Modélisation de l'information contenue dans les défenses des narvals dans le but d'estimer leur durée de vie

Yanis BEN BELGACEM, Vadim BERTRAND, Angélique SAILLET

Sommaire

Comme nous avons pu le voir précédemment, le narval est une espèce de cétacés vivant dans l'océan Arctique. Ces animaux d'une durée de vie moyenne de 50 ans, possèdent deux dents. Chez les femelles, les dents restent à l'intérieur de la boîte crânienne, tandis que pour les mâles, la canine gauche s'allonge et prend la forme d'une corne, comme le montre la Figure 1. Elle commence à pousser au travers de la lèvre supérieure gauche dès l'âge d'un an lors de la puberté et croît jusqu'à la maturité sexuelle, entre 8 et 9 ans. Cette défense torsadée possède des fonctionnalités et propriétés uniques dans la nature. Elle contient des millions de terminaisons nerveuses, ce qui en fait un organe sensoriel très développé [1].

Figure 1: Vue de face d'un narval et de sa dent. [2]

Certains chercheurs danois, comme Eva Garde, s'intéressent plus particulièrement à l'estimation de la durée de des narvals via l'information contenue dans cette dent. Pour mener cette étude, plusieurs découpes latérales des dents d'animaux décédés ont été réalisées. Comme nous pouvons le voir sur la Figure 2, ces découpes se présentent sous la forme d'une séquence de sillons ou de couches comportant des marqueurs saisonniers au cours de la croissance des dents. Ces derniers créent des motifs sinusoïdaux. La fréquence et la forme de ces sinusoïdes varient d'une année à l'autre selon la variabilité de la durée ou de l'intensité des saisons [3]. L'information portée par les motifs à l'intérieur des défenses est donc logiquement liée à la durée de vie de l'animal.

Figure 2: Présentation de l'allure d'une section en longueur d'une dent de narval. [3]

Le premier objectif pour cette problématique est le choix d'un modèle sinusoïdal pouvant représenter l'information contenue dans la dent de l'animal. À partir de cette forme de modèle et d'observations, le deuxième objectif sur lequel nous allons nous concentrer est celui de l'estimation des paramètres du modèle sinusoïdal. Nous présentons donc dans les parties suivantes, le modèle envisagé ainsi que notre démarche d'estimation de ces paramètres à partir d'un algorithme SAEM.

1 Modèle sinusoïdal

Comme nous l'avons évoqué précédemment, les motifs sinusoïdaux observés sont le reflet de la variabilité des saisons, ainsi ce motif n'est pas répété identiquement en fonction du temps. Ces variations complexifient donc la modélisation de cette information.

Les observations le long de la défense sont notées Y_i pour $i = 1, \dots, n$, avec la position correspondante sur la dent notée x_i. Le modèle est le suivant :

$$Y_i = f(x_i, \varphi) + \varepsilon_i$$

avec ε_i un bruit aléatoire suivant une loi normale de moyenne 0 et de variance ω^2.

La fonction de régression $f(x, \varphi)$ est une fonction périodique sinusoïdale telle que :

$$f(x, \varphi) = A \sin(g(x) + b) + B \sin(2g(x) + 2b + \frac{\pi}{2})$$

avec

$$g(x) = ax + \xi_x$$

et finalement ξ_x, un processus aléatoire d'Ornstein-Uhlenbeck, tel que :

$$d\xi_x = -\beta \xi_x dx + \sigma dW_x$$

Dont la solution est donnée par :

$$\xi_{x+\Delta} = \xi_x \psi + \int_x^{x+\Delta} \sigma e^{\beta(s-x)} dW_s$$

de sorte que la densité de transition est :

$$p(\xi_{x+\delta} | \xi_x) = \mathcal{N}(\xi_x \psi, \frac{\sigma^2}{2\beta}(1 - \psi^2))$$

Dans ce cadre, l'objectif est donc d'estimer les paramètres θ :

- $\varphi = (A, B, a, b)$,
- ω,
- $\psi = e^{-\beta\Delta}$, où Δ est l'intervalle de temps entre deux observations,
- $\gamma^2 = \frac{\sigma^2}{2\beta}(1 - \psi^2)$.

Une réalisation de ce modèle est présentée sur la Figure 3.

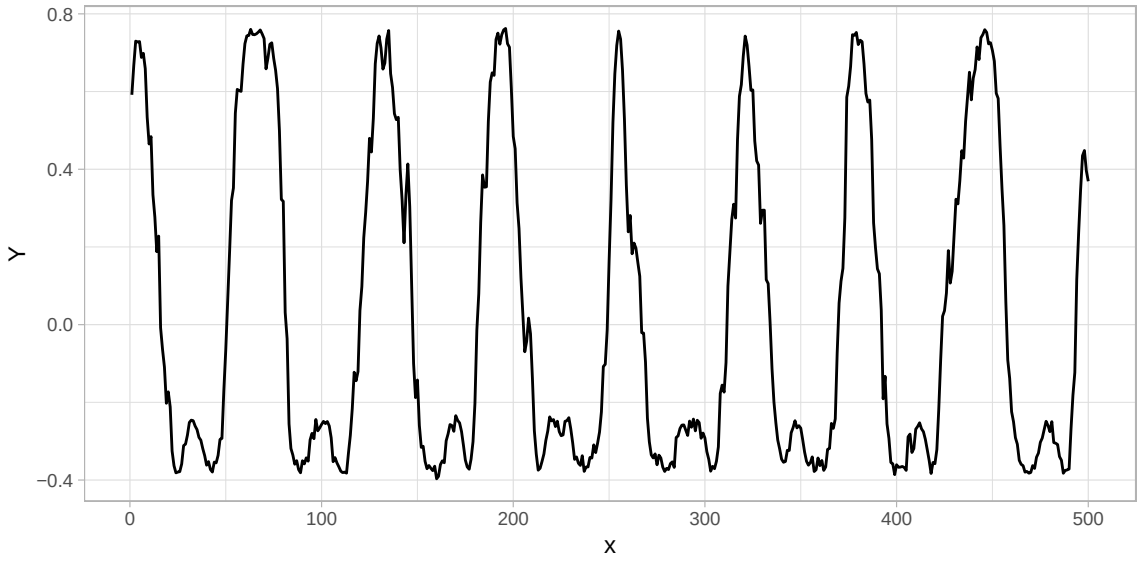

Figure 3: Simulation des observations Y, avec les paramètres suivants : $A = 0.5$, $B = -0.25$, $b = 1$, $a = 0.1$, $\beta = 0.05$, $\sigma = 0.1$, $\omega = 0.01$ et $\delta = 1$.

1.1 Identifiabilité

Afin de justifier l'intérêt de l'estimation des paramètres du modèle, nous nous sommes intéressé à son identifiabilité. Nous pouvons observer sur la Figure 4 une trajectoire du processus ξ_x cible ainsi que trois autres simulations de trajectoire du processus ξ_x pour des valeurs de paramètres ψ et γ variant, obtenues selon le procédé suivant :

$\xi_{1:n} \leftarrow 0$ ▷ Initialisation de la première valeur
for $i \in \{1, ..., n\}$ **do**
  $\xi_i = \xi_{i-1} * \psi + \epsilon_\xi$, avec $\epsilon_\xi \sim N(0, \gamma^2)$
end for

Nous observons sur la Figure 4 que les différentes réalisations du processus ξ_x conduisent à des trajectoires différentes. Comme attendu au regard de l'expression du processus, la modification de ψ impacte la valeurs moyenne de ξ_x, et γ sa variance.

