Approche bloc en ACP group-sparse: le package sparsePCA

Marie Chavent [1,2] Guy Chavent [3]

[1] Inria Bordeaux Sud-Ouest

[2] Université de Bordeaux

[3] Inria Paris

Références

- Article : *Group-sparse block PCA and explained variance*, arXiv :1705.00461
- Package R `sparsePCA` : https://github.com/chavent/sparsePCA

Plan

1. ACP et décomposition en valeurs singulières
2. Approche par déflation et formulations bloc du probème d'ACP
3. Méthode bloc d'ACP group-sparse
4. Résultats numériques

Matrice de données A de rang r :

- n observations
- p variables centrées ou centrées-réduites

Décomposition en Valeurs Singulières de A :

$$A = U\Sigma V^T$$

- $u_1 \dots u_r$: vecteurs singuliers de gauche (vecteurs propres de AA^T)
- $v_1 \dots v_r$ of V : vecteurs singuliers de droite (vecteurs propres de A^TA)
- $\sigma_1 \dots \sigma_r$: valeurs singulières (racines des valeurs propres de AA^T et AA^T)

Analyse en Composantes Principales

L'ACP cherche $m \leq r$ combinaisons $z_1, \ldots, z_m$ (loadings) des p variables telles que les variables $y_j = Az_j, j = 1 \ldots m$ (composantes) soient non corrélées et expliquent le maximum de la variance des données.

Loadings optimaux et composantes optimales :

$$\begin{aligned} z_j^* &= v_j \quad (m \text{ premiers vecteurs singuliers de droite de } A)\ , \\ y_j^* &= Av_j = \sigma_j u_j \quad (\text{prop. aux } m \text{ premiers vecteurs singuliers de gauche de } A) \end{aligned}$$

Remarque

Les colonnes de A sont centrées donc les composantes principales y_j sont centrées et donc

$$Var(y_j) = \|y_j\|^2 = \|Az_j\|^2.$$

Approche par déflation : les m vecteurs de loadings z_j sont calculés successivement par récurrence :

$$\begin{aligned} \text{Set } A_0 &= A \ , \ z_0 = 0 \ , \ \text{and compute, for } j = 1 \dots m : \\ A_j &= A_{j-1}(I_{|p|} - z_{j-1} z_{j-1}^T) \\ z_j &= \arg \max_{\|z\|=1} \|A_j z\|^2 \end{aligned}$$

Approche bloc : recherche simultanée des m vecteurs de loadings z_j.

Ecrire l'ACP comme un problème d'optimisation bloc.

Trois inconnues bloc :

$$\left\{ \begin{array}{lcll} Z & = & [z_1 \ \ldots z_m] \in \boldsymbol{R}^{p \times m} & \text{(tentative loadings)}, \\ Y & = & [y_1 \ \ldots y_m] \in \boldsymbol{R}^{n \times m} & \text{(tentative components)}, \\ X & = & [x_1 \ldots x_m] \in \boldsymbol{R}^{n \times m} & \text{(tentative normalized components)} \ . \end{array} \right.$$

Notations :

- $(\mathcal{B}^k)^m$: matrices $k \times m$ dont les colonnes sont normées.
- $\mathcal{S}_m^k$: matrices $k \times m$ dont les colonnes sont orthonormées (variété de Stiefel).

Trois formulations bloc du problème d'ACP :

$$\max_{Z \in \mathcal{S}_m^p} \sum_{j=1\dots m} \mu_j^2 \|Az_j\|^2 \quad \text{(loading formulation)} \tag{1}$$

$$\max_{X \in \mathcal{S}_m^n} \sum_{j=1\dots m} \mu_j^2 \|A^T x_j\|^2 \quad \text{(component formulation)} \tag{2}$$

$$\max_{X \in \mathcal{S}_m^n} \max_{Z \in (\mathcal{B}^p)^m} \sum_{j=1\dots m} \mu_j^2 (x_j^T A z_j)^2 \quad \text{(component/loading formulation)} \tag{3}$$

