu \geq 1$, and thus for any limit pair $\{X_*, Y_*\}$ of $\{X^{(\nu)}, Y^{(\nu)}\}_{\nu=1}^{\infty}$, $X_*^{\rm T} C Y_*$ is symmetric and positive semidefinite.

(iii) The sequence $\{f(X^{(\nu)}, Y^{(\nu)})\}_{\nu=1}^{\infty}$ is monotonically increasing and converges.

Efficiency of Algorithm 1 relies heavily on solving the sub-maximization problems at Lines 2 and 3.

Abstractly, they are of the following type

$$\max_{X \in \mathbb{O}^{n \times k}} \eta(X) \quad \text{with } \eta(X) := \frac{\operatorname{tr}(X^{\rm T} D)}{\sqrt{\operatorname{tr}(X^{\rm T} A X)}}, \tag{5}$$

where $0 \neq D \in \mathbb{R}^{n \times k}$ and $A \succ 0$.

Convergence

Convergence Theorem

Let $\{X_{\rm opt}, Y_{\rm opt}\}$ be the optimal solution to (4) and $\{X^{(\nu)}, Y^{(\nu)}\}$ be the νth approximation of Algorithm 1. Then

(i) $X_{\rm opt}^{\rm T} C Y_{\rm opt}$ is symmetric and positive semidefinite.

(ii) $(X^{(\nu)})^{\rm T} C Y^{(\nu)}$ is symmetric and positive semidefinite for $\nu \geq 1$, and thus for any limit pair $\{X_*, Y_*\}$ of $\{X^{(\nu)}, Y^{(\nu)}\}_{\nu=1}^{\infty}$, $X_*^{\rm T} C Y_*$ is symmetric and positive semidefinite.

(iii) The sequence $\{f(X^{(\nu)}, Y^{(\nu)})\}_{\nu=1}^{\infty}$ is monotonically increasing and converges.

Efficiency of Algorithm 1 relies heavily on solving the sub-maximization problems at Lines 2 and 3.
Abstractly, they are of the following type

$$\max_{X\in\mathbb{O}^{n\times k}} \eta(X) \quad \text{with } \eta(X) := \frac{\operatorname{tr}(X^{\rm T} D)}{\sqrt{\operatorname{tr}(X^{\rm T} A X)}}, \tag{5}$$

where $0 \neq D \in \mathbb{R}^{n\times k}$ and $A \succ 0$.

Local but non-global maximizers

Problem (5) may admit local but non-global maximizers.

Example. Consider the case with $n = 5,\ k = 2$,

$$A = \begin{bmatrix} 4 & 0 & -5 & -5 & 1 \\ 0 & 2 & 1 & -1 & 1 \\ -5 & 1 & 9 & 5 & 1 \\ -5 & -1 & 5 & 18 & 4 \\ -1 & 1 & 1 & 4 & 2 \end{bmatrix} \quad \text{and} \quad D = \begin{bmatrix} -1 & 1 \\ 0 & 0 \\ 0 & 2 \\ 0 & 0 \\ 1 & 0 \end{bmatrix}.$$

By calling MATLAB's `fmincon`, we find two (numerical) local maximizers:

$$X_+ = \begin{bmatrix} -0.358041496119094 & 0.770164268103322 \\ -0.453284095949462 & -0.326431512218038 \\ -0.091335437376569 & 0.497561512998402 \\ -0.269574025133855 & 0.008593213179154 \\ 0.765066989399257 & 0.229451880441015 \end{bmatrix},$$

Local but non-global maximizers (cont'd)

$$X_* = \begin{bmatrix} -0.506648923972689 & 0.664385053189626 \\ 0.619602876311725 & 0.312889763321350 \\ -0.337893503149209 & 0.384494340924914 \\ 0.103073503143856 & 0.210902556071053 \\ -0.484358314662567 & -0.518050876600301 \end{bmatrix}.$$

$$\eta(X_+) \approx 1.517 < \eta(X_*) \approx 3.187.$$

We argue that they are (numerical) local maximizers:

- $\|\operatorname{grad} \eta(X)\|_F \le 10^{-6}$ (on $\mathbb{O}^{n\times k}$) for $X = X_+$ or X_*;
- Second order sufficient condition: verified at 10^7 random tangent "vectors" in $\mathcal{T}_X\mathbb{O}^{n\times k}$ for both X_+ and X_*.

This example numerically shows (5) in general admits local but non-global maximizers.

Outline

SCF Iteration

$$\max_{X \in \mathbb{O}^{n\times k}} \eta(X) \quad \text{with } \eta(X) := \frac{\operatorname{tr}(X^{\mathrm{T}}D)}{\sqrt{\operatorname{tr}(X^{\mathrm{T}}AX)}}. \tag{5}$$

Gradient:

$$\operatorname{grad}\eta(X) = \Pi_X\left(\frac{\partial \eta(X)}{\partial X}\right) \in \mathcal{T}_X\mathbb{O}^{n\times k}, \tag{6}$$

where $\Pi_X(Z) = Z - X\,\operatorname{sym}(X^{\mathrm{T}}Z)$ for $Z \in \mathbb{R}^{n\times k}$. By calculations,

$$\frac{\partial \eta(X)}{\partial X} = \frac{1}{\sqrt{\operatorname{tr}(X^{\mathrm{T}}AX)}}\,D - \frac{\operatorname{tr}(X^{\mathrm{T}}D}{[\operatorname{tr}(X^{\mathrm{T}}AX)]^{3/2}}\,AX,$$

$$\frac{[\operatorname{tr}(X^{\mathrm{T}}AX)]^{3/2}}{\operatorname{tr}(X^{\mathrm{T}}D)}\operatorname{grad}\eta(X) = [\xi(X)D - AX] - X\Lambda(X) \in \mathbb{R}^{n\times k},$$

where $\xi(X) = \frac{\operatorname{tr}(X^{\mathrm{T}}AX)}{\operatorname{tr}(X^{\mathrm{T}}D)}$,

$$\Lambda(X) = \xi(X)\frac{1}{2}\left[X^{\mathrm{T}}D + D^{\mathrm{T}}X\right] - X^{\mathrm{T}}AX \in \mathbb{R}^{k\times k}. \tag{7}$$

SCF Iteration

$$\max_{X \in \mathbb{O}^{n\times k}} \eta(X) \quad \text{with } \eta(X) := \frac{\operatorname{tr}(X^{\mathrm{T}} D)}{\sqrt{\operatorname{tr}(X^{\mathrm{T}} A X)}}. \tag{5}$$

Gradient:

$$\operatorname{grad} \eta(X) = \Pi_X \left(\frac{\partial \eta(X)}{\partial X} \right) \in \mathcal{T}_X \mathbb{O}^{n\times k}, \tag{6}$$

where $\Pi_X(Z) = Z - X \operatorname{sym}(X^{\mathrm{T}} Z)$ for $Z \in \mathbb{R}^{n\times k}$. By calculations,

$$\frac{\partial \eta(X)}{\partial X} = \frac{1}{\sqrt{\operatorname{tr}(X^{\mathrm{T}} A X)}} D - \frac{\operatorname{tr}(X^{\mathrm{T}} D}{[\operatorname{tr}(X^{\mathrm{T}} A X)]^{3/2}} AX,$$

$$\frac{[\operatorname{tr}(X^{\mathrm{T}} A X)]^{3/2}}{\operatorname{tr}(X^{\mathrm{T}} D)} \operatorname{grad} \eta(X) = [\xi(X) D - AX] - X\Lambda(X) \in \mathbb{R}^{n\times k},$$

where $\xi(X) = \frac{\operatorname{tr}(X^{\mathrm{T}} A X)}{\operatorname{tr}(X^{\mathrm{T}} D)}$,

$$\Lambda(X) = \xi(X) \frac{1}{2} \left[X^{\mathrm{T}} D + D^{\mathrm{T}} X \right] - X^{\mathrm{T}} A X \in \mathbb{R}^{k\times k}. \tag{7}$$

SCF Iteration

$$\max_{X \in \mathbb{O}^{n\times k}} \eta(X) \quad \text{with } \eta(X) := \frac{\operatorname{tr}(X^{\mathrm{T}}D)}{\sqrt{\operatorname{tr}(X^{\mathrm{T}}AX)}}. \tag{5}$$

Gradient:

$$\operatorname{grad}\eta(X) = \Pi_X\left(\frac{\partial\eta(X)}{\partial X}\right) \in \mathcal{T}_X\mathbb{O}^{n\times k}, \tag{6}$$

where $\Pi_X(Z) = Z - X\,\operatorname{sym}(X^{\mathrm{T}}Z)$ for $Z \in \mathbb{R}^{n\times k}$. By calculations,

$$\frac{\partial\eta(X)}{\partial X} = \frac{1}{\sqrt{\operatorname{tr}(X^{\mathrm{T}}AX)}}\,D - \frac{\operatorname{tr}(X^{\mathrm{T}}D}{[\operatorname{tr}(X^{\mathrm{T}}AX)]^{3/2}}\,AX,$$

$$\frac{[\operatorname{tr}(X^{\mathrm{T}}AX)]^{3/2}}{\operatorname{tr}(X^{\mathrm{T}}D)}\operatorname{grad}\eta(X) = [\xi(X)D - AX] - X\Lambda(X) \in \mathbb{R}^{n\times k},$$

where $\xi(X) = \frac{\operatorname{tr}(X^{\mathrm{T}}AX)}{\operatorname{tr}(X^{\mathrm{T}}D)}$,

$$\Lambda(X) = \xi(X)\frac{1}{2}\left[X^{\mathrm{T}}D + D^{\mathrm{T}}X\right] - X^{\mathrm{T}}AX \in \mathbb{R}^{k\times k}. \tag{7}$$

First Order KKT Condition

Lemma 1. First Order KKT Condition

If X is a maximizer of (5), then $X^{\mathrm{T}}D = D^{\mathrm{T}}X$ and

$$\xi(X)D - AX = X\Lambda(X). \tag{8}$$

Condition (8) is a type of nonlinear Sylvester equation but with the orthogonality constraint $X^{\mathrm{T}}X = I_k$. Not clear how to solve.

