Un algorithme rapide à identification polynomiale multivariable pour la séparation aveugle d'images

Johan Thomas, Yannick Deville, Shahram Hosseini

Laboratoire d'Astrophysique de Toulouse-Tarbes - Université Paul Sabatier Toulouse 3 - CNRS
14 av. Edouard Belin, 31400 Toulouse, France
jthomas,ydeville,shosseini@ast.obs-mip.fr

Résumé – Cet article a pour objet la séparation rapide de mélanges linéaires instantanés dans le cas d'observations contenant un grand nombre d'échantillons. Cette configuration est fréquente notamment en séparation d'images en raison de l'augmentation actuelle du nombre de pixels des capteurs de types CCD ou CMOS. Nous présentons ici un algorithme de type point fixe inspiré de FastICA, qui, grâce à une première étape d'identification polynomiale, permet d'éviter le calcul de statistiques à chaque itération de l'algorithme d'optimisation sous contrainte du kurtosis.

Abstract – This article deals with fast separation of linear instantaneous mixtures when the observations contain a great number of samples. This configuration is frequent, especially in image separation because of the present increase of the number of pixels in CCD and CMOS-type light sensors. We here present a fixed-point algorithm inspired from FastICA, which avoids the computation of statistics at each step of the constrained optimization algorithm, thanks to a first step based on polynomial identification.

1 Introduction

Un mélange linéaire instantané [1] est modélisé par l'équation matricielle

$$\mathbf{x}(n) = \mathbf{A}\mathbf{s}(n) \tag{1}$$

où $\mathbf{A}$ est une matrice de mélange de taille $N \times N$ et $\mathbf{s}(n)$ et $\mathbf{x}(n)$ sont respectivement les vecteurs de taille N des sources et des observations.

En plus des hypothèses classiques relatives à l'Analyse en Composantes Indépendantes (indépendance et non - gaussianité des sources, matrice $\mathbf{A}$ de rang de colonne plein), nous supposons ici que les observations contiennent un grand nombre d'échantillons (typiquement plus de 100 000, ce qui est très souvent le cas des images).

Nous supposons aussi que le nombre de sources n'est pas trop élevé (jusqu'à 7 typiquement), ce qui est souvent le cas, à l'exception toutefois de diverses applications biomédicales qui mettent en jeu un grand nombre de canaux enregistrés.

2 L'algorithme FastICA

Résumons l'algorithme FastICA basé sur le kurtosis, dont nous nous inspirons. Il est composé d'une première étape de blanchiment (appelée " sphering") effectuée par l'opération

$$\mathbf{z}(n) = \mathbf{R}_{\mathbf{x}}^{-\frac{1}{2}}\mathbf{x}(n) \tag{2}$$

où $\mathbf{R}_{\mathbf{x}}$ est la matrice d'autocorrélation du vecteur d'observations et $\mathbf{z}(n)$ est le nouveau vecteur d'observations, blanchi, qui vérifie l'équation $\mathbf{R}_{\mathbf{z}} = \mathbf{I}_N$.

Après avoir effectué le blanchiment décrit ci-dessus, l'algorithme FastICA maximise la valeur absolue du kurtosis de $\mathbf{w}^t\mathbf{z}(n)$, $kurt(\mathbf{w}^t\mathbf{z}(n))$, par rapport au vecteur d'extraction $\mathbf{w}$ sous la contrainte $\|\mathbf{w}\| = 1$[1]. Pour cela, la version parallèle de FastICA développée par Hyvärinen et Oja [4] adapte en même temps tous les vecteurs $\mathbf{w}_i$ d'extraction associés aux N sources, fournissant ainsi des signaux de sortie $y_i(n) = \mathbf{w}_i^t\mathbf{z}(n)$, et orthogonalise ces vecteurs $\mathbf{w}_i$ pour éviter qu'ils ne convergent vers des points identiques (au signe près). Cette orthogonalisation peut être réalisée par l'opération symétrique $\mathbf{W} = \mathbf{W}\left(\mathbf{W}^t\mathbf{W}\right)^{-\frac{1}{2}}$ où $\mathbf{W} = [\mathbf{w}_1, \cdots, \mathbf{w}_N]$ contient les N vecteurs d'extraction arrangés par colonnes.