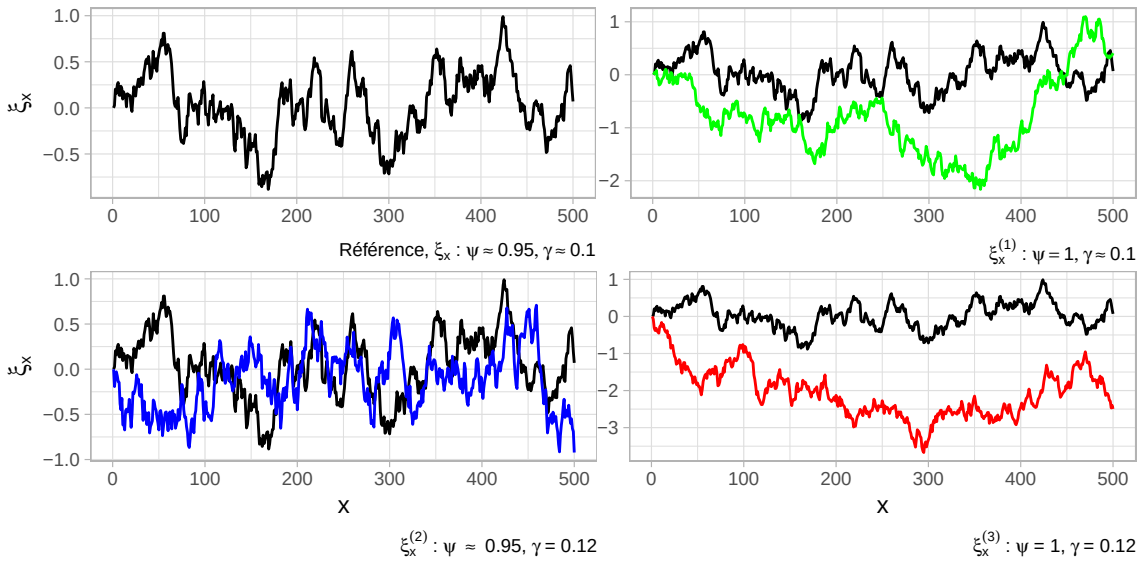

Figure 4: Présentation d'une trajectoire du processus ξ_x cible (en noir) et de trois autres trajectoires de ce processus simulées à partir de valeurs différentes de ψ et γ (en vert, bleu et rouge).

Ce résultat a un impact direct sur les observations Y puisque, comme nous pouvons le voir sur la Figure 5, les observations Y correspondant à des réalisations ξ_x pour des valeurs de ψ et γ différentes ne collent pas du tout à la distribution cible. De plus, une variation des paramètres A, B, a, et b entraine une différence encore plus forte : A et B jouant sur l'amplitude de la sinusoïde, a sur sa pulsation et b sur son décalage de phase.

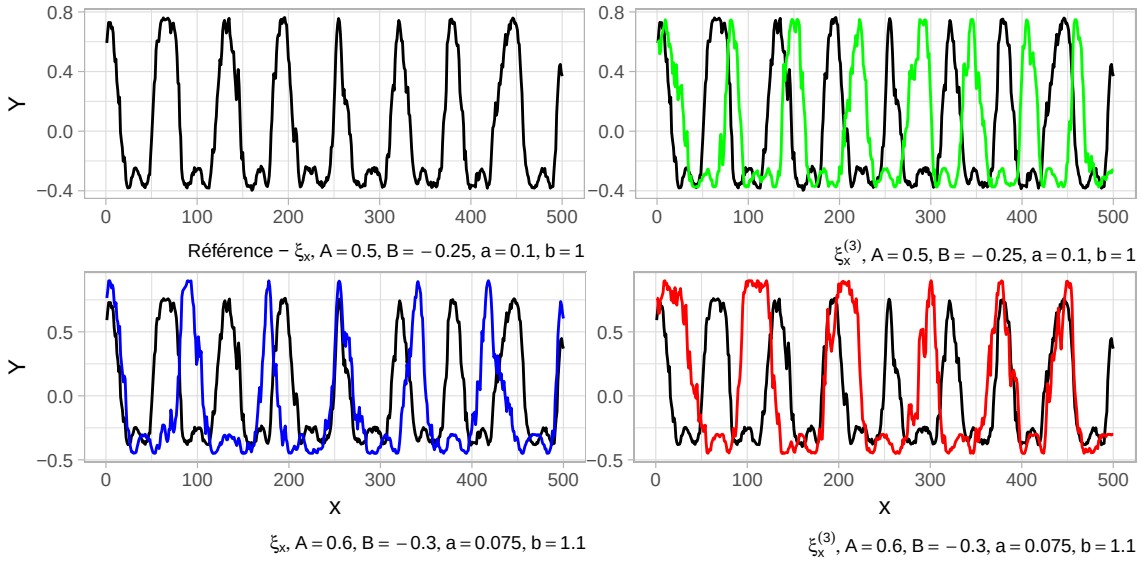

Figure 5: Distributions Y pour différentes trajectoires de ξ_x et différentes valeurs de A, B, a, b (en vert, bleu et rouge).

Les observations de Y sont sensibles à la trajectoire du processus ξ_x associé, ainsi qu'aux paramètres A, B, a, b. Le modèle sinusoïdal semble donc identifiable.

2 Estimation des paramètres à partir d'un algorithme SAEM

Afin d'estimer les paramètres θ du modèle présenté dans la partie précédente, nous avons implémenté une procedure reposant sur l'algorithme SAEM. Nous allons d'abord présenter le principe d'un algorithme EM [4], puis celui de son approximation stochastique : l'algorithme SAEM [5]. Nous détaillerons les étapes MCMC [6] et SMC [7] avant de présenter l'algorithme complet.

2.1 Algorithme EM

L'algorithme EM est basé sur la log-vraisemblance complète du modèle qui s'écrit de la manière suivante :

$$\begin{aligned}
\log L(Y, \xi_x, \theta) &= \sum_{i=1}^{n} \log p(Y_i|\xi_i) + \sum_{i=1}^{n} \log p(\xi_i|\xi_{i-1}) + \log p(\xi_1) \\
&= -\sum_{i=1}^{n} \frac{(Y_i - f(x_i, \varphi))^2}{2\omega^2} - \frac{n}{2}\log(\omega^2) \\
&\quad -\sum_{i=1}^{n} \frac{(\xi_i - \xi_{i-1}\psi)^2}{\frac{\sigma^2}{\beta}(1-\psi^2)} - \frac{n}{2}\log(\frac{\sigma^2}{2\beta}(1-\psi^2)) \\
&= -\sum_{i=1}^{n} \frac{(Y_i - f(x_i, \varphi))^2}{2\omega^2} - \frac{n}{2}\log(\omega^2) \\
&\quad -\sum_{i=1}^{n} \frac{(\xi_i - \xi_{i-1}\psi)^2}{2\gamma^2} - \frac{n}{2}\log(\gamma^2)
\end{aligned}$$

Pour chaque itération k, l'algorithme EM procède aux deux étapes suivantes, étant donné la valeur courante des paramètres $\theta^{(k)}$.

- étape E : calcul de $Q(\theta, \theta^{(k)})$, l'espérance conditionnelle de la log-vraisemblance du modèle : $Q(\theta, \theta^{(k)}) \leftarrow E[\log L(Y, \xi, \theta)|Y; \theta_{(k)}]$
- étape M : actualisation des paramètres $\theta^{(k+1)} = \arg\max_\theta Q(\theta, \theta^{(k)})$.

