Complément 1. La traque suivant un dictionnaire

Algorithmes de Matching Pursuit et décomposition orthogonale propre

Si l'orthogonalité est une notion clé dans tout processus d'analyse et de synthèse en théorie de l'information, c'est malheureusement une notion *fragile* : par exemple le procédé d'orthonormalisation de Gram-Schmidt dans $\mathbb{R}^n$ (proposition **??** dans ce chapitre),mis en œuvre à partir d'un système libre au sein duquel deux vecteurs sont *presque* colinéaires, peut générer des combinaisons de ces vecteurs de base, affectés d'énormes coefficients ! Pour prendre un exemple plus concret, il est souvent utile aujourd'hui, face aux questions de sécurité ou de copyright, d'authentifier une image en y surimprimant un code (une *marque*[1]) que seul son propriétaire est à même de déceler. Si la réalisation préalable de décompositions orthogonales permet de *signer* l'image en codant intelligemment les divers composants de l'une de ces décompositions (par exemple l'une des décompositions espace-échelles évoquées dans la sous-section III.2 du chapitre 10), il n'est aucunement évident que l'orthogonalité entre composants soit un tant soit peu préservée lorsque l'image subit un quelconque traitement (compression, modification géométrique, filtrage, etc.). Dès lors, le processus d'authentification de l'image perturbée par son propriétaire devient impossible ! Voici encore un second exemple : en imagerie médicale, il est fréquemment utile, plutôt que de tenter d'inverser le mécanisme tomographique $\mathcal{R}$ qui a permis d'obtenir une image I, de chercher à *pister* cette image contre un dictionnaire (éventuellement redondant, mais on veillera à limiter cela) d'images $\mathcal{R}[f_j]$, où $f_1, ..., f_N$ sont des états sources pathologiques auxquels on aimerait confronter l'état f dont on ne connaît que $\mathcal{R}[f]$. On pressent en effet que f s'apparente à une combinaison de ces états pathologiques f_j, $j = 1, ..., N$, combinaison qu'il serait judicieux de retrouver sous forme *organisée* : d'abord la pathologie (parmi les f_j) la mieux *corrélée* à f, puis celle qui, bien qu'aussi corrélée signicativement à f, l'est un peu moins, etc. Ce sont ces idées que nous allons esquisser dans ce complément.

1.1. Dictionnaires et Matching Pursuit

Étant donné, dans un espace de Hilbert, un élément dont on souhaite extraire une information (aux fins d'analyse, de classification ou de synthèse ultérieure), l'un des algorithmes les plus naïfs (mais de fait aussi les plus robustes) que l'on puisse imaginer est celui qui consiste à *pister* h contre un *dictionnaire* que l'on aura au préalable composé d'*éléments tests* dont on s'attend a priori que l'élément h proposé soit une combinaison linéaire. Il peut se révéler utile (on le verra plus loin) d'élaborer un dictionnaire d'*éléments tests* à partir de l'élément h que l'on prétend analyser. On donnera, dans cette section, deux versions (l'une élémentaire, l'autre un peu plus élaborée) de l'algorithme mathématique sous-tendant ce scénario, dit algorithme de *Matching Pursuit*[2].

Le cadre proposé est le cadre hilbertien des espaces $H = L^2_{\mathbb{K}}(\mathbb{R})$ ou $l^2_{\mathbb{K}}(\mathbb{Z})$ avec $\mathbb{K} = \mathbb{R}$ ou $\mathbb{C}$; on pourrait tout aussi bien envisager $L^2_{\mathbb{K}}(\mathbb{R}^n)$ ou $l^2_{\mathbb{K}}(\mathbb{Z}^n)$. Il faut aussi ajouter que, même dans le cadre de la dimension finie, $H = \mathbb{R}^N$ ou $H = \mathbb{C}^N$, le recours à pareille démarche se justifie.

On convient d'appeler *dictionnaire* une liste finie ou dénombrable d'éléments de H engendrant un sous-espace dense ; pour simplifier, on suppose tous les éléments du dictionnaire normalisés et de norme égale à 1. Étant donné un tel dictionnaire $\mathcal{D}$ et un élément h de H, l'algorithme le plus simple proposé (M. P. pour *Matching Pursuit*) repose sur la proposition suivante.

