Matching Pursuit et Orthogonal Matching Pursuit

Ch. Dossal

Avril 2013

1 Introduction

Le Matching Pursuit (MP) et sa variante orthogonale, le Matching Pursuit Orthogonal (OMP) sont deux méthodes classiques largement utilisées dans différents domaines du traitement du signal : approximation, débruitage, problèmes inverses, détection, séparation de sources etc ...
Le problème de base où apparait le MP est le suivant : On dispose d'un signal X et d'un dictionnaire de signaux $A = (a_i)_{i \leqslant p}$ de la taille de X et on souhaite écrire X comme une combinaison linéaire d'éléments a_i de A ou au moins approcher X par une telle combinaison linéaire.
Le MP propose une approche simple : On estime l'atome a_{i_0} le plus corrélé à X, on soustrait à X sa projection sur a_{i_0} et on recommence l'opération sur le résidu $X - \lambda a_{i_0}$ jusqu'à un critère d'arrêt fixé par l'utilisateur. Dans l'OMP on sélectionne également les atomes a_i un à un mais à chaque nouvelle sélection d'atome, on recalcule la projection de X sur l'espace vectoriel engendré par les vecteurs sélectionnés. Quand X s'écrit vraiment comme une combinaison linéaire des a_i, l'OMP peut converger rapidement vers une erreur nulle, l'erreur commise par le MP sera elle rarement nulle.
Nous proposons ici deux applications du MP. Dans un premier temps nous verrons comment cette technique permet d'effectuer de la séparation sur des petits signaux 1D et nous comparerons le MP et l'OMP. Dans un second temps nous utiliserons le MP pour de la compression et du débruitage de signaux musicaux.

2 MP et OMP pour la séparation

Le but de cette section est de séparer un signal S somme de Diracs et de sinusoides en une somme de Diracs d'un part et et une somme de sinusoides d'autre part. Nous évaluerons également la sensibilité au bruit.

1. Ecrire un programme GenereMatSinussoide

   ```
   A=GenereMatSinusoide(N)
   ```

   qui génère une matrice de taille $N \times N$ contenant une base de cosinus discrets **On pourra utiliser la dct.**

2. A l'aide de la fonction précédente et par concaténation avec la matrice identité I_N, créer une matrice **avec des colonnes normées**, de taille $N \times 2N$ une matrice dont les colonnes sont des Diracs ou des sinusoides.

3. Ecrire un programme

`X=Genevect(d,N1)`

qui génère un vecteur colonne de longueur $N1$, ayant exactement d composantes non nulles. Chaque composante étant la réalisation d'une variable aléatoire gaussienne.

4. A l'aide des deux fonctions précédentes créer un mélange d'un petit nombre de Diracs et de sinusoides. Afficher sur trois figures différentes la partie Diracs, la partie sinusoides et la sommes. On prendra de préférence des signaux de taille $N = 1024$.

5. Ecrire un programme

   `X=MP(Y,A,er)`

   qui prend en entrée, un vecteur Y de taille N, une matrice de taille $N \times 2N$ et un réel positif er qui renvoie un vecteur X de taille X tel que $\|Y - AX\|_2^2 \leqslant er$ en utilisant le MP.

6. Ecrire un programme utilisant le MP qui à partir d'un mélange de sinusoides et de Diracs, sépare et affiche les deux composantes et qui calcule l'erreur de reconstruction.

7. Tester la fonction précédente à partir d'un mélange de 3 Diracs ou sinusoides. Puis augmenter la taille du mélange pour voir jusqu'à quel niveau de mélange, le MP arrive Ãă faire la séparation.

8. Ecrire un programme

   `[er1,er2]=MPtest(Y,A,X,NBmax)`

   prenant en entrée un vecteur Y de taille N, une matrice A de taille $N \times 2N$, un vecteur X de taille $2N$ et un nombre d'étapes NBmax et qui renvoie et affiche les vecteurs er1 d'erreur de reconstruction par rapport aux observations Y et er2 d'erreur de reconstruction par rapport aux données *réelles* X en fonction du nombre d'étapes.

9. Tester la fonction précédente avec plusieurs tailles de mélanges et différents niveaux de bruit. On utilisera un modèle de bruit gaussien sur les observations.

10. A partir de ces courbes, peut on déduire un critère d'arrêt pertinent ?

11. Proposer un critère d'arrêt théorique et une manière pratique de le mettre en place. Ce critère permet il d'avoir de bon résultat ?
    D'une manière générale dans les méthodes itératives de reconstruction il est important de savoir quand s'arrêter. Si on s'arrête trop tôt on ne récupère pas assez de signal, si on s'arrête trop tard on a une représentation moins compacte et surtout quand on fait du débruitage on risque de récupérer trop de bruit.

12. Faire le travail correspondant pour l'OMP et comparer les deux méthodes à divers niveaux de parcimonie et différents niveaux de bruit. On commencera par rédiger la fonction

    `X=OMP(Y,A,er)`

    Pour l'OMP on a besoin de la pseudo inverse d'une matrice ...

13. Comparer les deux algorithmes.

3 Matching Pursuit pour l'inpainting

Le MP et l'OMP peuvent être utilisé dans de nombreuses applications, comme la déconvolution ou l'inpainting. Nous allons voir maintenant la manière de l'utiliser pour l'inpainting. L'idée du MP pour l'inpainting est de rechercher parmi tous les vecteurs qui correspondent aux observations, celui qui s'exprime avec le moins de composantes dans une famille de vecteurs donnée. Ainsi si le vecteur recherché s'exprime réellement avec peu de coefficients, le MP ou l'OMP aura une chance de le retrouver, si ce n'est pas le cas, algorithme fournira une solution qui peut ne pas avoir d'intérêt. Pour illustrer ce phénomène, on va chercher à reconstruire un vecteur masqué comme une somme d'un petit nombre de cosinus discrets. Si le vecteur original s'exprime de fait avec peu de coefficients dans cette base on peut espérer le retrouver. D'une manière plus générale si on veut utiliser le MP pour faire de l'inpainting, il faut avoir une idée d'une base ou d'une famille ou le vecteur cible est *parcimonieux*.
On considère un vecteur de taille N que l'on va masquer partiellement et aléatoirement.

14. A l'aide de la commande *rand* écrire un programme matlab qui prend en entrée un entier N et un réel a et qui renvoie un masque aléaotire M, de taille N dont les composantes valent 0 ou 1 et qui comprend environ une proportion a de 0.
Dans la suite on considère qu'on cherche à estimer un vecteur x à partir de $y = Mx$ où y est le signal masqué. Dans la suite on utilisera un vecteur x qui est une combinaison aléatoire d'un petit nombre de cosinus discrets que l'on peut construire avec la fonction *GeneVect*. On pourra commencer par une parcimonie (nombre de coefficients non nuls) de 5.

15. Construire une matrice A dont chaque colonne correspond à un vecteur de base de la dct masquée par le masque M.

16. Décomposer le vecteur y masqué dans la famille des colonnes de A à l'aide du MP ou de l'OMP.

17. Par une dct inverse reconstruire le signal. Comparer votre reconstruction au signal original.

18. Faire différents tests avec différents niveaux de masquage et différentes parcimonies.

19. Que se passe t il si le vecteur x original n'est pas assez parcimonieux ? ou si le masquage est trop important.

20. Proposer un code qui permet de faire de l'inpainting sur une image carré 8×8.

21. Proposer un code matlab qui fait un inpainting sur une image complète en utilisant une dct locale.

22. Faire un test avec Lenna et un un masque à 50 pourcents.

23. Proposer une version invariante par translation.