Pour actualiser les paramètres, nous avons besoin des statistiques exhaustives. Ces statistiques sont obtenues à partir du théorème de factorisation [?] et contiennent toute l'information de la vraisemblance. Leurs définitions sont les suivantes :

$$\begin{aligned}
S_1(\xi_i) &= \frac{1}{n}\sum_{i=1}^{n}(Y_i - f(x_i(\xi_i), \varphi))^2 \\
S_2(\xi_i) &= \sum_{i=1}^{n} \xi_{i-1}\xi_i \\
S_3(\xi_i) &= \sum_{i=1}^{n} \xi_{i-1}^2 \\
S_4(\xi_i) &= \sum_{i=1}^{n} \xi_i^2
\end{aligned}$$

L'actualisation des paramètres dépend directement de ces statistiques.

2.2 Simulation de ξ_x

Dans notre cas, la distribution conditionnelle $p(\xi_x|Y;\theta^{(k)})$ n'est pas explicite en raison de la non-linéarité de notre fonction de régression $f(x,\varphi)$. Nous pouvons donc utiliser un algorithme MCMC ou un algorithme SMC pour simuler selon cette distribution.

2.2.1 Algorithme MCMC

L'objectif de cet algorithme MCMC (Markov Chain Monte Carlo) est de simuler une trajectoire du processus ξ_x à partir des observations Y ainsi que des paramètres θ. L'algorithme programmé est plus précisément un algorithme de Gibbs - Metropolis Hasting avec marche aléatoire.

Effectivement, après l'initialisation d'une trajectoire $\xi^{(0)} = (\xi_1^{(0)}, ..., \xi_n^{(0)})$, l'algorithme procède à M itérations. La trajectoire du processus simulée peut donc s'écrire $\xi^{(M)} = (\xi_1^{(M)}, ..., \xi_n^{(M)})$.

Plus précisément, pour chaque itération k, on calcule pour chaque position x_i, une valeur courante candidate ξ_c avec une marche aléatoire : $\xi_i^{(c)} = \xi_i^{(k-1)} + N(0, \delta_i^2)$. Cela introduit un nouveau paramètre $\delta = (\delta_1, ..., \delta_n)$ contrôlant la variance de la marche aléatoire. Pour chacun de ses candidats, une log-probabilité d'acceptation est calculée de la façon suivante :

$$\log(\alpha) = min(\log(\frac{L(Y,\xi^{(c)})}{L(Y,\xi^{(k-1)})}), 1)$$

avec :

$$\begin{aligned}
\log(\frac{L(Y,\xi^{(c)})}{L(Y,\xi^{(k-1)})}) &= \log(L(Y,\xi^{(c)})) - \log(L(Y,\xi^{(k-1)})) \\
&= -\frac{1}{2\omega^2}\sum_{i=1}^{n}(Y_i - f_\varphi(\xi^{(c)}))^2 - \frac{1}{2\frac{\gamma^2}{2}}\sum_{i=1}^{n}(\xi^{(c)} - \xi^{(k-1)}\psi)^2 \\
&\quad + \frac{1}{2\omega^2}\sum_{i=1}^{n}(Y_i - f_\varphi(\xi^{(k-1)}))^2 + \frac{1}{2\frac{\gamma^2}{2}}\sum_{i=1}^{n}(\xi^{(k-1)} - \xi^{(k-2)}\psi)^2
\end{aligned}$$

À partir de la valeur de cette log-probabilité ainsi que d'une réalisation d'une loi uniforme prenant ses valeurs entre 0 et 1, le candidat est soit rejeté, soit accepté, auquel cas, il remplace la valeur considérée à l'itération $k-1$.

De plus, nous avons choisi de rendre le paramètre δ adaptatif en fonction du taux d'acceptation acc_rate_i pour chaque point au fil des itérations k. Cela ajoute donc une étape d'actualisation à l'algorithme précédent, ce qui donne finalement l'Algorithme 1 :

Algorithm 1 Algorithme MCMC de simulation d'une trajectoire du processus ξ_x.

$\xi_{1:n} \leftarrow 0$ ▷ Initialisation du processus
$\delta_{1:n} \leftarrow 0.05$ ▷ Initialisation du delta adaptatif
$\delta_{AR} \leftarrow 0.1$ ▷ Pas d'évolution du delta adaptatif
$acc\_rate_{1:n} \leftarrow 0$ ▷ Initialisation du vecteur de taux d'acceptation
$acc\_rate_{target} \leftarrow 0.23$ ▷ Taux d'acceptation visé
for $k \in \{1, ..., M\}$ **do**
  for $i \in \{1, ..., n\}$ **do**
    $\xi^{(c)} \leftarrow \xi$
    $\xi_i^{(c)} \sim \xi_i + \mathcal{N}(0, \delta_i^2)$ ▷ Simulation du candidat pour ξ_i
    $\alpha_{\log} \leftarrow min(\log(\frac{L(Y,\xi^{(c)})}{L(Y,\xi)}), 1)$ ▷ Calcul de la probabilité d'acceptation
    $u \sim \mathcal{U}(0, 1)$ ▷ Tirage d'une réalisation de loi uniforme
    if $\log u \leq \alpha_{\log}$ **then**
      $\xi \leftarrow \xi^{(c)}$
    end if
    $acc\_rate_i \leftarrow$ mise à jour du taux d'acceptation
    if $acc\_rate_i < acc\_rate_{target} * (1 - 0.1)$ **then**
      $\delta_i \leftarrow \delta_i * (1 - \delta_{AR})$ ▷ Réduction du delta adaptatif
    else if $acc\_rate_i > acc\_rate_{target} * (1 + 0.1)$ **then**
      $\delta_i \leftarrow \delta_i * (1 + \delta_{AR})$ ▷ Augmentation du delta adaptatif
    end if
  end for
end for
$\widehat{\xi_x} \leftarrow \xi$

2.2.2 Algorithme SMC

Comme pour l'algorithme MCMC, l'algorithme Sequential Monte-Carlo (SMC) - ou filtre particulaire - a pour objectif de simuler une trajectoire de ξ_x à partir des observations Y et des paramètres θ.

Cependant son fonctionnement est assez différent : plutôt que d'accepter ou de rejeter un candidat pour chaque instant du processus selon un rapport de vraisemblance comme le fait l'approche MCMC, le SMC propose pour chaque instant P réalisations (les particules) selon les P estimations réalisées à l'instant précédent et associe à chacune des nouvelles réalisations un poids égale à la probabilité d'observer la valeur de Y à l'instant courant conditionnellement à la réalisation simulée. Il est alors possible de tirer avec remise parmi les particules en utilisant les poids normalisés comme probabilités de tirage et de conserver ainsi les particules permettant d'observer avec les plus fortes probabilités Y. Une fois que le dernier instant du processus est atteint, il suffit de tirer un index selon les derniers poids calculés pour obtenir une trajectoire du processus. L'Algorithme 2 détaille chacune de ces étapes :

Algorithm 2 Algorithme SMC pour la simulation d'une trajectoire du processus ξ_x.