L'équivalence entre (2) et (3) s'obtient avec

$$\forall x \in \boldsymbol{R}^n \ , \quad \|A^T x\|^2 = \max_{z \in \mathcal{B}^{|p|}} (z^T A^T x)^2 = \max_{z \in \mathcal{B}^{|p|}} (x^T A z)^2 \ .$$

Remarque

- Si $\mu_1 = \cdots = \mu_m = 1$, Z^* est une base de l'espace propre $vect\{v_1 \dots v_m\}$ de A.
- Si $\mu_1 > \mu_2 > \cdots > \mu_m > 0$, Z^* est la base des vecteurs propres classés par valeurs propres décroissantes.

Exemple : données simulées selon le modèle de Shen & Huang (2008)

- $m = 2$ composantes, $p = 10$ variables,
- structuration des variables en trois groupes de taille 4,4,2.

Table: Z sous-jacent

z1	z2
0.422	0.000
0.422	0.000
0.422	0.000
0.422	0.000
0.000	0.489
0.000	0.489
0.000	0.489
0.000	0.489
0.380	-0.147
0.380	0.147

Table: ACP

z1	z2
0.395	0.244
0.338	0.208
0.446	0.027
0.359	0.031
-0.107	0.513
-0.183	0.532
-0.143	0.336
-0.015	0.353
0.417	-0.212
0.402	0.260

Table: Sparse

z1	z2
0.423	0.089
0.354	0.067
0.459	0.000
0.361	0.000
-0.001	0.524
-0.082	0.590
-0.058	0.390
0.000	0.381
0.397	-0.229
0.433	0.125

Table: Group-sparse

z1	z2
0.446	0.000
0.386	0.000
0.464	0.000
0.389	0.000
0.000	0.493
0.000	0.596
0.000	0.453
0.000	0.429
0.360	-0.098
0.395	0.039

On se donne une structuration des variables en groupes et on note p le nombre de groupes.

Afin de faire apparaître des zeros dans des groupes de loadings, on définit la norme :

$$\|z_j\|_1 = \sum_{i=1}^{p} \|z_{i,j}\| \ ,$$

où :

- $z_{i,j}$ est le vecteur de dimension p_i des loadings des variables du groupe i :

$$z_j^T = \left[z_{1,j}^T \dots z_{i,j}^T \dots z_{p,j}^T\right] \ ,$$

- $\|z_{i,j}\|$ est la norme Euclidienne sur $\boldsymbol{R}^{p_i}$

On choisit également des paramètres de régularisation :

$$\gamma_j > 0 \ , \ j = 1 \dots m \ .$$

Formulation bloc du problème d'ACP group-sparse.

$$\max_{X \in \mathcal{S}_m^n} \max_{Z \in (\mathcal{B}^{|p|})^m} \sum_{j=1\ldots m} \mu_j^2 \left[x_j^T A z_j - \gamma_j \|z_j\|_1 \right]_+^2 \tag{4}$$

avec $[t]_+ = t$ si $t \geq 0$ et $[t]_+ = 0$ si $t < 0$.

Inconvénient et avantages de cette formulation :

- Les loadings sparses z_j^* et les composantes principales $y_j^* = Az_j^*$ ne sont plus orthogonaux et les solutions x_j^* sont orthonormales mais ne coincident plus avec les composantes principales normalisées :

$$x_j^* \neq (Az_j^*)/\|Az_j^*\| \ , \ j = 1 \ldots m \ ,$$

+ Les difficultés numériques sont réparties entre X et Z : la contrainte d'orthonormalité est sur X et la non-différentiabilité de la norme est sur Z.