Turn it into nonlinear eigenvalue problem (NEPv):

$$E(X)X = X\widehat{\Lambda}(X), \tag{9}$$

where $\widehat{\Lambda}(X)^{\mathrm{T}} = \widehat{\Lambda}(X)$ and

$$E(X) := \xi(X)(DX^{\mathrm{T}} + XD^{\mathrm{T}}) - A.$$

Evidently, $E(X)$ is always symmetric. It is implied $\widehat{\Lambda}(X) = X^{\mathrm{T}}E(X)X \in \mathbb{R}^{k\times k}$.

Lemma 2. Equivalent KKT Condition

Suppose $X \in \mathbb{O}^{n\times k}$. Then X satisfies (8) if and only if X is an eigenbasis matrix of $E(X)$, i.e., X satisfies (9).

First Order KKT Condition

Lemma 1. First Order KKT Condition

If X is a maximizer of (5), then $X^{\mathrm{T}}D = D^{\mathrm{T}}X$ and

$$\xi(X)D - AX = X\Lambda(X). \tag{8}$$

Condition (8) is a type of nonlinear Sylvester equation but with the orthogonality constraint $X^{\mathrm{T}}X = I_k$. Not clear how to solve.

Turn it into nonlinear eigenvalue problem (NEPv):

$$E(X)X = X\widehat{\Lambda}(X), \tag{9}$$

where $\widehat{\Lambda}(X)^{\mathrm{T}} = \widehat{\Lambda}(X)$ and

$$E(X) := \xi(X)(DX^{\mathrm{T}} + XD^{\mathrm{T}}) - A.$$

Evidently, $E(X)$ is always symmetric. It is implied $\widehat{\Lambda}(X) = X^{\mathrm{T}}E(X)X \in \mathbb{R}^{k\times k}$.

Lemma 2. Equivalent KKT Condition

Suppose $X \in \mathbb{O}^{n\times k}$. Then X satisfies (8) if and only if X is an eigenbasis matrix of $E(X)$, i.e., X satisfies (9).

First Order KKT Condition

Lemma 1. First Order KKT Condition

If X is a maximizer of (5), then $X^{\mathrm{T}}D = D^{\mathrm{T}}X$ and

$$\xi(X)D - AX = X\Lambda(X). \tag{8}$$

Condition (8) is a type of nonlinear Sylvester equation but with the orthogonality constraint $X^{\mathrm{T}}X = I_k$. Not clear how to solve.

Turn it into nonlinear eigenvalue problem (NEPv):

$$E(X)X = X\widehat{\Lambda}(X), \tag{9}$$

where $\widehat{\Lambda}(X)^{\mathrm{T}} = \widehat{\Lambda}(X)$ and

$$E(X) := \xi(X)(DX^{\mathrm{T}} + XD^{\mathrm{T}}) - A.$$

Evidently, $E(X)$ is always symmetric. It is implied $\widehat{\Lambda}(X) = X^{\mathrm{T}}E(X)X \in \mathbb{R}^{k\times k}$.

Lemma 2. Equivalent KKT Condition

Suppose $X \in \mathbb{O}^{n\times k}$. Then X satisfies (8) if and only if X is an eigenbasis matrix of $E(X)$, i.e., X satisfies (9).

First Order KKT Condition

Lemma 1. First Order KKT Condition

If X is a maximizer of (5), then $X^{\mathrm{T}}D = D^{\mathrm{T}}X$ and

$$\xi(X)D - AX = X\Lambda(X). \tag{8}$$

Condition (8) is a type of nonlinear Sylvester equation but with the orthogonality constraint $X^{\mathrm{T}}X = I_k$. Not clear how to solve.

Turn it into nonlinear eigenvalue problem (NEPv):

$$E(X)X = X\widehat{\Lambda}(X), \tag{9}$$

where $\widehat{\Lambda}(X)^{\mathrm{T}} = \widehat{\Lambda}(X)$ and

$$E(X) := \xi(X)(DX^{\mathrm{T}} + XD^{\mathrm{T}}) - A.$$

Evidently, $E(X)$ is always symmetric. It is implied $\widehat{\Lambda}(X) = X^{\mathrm{T}}E(X)X \in \mathbb{R}^{k\times k}$.

Lemma 2. Equivalent KKT Condition

Suppose $X \in \mathbb{O}^{n\times k}$. Then X satisfies (8) if and only if X is an eigenbasis matrix of $E(X)$, i.e., X satisfies (9).

A self-consistent-field (SCF) iteration

Necessary condition of a global maximizer for (5)

If $X_{\rm opt}$ is a global maximizer to (5), then $X_{\rm opt}$ is an orthonormal eigenbasis matrix associated with the k largest eigenvalues of $E(X_{\rm opt})$.

Algorithm 2. An SCF iteration for solving (5)

Input: $X_{(0)} \in \mathbb{O}^{n\times k}$;
Output: approximate maximizer X to (5).

for $\nu = 1, 2, \ldots$ until convergence **do**
    construct $E_{(\nu)} = E(X_{(\nu-1)})$ as in (9);
    compute an orthonormal eigenbasis matrix $X_{(\nu)}$ associated with the k largest eigenvalues of $E_{(\nu)}$;
    compute SVD: $X_{(\nu)}^{\rm T} D = U\Sigma V^{\rm T}$ and update $X_{(\nu)} \leftarrow X_{(\nu)} U V^{\rm T}$;
end for
return. the last $X_{(\nu)}$ as a numerical maximizer of (5).

A self-consistent-field (SCF) iteration

Necessary condition of a global maximizer for (5)

If $X_{\rm opt}$ is a global maximizer to (5), then $X_{\rm opt}$ is an orthonormal eigenbasis matrix associated with the k largest eigenvalues of $E(X_{\rm opt})$.

Algorithm 2. An SCF iteration for solving (5)

Input: $X_{(0)} \in \mathbb{O}^{n\times k}$;
Output: approximate maximizer X to (5).

for $\nu = 1, 2, \ldots$ until convergence **do**
    construct $E_{(\nu)} = E(X_{(\nu-1)})$ as in (9);
    compute an orthonormal eigenbasis matrix $X_{(\nu)}$ associated with the k largest eigenvalues of $E_{(\nu)}$;
    compute SVD: $X_{(\nu)}^{\rm T} D = U\Sigma V^{\rm T}$ and update $X_{(\nu)} \leftarrow X_{(\nu)} U V^{\rm T}$;
end for
return. the last $X_{(\nu)}$ as a numerical maximizer of (5).

Comments on Algorithm 2 (SCF)

Use full eigen-decomposition of E for small n (e.g., ≤ 200); use an iterative method for large n such as LOBPCG, Inverse-free (Golub+Ye), LOBPECG, ...

The SCF iteration stops if

$$\frac{\operatorname{tr}(X^{\mathrm{T}}D)}{[\operatorname{tr}(X^{\mathrm{T}}AX)]^{3/2}}\frac{\|\operatorname{grad}\eta(G_{(\nu)})\|_1}{\|A\|_1+\|D\|_1}\leq\epsilon_{\mathrm{scf}}$$

with, e.g., $\epsilon_{\mathrm{scf}}=10^{-5}$.

Comments on Algorithm 2 (SCF)

Use full eigen-decomposition of E for small n (e.g., ≤ 200); use an iterative method for large n such as LOBPCG, Inverse-free (Golub+Ye), LOBPECG, ...

The SCF iteration stops if

$$\frac{\operatorname{tr}(X^{\mathrm{T}}D)}{[\operatorname{tr}(X^{\mathrm{T}}AX)]^{3/2}} \frac{\|\operatorname{grad}\eta(G_{(\nu)})\|_1}{\|A\|_1 + \|D\|_1} \leq \epsilon_{\mathrm{scf}}$$

with, e.g., $\epsilon_{\mathrm{scf}} = 10^{-5}$.

Convergence

Weak Convergence Theorem

Let $\{X_{(\nu)}\}$ be generated by the SCF iteration (Algorithm 2).