$w_{1:P} \leftarrow 1/P$ ▷ Initialisation des poids associés aux particules
$\xi_c^{(1:P)} \leftarrow 0$ ▷ Initialisation des particules au premier instant du processus
$\xi_{1:n} \leftarrow \xi_c$ ▷ Initialisation des trajectoires du processus
for $i \in \{2, ..., n\}$ **do**
  for $j \in \{1, ..., P\}$ **do**
    $\xi_c^{(j)} \sim \xi_{i-1}^{(j)} * \psi + \mathcal{N}(0, \gamma^2)$ ▷ Simulation d'une valeur courante selon $\xi_{i-1}^{(j)}$
    $w_j \leftarrow P(Y_i | f(i, \xi_c^{(j)}))$ ▷ Poids égale à la probabilité de Y_i conditionnellement à $\xi_c^{(j)}$
  end for
  for $j \in \{1, ..., P\}$ **do**
    $w_j \leftarrow \frac{w_j}{\sum_{k=1}^{P} w_k}$ ▷ Normalisation
  end for
  for $j \in \{1, ..., P\}$ **do**
    $idx \leftarrow$ tirage probabiliste d'une particule en fonction des poids w
    $\xi_i^{(j)} \leftarrow \xi_c^{(idx)}$ ▷ Conservation de la particule idx
  end for
end for
$idx \leftarrow$ tirage probabiliste d'une trajectoire en fonction des poids w
$\widehat{\xi_x} \leftarrow \xi^{(idx)}$

2.3 Algorithme SAEM

L'introduction d'une étape MCMC ou SMC conduit à la version stochastique de l'algorithme EM, à savoir l'algorithme SAEM. Cet algorithme utilise les étapes de l'algorithme EM auxquelles s'ajoute une étape d'approximation stochastique.

Effectivement, pour chaque itération k, les étapes sont les suivantes :

- étape E : simulation d'une nouvelle trajectoire de $\xi^{(k)}$ à l'aide d'un algorithme MCMC considérant $p(\xi|Y; \theta^{(k)})$ comme une distribution stationnaire,
- étape SA : approximation stochastique des statistiques exhaustives :

$$\begin{aligned} s_1^{(k)} &= s_1^{(k-1)} + \alpha_k(S_1(\xi^{(k)}) - s_1^{(k-1)}) \\ s_2^{(k)} &= s_2^{(k-1)} + \alpha_k(S_2(\xi^{(k)}) - s_2^{(k-1)}) \\ s_3^{(k)} &= s_3^{(k-1)} + \alpha_k(S_3(\xi^{(k)}) - s_3^{(k-1)}) \\ s_4^{(k)} &= s_4^{(k-1)} + \alpha_k(S_4(\xi^{(k)}) - s_4^{(k-1)}) \end{aligned}$$

- étape M : actualisation de θ^k, à partir des formules suivantes qui utilisent les statistiques exhaustives $s^{(k)}$:

$$
\begin{aligned}
\widehat{\varphi}^{(k)} &= \arg\min_{\varphi} \sum_{i=1}^{n} \left(Y_i - f(x_i(\xi_i^{(k)}), \varphi) \right)^2 \\
\widehat{\psi}^{(k)} &= \frac{s_2^{(k)}}{s_3^{(k)}} \\
\widehat{\omega^2}^{(k)} &= s_1^{(k)} \\
\widehat{\gamma^2}^{(k)} &= \frac{1}{n}(\widehat{\psi^2}^{(k)} s_3^{(k)} - 2\widehat{\psi}^{(k)} s_2^{(k)} + s_4^{(k)})
\end{aligned}
$$

Au principe général d'un algorithme SAEM à Q itérations, nous avons ajouté deux hyper-paramètres M_{max} & α_{min} :

- Pour chaque itération q de l'algorithme SAEM, au moins une itération de l'algorithme MCMC ou SMC est effectuée. Dans le cas de l'utilisation d'une étape MCMC, afin de pouvoir améliorer les performances au début de notre algorithme, nous avons décidé de fixer ce nombre d'itérations à 5 pour les M_{max} premières itérations du SAEM, puis ensuite l'algorithme n'accomplit plus qu'une seule itération du MCMC. Le paramètre M_{max} est donc un hyper-paramètre de l'algorithme SAEM à choisir au préalable.
- Le deuxième paramètre concerne l'approximation stochastique effectuée pour chacune des itérations q. Effectivement, durant les premières itérations les approximations sont relativement éloignées de la valeur cible et donc sensiblement différentes entre elles. Au contraire, dans les dernières itérations, étant donné le phénomène de convergence que nous devons observer, les approximations sont censées être plus proches de la valeur cible et également entre elles. Afin de prendre en compte ce phénomène, nous faisons varier la valeur du paramètre de mémoire α permettant de tenir compte des valeurs précédemment estimées au fil des itérations à partir du paramètre α_{min} de la façon suivante :
  - dans un premier temps $\alpha = 1$ pour les α_{min} premières itérations, ce qui permet, d'appliquer la formule complète d'approximation,
  - puis, pour les (Q-α_{min}) dernières itérations, les alphas sont calculés de la façon suivante :

$$\alpha_{\alpha_{min}:Q} = \frac{1}{l^{0.8}}, \text{ avec } l = 1 : (Q - \alpha_{min})$$

  Ainsi le paramètre α décroît au fil des itérations à partir de la α_{min} ième itération. Cette procédure permet de réduire l'importance du terme associé à α et d'augmenter donc l'influence de la valeur précédente pour ces itérations-là.

En prenant en compte ces deux paramètres supplémentaires, l'algorithme se présente comme l'Algorithme 3 :

Algorithm 3 Algorithme SAEM complet

$s_1 \leftarrow (s_1^{(1)}, ..., s_1^{(Q)})$
$s_2 \leftarrow (s_2^{(1)}, ..., s_2^{(Q)})$
$s_3 \leftarrow (s_3^{(1)}, ..., s_3^{(Q)})$
$s_4 \leftarrow (s_4^{(1)}, ..., s_4^{(Q)})$
$\widehat{\varphi} \leftarrow (\widehat{\varphi}^{(1)}, ..., \widehat{\varphi}^{(Q)})$
$\widehat{\psi} \leftarrow (\widehat{\psi}^{(1)}, ..., \widehat{\psi}^{(Q)})$
$\widehat{\omega^2} \leftarrow (\widehat{\omega^2}^{(1)}, ..., \widehat{\omega^2}^{(Q)})$
$\widehat{\gamma^2} \leftarrow (\widehat{\gamma^2}^{(1)}, ..., \widehat{\gamma^2}^{(Q)})$
$\alpha_{1:\alpha_{min}-1} \leftarrow 1$; $\alpha_{\alpha_{min}:Q} \leftarrow \frac{1}{l^{0.8}}$ avec $l = 1 : (Q - \alpha_{min} + 1)$ ▷ Initialisation du paramètre mémoire
$M_{1:M_{max}} \leftarrow 5$; $M_{M_{max}+1:Q} \leftarrow 1$ ▷ Initialisation du nombre d'itérations du MCMC
for $q \in \{2, ..., Q\}$ **do**
▷ Etape E
$\xi^{(q)} \leftarrow \xi$ avec $\theta^{(q-1)}$, par $M = M_q$ itérations MCMC (Algorithme 1), ou par SCM (Algorithme 2)
▷ Etape SA
$S_1 \leftarrow \frac{1}{n}\sum_{i=1}^{n}(Y_i - f(x_i(\xi_i^{(q)}), \varphi))^2$ ▷ Calcul des statistiques exhaustives
$S_2 \leftarrow \sum_{i=1}^{n} \xi_{i-1}^{(q)}\xi_i^{(q)}$
$S_3 \leftarrow \sum_{i=1}^{n}(\xi_{i-1}^{(q)})^2$
$S_4 \leftarrow \sum_{i=1}^{n}(\xi_i^{(q)})^2$
▷ Mise à jour des approximations stochastiques
$s_1^{(q)} \leftarrow s_1^{(q-1)} + \alpha_q(S_1(\xi^{(q)}) - s_1^{(q-1)})$
$s_2^{(q)} \leftarrow s_2^{(q-1)} + \alpha_q(S_2(\xi^{(q)}) - s_2^{(q-1)})$
$s_3^{(q)} \leftarrow s_3^{(q-1)} + \alpha_q(S_3(\xi^{(q)}) - s_3^{(q-1)})$
$s_4^{(q)} \leftarrow s_4^{(q-1)} + \alpha_q(S_4(\xi^{(q)}) - s_4^{(q-1)})$
▷ Etape M
$\widehat{\varphi}^{(q)} \leftarrow \arg\min_{\varphi} \sum_{i=1}^{n}\left(Y_i - f(x_i(\xi_i^{(q)}), \varphi)\right)^2$ ▷ Actualisation de $\theta^{(q)}$
$\widehat{\psi}^{(q)} \leftarrow \frac{s_2^{(q)}}{s_3^{(q)}}$
$\widehat{\omega^2}^{(q)} \leftarrow s_1^{(q)}$
$\widehat{\gamma^2}^{(q)} \leftarrow \frac{1}{n}(\widehat{\psi^2}^{(q)} s_3^{(q)} - 2\widehat{\psi}^{(q)} s_2^{(q)} + s_4^{(q)})$
end for

