# Estimation distribuée sur des réseaux de capteurs : une approche par bond

Louis DUVIVIER[1]

[1]Univ Lyon, ENS de Lyon, CNRS, Laboratoire de Physique, F-69342 Lyon, France

**Résumé –** Dans de nombreux domaines, des réseaux de capteurs sont utilisés pour mesurer un signal en différents points d'un milieu. Dans ce contexte, utiliser des algorithmes de consensus distibué permet d'améliorer la précision des estimations sans multiplier inutilement les communications. Quand tous les capteurs du réseau ne mesurent pas le même signal, chaque capteur doit d'abord déterminer avec lesquels de ses voisins il doit collaborer. Dans cet article, nous proposons un algorithme, dit « à bond » pour résoudre conjointement ce problème d'estimation des voisins avec lesquels collaborer et d'estimation coopérative, avec une objectif d'oubli rapide quand les mesures changent. Nous comparons ses performances numériques avec celles de l'algorithme de descente de gradient distribué Adapt Then Combine proposé auparavant.

**Abstract –** In many fields, sensor networks are used to measure a signal at different points in a medium. In this context, using distributed consensus algorithms can improve the accuracy of estimates without unnecessarily increasing communication. When not all sensors in the network measure the same signal, each sensor must first determine which of its neighbors to collaborate with. In this article, we propose an algorithm that we call "with leaps", jointly solve this problem of estimating neighbors to collaborate with and cooperative estimation, with an objective of fast forgetting when measurements change. We compare its numerical performance with that of the previously proposed Adapt Then Combine distributed gradient descent algorithm.

## 1 Introduction

Dans de nombreux domaines, des réseaux de capteurs sont utilisés pour mesurer un signal en différents points d'un milieu [2]. Dès que les mesures des différents capteurs ne sont pas indépendantes les unes des autres, il peut être utile de combiner ces mesures pour améliorer la précision ou la vitesse de convergence des estimations.

De nombreuses méthodes ont été étudiées pour permettre à ces capteurs de collaborer entre eux [2, 4] Néanmoins, la plupart de ces travaux ce sont concentrés sur le cas où tous les capteurs cherchent à estimer la même valeur [1, 3], ou bien où les capteurs connaissent les voisins qui cherchent à estimer la même valeur qu'eux [5]. Dans les deux cas, le problème de savoir avec lesquels de ses voisins un capteur doit collaborer ne se pose donc pas. Une exception notable est l'article de Xiaochuan Zhao et Ali Sayed [7], dans lequel ils présentent une adaptation des algorithmes Adapt Then Combine (ATC) et Combine Then Adapt (CTA) au cas où les capteurs estiment simultanément la valeur du signal et les voisins avec lesquels collaborer. C'est sur cette situation que notre travail se concentre également.

L'algorithme qu'ils proposent est basé sur une descente collaborative de gradient à pas constant. L'appartenance commune à un groupe est déterminée en se basant sur la proximité des estimations à chaque instant. Cet algorithme repose sur un compromis concernant la taille du pas de la descente de gradient : s'il est trop grand la convergence vers la valeur mesurée est imprécise, et s'il est trop petit, elle est lente.

Dans cet article, nous présentons une approche alternative, qui repose sur le calcul direct de l'estimateur. Nous la comparons avec l'algorithme de Zhao et Sayed et montrons qu'elle permet d'accélérer la vitesse de convergence, au prix d'une certaine instabilité des estimations.

## 2 Présentation de l'algorithme

Un réseau de capteur peut être modélisé par un graphe simple non-dirigé $G = (V, E)$ dont chacun des nœuds $V = [\![0, n-1]\!]$ est un capteur pouvant communiquer avec ses voisins, ainsi qu'indiqué par les arêtes $E$ du graphe (et uniquement eux). Chaque nœud $k$ mesure donc au temps $t$ un signal bruité

$$\mathbf{x}_k(t) = \mathbf{u}_k(t) + \epsilon_k(t)$$

où on adopte comme modèle que $\mathbf{u}_k(t) \in \mathbb{R}^s$ est la valeur que cherche à mesurer le capteur et $\epsilon_k(t) \sim \mathcal{N}(0, V_k)$ est un bruit blanc dont on suppose qu'il suit une loi normale centrée de matrice de covariance $V_k$. On suppose de plus qu'il existe une partition des nœuds $\mathcal{P}(t) = (p_1(t), \ldots, p_q(t))$ telle que

$$\forall t, \forall p_k \in \mathcal{P}(t), \forall i, j \in p_k(t), \mathbf{u}_i(t) = \mathbf{u}_j(t)$$