(i) For each $\nu \geq 1$, $D^{\mathrm{T}} X_{(\nu)} \succeq 0$ and $\mathrm{tr}(X_{(\nu)}^{\mathrm{T}} D) = \sum_{j=1}^{k} \sigma_j(X_{(\nu)}^{\mathrm{T}} D)$;

(ii) $\{\eta(X_{(\nu)})\}$ is monotonically increasing and convergent;

(iii) If
$$\mathrm{tr}(X_{(\nu)}^{\mathrm{T}} E(X_{(\nu-1)}) X_{(\nu)}) \geq \mathrm{tr}(X_{(\nu-1)}^{\mathrm{T}} E(X_{(\nu-1)}) X_{(\nu-1)}), \tag{10}$$
then $\eta(X_{(\nu-1)}) \leq \eta(X_{(\nu)})$; If (10) is strict, then also $\eta(X_{(\nu-1)}) < \eta(X_{(\nu)})$;

(iv) $\{X_{(\nu)}\}$ has a convergent subsequence $\{X_{(\nu)}\}_{\nu \in \mathcal{I}}$;

(v) Let $\{X_{(\nu)}\}_{\nu \in \mathcal{I}}$ be any convergent subsequence of $\{X_{(\nu)}\}$ with the accumulation point X_* satisfying
$$\zeta = \lambda_k(E(X_*)) - \lambda_{k+1}(E(X_*)) > 0. \tag{11}$$
Then X_* satisfies the first order optimality condition and also the necessary condition for a global minimizer.

Convergence

Weak Convergence Theorem

Let $\{X_{(\nu)}\}$ be generated by the SCF iteration (Algorithm 2).

(i) For each $\nu \geq 1$, $D^{\mathrm{T}} X_{(\nu)} \succeq 0$ and $\mathrm{tr}(X_{(\nu)}^{\mathrm{T}} D) = \sum_{j=1}^{k} \sigma_j(X_{(\nu)}^{\mathrm{T}} D)$;

(ii) $\{\eta(X_{(\nu)})\}$ is monotonically increasing and convergent;

(iii) If
$$\mathrm{tr}(X_{(\nu)}^{\mathrm{T}} E(X_{(\nu-1)}) X_{(\nu)}) \geq \mathrm{tr}(X_{(\nu-1)}^{\mathrm{T}} E(X_{(\nu-1)}) X_{(\nu-1)}), \tag{10}$$
then $\eta(X_{(\nu-1)}) \leq \eta(X_{(\nu)})$; If (10) is strict, then also $\eta(X_{(\nu-1)}) < \eta(X_{(\nu)})$;

(iv) $\{X_{(\nu)}\}$ has a convergent subsequence $\{X_{(\nu)}\}_{\nu \in \mathcal{I}}$;

(v) Let $\{X_{(\nu)}\}_{\nu \in \mathcal{I}}$ be any convergent subsequence of $\{X_{(\nu)}\}$ with the accumulation point X_* satisfying
$$\zeta = \lambda_k(E(X_*)) - \lambda_{k+1}(E(X_*)) > 0. \tag{11}$$
Then X_* satisfies the first order optimality condition and also the necessary condition for a global minimizer.

Convergence

Weak Convergence Theorem

Let $\{X_{(\nu)}\}$ be generated by the SCF iteration (Algorithm 2).

(i) For each $\nu \geq 1$, $D^{\mathrm{T}} X_{(\nu)} \succeq 0$ and $\mathrm{tr}(X_{(\nu)}^{\mathrm{T}} D) = \sum_{j=1}^{k} \sigma_j(X_{(\nu)}^{\mathrm{T}} D)$;

(ii) $\{\eta(X_{(\nu)})\}$ is monotonically increasing and convergent;

(iii) If

$$\mathrm{tr}(X_{(\nu)}^{\mathrm{T}} E(X_{(\nu-1)}) X_{(\nu)}) \geq \mathrm{tr}(X_{(\nu-1)}^{\mathrm{T}} E(X_{(\nu-1)}) X_{(\nu-1)}), \tag{10}$$

then $\eta(X_{(\nu-1)}) \leq \eta(X_{(\nu)})$; If (10) is strict, then also $\eta(X_{(\nu-1)}) < \eta(X_{(\nu)})$;

(iv) $\{X_{(\nu)}\}$ has a convergent subsequence $\{X_{(\nu)}\}_{\nu \in \mathcal{I}}$;

(v) Let $\{X_{(\nu)}\}_{\nu \in \mathcal{I}}$ be any convergent subsequence of $\{X_{(\nu)}\}$ with the accumulation point X_* satisfying

$$\zeta = \lambda_k(E(X_*)) - \lambda_{k+1}(E(X_*)) > 0. \tag{11}$$

Then X_* satisfies the first order optimality condition and also the necessary condition for a global minimizer.

Convergence (cont'd)

Strong Convergence Theorem

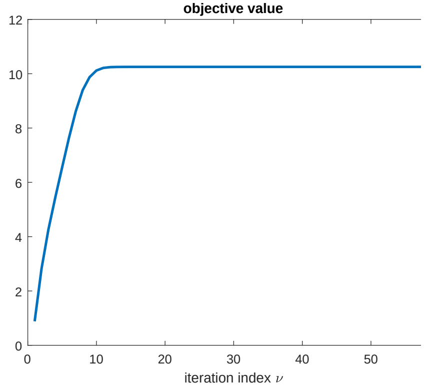
Let $\{X_{(\nu)}\}$ be generated by the SCF iteration (Algorithm 2), and let X_* be an accumulation point of $\{X_{(\nu)}\}$. Suppose that $\mathcal{R}(X_*)$ is an isolated accumulation point of $\{\mathcal{R}(X_{(\nu)})\}_{\nu=0}^{\infty}$.

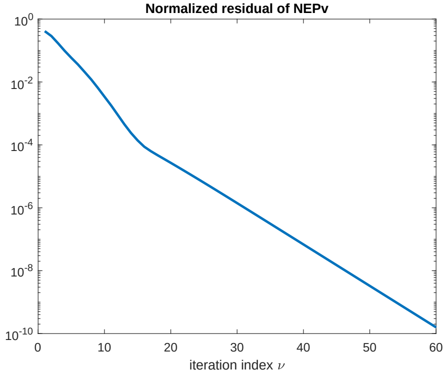
(i) $\{\mathcal{R}(X_{(\nu)})\}_{\nu=0}^{\infty}$ converges to $\mathcal{R}(X_*)$.

(ii) If also $\operatorname{rank}(X_*^{\mathrm{T}}D) = k$, then $\{X_{(\nu)}\}_{\nu=0}^{\infty}$ converges to X_*.

A random example for Algorithm 2

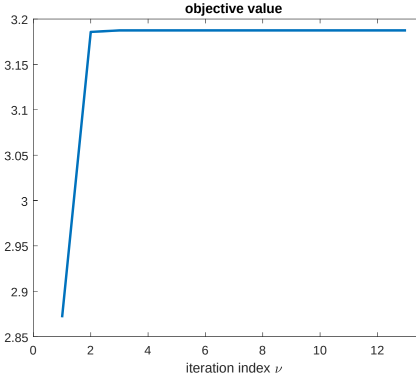
$$\eta(X) = \frac{\operatorname{tr}(X^{\mathrm{T}}D)}{\sqrt{\operatorname{tr}(X^{\mathrm{T}}AX)}}, \qquad \frac{\|E_{(\nu)}X_{(\nu)} - X_{(\nu)}\widehat{\Lambda}_{(\nu)}\|_2}{\|E_{(\nu)}\|_2}.$$

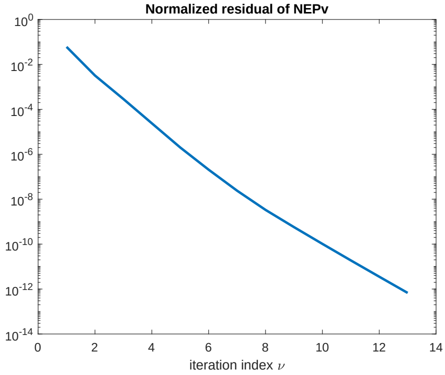




Earlier example with local minimizers

$$\eta(X) = \frac{\mathrm{tr}(X^{\mathrm{T}}D)}{\sqrt{\mathrm{tr}(X^{\mathrm{T}}AX)}}, \qquad \frac{\|E_{(\nu)}X_{(\nu)} - X_{(\nu)}\widehat{\Lambda}_{(\nu)}\|_2}{\|E_{(\nu)}\|_2}.$$





Digression: Compared to LDA

Fisher's linear discriminant analysis (LDA): given symmetric $B, A \in \mathbb{R}^{n\times n}$ and $A \succ 0$, solve

$$\max_{X\in\mathbb{O}^{n\times k}} \frac{\operatorname{tr}(X^{\mathrm{T}}BX)}{\operatorname{tr}(X^{\mathrm{T}}AX)}. \tag{12}$$

Equivalent to

$$H(X)X := \left(B - \frac{\operatorname{tr}(X^{\mathrm{T}}BX)}{\operatorname{tr}(X^{\mathrm{T}}AX)}A\right) = X(X^{\mathrm{T}}H(X)X) =: X\Lambda(X). \tag{13}$$

SCF:

$$H(X_{\nu-1})X_\nu = X_\nu\Lambda(X_\nu) \quad \text{for } \nu = 1, 2, \ldots \tag{14}$$

- (12) has global maximizers, but no local maximizer;
- X is a global maximizer if and only if the eigenvalues of $\Lambda(X)$ consist of largest k eigenvalues of $H(X)$;
- SCF (14) always converges and converges quadratically!