3 Simulation & résultats

Dans cette dernière partie, nous présentons les résultats d'estimation de ξ_x par les algorithmes MCMC et SMC ; et plus largement de l'ensemble de coefficients θ par l'algorithme SAEM, ainsi que du plan d'expérience mis en place pour les valider.

3.1 Estimation de ξ_x par MCMC

Nous avons, dans un premier temps, testé l'efficacité de l'algorithme MCMC programmé indépendamment de l'algorithme SAEM. Pour ce faire, nous avons fixé les valeurs des paramètres : $A = 0.5$, $B = -0.25$, $b = 1$, $a = 0.1$, $\beta = 0.05$, $\sigma = 0.1$, $\omega = 0.01$ et $\delta = 1$. Ces valeurs ont été utilisées pour simuler une trajectoire de ξ_x cible avec une distribution Y associée selon le modèle sinusoïdal. L'algorithme MCMC a été réalisé avec $M = 150$ itérations et obtient, à partir de ces mêmes valeurs de paramètres et des observations Y, les résultats présentés sur la Figure 6.

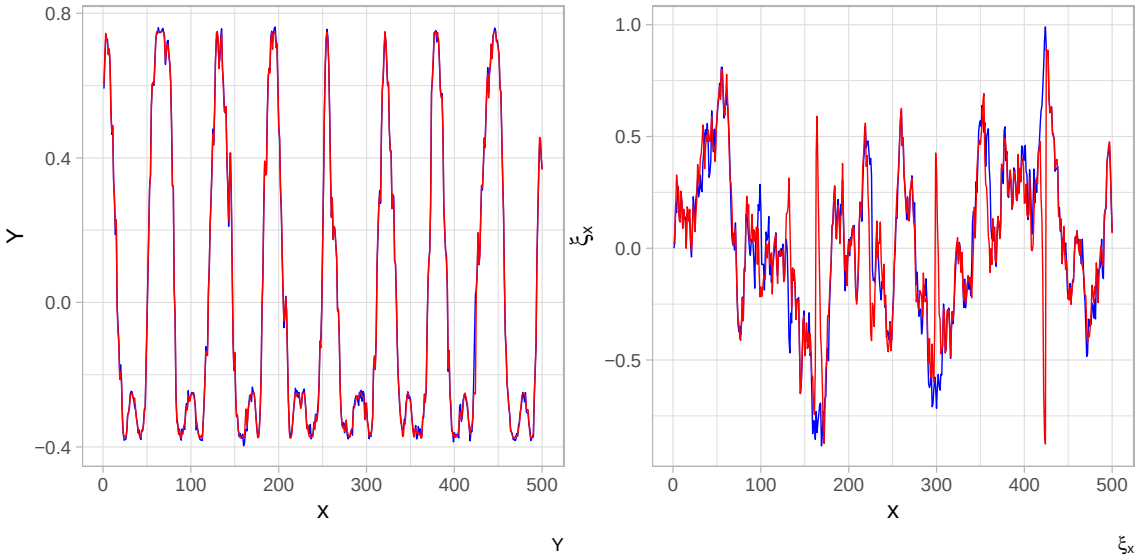

Figure 6: Superposition de la distribution Y cible et de la trajectoire ξ_x réellement utilisée (en bleu) ainsi que de la trajectoire de $\widehat{\xi_x}$ obtenue par l'algorithme MCMC et la distribution $\hat{Y}$ qui l'utilise (en rouge).

La trajectoire de ξ_x obtenue par le MCMC (en rouge) est très proche de celle à l'origine des observations Y. Nous remarquons cependant que quelques points de ξ_x sont très mal approchés par l'algorithme. Effectivement pour plusieurs positions x_i, la distance entre la valeur originelle de ξ_i et la valeur estimée par l'algorithme semble grande. Cependant, ces différences n'influencent que très peu l'allure du signal Y, qui reste très satisfaisant. Afin de comprendre la raison de ces erreurs, nous nous sommes intéressés au comportement du δ_i au fil des itérations, relativement à celui du taux d'acceptation acc_rate_i pour un des points concernés. Nous avons comparé leur évolution pour la position x_i où l'erreur est maximale et celle où elle est minimale.

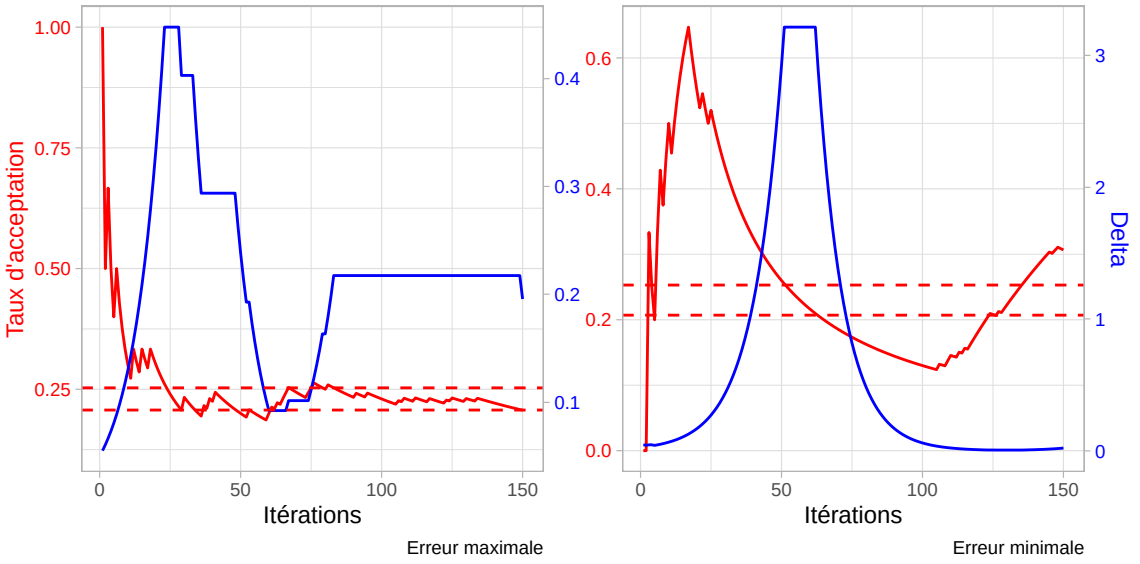

Figure 7: Distributions du taux d'acceptation et du delta adaptatif au fil des itérations pour deux positions différentes : celle pour laquelle l'erreur d'estimation est maximale et celle où elle est minimale.