L'algorithme FastICA symétrique peut donc être résumé par la procédure suivante :

- Blanchir les observations en utilisant (2).
- Initialiser $\mathbf{W}$ avec N vecteurs de taille N arrangés par colonnes (on peut choisir $\mathbf{W} = \mathbf{I}_N$).
- Répéter les étapes 1) et 2) ci-dessous jusqu'à la convergence :

  1) $\forall k = 1 \ldots N,\ \mathbf{w}_k \leftarrow E\left\{\mathbf{z}(n)(\mathbf{w}_k^t\mathbf{z}(n))^3\right\} - 3\mathbf{w}_k$
  $\propto \frac{\partial kurt(\mathbf{w}_k^t\mathbf{z}(n))}{\partial \mathbf{w}_k}$

  2) $\mathbf{W} \leftarrow \mathbf{W}\left(\mathbf{W}^t\mathbf{W}\right)^{-\frac{1}{2}}$

Cet algorithme est très populaire dans la communauté de séparation de sources en raison de sa rapidité et de son absence de paramètres d'optimisation. De plus, sa convergence a été prouvée rigoureusement par Oja et Yuan [5]. Cela explique son succès dans de nombreux domaines comme la séparation d'images en astronomie par exemple [6].

[1] Ce type de problème d'optimisation a été étudié dans [2, 3], mais pas spécifiquement dans le cas de longs enregistrements de mélanges.

Néanmoins, nous pouvons remarquer qu'une itération de l'algorithme à point fixe est très gourmande en termes de temps de calcul et d'espace mémoire, particulièrement lorsque le nombre d'échantillons T est grand (c'est souvent le cas en astronomie). Dans le logiciel Matlab FastICA [7], cette mise à jour est réalisée par l'opération matricielle suivante :

```
W = (Z * (( Z' * W) .^3)) / T - 3 * W
```  (3)

où `Z` est une matrice de taille $N \times T$ contenant les observations blanchies et `W` est une matrice de taille $N \times N$ contenant les vecteurs d'extraction. Comme l'opération (3) demande de stocker `Z` et `( Z' * W) .^3`, elle a besoin de $2TN$ cases mémoires. De plus on peut montrer que (3) nécessite le calcul de $T(4N^2+N)+2N^2$ opérations élémentaires (additions ou multiplications), ce qui est très pénalisant si le nombre d'échantillons T est grand.

3 Méthode proposée

Nous proposons ici une méthode qui évite cet important besoin en mémoire et calcul lors de l'optimisation. Notre méthode est basée sur l'identification d'un polynôme à plusieurs variables qui permet d'effectuer l'optimisation de type point fixe dans un espace de calcul plus simple. On évite aussi le calcul et le stockage du vecteur d'observations blanchies $\mathbf{z}(n)$ défini par (2).

Notons $y(n) = \mathbf{w}^t\mathbf{z}(n)$ une combinaison linéaire des observations blanchies de $\mathbf{z}(n)$ calculé dans l'algorithme FastICA (le calcul de $\mathbf{z}(n)$ sera évité dans notre approche comme expliqué plus bas). En supposant tous les signaux centrés, le kurtosis non normalisé de $y(n)$ s'écrit :

$$\begin{aligned} kurt(y(n)) &= kurt(\mathbf{w}^t\mathbf{z}(n)) \\ &= E\left\{(\mathbf{w}^t\mathbf{z}(n))^4\right\} - 3E\left\{(\mathbf{w}^t\mathbf{z}(n))^2\right\}^2 \end{aligned} \quad (4)$$

Comme nous avons

$$\begin{aligned} E\left\{(\mathbf{w}^t\mathbf{z}(n))^2\right\} &= E\left\{\mathbf{w}^t\mathbf{z}(n)\mathbf{z}^t(n)\mathbf{w}\right\} \\ &= \mathbf{w}^t E\left\{\mathbf{z}(n)\mathbf{z}^t(n)\right\}\mathbf{w} \\ &= \mathbf{w}^t\mathbf{I}\mathbf{w} \\ &= \|\mathbf{w}\|^2, \end{aligned} \quad (5)$$