Digression: Compared to LDA

Fisher's linear discriminant analysis (LDA): given symmetric $B, A \in \mathbb{R}^{n\times n}$ and $A \succ 0$, solve

$$\max_{X\in\mathbb{O}^{n\times k}} \frac{\operatorname{tr}(X^{\mathrm{T}}BX)}{\operatorname{tr}(X^{\mathrm{T}}AX)}. \tag{12}$$

Equivalent to

$$H(X)X := \left(B - \frac{\operatorname{tr}(X^{\mathrm{T}}BX)}{\operatorname{tr}(X^{\mathrm{T}}AX)}A\right) = X(X^{\mathrm{T}}H(X)X) =: X\Lambda(X). \tag{13}$$

SCF:

$$H(X_{\nu-1})X_\nu = X_\nu\Lambda(X_\nu) \quad \text{for } \nu = 1, 2, \ldots \tag{14}$$

- (12) has global maximizers, but no local maximizer;
- X is a global maximizer if and only if the eigenvalues of $\Lambda(X)$ consist of largest k eigenvalues of $H(X)$;
- SCF (14) always converges and converges quadratically!

Digression: Compared to LDA

Fisher's linear discriminant analysis (LDA): given symmetric $B, A \in \mathbb{R}^{n \times n}$ and $A \succ 0$, solve

$$\max_{X \in \mathbb{O}^{n \times k}} \frac{\operatorname{tr}(X^{\mathrm{T}} B X)}{\operatorname{tr}(X^{\mathrm{T}} A X)}. \tag{12}$$

Equivalent to

$$H(X)X := \left(B - \frac{\operatorname{tr}(X^{\mathrm{T}} B X)}{\operatorname{tr}(X^{\mathrm{T}} A X)} A\right) = X(X^{\mathrm{T}} H(X) X) =: X\Lambda(X). \tag{13}$$

SCF:

$$H(X_{\nu-1})X_\nu = X_\nu \Lambda(X_\nu) \quad \text{for } \nu = 1, 2, \ldots \tag{14}$$

- (12) has global maximizers, but no local maximizer;
- X is a global maximizer if and only if the eigenvalues of $\Lambda(X)$ consist of largest k eigenvalues of $H(X)$;
- SCF (14) always converges and converges quadratically!

Digression: Compared to LDA

Fisher's linear discriminant analysis (LDA): given symmetric $B, A \in \mathbb{R}^{n\times n}$ and $A \succ 0$, solve

$$\max_{X\in\mathbb{O}^{n\times k}} \frac{\operatorname{tr}(X^{\mathrm{T}}BX)}{\operatorname{tr}(X^{\mathrm{T}}AX)}. \tag{12}$$

Equivalent to

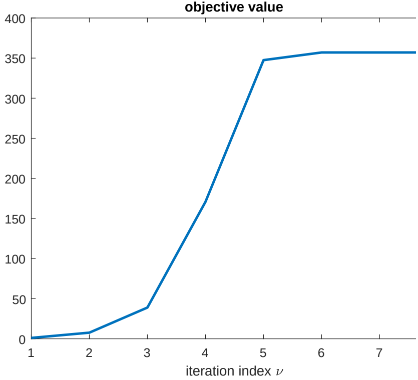
$$H(X)X := \left(B - \frac{\operatorname{tr}(X^{\mathrm{T}}BX)}{\operatorname{tr}(X^{\mathrm{T}}AX)}A\right) = X(X^{\mathrm{T}}H(X)X) =: X\Lambda(X). \tag{13}$$

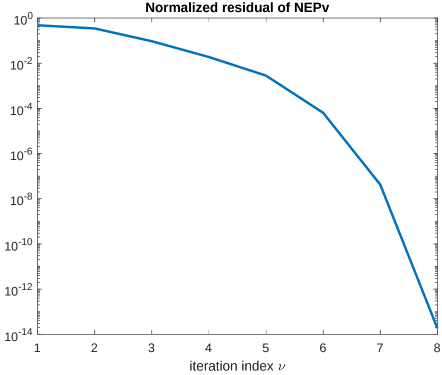
SCF:

$$H(X_{\nu-1})X_\nu = X_\nu\Lambda(X_\nu) \quad \text{for } \nu = 1, 2, \ldots \tag{14}$$

- (12) has global maximizers, but no local maximizer;
- X is a global maximizer if and only if the eigenvalues of $\Lambda(X)$ consist of largest k eigenvalues of $H(X)$;
- SCF (14) always converges and converges quadratically!

Digression: SCF for LDA





- L.-H. Zhang, L.-Z. Liao, and M. K. Ng. Fast algorithms for the generalized Foley-Sammon discriminant analysis. *SIAM J. Matrix Anal. Appl.*, 31(4):1584–1605, 2010.
- L.-H. Zhang, W. Yang, and L.-Z. Liao. A note on the trace quotient problem. *Opt. Lett.*, 8:1637–1645, 2014.
- Y. Cai, L.-H. Zhang, Z. Bai, and R.-C. Li, On an eigenvector-dependent nonlinear eigenvalue problem. *SIAM J. Matrix Anal. Appl.*, 39:1360–1382, 2018.

Outline

Single-Vector Multiset CCA (MCCA)

Multiset CCA (MCCA) is to analyze linear relationships among more than two canonical variates, as a generalization of traditional two-view CCA.

Widely used model: Given ℓ datasets in the form of matrices

$$S_i \in \mathbb{R}^{n_i \times q} \quad \text{for } i = 1, 2, \ldots, \ell, \tag{15}$$

where n_i is the number of features in the ith view, and q is the number of sample data points.

Assume all S_i are centered, i.e., $S_i \mathbf{1}_q = 0$ for all i.

(Cross-)Covariance: $C_{i,j} = S_i S_j^{\mathrm{T}}$ for $i,\, j = 1, \ldots, \ell$. MCCA seeks to solve

$$\max_{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_\ell} \sum_{i,\, j=1}^{\ell} \boldsymbol{x}_i^{\mathrm{T}} C_{i,j} \boldsymbol{x}_j \quad \text{subject to} \begin{cases} \text{either} & \sum_{i=1}^{\ell} \boldsymbol{x}_i^{\mathrm{T}} C_{i,i} \boldsymbol{x}_i = 1, \\ \text{or} & \boldsymbol{x}_i^{\mathrm{T}} C_{i,i} \boldsymbol{x}_i = 1,\, i = 1, \ldots, \ell. \end{cases}$$

Single-Vector Multiset CCA (MCCA)

Multiset CCA (MCCA) is to analyze linear relationships among more than two canonical variates, as a generalization of traditional two-view CCA.

Widely used model: Given ℓ datasets in the form of matrices

$$S_i \in \mathbb{R}^{n_i \times q} \quad \text{for } i = 1, 2, \ldots, \ell, \tag{15}$$

where n_i is the number of features in the ith view, and q is the number of sample data points.

Assume all S_i are centered, i.e., $S_i \mathbf{1}_q = 0$ for all i.

(Cross-)Covariance: $C_{i,j} = S_i S_j^{\mathrm{T}}$ for i, $j = 1, \ldots, \ell$. MCCA seeks to solve

$$\max_{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_\ell} \sum_{i,\, j=1}^{\ell} \boldsymbol{x}_i^{\mathrm{T}} C_{i,j} \boldsymbol{x}_j \quad \text{subject to} \begin{cases} \text{either} & \sum_{i=1}^{\ell} \boldsymbol{x}_i^{\mathrm{T}} C_{i,i} \boldsymbol{x}_i = 1, \\ \text{or} & \boldsymbol{x}_i^{\mathrm{T}} C_{i,i} \boldsymbol{x}_i = 1,\ i = 1, \ldots, \ell. \end{cases}$$

Single-Vector Multiset CCA (MCCA)

Multiset CCA (MCCA) is to analyze linear relationships among more than two canonical variates, as a generalization of traditional two-view CCA.

Widely used model: Given ℓ datasets in the form of matrices

$$S_i \in \mathbb{R}^{n_i \times q} \quad \text{for } i = 1, 2, \ldots, \ell, \tag{15}$$

where n_i is the number of features in the ith view, and q is the number of sample data points.

Assume all S_i are centered, i.e., $S_i \mathbf{1}_q = 0$ for all i.

(Cross-)Covariance: $C_{i,j} = S_i S_j^{\mathrm{T}}$ for $i,\, j = 1, \ldots, \ell$. MCCA seeks to solve

$$\max_{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_\ell} \sum_{i,\, j=1}^{\ell} \boldsymbol{x}_i^{\mathrm{T}} C_{i,j} \boldsymbol{x}_j \quad \text{subject to} \begin{cases} \text{either} & \sum_{i=1}^{\ell} \boldsymbol{x}_i^{\mathrm{T}} C_{i,i} \boldsymbol{x}_i = 1, \\ \text{or} & \boldsymbol{x}_i^{\mathrm{T}} C_{i,i} \boldsymbol{x}_i = 1,\ i = 1, \ldots, \ell. \end{cases}$$

Single-Vector Multiset CCA (MCCA)

Multiset CCA (MCCA) is to analyze linear relationships among more than two canonical variates, as a generalization of traditional two-view CCA.