Chaque nœud $k$ ignore initialement à quel groupe il appartient, donc avec lesquels de ses voisins il peut collaborer. Pour le déterminer, il réalise d'abord une estimation locale de la moyenne $\overline{\mathbf{x}_k}^{loc}(T)$ et de la covariance $\overline{V_k}(T)$ du signal pour pouvoir les comparer avec ceux de ses voisins.

On considère dans un premier temps que pour chaque nœud $k$, le signal $\mathbf{u}_k(t)$ est constant dans le temps. On peut alors l'estimer localement par la moyenne empirique de l'échantillon de mesure $(\mathbf{x}_k(0), \ldots \mathbf{x}_k(T))$ :

$$\overline{\mathbf{x}_k}^{loc}(T) = \frac{1}{T+1} \sum_{t=0}^{T} \mathbf{x}_k(t)$$

On peut également estimer la matrice de covariance :

$$\overline{V_k}(T) = \frac{1}{T} \sum_{t=0}^{T} (\mathbf{x}_k(t) - \overline{\mathbf{x}_k}) \cdot {}^t(\mathbf{x}_k(t) - \overline{\mathbf{x}_k})$$

L'intérêt de ces deux estimateurs est qu'ils peuvent être calculés à la volée, sans qu'il soit nécessaire de garder en mémoire l'ensemble des mesures $(\mathbf{x}_k(0) \dots \mathbf{x}_k(T))$. En effet, si on note

$$\Delta_k(T+1) = (\mathbf{x}_k(T+1) - \overline{\mathbf{x}_k}^{loc}(T)) \cdot {}^t(\mathbf{x}_k(T+1) - \overline{\mathbf{x}_k}^{loc}(T))$$

on a les relations suivantes :

$$\overline{\mathbf{x}_k}^{loc}(T+1) = \frac{1}{T+2} \cdot ((T+1) \cdot \overline{\mathbf{x}_k}^{loc}(T) + \mathbf{x}_k(T+1)$$
$$\overline{V_k}(T+1) = \frac{T}{T+1} \cdot \overline{V_k}(T) + \frac{1}{T+2} \cdot \Delta_k(T+1)$$

Chaque capteur n'a donc a retenir en mémoire que les estimations courantes $\overline{\mathbf{x}_k}^{loc}(T)$, $\overline{V_k}(T)$ et le nombre d'observations $T+1$. La mise à jour des estimations locales est réalisée par l'algorithme 1.

**Algorithme 1 :** Mise à jour locale (Màj_Locale)

**Data :** $T \in \mathbb{N}$, $\bar{\mathbf{x}}^{loc}(T) \in \mathbb{R}^s$, $\bar{V}(T) \in \mathcal{M}_s(\mathbb{R})$, $\mathbf{x} \in \mathbb{R}^s$
**Result :** $T+1 \in \mathbb{N}$, $\bar{\mathbf{x}}^{loc}(T+1) \in \mathbb{R}^s$, $\bar{V}(T+1) \in \mathcal{M}_s(\mathbb{R})$
$\Delta \leftarrow \left(\mathbf{x} - \bar{\mathbf{x}}^{loc}\right) \cdot {}^t\left(\mathbf{x} - \bar{\mathbf{x}}^{loc}\right)$;
$\bar{\mathbf{x}}^{loc} \leftarrow \frac{1}{T+1} \times \left(T \times \bar{\mathbf{x}}^{loc} + \mathbf{x}\right)$;
$\bar{V} \leftarrow \frac{T}{T+1} \times \bar{V} + \frac{1}{T+2} \times \Delta$;
$T \leftarrow T+1$;