La boucle d'optimisation intérieur sur Z dans (4) a une solution analytique pour tout $X \in \mathcal{S}_m^n$ ce qui conduit à la maximisation d'une fonction convexe différentiable :

$$F(X) \stackrel{\text{def}}{=} \sum_{j=1\ldots m} \mu_j^2 \sum_{i=1}^{p} \left[\|a_i{}^T x_j\| - \gamma_j\right]_+^2 = \sum_{j=1\ldots m} \mu_j^2 \|t_j\|^2 \ . \tag{5}$$

où les a_i sont les matrices de données de dimension $n \times p_i$ lorsque la matrice A de dimension $n \times |p|$ s'écrit :

$$A = \left[a_1 \ldots a_i \ldots a_p\right] \ .$$

Interprétation de $F(X)$:

- $a_i^T x_j \in \mathbf{R}^{p_i}$ s'interprète comme le vecteur des corrélations entre les p_i variables du groupe i et la $j$ème composante normalisée x_j.
- $\|t_j\|^2 = \sum_{i=1}^{p} \left[\|a_i^T x_j\| - \gamma_j\right]_+^2$ avec $t_j^T = \left[t_{1,j}^T \ldots t_{p,j}^T\right]$ et

$$t_{ij} = \frac{a_i^T x_j}{\|a_i^T x_j\|} [\|a_i^T x_j\| - \gamma_j]_+ \in \mathbf{R}^{p_i} \ . \tag{6}$$

- t_{ij} est donc obtenu par seuillage doux du vecteur $a_i^T x_j$ qui consiste à le mettre à zéro si sa norme est inférieur à γ_j et à diminuer sa longeur de γ_j sinon.

La solution X^* s'obtient en appliquant l'algorithme de Journée et al. (JMLR, 2010) pour maximiser $F(X)$ sur la variété de Stiefel $\mathcal{S}_m^n$.

Le gradient de F est :

$$\nabla_X F(X) = 2ATN^2 \in \boldsymbol{R}^{n \times m}$$

avec $T = \left[t_1 \dots t_m\right] \in \boldsymbol{R}^{|p| \times m}$ et $N = \mathrm{diag}\{\mu_1, \dots, \mu_m\}$.

L'algorithme s'écrit alors :

input : $X_0 \in \mathcal{S}_m^n$
output : X_n (approximate solution)
begin
  $0 \longleftarrow k$
  repeat
    $T_k \longleftarrow$ (6)
    $G_k \longleftarrow \nabla_X F(X_k) = 2AT_kN^2$
    $X_{k+1} \longleftarrow \mathrm{polar}(G_k)$
    $k \longleftarrow k+1$
  until a stopping criterion is satisfied
end

Avec le package `sparsePCA` :

```
groupsparsePCA(A, m, lambda, index = 1:ncol(A),
                block = 1, mu = 1/1:m,
                center = TRUE, scale = TRUE)
```

- `A` est la matrice des données,
- `m` est le nombre de composantes,
- `lambda` est le vecteur des m paramètres λ_j de sparsité réduits,
- `index` définit les groupes de variables,
- `block` définit l'approche utilisée (`block=0` pour déflation et `block=1` pour bloc).
- `mu` est le vecteur des poids utilisés dans l'approche bloc.

Remarque

Les paramètres λ_j de la fonction `groupsparsePCA` sont des paramètres de sparsité réduit prenant leurs valeurs dans $[0, 1]$:

$$\lambda_j = \gamma_j / \gamma_{j,max} \quad , \quad j = 1 \dots m \ .$$

avec $\gamma_{j,max}$ le *paramètre de sparsité nominale* de chaque composante :

$$\gamma_{j,max} = \frac{\sigma_j}{\sigma_1} \gamma_{max} \quad \text{where} \quad \gamma_{max} \stackrel{\text{def}}{=} \max_{i=1\dots p} \|a_i\|_2 \ ,$$

Autres fonctions du package :

- la fonction `simuPCA`,
- la fonction `sparsePCA`,
- la fonction `explainedVar` (implémente 5 définitions de la variance expliquée par des composantes principales sparse),
- la fonction `pev` qui donne le pourcentage de la variance des données expliquée par chaque composante sparse,

Exemple : données simulées selon le modèle de Shen & Huang (2008)