Widely used model: Given ℓ datasets in the form of matrices

$$S_i \in \mathbb{R}^{n_i \times q} \quad \text{for } i = 1, 2, \ldots, \ell, \tag{15}$$

where n_i is the number of features in the ith view, and q is the number of sample data points.

Assume all S_i are centered, i.e., $S_i \mathbf{1}_q = 0$ for all i.

(Cross-)Covariance: $C_{i,j} = S_i S_j^{\mathrm{T}}$ for $i,\, j = 1, \ldots, \ell$. MCCA seeks to solve

$$\max_{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_\ell} \sum_{i,\, j=1}^{\ell} \boldsymbol{x}_i^{\mathrm{T}} C_{i,j} \boldsymbol{x}_j \quad \text{subject to} \left\{ \begin{array}{rl} \text{either} & \sum_{i=1}^{\ell} \boldsymbol{x}_i^{\mathrm{T}} C_{i,i} \boldsymbol{x}_i = 1, \\ \text{or} & \boldsymbol{x}_i^{\mathrm{T}} C_{i,i} \boldsymbol{x}_i = 1,\ i = 1, \ldots, \ell. \end{array} \right.$$

Orthogonal Multiset CCA (OMCCA)

We seek Canonical Weight Matrices $X_i \in \mathbb{R}^{n_i \times k}$ that solve

$$\max_{\{X_i\}} f(\{X_i\}), \quad \text{s.t. } X_i^{\mathrm{T}} X_i = I_k,\ i = 1, \ldots, \ell, \tag{16}$$

where $1 \le k \le \min\{n_1, \ldots, n_\ell, q\}$, and

$$f(\{X_i\}) = \sum_{\substack{i,\, j=1 \\ i \ne j}}^{\ell} \rho_{ij} \frac{\mathrm{tr}(X_i^{\mathrm{T}} C_{i,j} X_j)}{\sqrt{\mathrm{tr}(X_i^{\mathrm{T}} C_{i,i} X_i)} \sqrt{\mathrm{tr}(X_j^{\mathrm{T}} C_{j,j} X_j)}}, \tag{17}$$

with some weighting factors $\rho_{ij} \ge 0$ that turn out to be extremely important.

- $\{\rho_{ij}\}$ dictate the contribution of the correlation between S_i and S_j to the total $f(\{X_i\})$;
- sparse $\{\rho_{ij}\}$ dramatically reduce the number terms in $f(\{X_i\})$ and thus speed up computations;
- judiciously chosen ρ_{ij} with only a few of them nonzero can in fact improve the performances of muti-view tasks (as verified by experiments).

Orthogonal Multiset CCA (OMCCA)

We seek Canonical Weight Matrices $X_i \in \mathbb{R}^{n_i \times k}$ that solve

$$\max_{\{X_i\}} f(\{X_i\}), \quad \text{s.t. } X_i^{\mathrm{T}} X_i = I_k,\ i = 1, \ldots, \ell, \tag{16}$$

where $1 \le k \le \min\{n_1, \ldots, n_\ell, q\}$, and

$$f(\{X_i\}) = \sum_{\substack{i,\, j=1 \\ i \ne j}}^{\ell} \rho_{ij} \frac{\operatorname{tr}(X_i^{\mathrm{T}} C_{i,j} X_j)}{\sqrt{\operatorname{tr}(X_i^{\mathrm{T}} C_{i,i} X_i)} \sqrt{\operatorname{tr}(X_j^{\mathrm{T}} C_{j,j} X_j)}}, \tag{17}$$

with some weighting factors $\rho_{ij} \ge 0$ that turn out to be extremely important.

- $\{\rho_{ij}\}$ dictate the contribution of the correlation between S_i and S_j to the total $f(\{X_i\})$;
- sparse $\{\rho_{ij}\}$ dramatically reduce the number terms in $f(\{X_i\})$ and thus speed up computations;
- judiciously chosen ρ_{ij} with only a few of them nonzero can in fact improve the performances of muti-view tasks (as verified by experiments).

Choosing weights ρ_{ij}

To begin with, we define

$$\widehat{\rho}_{ij} = \frac{\sum_{r=1}^{\operatorname{rank}(C_{i,j})} \sigma_r(C_{i,j})}{\sqrt{\operatorname{tr}(C_{i,i})\operatorname{tr}(C_{j,j})}}, \quad \text{for } i,j = 1,\dots,\ell. \tag{18}$$

It is known $0 \le \widehat{\rho}_{ij} \le 1$.

Envision a graph of ℓ nodes corresponding to dataset matrices X_i, respectively, with every two nodes connected with an edge whose weight ρ_{ij} to be determined.

Three heuristic strategies to select the weights $\rho_{ij} = \rho_{ji}$:

1. uniform weighting: use $\rho_{ij} = 1 \, \forall i,j$;
2. tree weighting: find the minimal spanning tree of the graph with the same nodes but the edge (i,j) having weight $1 - \widehat{\rho}_{ij}$, and then let $\rho_{ij} = \widehat{\rho}_{ij}$ if the the edge (i,j) is on the tree and 0 otherwise.
3. top-p weighting: let $\widetilde{\rho}_{ij} = \widehat{\rho}_{ij}$ for the p largest $\widehat{\rho}_{ij}$ for $i > j$ and all other $\widetilde{\rho}_{ij} = 0$, and then apply the soft-max function to $\widetilde{\rho}_{ij}$ to yield ρ_{ij}.

Choosing weights ρ_{ij}

To begin with, we define

$$\widehat{\rho}_{ij} = \frac{\sum_{r=1}^{\mathrm{rank}(C_{i,j})} \sigma_r(C_{i,j})}{\sqrt{\mathrm{tr}(C_{i,i})\,\mathrm{tr}(C_{j,j})}}, \quad \text{for } i,j = 1,\ldots,\ell. \tag{18}$$

It is known $0 \le \widehat{\rho}_{ij} \le 1$.

Envision a graph of ℓ nodes corresponding to dataset matrices X_i, respectively, with every two nodes connected with an edge whose weight ρ_{ij} to be determined.

Three heuristic strategies to select the weights $\rho_{ij} = \rho_{ji}$:

1. uniform weighting: use $\rho_{ij} = 1\,\forall i,j$;
2. tree weighting: find the minimal spanning tree of the graph with the same nodes but the edge (i,j) having weight $1-\widehat{\rho}_{ij}$, and then let $\rho_{ij} = \widehat{\rho}_{ij}$ if the the edge (i,j) is on the tree and 0 otherwise.
3. top-p weighting: let $\widetilde{\rho}_{ij} = \widehat{\rho}_{ij}$ for the p largest $\widehat{\rho}_{ij}$ for $i > j$ and all other $\widetilde{\rho}_{ij} = 0$, and then apply the soft-max function to $\widetilde{\rho}_{ij}$ to yield ρ_{ij}.

Choosing weights ρ_{ij}

To begin with, we define

$$\widehat{\rho}_{ij} = \frac{\sum_{r=1}^{\operatorname{rank}(C_{i,j})} \sigma_r(C_{i,j})}{\sqrt{\operatorname{tr}(C_{i,i})\operatorname{tr}(C_{j,j})}}, \quad \text{for } i, j = 1, \ldots, \ell. \tag{18}$$

It is known $0 \le \widehat{\rho}_{ij} \le 1$.

Envision a graph of ℓ nodes corresponding to dataset matrices X_i, respectively, with every two nodes connected with an edge whose weight ρ_{ij} to be determined.

Three heuristic strategies to select the weights $\rho_{ij} = \rho_{ji}$:

1. uniform weighting: use $\rho_{ij} = 1 \, \forall i, j$;
2. tree weighting: find the minimal spanning tree of the graph with the same nodes but the edge (i, j) having weight $1 - \widehat{\rho}_{ij}$, and then let $\rho_{ij} = \widehat{\rho}_{ij}$ if the the edge (i, j) is on the tree and 0 otherwise.
3. top-p weighting: let $\widetilde{\rho}_{ij} = \widehat{\rho}_{ij}$ for the p largest $\widehat{\rho}_{ij}$ for $i > j$ and all other $\widetilde{\rho}_{ij} = 0$, and then apply the soft-max function to $\widetilde{\rho}_{ij}$ to yield ρ_{ij}.

SCF algorithm for OMCCA (1)

$$\max_{\{X_i\}} f(\{X_i\}), \quad \text{s.t. } X_i^{\mathrm{T}} C_{i,i} X_i = I_k,\ i = 1, \ldots, \ell,$$

where

$$f(\{X_i\}) = \sum_{\substack{i,\, j=1 \\ i \neq j}}^{\ell} \rho_{ij} \frac{\operatorname{tr}(X_i^{\mathrm{T}} C_{i,j} X_j)}{\sqrt{\operatorname{tr}(X_i^{\mathrm{T}} C_{i,i} X_i)} \sqrt{\operatorname{tr}(X_j^{\mathrm{T}} C_{j,j} X_j)}}.$$

Plan to optimize $f(\{X_i\})$ cyclically over each matrix variable X_i in the styles similar to either Jacobi or Gauss-Seidel updating for linear systems.