---

1. C'est ce que l'on appelle le *watermarking*.

2. La terminologie anglo-saxonne résume l'objectif : *traquer* (ou *poursuivre*) en essayant d'*ajuster* aux combinaisons d'éléments du dictionnaire (c'est la phase de *matching*).

Proposition 9.1. *Soient $\mathcal{D}$ un dictionnaire d'éléments de H et h un élément de H. On suppose que l'on peut construire de manière itérative une suite d'éléments $d_1, \dots, d_k, \dots$ du dictionnaire $\mathcal{D}$ (les d_j dépendant de h), une suite de scalaires $\lambda_1, \dots, \lambda_k, \dots$ (dépendant aussi de h) tels que $|\langle h\,,\, d_1\rangle| = \max_{d\in\mathcal{D}} |\langle h\,,\, d\rangle|$, $\lambda_1 d_1 = \mathrm{Proj}_{\mathbb{K}d_1}(h)$ et, pour tout $k \geq 1$,*

$$\left|\left\langle h - \sum_{l=1}^{k} \lambda_l d_l\,,\, d_{k+1}\right\rangle\right| = \max_{d\in\mathcal{D}} \left|\left\langle h - \sum_{l=1}^{k} \lambda_l d_l\,,\, d\right\rangle\right| \text{ et } \lambda_{k+1} d_{k+1} = \mathrm{Proj}_{\mathbb{K}d_{k+1}}\left(h - \sum_{l=1}^{k} \lambda_l d_l\right).$$

Alors, la suite

$$(h_k)_{k\geq 1} \;:=\; \left(\sum_{l=1}^{k} \lambda_l d_l\right)_{k\geq 1}$$

converge vers h dans H ; on dit qu'elle réalise le matching de h suivant le dictionnaire $\mathcal{D}$.

Remarque. Comme le dictionnaire $\mathcal{D}$ peut fort bien être redondant (on a d'ailleurs tout intérêt à ce qu'il en soit ainsi), la décomposition de h suivant $\mathcal{D}$ n'a rien d'unique. La méthode proposée dans ce scénario algorithmique fournit simplement une décomposition efficace en *traquant* (de manière *hiérarchique*) les éléments tests du dictionnaire qui comptent le plus dans la représentation de h.

Preuve. Posons $\nu_l := \lambda_l d_l$ pour $l \in \mathbb{N}^*$ et notons r_k le reste de la décomposition de h une fois l'étape k de l'algorithme effectuée, soit $r_k := h - \sum_{l=1}^{k} \nu_l$, partant de $r_0 = h$. Si $1 \leq k < p$, on a $r_k = r_p + \sum_{l=k+1}^{p} \nu_l$. Comme on effectue à chaque cran de l'algorithme une projection orthogonale unidimensionnelle sur la droite vectorielle engendrée par le nouvel élément du dictionnaire trouvé, on a, pour tout $l \in \mathbb{N}^*$, $\|r_k\|^2 = \|r_{k-1}\|^2 - |\langle r_{k-1}\,,\, d_k\rangle|^2$. La suite $(\|r_k\|^2)_{k\geq 0}$ est donc une suite décroissante minorée, donc convergente ; on en déduit aussi

$$\sum_{l=1}^{\infty} \|\nu_l\|^2 < \infty\,. \tag{9.1}$$

Pour montrer que la suite $(r_k)_{k\geq 0}$ converge, nous allons prouver qu'elle est de Cauchy. Fixons donc $\epsilon > 0$. On a, si $1 \leq k < p$, $\|r_k - r_p\|^2 = \|r_k\|^2 + \|r_p\|^2 - 2\|r_p\|^2 - 2\sum_{l=k+1}^{p} \mathrm{Re}\,\langle r_p\,,\, \nu_l\rangle$; mais on a, pour tout l entre $k+1$ et p,

$$|\langle r_p\,,\, \nu_l\rangle| \leq \|\nu_l\| \times \|\nu_{p+1}\|\,. \tag{9.2}$$