(4) devient

$$\begin{aligned} &kurt(y(n)) \\ &= E\left\{\textstyle\sum_{i_1,..,i_4=1}^N \prod_{k=1}^4 w(i_k)z_{i_k}(n)\right\} - 3\left(\textstyle\sum_{i=1}^N w(i)^2\right)^2 \\ &= \sum_{i_1,..,i_4=1}^N E\left\{\prod_{k=1}^4 z_{i_k}(n)\right\}\prod_{k=1}^4 w(i_k) - 3\sum_{i_1,i_2=1}^N w(i_1)^2 w(i_2)^2. \end{aligned} \quad (6)$$

Comme (6) est un polynôme d'ordre 4 relativement aux variables $w(1),\cdots,w(N)$, nous pouvons définir un jeu de coefficients $(\alpha_{\mathbf{d}})_{\mathbf{d}\in D}$ avec $D = \left\{\mathbf{d}\in\{0,\cdots,4\}^N \setminus \sum_{k=1}^N d(k) = 4\right\}$ tel que

$$kurt\left(y(n)\right) = \sum_{\mathbf{d}\in D}\alpha_{\mathbf{d}}\prod_{k=1}^N w_k^{d(k)} \quad (7)$$

ce qui correspond au développement canonique du kurtosis non normalisé. Notons R le cardinal[2] de D, $\mathbf{d}_1,\cdots,\mathbf{d}_R$ ses éléments et $\alpha_1,\cdots,\alpha_R$ les valeurs de $(\alpha_{\mathbf{d}})_{\mathbf{d}\in D}$.

On a alors

$$kurt(y(n)) = \sum_{r=1}^R \alpha_r \prod_{k=1}^N w_k^{d_r(k)}. \quad (8)$$

Il serait possible de calculer directement les coefficients α_r qui dépendent de moments croisés d'ordre 4 des signaux $z_1(n),\cdots,z_N(n)$. Nous proposons une méthode plus rapide basée sur l'observation suivante. Notons $(\mathbf{v}_i)_{i=1..R}$ une famille de R vecteurs de taille N. D'après (8),

$$\forall i = 1\ldots R,\quad kurt(\mathbf{v}_i^t\mathbf{z}(n)) = \sum_{j=1}^R \alpha_j \prod_{k=1}^N \mathbf{v}_i(k)^{d_j(k)} \quad (9)$$

ce qui peut être réécrit sous la forme :

$$\mathbf{M}\boldsymbol{\alpha} = \mathbf{k} \quad (10)$$

où $\boldsymbol{\alpha}$ est le vecteur des coefficients $(\alpha_r)_{r=1..R}$ et où la matrice $\mathbf{M} = [m_{ij}]_{i,j=1..R}$ et le vecteur colonne $\mathbf{k} = (k_i)_{i=1..R}$ sont définis par

$$\begin{cases} \forall i,j = 1\ldots R, \quad m_{ij} = \prod_{k=1}^N v_i(k)^{d_j(k)} \\ \forall i = 1\ldots R, \quad k_i = kurt(\mathbf{v}_i^t\mathbf{z}(n)). \end{cases} \quad (11)$$

En choisissant R vecteurs d'extraction $\mathbf{v}_i$ de taille N qui fournissent une matrice $\mathbf{M}$ non singulière et en calculant les kurtosis des R signaux $\mathbf{v}_i^t\mathbf{z}(n)$, on peut donc identifier les R coefficients $(\alpha_r)_{r=1..R}$ grâce à l'inverse de (10), c.-à-d. $\boldsymbol{\alpha} = \mathbf{M}^{-1}\mathbf{k}$. Grâce aux propriétés du vecteur blanchi $\mathbf{z}(n)$ qui entraînent $E\left\{(\mathbf{v}_i^t\mathbf{z}(n))^2\right\} = \|\mathbf{v}_i\|^2$ comme démontré par (5), le kurtosis de $\mathbf{v}_i^t\mathbf{z}(n)$ vaut $kurt(\mathbf{v}_i^t\mathbf{z}(n)) = E\{(\mathbf{v}_i^t\mathbf{z}(n))^4\} - 3\|\mathbf{v}_i\|^4$ de telle sorte que l'on a un seul auto-moment d'ordre 4 à estimer pour chacun des R vecteurs d'extraction. De plus, la relation

$$\mathbf{v}_i^t\mathbf{z}(n) = \mathbf{v}_i^t\mathbf{R}_{\mathbf{x}}^{-1/2}\mathbf{x}(n) = \mathbf{u}_i^t\mathbf{x}(n) \quad (12)$$

avec $\mathbf{u}_i^t = \mathbf{v}_i^t\mathbf{R}_{\mathbf{x}}^{-1/2}$ permet d'éviter le calcul préalable de $\mathbf{z}(n)$. Dans la suite, nous noterons $\mathbf{V}$ et $\mathbf{U}$ les matrices contenant respectivement les ensembles de vecteurs $(\mathbf{v}_i)_{i=1..R}$ et $(\mathbf{u}_i)_{i=1..R}$ arrangés par colonnes.