Specifically, an inner-outer iterative method:

- outer iteration – each step called a cycle – generates from the current approximation $\{X_i^{(\nu)}\}_{i=1}^{\ell}$ to the next $\{X_i^{(\nu+1)}\}_{i=1}^{\ell}$ of the maximizer of $f(\{X_i\})$;
- inner iteration – an either Jacobi-style or Gauss-Seidel-style updating scheme that relies on the proposed SCF iteration for solving a series of subproblems in the form of (5).

SCF algorithm for OMCCA (1)

$$\max_{\{X_i\}} f(\{X_i\}), \quad \text{s.t. } X_i^{\mathrm{T}} C_{i,i} X_i = I_k,\ i = 1, \ldots, \ell,$$

where

$$f(\{X_i\}) = \sum_{\substack{i,\, j=1 \\ i \neq j}}^{\ell} \rho_{ij} \frac{\mathrm{tr}(X_i^{\mathrm{T}} C_{i,j} X_j)}{\sqrt{\mathrm{tr}(X_i^{\mathrm{T}} C_{i,i} X_i)} \sqrt{\mathrm{tr}(X_j^{\mathrm{T}} C_{j,j} X_j)}}.$$

Plan to optimize $f(\{X_i\})$ cyclically over each matrix variable X_i in the styles similar to either Jacobi or Gauss-Seidel updating for linear systems.

Specifically, an inner-outer iterative method:

- outer iteration – each step called a cycle – generates from the current approximation $\{X_i^{(\nu)}\}_{i=1}^{\ell}$ to the next $\{X_i^{(\nu+1)}\}_{i=1}^{\ell}$ of the maximizer of $f(\{X_i\})$;
- inner iteration – an either Jacobi-style or Gauss-Seidel-style updating scheme that relies on the proposed SCF iteration for solving a series of subproblems in the form of (5).

SCF algorithm for OMCCA (1)

$$\max_{\{X_i\}} f(\{X_i\}), \quad \text{s.t. } X_i^{\mathrm{T}} C_{i,i} X_i = I_k,\ i = 1, \ldots, \ell,$$

where

$$f(\{X_i\}) = \sum_{\substack{i,\,j=1 \\ i \neq j}}^{\ell} \rho_{ij} \frac{\operatorname{tr}(X_i^{\mathrm{T}} C_{i,j} X_j)}{\sqrt{\operatorname{tr}(X_i^{\mathrm{T}} C_{i,i} X_i)} \sqrt{\operatorname{tr}(X_j^{\mathrm{T}} C_{j,j} X_j)}}.$$

Plan to optimize $f(\{X_i\})$ cyclically over each matrix variable X_i in the styles similar to either Jacobi or Gauss-Seidel updating for linear systems.

Specifically, an inner-outer iterative method:

- outer iteration – each step called a cycle – generates from the current approximation $\{X_i^{(\nu)}\}_{i=1}^{\ell}$ to the next $\{X_i^{(\nu+1)}\}_{i=1}^{\ell}$ of the maximizer of $f(\{X_i\})$;
- inner iteration – an either Jacobi-style or Gauss-Seidel-style updating scheme that relies on the proposed SCF iteration for solving a series of subproblems in the form of (5).

SCF algorithm for OMCCA (2)

Let the SVDs of S_i be ($r_i = \mathrm{rank}(S_i)$)

$$S_i = U_i \Sigma_i V_i^{\mathrm{T}},\ U_i \in \mathbb{R}^{n_i \times r_i},\ V_i \in \mathbb{R}^{q \times r_i},\ \Sigma_i \in \mathbb{R}^{r_i \times r_i}. \tag{19}$$

$$X_i^{\mathrm{T}} S_i S_j^{\mathrm{T}} X_j = X_i^{\mathrm{T}} U_i \Sigma_i V_i^{\mathrm{T}} V_j \Sigma_j U_j^{\mathrm{T}} X_j =: \widehat{X}_i^{\mathrm{T}} \Sigma_i V_i^{\mathrm{T}} V_j \Sigma_j \widehat{X}_j,$$

where $\widehat{X}_i = U_i^{\mathrm{T}} X_i \in \mathbb{R}^{r_i \times k}$. $X_i = U_i \widehat{X}_i$ by $\mathcal{R}(X_i) \subset \mathcal{R}(S_i)$.
The function $f(\{X_i\})$ is then transformed into

$$\sum_{i \neq j} \rho_{ij} \frac{\mathrm{tr}(\widehat{X}_i^{\mathrm{T}} \Sigma_i V_i^{\mathrm{T}} V_j \Sigma_j \widehat{X}_j)}{\sqrt{\mathrm{tr}(\widehat{X}_i^{\mathrm{T}} \Sigma_i^2 \widehat{X}_i)} \sqrt{\mathrm{tr}(\widehat{X}_j^{\mathrm{T}} \Sigma_j^2 \widehat{X}_j)}} =: g(\{\widehat{X}_i\}),$$

and, thus

$$\max_{X_i \in \mathbb{O}^{n_i \times k},\, \mathcal{R}(X_i) \subset \mathcal{R}(S_i),\ \forall i} f(\{X_i\}) = \max_{\widehat{X}_i \in \mathbb{O}^{r_i \times k},\, \forall i} g(\{\widehat{X}_i\}).$$

SCF algorithm for OMCCA (2)

Let the SVDs of S_i be ($r_i = \mathrm{rank}(S_i)$)

$$S_i = U_i \Sigma_i V_i^{\mathrm{T}},\ U_i \in \mathbb{R}^{n_i \times r_i},\ V_i \in \mathbb{R}^{q \times r_i},\ \Sigma_i \in \mathbb{R}^{r_i \times r_i}. \tag{19}$$

$$X_i^{\mathrm{T}} S_i S_j^{\mathrm{T}} X_j = X_i^{\mathrm{T}} U_i \Sigma_i V_i^{\mathrm{T}} V_j \Sigma_j U_j^{\mathrm{T}} X_j =: \widehat{X}_i^{\mathrm{T}} \Sigma_i V_i^{\mathrm{T}} V_j \Sigma_j \widehat{X}_j,$$

where $\widehat{X}_i = U_i^{\mathrm{T}} X_i \in \mathbb{R}^{r_i \times k}$. $X_i = U_i \widehat{X}_i$ by $\mathcal{R}(X_i) \subset \mathcal{R}(S_i)$.
The function $f(\{X_i\})$ is then transformed into

$$\sum_{i \neq j} \rho_{ij} \frac{\mathrm{tr}(\widehat{X}_i^{\mathrm{T}} \Sigma_i V_i^{\mathrm{T}} V_j \Sigma_j \widehat{X}_j)}{\sqrt{\mathrm{tr}(\widehat{X}_i^{\mathrm{T}} \Sigma_i^2 \widehat{X}_i)}\sqrt{\mathrm{tr}(\widehat{X}_j^{\mathrm{T}} \Sigma_j^2 \widehat{X}_j)}} =: g(\{\widehat{X}_i\}),$$

and, thus

$$\max_{X_i \in \mathbb{O}^{n_i \times k},\, \mathcal{R}(X_i) \subset \mathcal{R}(S_i),\ \forall i} f(\{X_i\}) = \max_{\widehat{X}_i \in \mathbb{O}^{r_i \times k},\, \forall i} g(\{\widehat{X}_i\}).$$

SCF algorithm for OMCCA (2)

Let the SVDs of S_i be ($r_i = \mathrm{rank}(S_i)$)

$$S_i = U_i \Sigma_i V_i^{\mathrm{T}},\ U_i \in \mathbb{R}^{n_i \times r_i},\ V_i \in \mathbb{R}^{q \times r_i},\ \Sigma_i \in \mathbb{R}^{r_i \times r_i}. \tag{19}$$

$$X_i^{\mathrm{T}} S_i S_j^{\mathrm{T}} X_j = X_i^{\mathrm{T}} U_i \Sigma_i V_i^{\mathrm{T}} V_j \Sigma_j U_j^{\mathrm{T}} X_j =: \widehat{X}_i^{\mathrm{T}} \Sigma_i V_i^{\mathrm{T}} V_j \Sigma_j \widehat{X}_j,$$

where $\widehat{X}_i = U_i^{\mathrm{T}} X_i \in \mathbb{R}^{r_i \times k}$. $X_i = U_i \widehat{X}_i$ by $\mathcal{R}(X_i) \subset \mathcal{R}(S_i)$.
The function $f(\{X_i\})$ is then transformed into

$$\sum_{i \neq j} \rho_{ij} \frac{\mathrm{tr}(\widehat{X}_i^{\mathrm{T}} \Sigma_i V_i^{\mathrm{T}} V_j \Sigma_j \widehat{X}_j)}{\sqrt{\mathrm{tr}(\widehat{X}_i^{\mathrm{T}} \Sigma_i^2 \widehat{X}_i)} \sqrt{\mathrm{tr}(\widehat{X}_j^{\mathrm{T}} \Sigma_j^2 \widehat{X}_j)}} =: g(\{\widehat{X}_i\}),$$

and, thus

$$\max_{X_i \in \mathbb{O}^{n_i \times k},\, \mathcal{R}(X_i) \subset \mathcal{R}(S_i),\ \forall i} f(\{X_i\}) = \max_{\widehat{X}_i \in \mathbb{O}^{r_i \times k},\, \forall i} g(\{\widehat{X}_i\}).$$

SCF algorithm for OMCCA (3)

The key step to maximize $g(\{\widehat{X}_i\})$ by either the Jacobi- or Gauss-Seidel-style updating scheme is to maximize it, for any $s \in \{1, \cdots, \ell\}$, over $\widehat{X}_s$ while keeping all other $\widehat{X}_j$ for $j \neq s$ constant.