Table: Z sous-jacent

z1	z2
1	0
1	0
1	0
1	0
0	1
0	1
0	1
0	1
0.9	-0.3
0.9	0.3

Table: Z normalisé

z1	z2
0.422	0.000
0.422	0.000
0.422	0.000
0.422	0.000
0.000	0.489
0.000	0.489
0.000	0.489
0.000	0.489
0.380	-0.147
0.380	0.147

```
library(sparsePCA)
z1 <- c(1,1,1,1,0,0,0,0,0.9,0.9)
z2 <- c(0,0,0,0,1,1,1,1,-0.3,0.3)
valp <- c(200,100,50,50,6,5,4,3,2,1)
A <- simuPCA(n=100,cbind(z1,z2),valp,seed=1) # matrice 100*10
A <- scale(A,center=T,scale=F)
index <- c(1,1,1,1,2,2,2,2,3,3) #définit les groupes de variables
```

Cas où $m = 1$ composante.

On prend comme paramètre de sparsité réduit $\lambda_1 = 0.3$.

```
groupsparsePCA(A,index,lambda=0.3,
               m=1)$Z
```

```
##         [,1]
##  [1,] 0.441
##  [2,] 0.430
##  [3,] 0.473
##  [4,] 0.445
##  [5,] 0.000
##  [6,] 0.000
##  [7,] 0.000
##  [8,] 0.000
##  [9,] 0.309
## [10,] 0.321
```

Les loadings du second groupe de variables sont bien mis à zéro et les loadings des autres groupes sont bien retrouvés.

Cas où $m = 2$ composantes et approche déflation.

On prend comme paramètre de sparsité réduit $\lambda_1 = 0.3$ et $\lambda_2 = 0.3$.

```
groupsparsePCA(A,index,lambda=c(0.3,0.3),
               m=2,block=0)$Z
```

```
##         [,1]  [,2]
##  [1,] 0.441 0.000
##  [2,] 0.430 0.000
##  [3,] 0.473 0.000
##  [4,] 0.445 0.000
##  [5,] 0.000 0.468
##  [6,] 0.000 0.511
##  [7,] 0.000 0.532
##  [8,] 0.000 0.487
##  [9,] 0.309 0.000
## [10,] 0.321 0.000
```

```
groupsparsePCA(A,index,lambda=0.3,
               m=1)$Z
```

```
##         [,1]
##  [1,] 0.441
##  [2,] 0.430
##  [3,] 0.473
##  [4,] 0.445
##  [5,] 0.000
##  [6,] 0.000
##  [7,] 0.000
##  [8,] 0.000
##  [9,] 0.309
## [10,] 0.321
```

Le premier vecteur de loadings est identique à celui trouvé avec $m = 1$. Le second met à zéro les loadings du premier groupe (à raison) et ceux des du troisième groupe (à tort).

Cas où $m = 2$ composantes et approche bloc.

On prend comme paramètres de sparsités réduit $\lambda_1 = 0.3$ et $\lambda_2 = 0.3$ et comme poids $\mu_1 = \mu_2 = 1$. On parle d'approche "block mu égaux".

```
groupsparsePCA(A,index,lambda=c(0.3,0.3),
               mu=c(1,1), m=2,block=1)$Z
```

```
##         [,1]  [,2]
##  [1,] 0.443 0.000
##  [2,] 0.432 0.000
##  [3,] 0.473 0.000
##  [4,] 0.446 0.000
##  [5,] 0.000 0.468
##  [6,] 0.000 0.508
##  [7,] 0.000 0.528
##  [8,] 0.000 0.495
##  [9,] 0.301 0.000
## [10,] 0.322 0.000
```

```
groupsparsePCA(A,index,lambda=0.3,
               m=1)$Z
```

```
##         [,1]
##  [1,] 0.441
##  [2,] 0.430
##  [3,] 0.473
##  [4,] 0.445
##  [5,] 0.000
##  [6,] 0.000
##  [7,] 0.000
##  [8,] 0.000
##  [9,] 0.309
## [10,] 0.321
```

On trouve des résultats très proches de ceux obtenus avec l'approche déflation. Mais avec l'approche bloc, le premier vecteur de loadings n'est plus exactement identique à celui trouvé avec $m = 1$.