Remarquons que $\mathbf{M}$ est indépendante des sources et du mélange. On peut donc calculer son inverse une fois pour toutes, pour chaque valeur considérée de N (jusqu'à 7 pour ce qui nous intéresse) et les stocker. Nous proposons dans l'Annexe un choix particulier de vecteurs $(\mathbf{v}_i)_{i=1..R}$ qui implique un bon conditionnement de la matrice $\mathbf{M}$.

Après avoir identifié le jeu de coefficients $(\alpha_r)_{r=1..R}$, nous cherchons à maximiser la valeur absolue de $kurt(\mathbf{w}^t\mathbf{z}(n)) = \sum_{r=1}^R \alpha_r \prod_{k=1}^N w_k^{d_r(k)}$ par rapport à $\mathbf{w}$ sous la contrainte $\|\mathbf{w}\| = 1$, comme dans le critère de FastICA. Ceci est réalisé d'une façon similaire à FastICA à l'aide d'un algorithme de type point fixe, mais dans un

[2] On peut montrer que $card(D) = R = N + \frac{3N(N-1)}{2} + \frac{N(N-1)(N-2)}{2} + \frac{N(N-1)(N-2)(N-3)}{24}$.

espace de calcul différent. Le gradient de $kurt(\mathbf{w}^t\mathbf{z}(n))$ par rapport à $\mathbf{w}$ dans notre espace s'écrit

$$\frac{\partial kurt(\mathbf{w}^t\mathbf{z}(n))}{\partial \mathbf{w}} = \sum_{r=1}^{R} \alpha_r \frac{\partial \prod_{k=1}^{N} w(k)^{d_r(k)}}{\partial \mathbf{w}} \qquad (13)$$

avec $\dfrac{\partial \prod_{k=1}^{N} w(k)^{d_r(k)}}{\partial w(j)} =$

$$\begin{cases} d_r(j) w(j)^{d_r(j)-1} \prod_{k \neq j} w(k)^{d_r(k)} & , \text{ si } \quad d_r(j) \geq 1 \\ 0 & , \text{ si } \quad d_r(j) = 0 \end{cases} \qquad (14)$$

Résumons maintenant notre algorithme parallèle de type point fixe optimisé pour les signaux longs, que nous baptisons O-FICA :

- Estimer la matrice d'autocorrélation $\mathbf{R_x}$ du vecteur d'observations $\mathbf{x}(n)$.
- Calculer $\mathbf{U} = \mathbf{R_x}^{-1/2}\mathbf{V}$ où $\mathbf{V}$ est formée de R vecteurs d'extraction $\mathbf{v}_i$ arrangés par colonnes (dans l'Annexe, nous proposons un jeu de vecteurs particulier).
- Calculer les kurtosis $(k_i)_{i=1..R}$ des R signaux $\mathbf{u}_i^t\mathbf{x}(n)$, $i = 1..R$ et déterminer ensuite le jeu de coefficients $\boldsymbol{\alpha} = (\alpha_r)_{r=1..R}$ grâce à la relation $\boldsymbol{\alpha} = \mathbf{M}^{-1}\mathbf{k}$ (au moyen de l'inverse de $\mathbf{M}$ calculée préalablement).
- Initialiser $\mathbf{W} = [w_{ij}]_{i,j=1..N}$ avec N vecteurs $\mathbf{w}_i$ de taille N arrangés par colonnes puis répéter 1) et 2) jusqu'à la convergence :

1) $\forall i, j = 1 \ldots N$,
$w_{ji} \leftarrow \sum_{r=1}^{R} \alpha_r d_r(j) w_{ji}^{max(d_r(j)-1,0)} \prod_{k \neq j} w_{ki}^{d_r(k)}$

ce qui est équivalent à faire $\forall i$, $\mathbf{w}_i \leftarrow \frac{\partial kurt(\mathbf{w}_i^t\mathbf{z}(n))}{\partial \mathbf{w}_i}$.