Nous pouvons voir sur la Figure 7, que dans les deux cas, le delta adaptif évolue correctement. Effectivement le taux d'acceptation est premièrement supérieur à la borne supérieure de notre seuil ce qui entraine l'augmentation de δ. Cette augmentation permet alors au taux de diminuer et donc de rentrer dans nos deux bornes de seuil, et δ stagne durant cette période. Le taux d'acceptation devient ensuite trop bas, et alors la valeur de δ diminue afin de le faire augmenter à nouveau. Nous remarquons seulement que, dans le cas concernant l'erreur maximale, l'évolution du taux d'acceptation est moins rapide que dans le cas de l'erreur minimale, par exemple il atteint l'intervalle de seuil environ vers la $40^{\text{ième}}$ itération contre environ la $20^{\text{ième}}$ pour le point où l'erreur est minimale. Cela entraine une valeur maximale de δ bien plus élevée, d'environ 1.6 contre 0.3. Malgré cette observation, nous n'identifions pas la source de ces points aberrants. Cependant, étant donné que ceux-ci n'impactent pas l'estimation de la distribution des observations Y, l'algorithme MCMC programmé reste selon nous satisfaisant.

3.2 Estimation de ξ_x par SMC

Nous avons repris la trajectoire ξ_x à estimer dans la partie précédente et nous cette fois-ci tenté de la générer par notre algorithme SMC en utilisant $P = 500$ particules.

La Figure 8 permet de constater que cette approche semble plus précise que la MCMC, les points aberrants ont disparu et l'allure de ξ_x est bien reconstruite.

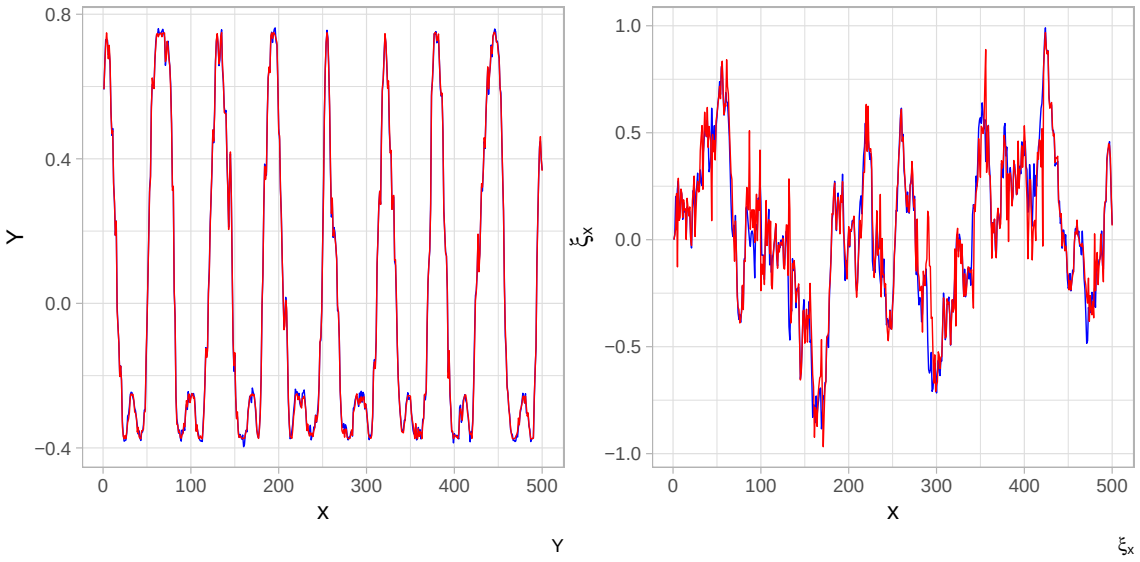

Figure 8: Superposition de la distribution Y cible et de la trajectoire ξ_x réellement utilisée (en bleu) ainsi que de la trajectoire de $\widehat{\xi_x}$ obtenue par l'algorithme SMC et la distribution $\hat{Y}$ qui l'utilise (en rouge).

3.3 Estimation de θ par SAEM

En gardant l'objectif d'estimer optimalement les paramètres utilisés pour obtenir une réalisation des observations Y cibles, nous avons effectué $Q = 500$ itérations de l'algorithme SAEM présenté précédemment. Les paramètres α_{min} et M_{max} ont été fixés après plusieurs tests et analyses graphiques aux valeurs suivantes : $\alpha_{min} = 90$ et $M_{max} = 20$, ce qui correspond à des valeurs retrouvées régulièrement dans la littérature. La trajectoire de ξ_x obtenue après les itérations ainsi que les observations Y l'utilisant sont présentés sur la Figure 9. Nous observons sur cette figure un léger décalage pour les premières positions de la trajectoire de ξ_x estimée. Malgré ce décalage, les deux courbes sont assez proches et nous retrouvons ce résultat pour la distribution de Y.

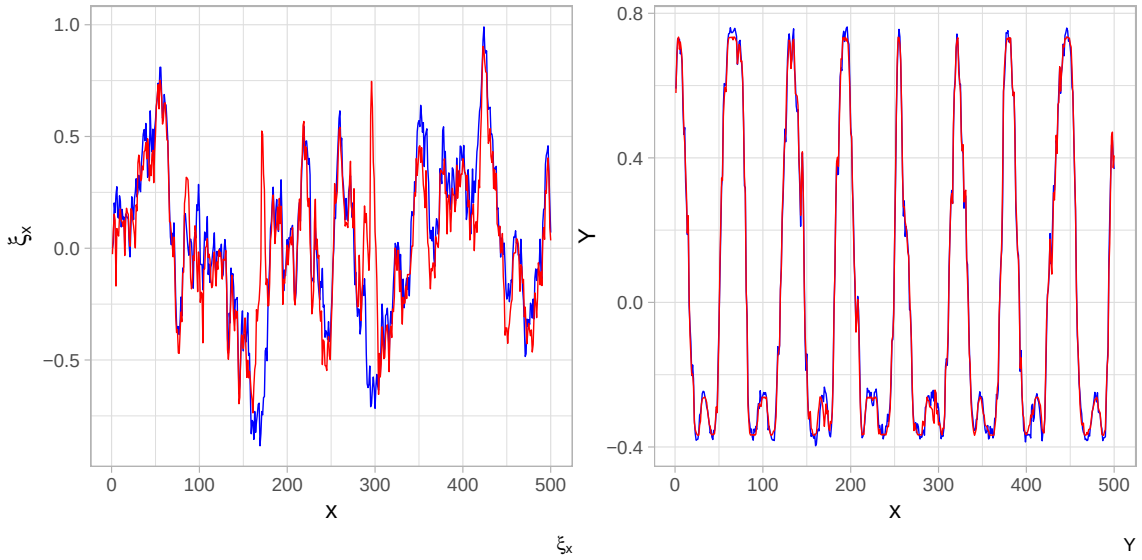

Figure 9: Superposition de la distribution Y cible et de la trajectoire ξ_x réellement utilisée (en bleu) ainsi que de la trajectoire de ξ_x obtenue par l'algorithme SAEM et la distribution Y qui l'utilise (en rouge).

Concernant l'estimation des autres paramètres du modèle, l'évolution des valeurs obtenues au fil des itérations pour ω, ψ, γ est présentée sur la Figure 10. Nous avons fait le choix d'initialiser ces 3 valeurs à 0.5 pour la première itération. Comme nous pouvons le voir sur cette figure, les paramètres ω et ψ convergent correctement vers les vraies valeurs des paramètres. Cependant, celle du paramètre γ ne converge pas exactement vers la valeur attendue et est légèrement surestimée.