On prend maintenant $\mu_1 = 1$ et $\mu_2 = 1/2$. On parle d'approche "block mu différents".

```
groupsparsePCA(A,index,lambda=c(0.3,0.3),
          mu=c(1,1/2), m=2,block=1)$Z
```

```
##         [,1]  [,2]
##  [1,] 0.441 0.000
##  [2,] 0.431 0.000
##  [3,] 0.473 0.000
##  [4,] 0.446 0.000
##  [5,] 0.000 0.467
##  [6,] 0.000 0.506
##  [7,] 0.000 0.526
##  [8,] 0.000 0.499
##  [9,] 0.306 0.000
## [10,] 0.321 0.000
```

Les résultats très proches de ceux obtenus avec l'approche "block mu égaux".

Données simulées selon le modèle de Chavent & Chavent (2017) avec les valeurs propres $(200, 180, 150, 130, 1 \ldots 1)$ et les loadings suivants :

Table: Ztrue

z1	z2	z3	z4
6	0	0	6
-6	0	0	6
6	0	0	6
-6	0	0	6
0	12	12	0
0	12	12	0
0	-12	12	0
0	-12	12	0
-5	8	0	5
-5	8	0	-5
5	8	0	5
5	8	0	-5
4	0	0	-10
4	0	0	-10
4	0	0	-10
4	0	0	-10
8	5	8	5
8	5	-8	5
8	-5	8	5
8	-5	-8	5

Dans ce modèle :

- $|p| = 20$ variables,
- $p = 5$ groupes de variables de taille 4,
- $m = 4$ composantes.

On va comparer la capacité des méthodes d'ACP group-sparse à retrouver les loadings nuls ?

On compare les approches "deflation", "block mu égaux", "block mu différents".

1. On simule avec la fonction `simuPCA` :
   - 100 matrices A de taille $n = 300$,
   - 100 matrices A de taille $n = 3000$.
2. On applique les trois approches avec la fonction `sparsegroupPCA` avec comme paramètres de sparsité :
   $$\lambda = \lambda_1 = \cdots = \lambda_m \ ,$$
   pour λ variant de 0 à 1 par pas de 0.01.
3. On compare les capacités des approches à retrouver les loadings nuls en calculant pour chaque dimension :
   - le *true positive rate* (tpr) : proportion de zéros dans Z_{true} retrouvés à 0 dans Z,
   - le *false positive rate* (fpr) : proportion de non zéros dans Z_{true} retrouvés à 0 dans Z.