La précision de l'estimation est d'autant plus importante que le nombre d'observations prises en compte est élevé. Pour augmenter cette précision, chaque nœud a donc intérêt à utiliser non seulement ses propres observations mais aussi celles de ses voisins, à condition que les signaux mesurés par chacun aient la même moyenne. Pour deux nœuds voisins $j$ et $k$, on peut tester si $\mathbf{u}_j = \mathbf{u}_k$ à l'aide d'un test $T^2$ de Hotelling avec un niveau de confiance $\alpha$, qu'on note $\mathrm{Test\_T}^2{}_\alpha(\mathbf{u}_j, \mathbf{u}_k)$ [6]. Si on ne peut pas rejeter l'égalité avec un niveau de confiance $\alpha$ (ce qu'on note $\mathbf{u}_j \overset{T^2}{=} \mathbf{u}_k$) les nœuds $j$ et $k$ mettent en commun leurs observations. Si on note

$$\mathcal{N}_k = \{j \in V \mid (k,j) \in E\}$$
$$\mathcal{N}_k^\bullet = \{j \in \mathcal{N}_k \mid \mathbf{u}_j \overset{T^2}{=} \mathbf{u}_k\} \cup \{k\}$$

On peut alors définir un nouvel estimateur distribué

$$\overline{\mathbf{x}_k}^{dis}(T) = \frac{\sum_{j \in \mathcal{N}_k^\bullet} (T_j + 1) \times \overline{\mathbf{x}_j}^{loc}(T_j)}{\sum_{j \in \mathcal{N}_k^\bullet} (T_j + 1)} \qquad (1)$$

La mise à jour de l'estimation distribuée est réalisée par l'algorithme 2. Un avantage de cet estimateur distribué est qu'il est calculé à partir des estimations locales et pas de l'ensemble des mesures des voisins. À chaque itération de l'algorithme, la communication du nœud $j$ au nœud $k$ se limite donc à $(T_j, \overline{\mathbf{x}_j}^{loc}(T_j), \overline{V_j}(T_j))$.

À chaque instant, un nœud dispose donc de deux estimations : une réalisée sur la base de ses mesures locales, qui lui sert à déterminer avec lesquels de ses voisins il collabore, et une réalisée sur la base de ces échanges avec ses voisins qui est basée sur un échantillon plus large et est donc plus précis. Cette manière de maintenir deux estimations en parallèle

**Algorithme 2 :** Mise à jour distribuée (Màj_Dist)

**Data :** $\overline{\mathbf{x}_k}^{dis}(T_k) \in \mathbb{R}, (T_j, \overline{\mathbf{x}_j}^{loc}(T_j), \overline{V_j}(T_j))_{j \in \mathcal{N}_k}$
$\mathcal{G} \leftarrow \{k\}$;
**for** $j \in \mathcal{N}_k$ **do**
  **if** $\mathrm{Test\_T}^2_\alpha$ $(m_k, \overline{\mathbf{x}_k}, \overline{V_k}, m_j, \overline{\mathbf{x}_j}, \overline{V_j}, \alpha)$ **then**
    $\mathcal{G} \leftarrow \mathcal{G} \cup \{j\}$;
  **end**
**end**
$\bar{\mathbf{x}}^{dis} \leftarrow \left(\sum_{j \in \mathcal{G}} m_j \times \overline{\mathbf{x}_j}\right) / \left(\sum_{j \in \mathcal{G}} m_j\right)$;

est similaire à ce qui est fait dans l'algorithme d'estimation distribué proposé dans [7].

Si on suppose maintenant que la valeur de $\mathbf{u}_k(t)$ varie au cours du temps, il faut à chaque nouvelle mesure $\mathbf{x}_k(t)$ vérifier si l'on est toujours en train de mesurer la même valeur ou si le signal a changé. Là aussi, on peut le faire à l'aide d'un test statistique en vérifiant si la nouvelle mesure appartient à l'ellipsoïde de confiance pour un certain niveau $\alpha$. Dans le cas où le signal a changé, on oublie les anciennes mesures et on recommence à estimer le signal à partir de la nouvelle mesure. Dans le cas contraire, on met simplement à jour les estimateurs.

Lorsqu'un nœud oublie ses anciennes mesures et reprend son estimation à zéro, il lui faut un peu de temps avant d'obtenir une estimation précise. Nous imposons donc un temps minimum $\tau$ après l'oubli pendant lequel il ne collabore pas avec ses voisins, pour éviter qu'il ne soit influencé par leurs mesures avant d'avoir pu obtenir une estimation fiable de son propre signal.