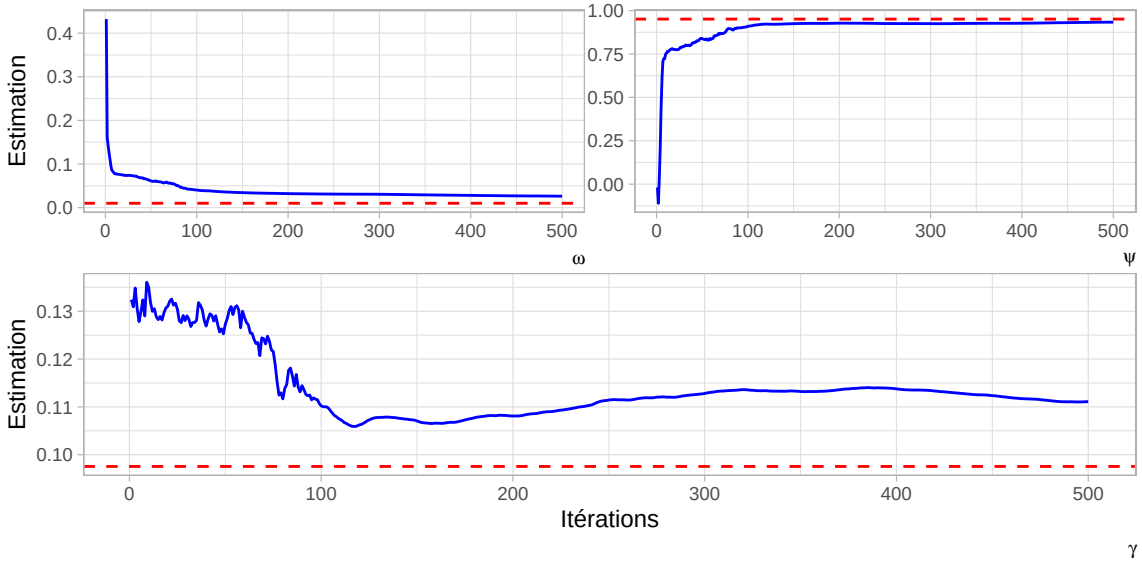

Figure 10: Présentation de l'estimation des coefficients ω, ψ et γ par l'algorithme SAEM, ainsi que des droites représentant la valeur cible de ces paramètres (en rouge).

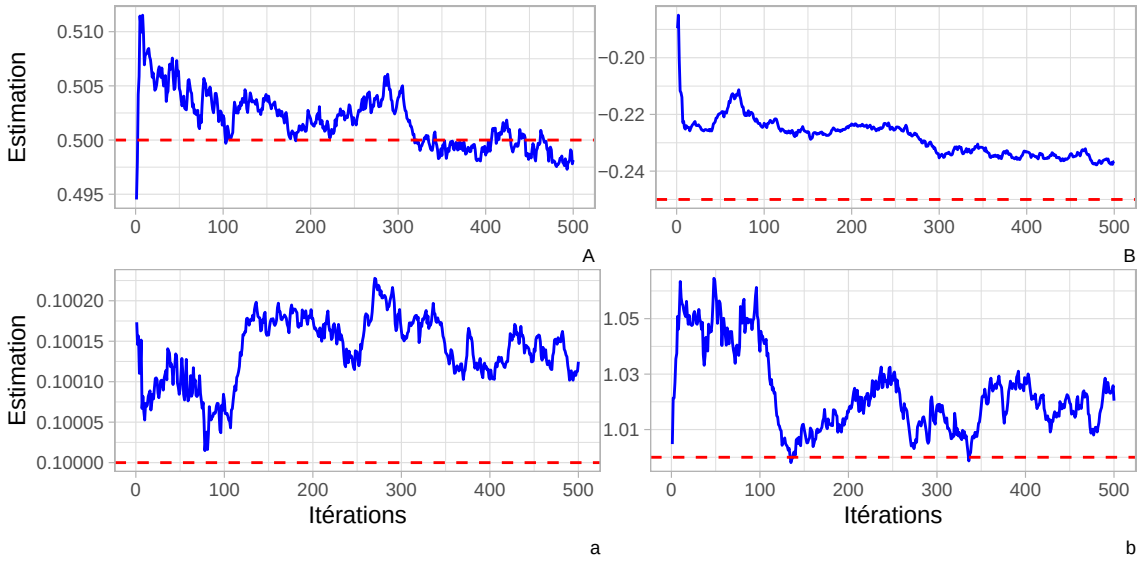

Figure 11: Présentation de l'estimation des coefficients A, B, a et b par l'algorithme SAEM, ainsi que des droites représentant la valeur cible de ces paramètres (en rouge).

A chaque itération de l'algorithme, les paramètres A, B, a, b sont estimés en minimisant les moindres carrés non linéaires. Sous **R**, cela se fait au moyen de la fonction *nls* qui nécessite des valeurs initiales des paramètres à estimer. Afin de favoriser la convergence de l'algorithme Gauss-Newton de résolution du problème des moindres carrées non linéaires, nous avons tenu compte de l'influence des paramètres A et B sur l'amplitude de Y en les initialisant selon une réalisation de loi uniforme $\mathcal{U}(-max(|Y|)-\omega, max(|Y|)+\omega)$. Dans le même objectif, puisque le paramètre a est lié à la pulsation de la sinusoïde, il est initialisé selon $a_{init} = 2*pi*f_{max}$ où f_{max} est la fréquence de Y avec la plus grande densité spectrale. Le paramètre b influe lui sur le décalage de phase de la sinusoïde, nous pouvons donc aider l'algorithme en l'initialisant par $b_{init} = (7*pi/8) - x_0 * a_{init}$ où x_0 correspond au plus petit x_i pour lequel Y est nul. Comme nous pouvons le voir sur la Figure 11, l'estimation du coefficient A converge assez normalement vers 0.5. Bien qu'elle se rapproche de -0.25, celle de B semble avoir du mal à converger réellement et stagne au-dessus de ce que nous souhaiterions. Par ailleurs, nous observons, que l'estimation du paramètre a est très proche de la "vraie" valeur, mais nous ne voyons pas de convergence à proprement parler. Pour finir, l'estimation de b semble plus difficile à accomplir : nous ne remarquons pas de convergence vers la valeur attendue et l'estimation finale apparaît un peu éloignée de la valeur fixée initialement.

3.3.1 Plan d'expérience

Les estimations de θ et ξ_x comportent une part d'aléatoire, ainsi afin d'évaluer les performances de notre algorithme SAEM il convient de répéter un grand nombre de fois ces estimations à partir de réalisations de ξ_x différentes. En effet, notre objectif est d'estimer nos paramètres sur un grand nombre d'itérations, afin d'analyser leurs variations autour des valeurs fixées. Nous avons choisi d'effectuer $M = 1000$ estimations de chacun des paramètres $\gamma, \psi, \omega, A, B, a, b$, et de calculer la racine carrée de l'erreur quadratique moyenne ($RMSE$), l'erreur en pourcentage (PE) et sa moyenne en valeur absolue ($MAPE$), à partir des formules suivantes :

$$RMSE = \sqrt{\frac{\sum_{i=1}^{M}(\hat{\theta}_i - \theta)^2}{M}}$$

$$MAPE = \frac{\sum_{i=1}^{M}|PE_i|}{M}$$

$$PE_i = \frac{\hat{\theta}_i - \theta}{\theta}$$

Avec $\hat{\theta}$ les paramètres estimés et θ les paramètres initiaux.

Nous avons également utilisé ce plan d'expérience pour comparer les résultats obtenus en utilisant une étape MCMC ou une étape SMC pour estimer ξ_x.