2) $\mathbf{W} \leftarrow \mathbf{W}\left(\mathbf{W}^t\mathbf{W}\right)^{-\frac{1}{2}}$

Contrairement à l'algorithme FastICA, notre méthode ne calcule pas de statistiques à chaque itération de l'optimisation à point fixe. On ne manipule ici que les coefficients de polynôme α_r, ce qui est plus rapide pour T grand, car leur nombre est indépendant de T (ces coefficients ne doivent pas être trop nombreux à identifier toutefois, ce qui est le cas pour $N \leq 7$ typiquement).

De plus on évite de stocker `Z` et `(Z' *W).^3` qui représentent $2TN$ cases mémoire dans l'algorithme FastICA. Dans notre cas, on peut montrer que nous n'utilisons que $R + RN + R^2 + N^2$ cases mémoire, valeur indépendante du nombre d'échantillons T.

Notre algorithme effectuant la même opération de mise à jour que FastICA (dans un espace de calcul différent), il atteint à chaque itération le même point pour chacun des vecteurs d'extraction et permet donc d'obtenir les mêmes performances de séparation en un temps plus court.

La version à déflation de l'algorithme O-FICA peut être obtenue facilement en remplaçant dans la procédure ci-dessus la matrice $\mathbf{W} = [w_{ij}]_{i,j=1..N}$ par un unique vecteur $\mathbf{w} = (w_j)_{j=1..N}$. Après avoir estimé une source au moyen de ce vecteur d'extraction, ses contributions dans les observations sont estimées par le calcul de coefficients d'intercorrélation entre signal extrait et observations, puis soustraites des observations afin de réitérer la procédure pour un mélange comportant une source de moins.

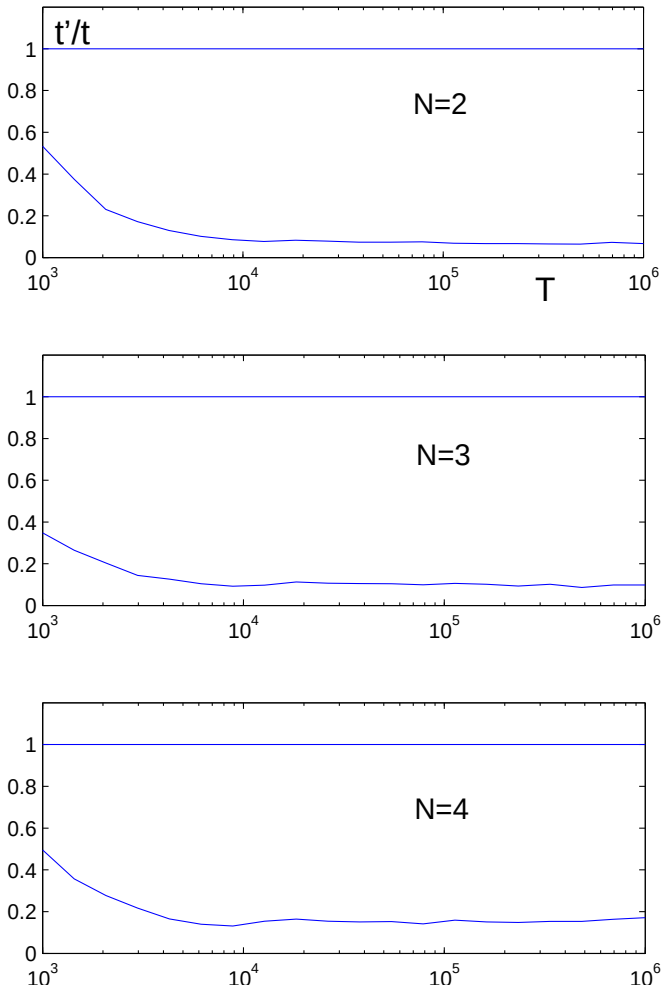

FIG. 1: Rapport des temps de calcul en fonction du nombre T d'échantillons pour un nombre N de sources variant de 2 à 4.

