ING2-MI

Mini Projet Compressive Sensing

Département : **Mathématiques**	*Date de rendu :* **26 mars 2024**
Nombre de pages : **5**	*Heure limite :* **23h59**

Le but de ce mini-projet est de montrer la compréhension du procédé de Compressive Sensing.

1 Partie théorique

1. Rappeler la signification de la **ralaxation convexe** appliquée au problème

$$(\mathcal{P}_0) \ : \quad min \ \|\alpha\|_0 \quad \text{sous contrainte} \quad D\alpha = x.$$

   Vous devez présenter le nouveau problème $(\mathcal{P}_1)$, et expliquer pourquoi il est plus intéressant.

2. Justifier que l'algorithme IRLS peut être utilisé pour résoudre le problème relaxé convexe obtenu dans la question précédente.

3. Quel est l'intérêt de traiter ce problème là plutôt que

$$(\mathcal{P}_p) \ : \quad min \ \|\alpha\|_p \quad \text{sous contrainte} \quad D\alpha = x$$

   avec $0 < p < 1$?

4. Indiquer une autre méthode pour résoudre le problème relaxé convexe. Vous présenterez la réécriture du problème permettant d'appliquer cette méthode, notamment les changements de variables, de fonction objectif et de contraintes.

2 Codage de base

1. Coder les 4 algorithmes de de codage parcimonieux, OMP, StOMP, CoSaMP et IRLS.
   Attention, dans IRLS, $w_i = (|\alpha_i^{(k-1)}| + \varepsilon)^{\frac{p}{2}-1}$. Et le critère d'arrêt est $\|\alpha^{(k)} - \alpha^{(k-1)}\| < \frac{\sqrt{\varepsilon}}{100}$.

2. Votre code doit avoir la forme de fonctions portant le nom de chaque algorithme avec comme arguments le signal x, le dictionnaire D, le nombre maximal d'itérations iterMax ainsi que les autres arguments ε, t ou s.

3. Vous préciserez sur votre rapport les sorties de chacune de ces fonctions, ainsi que les modifications et améliorations que vous auriez pu apporter par rapport à ce que nous avons vu en séance.

4. Coder également les 4 matrices de mesure vues en cours. Vous pourrez, si vous le souhaitez, en rajouter d'autres en justifiant vos choix.

5. Coder une fonction permettant de calculer la cohérence mutuelle entre 2 matrices.

3 Génération d'un dictionnaire et de 4 signaux

1. Générer aléatoirement un dictionnaire de taille 200×350. Vous préciserez sur votre rapport le choix de la méthode. Attention, les atomes doivent être normalisés.

2. Générer 4 signaux 5-parcimonieux dans ce dictionnaire. Vous déposerez le dictionnaire et les 4 signaux dans des fichiers *Dico.csv* et *signaux.csv*.

4 Comparaison des algorithmes de codage parcimonieux

1. Tester les 4 algorithmes de codage parcimonieux sur les 4 signaux.
2. Définir un critère de comparaison d'efficacité de ces algos, et donner votre conclusion.

5 Choix de la matrice de mesure

Afin de Tester le procédé complet de compressive sensing, vous devez choisir une matrice de mesure parmi les 4 déjà codées et pour des tailles de mesure m variant entre 5%, 10%, 25%, 40% et 50% de la taille du signal.

1. Expliquer la méthode que vous allez suivre pour décider.
2. Effectuer vos calculs et afficher vos résultats.

6 Procédé complet de compessive sensing

1. Utiliser la matrice de mesure sélectionnée pour générer les 4 vecteurs de mesure associés aux 4 signaux générés. Vous les déposerez dans un fichier *mesures.csv*.
2. Expliquer comment reconstruire les signaux à partir de ces mesures.
3. Effectuer cette reconstruction et expliquer comment évaluer la qualité de celle-ci.

7 Rendus attendus

Votre rendu doit se faire sous la forme d'un seul fichier : une archive .zip, portant les noms des membres du groupe.
Elle doit contenir :

- un rapport pdf contenant les réponses à la partie théorique, les justifications des choix de paramètres ainsi que les résultats et discussions menant à décision.
- un ou des fichiers .py contenant les fonctions *omp, stomp, cosamp* et *irls* pour les algos de codage parcimonieux, les générations de matrices de mesure, du dictionnaire et des signaux, le calcul de cohérence et la reconstruction.
- les fichiers *Dico.csv*, *signaux.csv* et *mesures.csv* ainsi que *Signaux-reconstruits.csv* contenant les 4 signaux reconstruits.