That is equivalent to

$$\max_{\widehat{X}_s \in \mathbb{O}^{n_s \times k}} \frac{\operatorname{tr}(\widehat{X}_s^{\mathrm{T}} D_s)}{\sqrt{\operatorname{tr}(\widehat{X}_s^{\mathrm{T}} \Sigma_s^2 \widehat{X}_s)}}, \tag{20}$$

where $D_s(\{\widehat{X}_i\}_{i\neq s}) = \Sigma_s V_s^{\mathrm{T}} \sum_{j\neq s} \rho_{sj} \frac{V_j \Sigma_j \widehat{X}_j}{\sqrt{\operatorname{tr}(\widehat{X}_j^{\mathrm{T}} \Sigma_j^2 \widehat{X}_j)}}$.

Problem (20) is equivalent to solving:

$$\max_{\widehat{X}_s \in \mathbb{O}^{n_s \times k}} \frac{\operatorname{tr}^2(\widehat{X}_s^{\mathrm{T}} D_s)}{\operatorname{tr}(\widehat{X}_s^{\mathrm{T}} \Sigma_s^2 \widehat{X}_s)}. \tag{21}$$

SCF algorithm for OMCCA (3)

The key step to maximize $g(\{\widehat{X}_i\})$ by either the Jacobi- or Gauss-Seidel-style updating scheme is to maximize it, for any $s \in \{1, \cdots, \ell\}$, over $\widehat{X}_s$ while keeping all other $\widehat{X}_j$ for $j \neq s$ constant.

That is equivalent to

$$\max_{\widehat{X}_s \in \mathbb{O}^{n_s \times k}} \frac{\operatorname{tr}(\widehat{X}_s^{\mathrm{T}} D_s)}{\sqrt{\operatorname{tr}(\widehat{X}_s^{\mathrm{T}} \Sigma_s^2 \widehat{X}_s)}}, \tag{20}$$

where $D_s(\{\widehat{X}_i\}_{i \neq s}) = \Sigma_s V_s^{\mathrm{T}} \sum_{j \neq s} \rho_{sj} \frac{V_j \Sigma_j \widehat{X}_j}{\sqrt{\operatorname{tr}(\widehat{X}_j^{\mathrm{T}} \Sigma_j^2 \widehat{X}_j)}}$.

Problem (20) is equivalent to solving:

$$\max_{\widehat{X}_s \in \mathbb{O}^{n_s \times k}} \frac{\operatorname{tr}^2(\widehat{X}_s^{\mathrm{T}} D_s)}{\operatorname{tr}(\widehat{X}_s^{\mathrm{T}} \Sigma_s^2 \widehat{X}_s)}. \tag{21}$$

SCF algorithm for OMCCA (3)

The key step to maximize $g(\{\widehat{X}_i\})$ by either the Jacobi- or Gauss-Seidel-style updating scheme is to maximize it, for any $s \in \{1, \cdots, \ell\}$, over $\widehat{X}_s$ while keeping all other $\widehat{X}_j$ for $j \neq s$ constant.

That is equivalent to

$$\max_{\widehat{X}_s \in \mathbb{O}^{n_s \times k}} \frac{\operatorname{tr}(\widehat{X}_s^{\mathrm{T}} D_s)}{\sqrt{\operatorname{tr}(\widehat{X}_s^{\mathrm{T}} \Sigma_s^2 \widehat{X}_s)}}, \tag{20}$$

where $D_s(\{\widehat{X}_i\}_{i \neq s}) = \Sigma_s V_s^{\mathrm{T}} \sum_{j \neq s} \rho_{sj} \frac{V_j \Sigma_j \widehat{X}_j}{\sqrt{\operatorname{tr}(\widehat{X}_j^{\mathrm{T}} \Sigma_j^2 \widehat{X}_j)}}$.

Problem (20) is equivalent to solving:

$$\max_{\widehat{X}_s \in \mathbb{O}^{n_s \times k}} \frac{\operatorname{tr}^2(\widehat{X}_s^{\mathrm{T}} D_s)}{\operatorname{tr}(\widehat{X}_s^{\mathrm{T}} \Sigma_s^2 \widehat{X}_s)}. \tag{21}$$

Algorithm 3. RCOMCCA: Range Constrained OMCCA

Input: $\{S_i \in \mathbb{R}^{n_i \times q}\}$ (each S_i centered), integer k, and tolerance ϵ;
Output: $\{X_i \in \mathbb{O}^{n_i \times k}\}$ that maximizes $f(\{X_i\})$.

compute SVDs in (19);
pick an initial approximation $\widehat{X}_1^{(0)}$;
$\nu = 0$, $g = 0$;
repeat
  $g_0 = g$; $g = 0$;
  for $s = 1$ to ℓ **do**
    compute the next $\{\widehat{X}_s^{(\nu+1)}\}$ by solving (21), where either $D_s = D_s(\{\widehat{X}_i^{(\nu)}\}_{i \neq s})$ for Jacobi-style updating, or $D_s = D_s(\widehat{X}_1^{(\nu+1)}, \ldots, \widehat{X}_{s-1}^{(\nu+1)}, \widehat{X}_{s+1}^{(\nu)}, \ldots, \widehat{X}_\ell^{(\nu)})$ for Gauss-Seidel-style updating;
    $g = g + g_s$, where g_s is the computed optimal objective value of (21).
  end for
  $\nu = \nu + 1$;
until $|g - g_0| \leq \epsilon g$;
return $X_i = U_i \widehat{X}_i^{(\nu)}$ for $1 \leq i \leq \ell$.

Outline

1. Introduction to CCA
2. Orthogonal CCA (OCCA)
3. Submaximization Problem
4. Orthogonal Multiset CCA (OMCCA)
5. Applications
6. Summary

Application 1: Multi-label classification

Multi-class classification: assign an object (vector) $\boldsymbol{x}$ to one of n_c classes, often by attaching a label $y \in \{1, 2, \dots, n_c\}$.

Multi-label classification: assign an object (vector) $\boldsymbol{x}$ to one or more of n_c classes, often by attaching an indicator vector $\boldsymbol{y} \in \mathbb{R}^{n_c}$ of 0s and 1s in such a way that $\boldsymbol{x}$ belongs to class i if $\boldsymbol{y}_{(i)} = 1$ and doesn't otherwise.

$X \in \mathbb{R}^{n \times q}$ contains q vectors of size n, and $Y \in \mathbb{R}^{n_c \times q}$ consists of q corresponding indicator vectors. CCA for multi-label classification popularly treats X as one view and Y as the other.

We will use ML-kNN[1] as our backend multi-label classifier.

Table: Multi-label classification datasets

Dataset	Samples (q)	Attributes (n)	labels (n_c)
birds	645	260	19
emotions	593	72	6

[1] http://lamda.nju.edu.cn/files/MLkNN.rar

Application 1: Multi-label classification

Multi-class classification: assign an object (vector) $\boldsymbol{x}$ to one of n_c classes, often by attaching a label $y \in \{1, 2, \ldots, n_c\}$.

Multi-label classification: assign an object (vector) $\boldsymbol{x}$ to one or more of n_c classes, often by attaching an indicator vector $\boldsymbol{y} \in \mathbb{R}^{n_c}$ of 0s and 1s in such a way that $\boldsymbol{x}$ belongs to class i if $\boldsymbol{y}_{(i)} = 1$ and doesn't otherwise.

$X \in \mathbb{R}^{n \times q}$ contains q vectors of size n, and $Y \in \mathbb{R}^{n_c \times q}$ consists of q corresponding indicator vectors. CCA for multi-label classification popularly treats X as one view and Y as the other.

We will use ML-kNN[1] as our backend multi-label classifier.

Table: Multi-label classification datasets

Dataset	Samples (q)	Attributes (n)	labels (n_c)
birds	645	260	19
emotions	593	72	6

[1] http://lamda.nju.edu.cn/files/MLkNN.rar

Application 1: Multi-label classification

Multi-class classification: assign an object (vector) $\boldsymbol{x}$ to one of n_c classes, often by attaching a label $y \in \{1, 2, \ldots, n_c\}$.