On obtient donc l'algorithme 3 pour chacun des nœuds du réseau. Nous appelons cet algorithme "par bond", car lorsque le signal mesuré par un capteur change il oublie ses anciennes estimations et reprend l'estimation à zéro à partir des nouvelles valeurs mesurées. Il saute ainsi directement d'une estimation à une autre, contrairement aux algorithmes basés sur une descente de gradient qui passent progressivement de l'une à l'autre.

**Algorithme 3 :** Estimation par bond

**Data :** $k \in [\![0, n-1]\!]$, $s \in \mathbb{N}^+$, $\alpha \in [0,1]$, $\tau \in \mathbb{N}^+$
Initialisation($m$, $\bar{\mathbf{x}}^{loc}$, $\bar{\mathbf{x}}^{dis}$, $\bar{V}$);
**while** True **do**
  $\mathbf{x} \leftarrow$ Nouvelle_Mesure();
  **if** $m \leq \tau$ **then**
    Màj_Locale($m$, $\bar{\mathbf{x}}^{loc}$, $\bar{V}$, $\mathbf{x}$);
  **else**
    **if** $m > \tau \wedge \mathbf{x} \notin \mathcal{E}_\alpha(\bar{\mathbf{x}}^{loc}, \bar{V})$ **then**
      Initialisation($m$, $\bar{\mathbf{x}}^{loc}$, $\bar{\mathbf{x}}^{dis}$, $\bar{V}$);
      Màj_Locale($m$, $\bar{\mathbf{x}}^{loc}$, $\bar{V}$, $\mathbf{x}$);
    **else**
      Màj_Locale($m$, $\bar{\mathbf{x}}^{loc}$, $\bar{V}$, $\mathbf{x}$);
      Envoyer_Voisins($m$, $\bar{\mathbf{x}}^{loc}$, $\bar{V}$);
      Recevoir_Voisins($(m_j, \overline{\mathbf{x}_j}, \overline{V_j})_{j \in \mathcal{N}_k}$);
      Màj_Dist($\bar{\mathbf{x}}^{dis}$, $(m_j, \overline{\mathbf{x}_j}, \overline{V_j})_{j \in \mathcal{N}_k}$);
    **end**
  **end**
**end**

## 3 Analyse de l'algorithme

Pour étudier la précision de l'estimation en régime stationnaire, on suppose que pour chaque nœud $k$, la matrice de covariance du bruit est la matrice identité $V_k = I_s$ et le signal est nul $u_k(t) = 0$. Si ce n'est pas le cas, on peut se ramener à cette situation par le changement de variable

$$\mathbf{y}_k(t) = \sqrt{V_k}^{-1} \cdot (\mathbf{x}_k(t) - \mathbf{u}_k(t))$$

On a donc $\mathbf{x}_k(t) \sim \mathcal{N}(0, I_s)$, et pour chaque composante $i \in [\![0, s-1]\!]$, $x_k(i)(t) \sim \mathcal{N}(0,1)$. On peut mesurer la précision de l'estimation par la largeur de l'intervalle de confiance de niveau $\beta$ autours de la moyenne empirique $\overline{x_k(i)}(T)$ sur un échantillon de $T+1$ observations. Cette largeur vaut

$$\delta_\beta = 2\frac{\Phi^{-1}(\beta)}{\sqrt{T+1}} \tag{2}$$

où $\Phi(l) = F^{-1}_{\mathcal{N}(0,1)}(l) - F^{-1}_{\mathcal{N}(0,1)}(-l)$. Le nombre d'observations prises en compte par l'estimateur distribué (1) vaut $\sum_{j \in \mathcal{N}_k^\bullet}(T_j + 1)$. Si on note $\overline{T+1}$ le nombre moyen de mesures prises en compte pour chaque nœud, $\overline{\deg}^\bullet$ le nombre moyen de voisins mesurant le même signal, et qu'on suppose le graphe suffisament homogène pour pouvoir faire l'approximation $\sum_{j \in \mathcal{N}_k^\bullet}(T_j + 1) \approx \overline{T+1} \times \overline{\deg}^\bullet$ on obtient

$$\delta_\beta = \frac{2\Phi^{-1}(\beta)}{\sqrt{\overline{\deg}^\bullet \times \overline{T+1}}}$$