En effet, toujours pour l entre $k+1$ et p, on a

$$|\langle r_p\,,\, \nu_l\rangle| = \left|\left\langle r_p\,,\, \langle r_{l-1}\,,\, d_l\rangle\, d_l\right\rangle\right| = |\langle r_p\,,\, d_l\rangle| \times |\langle r_{l-1}\,,\, d_l\rangle| = \|\nu_l\|\,|\langle r_p\,,\, d_l\rangle| \leq \|\nu_l\| \times \|\nu_{p+1}\|$$

du fait même du principe de l'algorithme, puisque $|\langle r_p\,,\, d_{p+1}\rangle|$ (qui est par définition la norme de ν_{p+1}) maximise tous les $|\langle r_p\,,\, d\rangle|$ avec $d \in \mathcal{D}$. On a donc

$$\|r_k - r_p\|^2 \leq \|r_k\|^2 - \|r_p\|^2 + 2\|\nu_{p+1}\| \sum_{l=k+1}^{p} \|\nu_l\|\,, \tag{9.3}$$

or $\|\nu_{p+1}\| \sum_{l=k+1}^{p} \|\nu_l\| \leq \|\nu_{p+1}\| \sum_{l=1}^{p+1} \|\nu_l\|$; mais la clause (9.1) implique

$$\lim_{p\to\infty} \left(\inf_{q\geq p} \left(\|\nu_q\| \sum_{l=1}^{q} \|\nu_l\|\right)\right) = 0\,. \tag{9.4}$$

Si k est assez grand, on est donc certain que $\|r_k\|^2 - \|r_p\|^2 \leq \epsilon$; on choisira k assez grand pour qu'il en soit ainsi. Il résulte de (9.4) qu'il existe $q > p$ tel que $\|\nu_q\| \sum_{l=1}^{q} \|\nu_l\| \leq \epsilon$, mais on a $\|r_k - r_p\| \leq \|r_k - r_q\| + \|r_p - r_q\|$. En appliquant (9.3) (mais cette fois avec les couples (k,q) et (p,q) au lieu de (p,k)), on voit que $\max(\|r_k - r_q\|^2, \|r_p - r_q\|^2) \leq 3\epsilon$. En mettant tout ensemble, on conclut que $\|r_k - r_p\| \leq 2\sqrt{3\epsilon}$ pour ce choix de k, ce qui implique bien que la suite $(r_k)_{k\geq 0}$ est de Cauchy, donc convergente. Mais on a aussi la convergence vers 0 de $\|\nu_k\|$ lorsque k tend vers l'infini ; on a donc $\lim_{k\to\infty} |\langle r_k \, , \, d_{k+1}\rangle| = 0$. Le principe de l'algorithme (appliqué une fois de plus) nous assure que pour tout élément d du dictionnaire, $\lim_{k\to\infty} |\langle r_k \, , \, d\rangle| = 0$. Comme le dictionnaire forme une partie totale, la limite de la suite $(r_k)_{k\geq 0}$ doit être orthogonale à tous les atomes du dictionnaire, est par conséquent nulle, et la proposition est démontrée. ■

Cet algorithme est entaché d'un défaut : il faut à chaque étape travailler avec le dictionnaire complet et l'on ne peut se permettre d'éliminer les *éléments tests* dès lors qu'ils sont apparus. Pour corriger cet état de fait et transformer l'algorithme de Matching Pursuit en un algorithme *glouton*, on peut en introduire une version orthogonale en le couplant avec le procédé d'orthonormalisation de Gram-Schmidt : c'est l'algorithme *Matching Pursuit Orthogonal* (M. P. O.) dont on présente ici la description. Il s'agit, outre la démarche décrite dans la proposition 9.1, d'imposer à chaque itération (à savoir la détection de d_{k+1}) l'orthogonalité du reste avec les éléments du dictionnaire précédemment sélectionnés. Pareil algorithme nécessite bien sûr, dès cet élément test d_{k+1} déterminé, un réajustement des coefficients du *résumé* $\sum_{l=1}^{k} \alpha_l^{(k)} d_l$ dont on disposait avant que la recherche de d_{k+1} ne soit mise en route.