Multi-label classification: assign an object (vector) $\boldsymbol{x}$ to one or more of n_c classes, often by attaching an indicator vector $\boldsymbol{y} \in \mathbb{R}^{n_c}$ of 0s and 1s in such a way that $\boldsymbol{x}$ belongs to class i if $\boldsymbol{y}_{(i)} = 1$ and doesn't otherwise.

$X \in \mathbb{R}^{n \times q}$ contains q vectors of size n, and $Y \in \mathbb{R}^{n_c \times q}$ consists of q corresponding indicator vectors. CCA for multi-label classification popularly treats X as one view and Y as the other.

We will use ML-kNN[1] as our backend multi-label classifier.

Table: Multi-label classification datasets

Dataset	Samples (q)	Attributes (n)	labels (n_c)
birds	645	260	19
emotions	593	72	6

[1] http://lamda.nju.edu.cn/files/MLkNN.rar

Application 1: Multi-label classification

Multi-class classification: assign an object (vector) $\boldsymbol{x}$ to one of n_c classes, often by attaching a label $y \in \{1, 2, \ldots, n_c\}$.

Multi-label classification: assign an object (vector) $\boldsymbol{x}$ to one or more of n_c classes, often by attaching an indicator vector $\boldsymbol{y} \in \mathbb{R}^{n_c}$ of 0s and 1s in such a way that $\boldsymbol{x}$ belongs to class i if $\boldsymbol{y}_{(i)} = 1$ and doesn't otherwise.

$X \in \mathbb{R}^{n \times q}$ contains q vectors of size n, and $Y \in \mathbb{R}^{n_c \times q}$ consists of q corresponding indicator vectors. CCA for multi-label classification popularly treats X as one view and Y as the other.

We will use ML-kNN[1] as our backend multi-label classifier.

Table: Multi-label classification datasets

Dataset	Samples (q)	Attributes (n)	labels (n_c)
birds	645	260	19
emotions	593	72	6

[1] http://lamda.nju.edu.cn/files/MLkNN.rar

Table: Results on two datasets by 5 methods (40% for training and 60% for testing over 10 random splits). Best results are in bold.

dataset	method	OneError	Average_Precision
birds	OCCA-scf	**0.4964** $\pm$ **0.0201**	**0.5452** $\pm$ **0.0118**
	CCA	0.8110 $\pm$ 0.0302	0.3087 $\pm$ 0.0192
	LS-CCA	0.8110 $\pm$ 0.0302	0.3084 $\pm$ 0.0191
	OCCA-SSY	0.5978 $\pm$ 0.0269	0.4722 $\pm$ 0.0182
	ML-kNN	0.7101 $\pm$ 0.0136	0.3942 $\pm$ 0.0108
emotions	OCCA-scf	**0.3258** $\pm$ **0.0201**	**0.7640** $\pm$ **0.0118**
	CCA	0.3497 $\pm$ 0.0169	0.7443 $\pm$ 0.0126
	LS-CCA	0.3385 $\pm$ 0.0182	0.7553 $\pm$ 0.0154
	OCCA-SSY	0.3860 $\pm$ 0.0274	0.7190 $\pm$ 0.0172
	ML-kNN	0.3983 $\pm$ 0.0169	0.6960 $\pm$ 0.0085

- OneError: the average number of times the top-ranked label is not in the set of proper labels of the instance (the smaller the better)
- Average_Precision: the average precision of labels ranked above a particular label in the same label set. (the bigger the better)

Application 2: Multi-view feature extraction

1-nearest neighbor classifier for evaluating classification accuracy performance.

CCA methods with varying $k \in \{3, 4, 5, 6\}$ for `mfeat`, and $k \in \{3, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50\}$ for other datasets.

Split data into training and testing with ratio 30/70. Results are based on the average of 10 randomly drawn splits.

Six variants of RCOMCCA in total based on different weighting rule. For the top-p weighting scheme, $p \in \{1, 3, 6\}$ is used, and best results are reported.

We compare with MCCA (Nielsen, 2002) and OMCCA-SS (Shen and Sun, 2015).

Table: Multi-view datasets

Dataset	Samples	Multiple views	classes
mfeat	2000	216;76;64;6;240;47	10
Caltech101-7	1474	254;512;1180;1008;64;1000	7
Caltech101-20	2386	254;512;1180;1008;64;1000	20

Table: Means and standard deviations of accuracy (Parameter k used by CCA methods to achieve the best accuracy is shown in the bracket).

	mfeat	Caltech101-7
view1	0.9513 ± 0.0053	0.9259 ± 0.0049
view2	0.7604 ± 0.0104	0.9443 ± 0.0051
view3	0.9293 ± 0.0043	0.9415 ± 0.0070
view4	0.6780 ± 0.0064	0.9287 ± 0.0105
view5	0.9630 ± 0.0025	0.7759 ± 0.0133
view6	0.7814 ± 0.0077	0.9152 ± 0.0059
MCCA	0.8679 ± 0.0073 (6)	0.8865 ± 0.0072 (15)
OMCCA-SS	0.8298 ± 0.0089 (6)	0.9493 ± 0.0024 (45)
RCOMCCA-G (uniform)	0.7634 ± 0.0134 (5)	0.8880 ± 0.0052 (50)
RCOMCCA-G (top-p)	$\mathbf{0.9696} \pm \mathbf{0.0035}$ (5)	$\mathbf{0.9664} \pm \mathbf{0.0060}$ (35)
RCOMCCA-G (tree)	0.9566 ± 0.0031 (6)	0.9392 ± 0.0043 (45)
RCOMCCA-J (uniform)	0.7540 ± 0.0121 (5)	0.8868 ± 0.0068 (30)
RCOMCCA-J (top-p)	0.9692 ± 0.0038 (5)	0.9649 ± 0.0029 (15)
RCOMCCA-J (tree)	0.9581 ± 0.0055 (6)	0.9474 ± 0.0041 (45)

Accuracy, CPU time and Training ratio

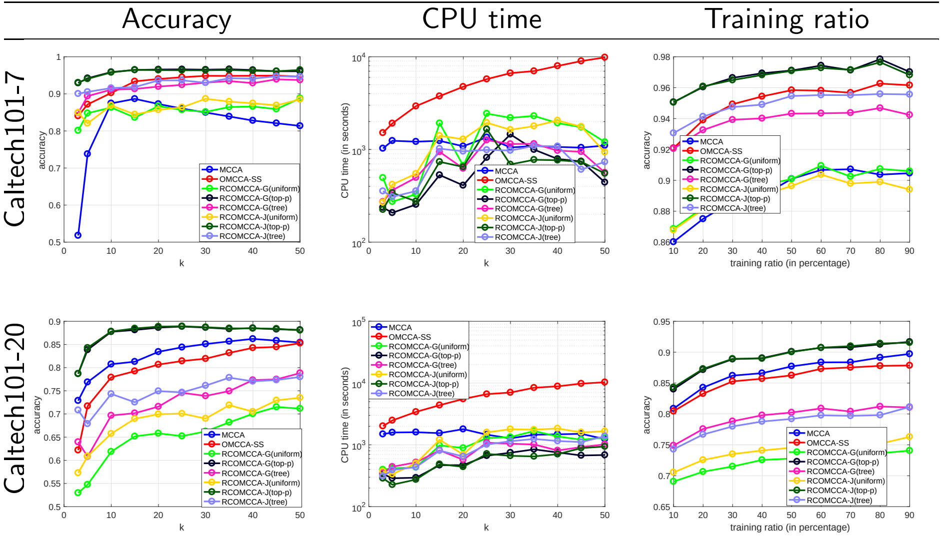


Figure: Accuracy and CPU time of MCCA methods on two datasets by varying the reduced dimension k and the training ratio.

Outline

Summary

- An OCCA-SCF algorithm for solving trace-fractional matrix optimization problem:

$$\max_{G\in\mathbb{O}^{n\times k}} \eta(G) \quad \text{with } \eta(G) := \frac{\operatorname{tr}(G^{\mathrm{T}}D)}{\sqrt{\operatorname{tr}(G^{\mathrm{T}}AG)}},$$

  where Stiefel manifold: $\mathbb{O}^{n\times k} = \{X \in \mathbb{R}^{n\times k} : X^{\mathrm{T}}X = I_k\}$.

- An alternating iterative method for solving Orthogonal Canonical Correlation Analysis (OCCA):

$$\max_{X\in\mathbb{O}^{n\times k},\, Y\in\mathbb{O}^{m\times k}} \frac{\operatorname{tr}(X^{\mathrm{T}}CY)}{\sqrt{\operatorname{tr}(X^{\mathrm{T}}AX)}\sqrt{\operatorname{tr}(Y^{\mathrm{T}}BY)}}.$$

- A new orthogonal multiset OCCA (OMCCA) model with integrated weights for each pair of views and trace-fractional objective for correlations between any two views.
- Applications to two real world applications: multi-label classification and multi-view feature extraction.

Related Reference

Leihong Zhang, Li Wang, Zhaojun Bai and Ren-Cang Li.
A Self-consistent-field Iteration for Orthogonal Canonical Correlation Analysis.
IEEE Transactions on Pattern Analysis and Machine Intelligence,
DOI: `10.1109/TPAMI.2020.3012541`, 2020.
(with a supplement of 13 pages for proofs)