En régime stationnaire, à chaque instant $t$, un nœud a une probabilité $(1-\alpha)$ de considérer par erreur qu'une mesure n'appartient pas à l'ellipsoide de confiance $\mathcal{E}_\alpha(\bar{x}^{loc}, \bar{V})$. Donc, si le temps minimal $\tau$ durant lequel on empêche l'oubli est faible, le nombre d'observations prises en compte peut être approximé par une loi géométrique de paramètre $(1-\alpha)$ dont l'espérance vaut $\overline{T+1} = (1-\alpha)^{-1}$. Augmenter $\alpha$ permet de prendre en compte d'avantage de mesure, et donc d'améliorer la précision de l'estimation, mais cela réduit la sensibilité de l'algorithme. En effet, il existe une distance $\lambda_\alpha$ telle que si $\|\mathbf{u}_k(t) - \mathbf{u}_j(t)\| < \lambda_\alpha$, le test $T^2$ n'est plus capable de distinguer les signaux $\mathbf{u}_k(t)$ et $\mathbf{u}_j(t)$.

Si on note $\lambda_\alpha = \Psi^{-1}(\alpha)$, on obtient finalement la relation suivante entre la précision $\delta_\beta$ de l'algorithme, sa sensibilité $\lambda_\alpha$ et le nombre moyen de nœuds mesurant le même signal $\overline{\deg}^\bullet$ :

$$\delta_\beta = 2\sqrt{\frac{1-\Psi(\lambda_\alpha)}{\overline{\deg}^\bullet}}\Phi^{-1}(\beta) \tag{3}$$

## 4 Estimation des performances

Pour estimer les performances de l'algorithme proposé, nous l'avons comparé avec l'algorithme Adapt Then Combine (ATC) proposé par Zhao et Sayed dans [7], qui est basé sur une descente de gradient collaborative entre les nœuds du réseau. Nous nous sommes particulièrement intéressés à la réaction du réseau lorsque le signal mesuré par les capteurs est brusquement modifié.

Pour cela, nous réalisons des simulations sur un réseau $G = (V, E)$ constitué d'une grille de 10 capteurs par 10. Pendant les 100 premiers pas de temps, tous les capteurs mesurent le même signal $\mathbf{u}_k(t) = (0,0)$ avec un bruit $\epsilon_k(t)$ dont la matrice de covariance est

$$V_k = \begin{pmatrix} 10 & 0 \\ 0 & 10 \end{pmatrix}$$

À $t_0 = 100$, le signal change. Pour les 50 nœuds situés dans les colonnes 0 à 4 de la grille, $\mathbf{u}_k(t) = (10, 10)$. Pour les 50 nœuds situés dans les colonnes 5 à 9, il devient $\mathbf{u}_k(t) = (10, -10)$. La covariance du bruit reste identique.

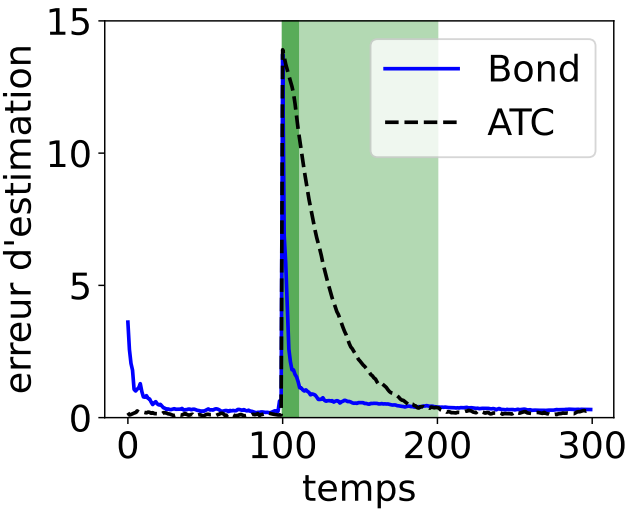


FIGURE 1 : Erreur d'estimation moyenne en fonction du temps pour le réseau de capteur $G$. La ligne bleue continue correspond à l'algorithme par bond que nous proposons, la ligne noire pointillée à l'algorithme Adapt Then Combine (ATC)