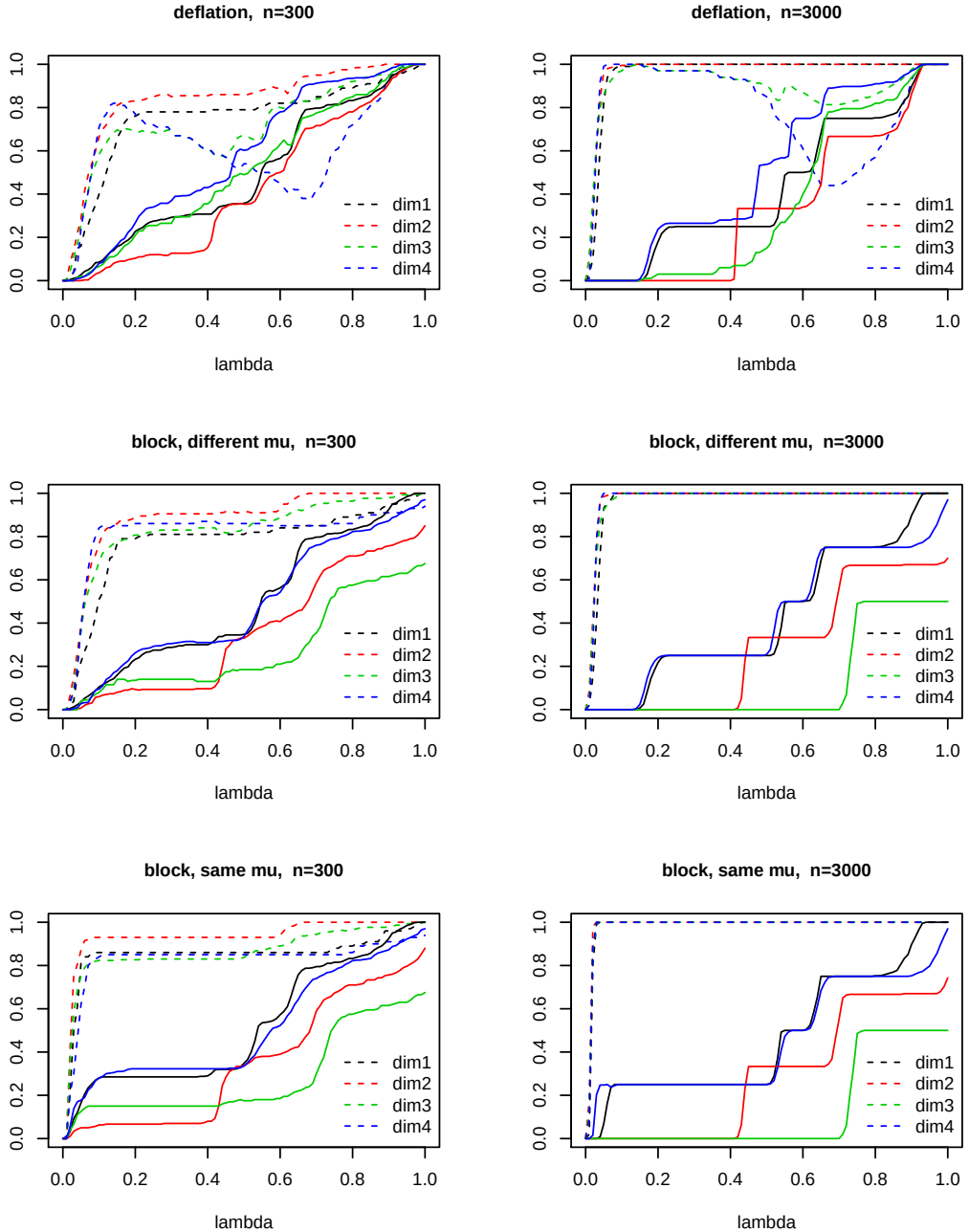

Figure: Mean true positive rates (dotted lines) and false positive rates (full lines) for each sparse loading versus reduced sparsity parameter λ.

Conclusion

- Inclure dans le package des fonctions `predict` pour pouvoir utiliser les composantes sparse dans le cadre de la régression sur composante principales.
- Inclure des métriques pour généraliser des données qualitatives ou mixtes (mélange de quantitatives et qualitatives).

Simulation d'une matrice A de dimension $n \times p$ selon un modèle de DVS :

1. On se donne $m \leq p$ vecteurs propres $v_1, \ldots, v_m$ (vecteurs singuliers de droite) et les valeurs propres $\sigma_1^2 \ldots \sigma_p^2$ d'une matrice de covariance C que l'on construit de la manière suivante :

$$C = V_{true} \Sigma_{true}^2 V_{true}^T$$

où :

- $\Sigma_{true}^{\mathbf{2}} = \mathrm{diag}(\sigma_{\mathbf{1}}^{\mathbf{2}} \ldots \sigma_{|p|}^{\mathbf{2}})$ est la matrice diagonale des valeurs propres,
- V_{true} est la matrice des vecteurs propres, obtenue en effectuant par la décomposition QR suivante

$$[v_{\mathbf{1}}, \ldots, v_m, U] = V_{true} R,$$

où U de dimension $p \times (p - m)$ est tiré aléatoirement selon une loi $U(0, 1)$.

2. On tire aléatoirement n observations selon une loi $N(0_p, C)$ pour obtenir A.

Remarque

La fonction `simuPCA` du package `sparsePCA` permet de simuler des données selon ce schéma.