Voilà en quelques points la démarche algorithmique que génèrent les diverses opérations impliquées dans cette variante par la détection du $(k+1)$-ième atome. Supposons que, précédant cette étape, on dispose d'une liste $d_1, ..., d_k$ d'éléments du dictionnaire sélectionnés (aux k-crans précédents), d'un *résumé* $R_k := \sum_{l=1}^{k} \alpha_l^{(k)} d_l$, et de l'inverse de la matrice de Gram G_k des atomes $d_1, ..., d_k$ [3]. On effectue alors la détection de l'élément test d_{k+1} suivant la règle

$$\left|\left\langle h - \sum_{l=1}^{k} \alpha_l^{(k)} d_l \, , \, d_{k+1} \right\rangle\right| = \max_{d\in\mathcal{D}, d\neq d_1,\ldots,d_k} \left|\left\langle h - \sum_{l=1}^{k} \alpha_l^{(k)} d_l \, , \, d \right\rangle\right| .$$

On calcule ensuite le vecteur-ligne ${}^tC_k$ des corrélations (i.e. produits scalaires) de ce nouvel élément test d_{k+1} avec les k éléments $d_1, ..., d_k$ du dictionnaire précédemment sélectionnés,

$${}^tC_k := (\langle d_{k+1} \, , \, d_1\rangle, \ldots, \langle d_{k+1} \, , \, d_k\rangle) ,$$

puis le vecteur-colonne $B_k = G_k^{-1} C_k$ de coordonnées $B_k(l)$, $l = 1, ..., k$; notons que B_k et C_k seront stockés en mémoire. On calcule enfin le coefficient $\alpha_{k+1}^{(k+1)}$ appelé à affecter le nouvel atome sélectionné d_{k+1} dans la nouvelle version R_{k+1} du résumé, selon la formule

$$\alpha_{k+1}^{(k+1)} = \frac{\langle h - R_k \, , \, d_{k+1}\rangle}{1 - \sum_{l=1}^{k} B_k(l) \langle d_l \, , \, d_{k+1}\rangle} ,$$

puis on effectue les réajustements nécessaires sur les coefficients $\alpha_1^{(k)}, ..., \alpha_k^{(k)}$ pour obtenir l'orthogonalité du nouveau reste $h - R_{k+1}$ avec les *éléments tests* $d_1, ..., d_{k+1}$. Ces réajustements conduisent à l'expression du nouveau résumé

$$R_{k+1} = \sum_{l=1}^{k} (\alpha_l^{(k)} - \alpha_{k+1}^{(k+1)} B_k(l)) d_l + \alpha_{k+1}^{(k+1)} d_{k+1} .$$

3. Ce calcul, on le verra plus loin, aura été effectué de manière récursive (via une information mémorisée lors de l'itération précédente).

Avant de procéder à l'étape suivante, on injecte les données jusque-là stockées B_k, C_k, G_k^{-1} pour préparer l'itération ultérieure, en l'occurrence calculer l'inverse de la matrice de Gram G_{k+1} des *éléments tests* $d_1, ..., d_{k+1}$; ce calcul s'effectue suivant la formule

$$G_{k+1}^{-1} = \begin{pmatrix} G_k^{-1} + \rho_k B_k B_k^* & -\rho_k B_k \\ -\rho_k B_k^* & \rho_k \end{pmatrix}, \quad \text{avec} \quad \rho_k := \frac{1}{1 - C_k^* B_k}.$$

Remarque. Des routines (`mp`, `mpo`, `mpo1`), écrites sous l'environnement MATLAB, et correspondant à ces deux algorithmes (M. P. et M. P. O.), sont proposées dans le dossier `Chap01-C1/MATCH` que l'on trouvera dans le support accompagnant cet ouvrage.

1.2. La décomposition orthogonale propre

Comme nous l'avons mentionné dans la section précédente, la réalisation préalable d'un dictionnaire, à partir précisément de l'élément que l'on prétend analyser, peut être un outil utile. C'est ce dictionnaire qui servira à la mise en œuvre des algorithmes décrits dans la section précédente.