4 Résultats expérimentaux

Dans cette section, nous présentons des résultats expérimentaux qui comparent la vitesse de notre algorithme O-FICA avec celle de l'algorithme FastICA.

Nous avons mesuré les temps de calcul des algorithmes jusqu'à ce qu'ils atteignent le critère d'arrêt qui est utilisé dans le logiciel Matlab FastICA. Ce critère compare les directions des vecteurs d'extraction après les deux dernières itérations de l'algorithme d'optimisation et stoppe l'optimisation si ces deux directions diffèrent moins qu'un certain seuil (nous avons choisi comme seuil celui pris par défaut dans le logiciel à savoir 10^{-4}).

Les Figures (1) et (2) représentent la valeur moyenne de t'/t en fonction du nombre d'échantillons T pour différentes valeurs de N, où t' et t sont respectivement les temps de calcul des algorithmes O-FICA et FastICA. Pour chaque configuration, nous avons fait 100 tests de Monte-Carlo et avons moyenné les valeurs de t'/t obtenues. Nous pouvons voir sur les Figures (1) et (2) que pour un nombre de sources N inférieur ou égal à 7, le rapport t'/t est inférieur à 1 quand le nombre d'échantillons T est assez grand et décroît (jusqu'à 1/15 dans certains cas) quand T augmente.

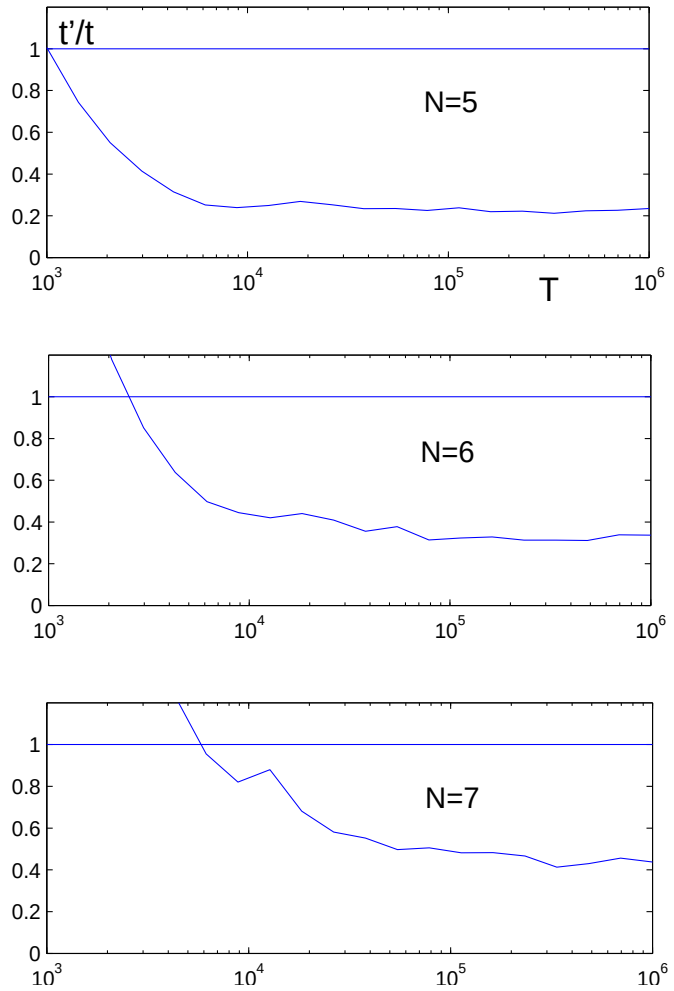

FIG. 2: Rapport des temps de calcul en fonction du nombre T d'échantillons pour un nombre N de sources variant de 5 à 7.

5 Conclusion

Nous avons présenté une nouvelle méthode pour optimiser le kurtosis non normalisé sous contrainte de puissance unité. Nous basons notre approche sur une étape d'identification polynomiale multivariable qui nous permet d'utiliser un algorithme de type point fixe dans un espace de calcul plus performant. Notre algorithme O-FICA est intéressant en particulier quand les signaux enregistrés contiennent un grand nombre d'échantillons, ce qui est souvent le cas en séparation aveugle d'images. Comparé à l'algorithme FastICA, notre algorithme est jusqu'à 15 fois plus rapide pour certaines configurations.