On considère les estimations réalisées par les capteurs avec les deux algorithmes comparés. Pour l'algorithme par bond, les paramètres utilisés sont : un niveau de confiance pour les tests statistiques fixé à $\alpha = 0.99$, et un temps d'isolement après oubli $\tau = 5$. Pour l'algorithme Adapt Then Combine, les paramètres utilisés sont fixé d'après les valeurs utilisées dans l'article original : un pas $\mu = 0.02$, et un seuil de test $\theta = 5$. On considère que les capteurs n'ont initialement aucune information à propos des voisins avec lesquels ils ont intérêt à collaborer ou non.

L'erreur moyenne d'estimation sur tous les capteurs à un instant $t$ est donnée par

$$\bar{\delta}(t) = \frac{1}{n}\sum_{k=0}^{n-1} \left\|\bar{\mathbf{x}}_k^{dis}(t) - \mathbf{u}_k(t)\right\|$$

Elle est tracée en fonction du temps pour chaque algorithme sur la figure 1.

On observe qu'après avoir convergé vers le premier signal observé jusqu'à $t_0 = 100$, l'erreur augmente brusquement. C'est une conséquence du fait que le signal mesuré a changé brutalement alors que les estimations n'ont pas eu le temps de se mettre à jour. Après cette perturbation, les estimations convergent vers la nouvelle valeur du signal mesuré. Les deux algorithmes proposés se distinguent sur deux points. Premièrement, le rythme de convergence : l'algorithme Adapt Then Combine est nettement plus lent que l'algorithme par bond. Au bout de 10 itérations, l'algorithme par bond a déjà une erreur moyenne inférieure à 1 alors qu'il faut presque 100 itérations

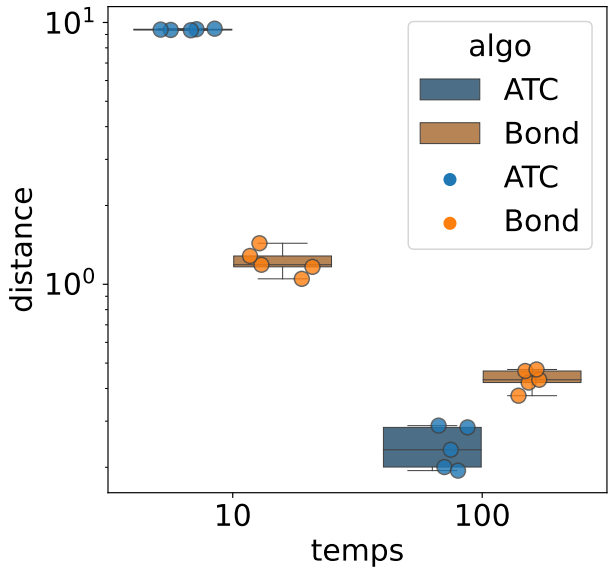


FIGURE 2 : Erreur d'estimation moyenne pour les algorithmes par bond et ATC suite à un changement du signal. Les deux distributions sur la gauche (temps = 10) correspondent à l'erreur après 10 itérations (ATC à gauche, Bond à droite). Les deux distributions sur la droite (temps = 100) à l'erreur après 100 itérations (ATC à gauche, Bond à droite). Les distributions empiriques indiquées correspondent à 10 réalisations de chaque algorithme.

à l'algorithme ATC pour atteindre cette précision. Cependant, en régime stationnaire, la précision atteinte par l'algorithme ATC est supérieure à celle de l'algorithme par bond.

Cela s'observe d'autant mieux si on considère l'erreur moyenne à $t_0 + 10$ et à $t_0 + 100$ pour plusieurs réalisations de la simulation. Ces valeurs sont affichées sur la figure 2 pour 10 réalisations des deux algorithmes. On voit qu'à $t_0 + 10$ l'erreur d'estimation moyenne de l'algorithme par bond est systématiquement inférieure d'un ordre de grandeur à celle de l'algorithme ATC. Par contre, à $t_0 + 100$, l'erreur d'estimation moyenne pour l'algorithme ATC est inférieure à celle de l'algorithme par bond, même si l'écart est moindre.