8 Annexes

Algorithme StOMP :

$\alpha = 0_{\mathbf{K},1}$, $\mathbf{R} = \mathbf{x}$; $\mathbf{R}$: le résiduel

À l'itération $k \geq 1$, l'algorithme StOMP procède selon les étapes suivantes:

1. Calcul du vecteur des contributions de tous les atomes $\mathbf{d}_j$, $j = 1, \ldots, \mathbf{K}$ au résiduel courant,

$$\mathbf{C}_j = \frac{\left|\langle \mathbf{d}_j, \mathbf{R}^{(k-1)} \rangle\right|}{\|\mathbf{d}_j\|_2}.$$

2. **La sélection:**

   (a) On calcule le seuillage $\mathbf{S}^{(k)}$: $\mathbf{S}^{(k)} = \mathbf{t}\dfrac{\|\mathbf{R}^{(k-1)}\|_2}{\sqrt{\mathbf{K}}}$, $2 \leq \mathbf{t} \leq 3$.

   (b) On sélectionne l'ensemble Λ_k des indices des atomes dont la contribution est supérieure au seuillage $\mathbf{S}^{(k)}$

$$\Lambda_k = \left\{ j \in \{1, \ldots, \mathbf{K}\},\ \mathbf{C}_j > \mathbf{S}^{(k)} \right\}.$$

3. **La mise à jour:** On met à jour l'ensemble des indices $\mathbf{P}_k = \mathbf{P}_{k-1} \cup \Lambda_k$.

4. On construit la matrice Φ_k en considérant les colonnes $\mathbf{d}_p$, $p \in \mathbf{P}_k$.

$$\Phi_k = [\mathbf{d}_p, p \in \mathbf{P}_k].$$

5. On résout le problème d'optimisation $\alpha^{(k)}_{\mathbf{P}_k} = \arg\min \|x - \Phi_k \alpha_{\mathbf{P}_k}\|_2$ par la méthode des moindres carrées.

6. On met à jour le résiduel, $\mathbf{R}^{(k)} = \mathbf{x} - \Phi_k \alpha^{(k)}_{\mathbf{P}_k}$.

Algorithme CoSaMP :

$\alpha = 0_{\mathbf{K},1}$, $\mathbf{R} = \mathbf{x}$; $\mathbf{R}$: le résiduel

$\mathbf{Supp} = \emptyset$ $\mathbf{Supp}$: le support de la solution parcimonieuse

Le CoSaMP suit cinq étapes principales et repose sur la connaissance a priori de l'ordre **s** de parcimonie de la solution:

Tant que le critère d'arrêt n'est pas satisfait faire, $k \geq 1$,

1. La sélection: Déterminer les 2**s** atomes de plus grande contribution au résiduel et en déduire le vecteur des positions de ces atomes, on le notera le support de la sélection **supp1**.
2. Mise à jour du support: Fusionner le support de la sélection **supp1** avec le support de la solution de l'itération précédente **supp** :

   $$\mathbf{supp} = \mathbf{supp} \cup \mathbf{supp1}$$

   On note $\mathbf{AS} = \mathbf{D}(:, \mathbf{supp})$ la matrice des atomes actifs sélectionnés.
3. Estimation: Utiliser la méthode des moindres carrés pour estimer les coefficients $\mathbf{z}_i$ de la solution parcimonieuse α correspondant au support mis à jour.
4. Rejet: Considérer les **s** plus grands coefficients de **z**.
5. Mise à jour du résiduel: $\mathbf{R} = \mathbf{x} - \mathbf{D}\alpha$;

Algorithme IRLS :

Algorithme IRLS

1. On initialise le compteur d'itération k = 0, le nombre maximum d'itérations kmax, le coefficient de régularisation à $\varepsilon = 0,1$, et la solution de départ:

   $$\alpha^{(0)} = \mathbf{D}^{'}\left(\mathbf{D}\mathbf{D}^{'}\right)^{-1}\mathbf{x}.$$

2. On calcule les poids $w_i = \left(\left|\alpha_i^{(k-1)}\right|^2 + \varepsilon\right)^{\frac{p}{2}-1}$;
3. On calcule la prochaine itération $\alpha^{(k)} = Q\mathbf{D}^{'}\left(\mathbf{D}Q\mathbf{D}^{'}\right)^{-1}\mathbf{x}$.
   (a) Si $\left|\|\alpha^{(k)}\|_2 - \|\alpha^{(k-1)}\|_2\right| > \frac{\sqrt{\varepsilon}}{100}$ et k<kmax on retourne à l'étape 2 après avoir incrémenté le compteur k.
   (b) Si $\left|\|\alpha^{(k)}\|_2 - \|\alpha^{(k-1)}\|_2\right| < \frac{\sqrt{\varepsilon}}{100}$ et $\varepsilon > 10^{-8}$ alors on modifie $\varepsilon = \frac{\varepsilon}{10}$:
      i. Si k<kmax, on incrémente k et on retourne à l'étape 2,
      ii. sinon, on termine.