Puisque l'estimation de chacun de nos paramètres doit se rapprocher de notre valeur initiale, nous devrions avoir des estimations $\hat{\theta}$ autour de nos θ, soit des PE relativement proches de 0. À partir des résultats obtenus pour les 1000 itérations, nous avons représenté sur la Figure 12 les boxplots des PE pour chacun des 7 paramètres et pour les deux méthodes de simulation de ξ_x.

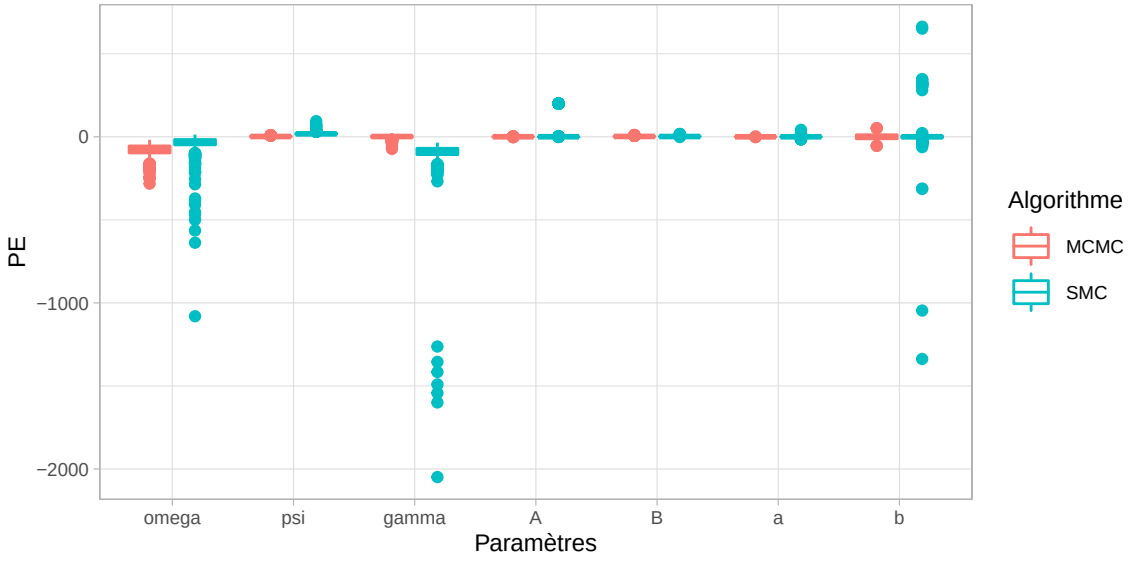

Figure 12: Boxplot des PE des 1000 estimations de chacun des paramètres.

Nous pouvons remarquer qu'avec la version MCMC d'estimation de ξ_x, hormis ω, les PE de l'ensemble des paramètres varient autour de 0. En effet, ω semble être sous-estimé et avoir une variation plus grande. Nous pouvons penser que cette variation est due au fait que la valeur de ω initiale de 0.01 est faible, ce qui rend sa source de bruit difficile à distinguer des autres, plus importantes, qui sont elles bien estimées.

En revanche, il semblerait qu'avec la méthode SMC d'estimation de ξ_x, l'estimation de ω est meilleure, mais celles de ψ et γ est nettement moins bonne. L'erreur réalisée sur l'estimation des paramètres A, B, a et b paraît similaire pour les deux approches d'estimation de ξ_x.

Table 1: Valeurs attendues, estimées en moyenne et erreurs associées de chacun des paramètres, en utilisant une étape MCMC dans l'algorithme SAEM.

	ω	ψ	γ	A	B	a	b
θ	0.010	0.951	0.098	0.500	-0.250	0.100	1.000
$\hat{\theta}$	0.018	0.932	0.097	0.499	-0.245	0.100	0.999
MAPE	79.859	2.143	6.153	0.486	2.100	0.467	13.911
RMSE	0.009	0.026	0.008	0.003	0.006	0.001	0.173

Au regard de la Table 1, il convient de relativiser l'erreur réalisée sur le paramètre ω avec la méthode MCMC : la valeur attendue est 0.01 et celle estimée en moyenne est égale à 0.014. La $MAPE$ peut sembler élevée,

mais encore une fois, il faut observer qu'en valeur absolue les estimations sont proches de la valeur à estimer, comme l'indique la $RMSE$. Nous pouvons noter que c'est bien le paramètre b dont l'estimation est la plus éloignée de la valeur attendue, ce qui rejoint les observations faites à partir des Figures 10 et 11.

Table 2: Valeurs attendues, estimées en moyenne et erreurs associées de chacun des paramètres, en utilisant une étape SMC dans l'algorithme SAEM.

	ω	ψ	γ	A	B	a	b
θ	0.010	0.951	0.098	0.500	-0.250	0.100	1.000
$\hat{\theta}$	0.014	0.773	0.197	0.478	-0.246	0.100	0.980
MAPE	40.939	18.723	101.959	4.579	1.752	0.453	16.310
RMSE	0.007	0.188	0.158	0.145	0.006	0.002	0.766

La Table 2 confirme les observations visuelles : la $MAPE$ des paramètres ψ et γ augmentent fortement en estimant ξ_x via un SMC par rapport à la méthode MCMC, tandis que celle du paramètre ω diminue légèrement et que pour les autres paramètres nous n'observons pas de réelle différence.

Au regard de ces résultats, la version MCMC semble être à privilégier par rapport à la SMC. Peut-être qu'il serait possible d'améliorer les estimations de la version SMC en jouant sur ses hyper-paramètres, mais cette démarche serait très coûteuse en temps de calcul, et potentiellement sensible aux choix des valeurs fixées des paramètres de la simulation.

Références

[1] Wikipedia contributors. *Narval*. 2022. URL: https://fr.wikipedia.org/wiki/Narval (visited on 02/02/2023).

[2] Jean-Pierre Sylvestre. *Dans l'Arctique canadien avec la licorne de mer*. 2018. URL: https://www.peuple-animal.com/article,lecture,1160_dans-l-arctique-canadien-avec-la-licorne-de-mer.html (visited on 02/02/2023).

[3] Peter Bondo. *The narwhal's tusk reveals its past living conditions*. 2021. URL: https://phys.org/news/2021-03-narwhal-tusk-reveals-conditions.html (visited on 02/02/2023).

[4] A. P. Dempster, N. M. Laird, and D. B. Rubin. "Maximum Likelihood from Incomplete Data Via the EM Algorithm". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 39.1 (1977), pp. 1–22. DOI: https://doi.org/10.1111/j.2517-6161.1977.tb01600.x. eprint: https://rss.onlinelibrary.wiley.com/doi/pdf/10.1111/j.2517-6161.1977.tb01600.x. URL: https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.2517-6161.1977.tb01600.x.

[5] Bernard Delyon, Marc Lavielle, and Eric Moulines. "Convergence of a Stochastic Approximation Version of the EM Algorithm". In: *The Annals of Statistics* 27.1 (1999), pp. 94–128. URL: http://www.jstor.org/stable/120120 (visited on 02/21/2023).

[6] Estelle Kuhn and Marc Lavielle. "Coupling a stochastic approximation version of EM with an MCMC procedure". In: *ESAIM: Probability and Statistics* 8 (2004), pp. 115–131. DOI: 10.1051/ps:2004007. URL: http://www.numdam.org/articles/10.1051/ps:2004007/.

[7] Sophie Donnet and Adeline Samson. "Using PMCMC in EM algorithm for stochastic mixed models: theoretical and practical issues". In: *Journal de la société française de statistique* 155.1 (2014), pp. 49–72. URL: http://www.numdam.org/item/JSFS_2014___155_1_49_0/.