Annexe : Choix de la famille de vecteurs d'extraction

Définissons les 5 ensembles suivants :

$$E_1 = \{\mathbf{w} \in \{0,1\}^N \ \setminus \ \exists i_1 \quad t.q. \\ w(i_1) = 1, \forall i \neq i_1 \ \ w(i) = 0\}$$

$$E_2 = \{\mathbf{w} \in \{0,1\}^N \ \setminus \ \exists i_1, i_2 \quad t.q. \\ w(i_1) = w(i_2) = 1, \forall i \neq i_1, i_2 \ w(i) = 0\}$$

$$E_3 = \{\mathbf{w} \in \{0,1,2\}^N \ \setminus \ \exists i_1, i_2 \quad t.q. \\ w(i_1) = 2w(i_2) = 2, \forall i \neq i_1, i_2 \ w(i) = 0\}$$

$$E_4 = \{\mathbf{w} \in \{0,1,2\}^N \ \setminus \ \exists i_1, i_2, i_3 \quad t.q. \\ w(i_1) = 2w(i_2) = 2w(i_3) = 2, \forall i \neq i_1, i_2, i_3 \ w(i) = 0\}$$

$$E_5 = \{\mathbf{w} \in \{0,1\}^N \ \setminus \ \exists i_1, \cdots, i_4 \quad t.q. \\ w(i_1) = \ldots = w(i_4) = 1, \forall i \neq i_1, \cdots, i_4 \ w(i) = 0\}.$$

Les cardinaux de $E_1, \ldots, E_5$ sont respectivement N, $\frac{N(N-1)}{2}$, $N(N-1)$, $\frac{N(N-1)(N-2)}{2}$, $\frac{N(N-1)(N-2)(N-3)}{24}$. En notant $E = \cup_{i=1}^5 E_i$, $\texttt{card}(E) = \sum_{i=1}^5 card(\mathbf{E}_i) = N + \frac{3N(N-1)}{2} + \frac{N(N-1)(N-2)}{2} + \frac{N(N-1)(N-2)(N-3)}{24} = \texttt{card}(D)$.

De plus, nous avons vérifié numériquement que cette famille donne une matrice $\mathbf{M}$ définie par (11) non singulière pour un nombre de sources $N \leq 7$. Les conditionnements respectifs de $\mathbf{M}$ (définis par le rapport de la plus grande et de la plus petite valeurs propres de $\mathbf{M}$) pour $N = 2, \cdots, 7$ valent en effet $182, 414, 844, 1605, 2758, 4344$, ce qui assez faible pour les valeurs de R associées (respectivement $5, 15, 35, 70, 126, 210$). Par exemple, le conditionnement moyen de matrices de dimension 210 avec des coefficients uniformément distribués entre 0 et 1 est supérieur à 50000. Nous avons donc une famille de vecteurs qui peut être utilisée pour identifier le jeu de coefficients $(\alpha_r)_{r=1..R}$ défini par (8).

References

[1] A. Hyvärinen, J. Karhunen et E. Oja, "Independent Component Analysis", John Wiley & Sons, 2001.

[2] E. Moreau, "Criteria for Complex Sources Separation", Ds Actes *European Signal Processing Conference (EUSIPCO'96), Trieste, Italie*, vol. 2, p. 931-934, Sept. 1996.

[3] V. Zarzoso et P. Comon, "How fast is FastICA?", Ds Actes *European Signal Processing Conference (EUSIPCO'06), Florence, Italie*, Sept. 2006.

[4] A. Hyvärinen et E. Oja, "A Fast Fixed-Point Algorithm for Independent Component Analysis", *Neural Computation*, vol. 9, p. 1483-1492, Oct. 1997.

[5] E. Oja et Z. Yuan, "The FastICA Algorithm Revisited: Convergence Analysis", *IEEE Trans. on Neural Networks*, vol. 17, p. 1370-1381, Nov. 2006.

[6] M. Funaro, E. Oja et H. Valpola, "Independent component analysis for artefact separation in astrophysical images", *Neural Networks*, vol. 16, p. 469-478, 2003.

[7] H. Gävert, J. Hurri, J. Särelä et A. Hyvärinen, "FastICA for Matlab 7.x and 6.x", Version 2.5, Oct. 2005. http://www.cis.hut.Fi/projects/ica/fastica.