## 5 Discussion et conclusion

Ces résultats s'interprètent aisément si on considère les mécanismes utilisés par chaque algorithme pour évaluer la valeur du signal. L'algorithme ATC est basé sur une descente de gradient à pas constant. Cela implique que pour obtenir une estimation précise, il est nécessaire de fixer une valeur de pas faible, ce qui entraîne une convergence lente de l'estimation. On pourrait utiliser un pas variable dont la taille décroît au fur et à mesure des itérations pour résoudre ce problème, mais cela aurait pour conséquence de réduire l'adaptabilité de l'algorithme. Si le signal change au bout d'un grand nombre d'itérations, le pas sera trop faible pour mettre à jour l'estimation.

L'algorithme par bond contourne ce problème en ne gardant pas en mémoire toutes les mesures passées quand un changement est détecté. L'estimation se fait uniquement sur les mesures pertinentes pour l'estimation courante, ce qui permet de converger plus rapidement vers une nouvelle mesure. Cela se paie néanmoins par une plus grande instabilité. En effet, il arrive qu'une valeur inattendue due au bruit soit mesurée, alors que le signal n'a pas changé. Dans ce cas, la mémoire est réinitialisée et l'estimation repart à zéro. Ce phénomène est rare mais possible, et il est rapidement compensé par la collaboration entre voisins. Néanmoins il suffit à détériorer la précision de l'estimation en régime stationnaire. Une possibilité pour réduire encore ce risque est de ne pas réinitialiser la mémoire après une seule mesure inattendue, mais seulement après plusieurs mesures consécutives.

En conclusion, pour ce qui concerne la précision de l'estimation, l'intérêt relatif de chaque algorithme dépend essentiellement du contexte d'utilisation. En particulier, il faut considérer le rapport entre la fréquence à laquelle le signal change et la fréquence d'exécution de l'algorithme. Si l'algorithme a le temps de réaliser plus de 100 itérations entre deux modifications du signal, Adapt Then Combine donnera une meilleure précision d'estimation en moyenne. Si la fréquence de changement du signal est plus importante, l'algorithme par bond sera plus précis.

Ces résultats ne portent que sur l'estimation de la valeur du signal, mais il serait également intéressant de considérer la précision de l'estimation de la partition des nœuds. De même, on a considéré lors des simulations uniquement une topologie en grille pour le réseau de capteur, mais cette topologie a un impact important sur les possibilité de communication et donc de collaboration des capteurs. Il serait donc intéressant de considérer, pour ces mêmes algorithmes, le comportement de réseaux ayant d'autres topologies.

## Références

[1] Jie CHEN, Cédric RICHARD et Ali H SAYED : Multitask diffusion adaptation over networks. *IEEE Transactions on Signal Processing*, 62(16):4129–4144, 2014.

[2] Chee-Yee CHONG : Forty years of distributed estimation. *In 2017 Sensor Data Fusion : Trends, Solutions, Applications (SDF)*, pages 1–10. IEEE, 2017.

[3] Laurent JACOB, Jean-philippe VERT et Francis BACH : Clustered multi-task learning. *Advances in neural information processing systems*, 21, 2008.

[4] Cassio G LOPES et Ali H SAYED : Diffusion least-mean squares over adaptive networks. *IEEE Transactions on Signal Processing*, 56(7):3122–3136, 2008.

[5] Roula NASSIF, Stefan VLASKI, Cédric RICHARD, Jie CHEN et Ali H SAYED : Multitask learning over graphs. *IEEE Signal Processing Magazine*, 37(3):14–25, 2020.

[6] Dennis WACKERLY, William MENDENHALL et Richard L SCHEAFFER : *Mathematical statistics with applications*. Cengage Learning, 2014.

[7] Xiaochuan ZHAO et Ali H SAYED : Distributed clustering and learning over networks. *IEEE Transactions on Signal Processing*, 63(13):3285–3300, 2015.

## Remerciements

Travail financé par l'ANR DARLING ANR-19-CE48-0002 à l'ENS de Lyon, suite aux discussions avec Pierre Borgnat.