Exemple 9.2. Si s est un signal digital $(s(1), ..., s(N))$ de longueur N, on peut, en s'inspirant de la démarche conduisant par exemple au calcul de la fonction d'autocorrélation et de la densité spectrale de puissance (méthode de Welch décrite dans la sous-section II.1.4 du chapitre 10), considérer comme dictionnaire la liste des signaux digitaux $(0, ..., 0, s(k), ..., s(k + M - 1), 0, ..., 0)$ obtenus en tronquant s hors d'une fenêtre de longueur $M << N$ que l'on translate (avec un pas suffisamment petit, la valeur idéale – mais coûteuse – en étant 1) dans l'intervalle $\{1, ..., N\}$. De même, pour une image, on peut, en s'inspirant du classique algorithme JPEG (voir chapitre 10), envisager des troncatures de l'image par un masque de 8 pixels sur 8 pixels que l'on translate chaque fois d'au plus d'un cran suivant l'une ou l'autre direction ; on parle, à propos de dictionnaires obtenus de cette manière, de dictionnaires de *snapshots* (on pourrait dire des *instantanés* du signal ou de l'image digitale).

L'un des premiers objectifs à réaliser lorsque l'on construit un dictionnaire à partir d'un élément h donné est que les éléments du dictionnaire soient beaucoup plus faciles à stocker que l'élément h lui-même (dans l'exemple 9.2 des dictionnaires de *snapshots*, stocker un élément du dictionnaire revient à stocker $M << N$ entrées non nulles ainsi que l'indice initial de la fenêtre).

Si $H = H_N = l^2_{\mathbb{K}}(\{1, ..., N\}$ ($\mathbb{K} = \mathbb{R}$ ou $\mathbb{C}$) et que le dictionnaire $\mathcal{D}[h]$ (de cardinal $D = N - M$) construit pour analyser h est composé d'instantanés de h de longueur $M << N$, pris sur une fenêtre de longueur M, que l'on décale chaque fois d'une unité (chacun de ces instantanés étant assimilé à un élément de $H_M := l^2_{\mathbb{K}}(\{1, ..., M\})$), on exploite les idées inspirées par le théorème de décomposition en valeurs singulières (théorème **??** de ce chapitre) pour réaliser à partir de $\mathcal{D}[h]$ un dictionnaire orthogonal cette fois, donc non redondant, plus efficace et moins coûteux que le dictionnaire $\mathcal{D}[h]$ originel, si on l'utilise pour implémenter la *traque* de h via l'algorithme M. P. décrit dans la section précédente. La redondance de $\mathcal{D}[h]$ n'est cependant pas ignorée ici : elle est fondamentalement prise en compte pour la construction, précisément de ce nouveau dictionnaire. L'idée est en effet de considérer la matrice de Gram $D \times D$ définie par

$$G := \left(\langle d_j \, , \, d_k \rangle_{H_M} \right)_{1 \leq j,k \leq D}$$

et d'en calculer la liste des valeurs propres $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_M \geq \lambda_{M+1} = 0 \geq \cdots \geq \lambda_D = 0$. On notera $U_1, ..., U_M$ le système orthonormé de $\mathbb{K}^D$ constitué de vecteurs propres de G (respectivement associés aux valeurs propres $\lambda_1, \lambda_2, ..., \lambda_M$). Les premiers de ces vecteurs propres $U_1, U_2, ...$, indiquent en quelque sorte les *directions principales* statistiquement les plus probables pour le sous-espace de H_N engendré par les instantanés $d_1, ..., d_D$, considérés cette fois comme des éléments de H_N et tronqués par des zéros hors d'une fenêtre de longueur M *fixée*,

prise dans la liste des fenêtres choisies pour la réalisation des *snapshots*. Pour chaque $j = 1, ..., D$, on posera donc [4]

$$D_j := \sum_{k=1}^{D} u_{jk} d_k \ , \ j = 1, ..., M$$

et on réalisera, après avoir choisi un indice r entre 1 et D, un nouveau dictionnaire $\mathcal{D}_r[h]$ en concaténant les r dictionnaires $\mathcal{D}_{r,j}[h]$, $j = 1, ..., r$, obtenus en positionnant D_j dans $[N/M]$ fenêtres consécutives (sans chevauchement cette fois) de l'intervalle $\{1, ..., N\}$. Ce dictionnaire $\mathcal{D}_r[h]$ est de cardinal $[N/M] \times r$, significativement inférieur à $N - M$ si r est petit devant M. Les éléments de ce nouveau dictionnaire $\mathcal{D}_r[h]$ (qui sont cette fois deux à deux orthogonaux dans H_N) sont dits *modes propres* de h et le *matching* de h contre ce dictionnaire (d'autant plus intéressant que l'on peut choisir $r << M$, ce qui est d'autant plus possible que la suite (λ_k) décroît rapidement) est un exemple de *décomposition orthogonale propre* [5]. Si les r modes propres $D_1, ..., D_r$ de h sont ainsi stockés, on espère pouvoir proposer dans chacune des $[N/M]$ fenêtres consécutives une version *résumée* de h sous la forme d'une combinaison linéaire de $D_1, ..., D_r$. Le nombre d'entrées nécessaires pour stocker cette version comprimée sera donc $r \times [N/M]$ (il y a $[N/M]$ fenêtres consécutives) et la compression sera d'autant plus intéressante que le taux r/M pourra être pris petit.

1.3. P. O. D. et non-stationnarité

La construction d'un dictionnaire $\mathcal{D}_r[h]$, $1 \leq r \leq M$ à partir d'un dictionnaire exhaustif d'instantanés (de longueur M) de h, du fait que son principe se fonde sur des idées statistiques, ne se révèle réellement efficace que si h répond à des critères de stationnarité (voir le chapitre 10 pour cette notion, sous l'angle déterministe ou stochastique). Toutefois, en l'absence de tels critères, la détermination préalable des modes propres $D_1, ..., D_M$ (à partir d'un dictionnaire exhaustif d'instantanés), couplée avec l'algorithme M. P. O., ouvre une voie pour isoler les structures cohérentes au sein de h : il est naturel de lancer l'exploration de h avec l'algorithme M. P. O. en utilisant, pour $1 \leq r \leq M$, le dictionnaire $\widetilde{\mathcal{D}}_r[h]$ (de cardinal toujours $\simeq D$) dont les éléments sont obtenus en positionnant l'un des D_j, $j = 1, ..., r$ dans l'une des $[D/r]$ fenêtres de longueur M de l'intervalle $\{1, ..., M\}$, prises de r en r.

Exemple 9.3. On propose ici un thème d'exercice applicatif à partir du matériel fourni sur le support accompagnant l'ouvrage. Les signaux temporels générés par les équations de Navier-Stokes (qui régissent par exemple les écoulements turbulents, voir le complément 3 du chapitre 10) constituent typiquement des signaux non stationnaires auxquels on peut envisager d'utiliser, à titre d'exercice, ces techniques. On trouvera dans le dossier `Chap01-C1/POD` deux signaux digitaux `w` et `p` (de longueur $N = 8\,192$) correspondant respectivement aux enregistrements (en fonction du temps) de la vorticité (module du rotationnel du champ de vitesse) et de la pression en un point pour l'écoulement d'un fluide (en présence d'obstacles induisant la turbulence) dans un canal 2D. Les quatres premiers modes propres D_1, D_2, D_3 et D_4 correspondant au dictionnaire des instantanés de `w` de longueur $M = 256$ sont aussi proposés (fichier `modepropre`), permettant (grâce à la routine `dictionnaire`) la construction du dictionnaire $\widetilde{\mathcal{D}}_4[w]$ et l'analyse de `w` ou `p` via M. P. ou (c'est préférable) M. P. O.

4. Notons que les D_j, $j = 1, ..., M$ construits ainsi peuvent aussi s'obtenir comme les colonnes de la matrice U dans la décomposition en valeurs singulières `[U,Diag,V]=svd(D)`, où `D` est la matrice $M \times D$ dont les colonnes sont les éléments $d_1, ..., d_D$ de H_M.

5. P. O. D. pour *Proper Orthogonal Decomposition* dans la terminologie anglo-saxonne ; on parle aussi d'*analyse en composantes principales* (voir aussi le chapitre 12 pour un point de vue d'obédience statistique).