SIGMA203b Mini Projet

Yichen ZHU

April 24, 2017

1 Questions lab

1.1 Exercice 1. Débruitage et interpolation sinusoidal avec Filtre Kalman

1.1.1 Génération des données

On a un modèle HMM qui représente la phase d'un signal sinusoidal :

$$\begin{aligned} X_0 &= X_0 \sim \mathcal{N}(0, S) \\ X_t|X_{t-1} &= x_{t-1} \sim \mathcal{N}(Ax_{t-1}, Q) \\ Y_t|X_t &= x_t \sim \mathcal{N}(Bx_t, R) \\ A &= exp(-\gamma) \begin{pmatrix} \cos(\omega) & -\sin(\omega) \\ \sin(\omega) & \cos(\omega) \end{pmatrix} \\ B &= \begin{pmatrix} \sqrt{2} & 0 \end{pmatrix} \end{aligned}$$

On génère X et Y selon ce modèle en utilisant la fonction *lingauss_simul* et

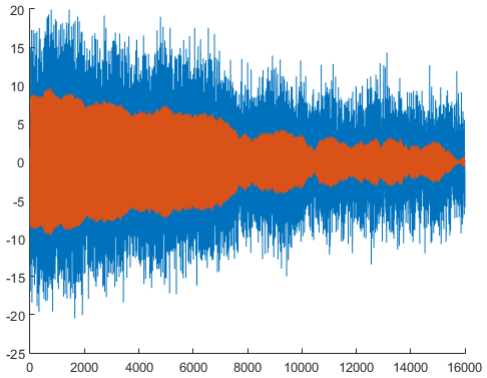

Figure 1: Observation en bleu et vrai etat en rouge

on s'intéresse aux paramètres ω,γ,Q et R. ω est donc le paramètre de fréquence

et γ permet de faire décroitre l'energie ou autrement dit l'amplitude du signal. Plus γ est grand, plus vite l'energie décroit. Q est le bruit d'évolution quand il est grand, la variance c'est à dire l'incertitude de X est grand. R est le bruit d'observation, on observe X avec un bruit, plus l'observation est bruitée, moins le résultat est précis.

1.1.2 Débruitage

On implémente le filtre Kalman pour débruiter l'observation et estimer X. Le principe de cette méthode est de prendre la moyenne de la prédiction et l'observation comme si on a deux mesures pour préciser la valeur estimée.

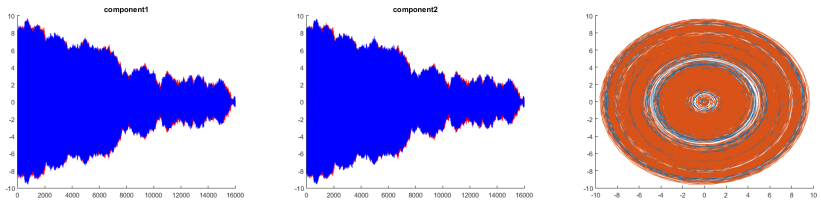

Figure 2: Projection du signal sur trois axes

Quand on fait augmenter le bruit d'observation, on calcule le SNR pour evaluer l'efficacité du filtre Kalman. Il n'y a pas beaucoup de différence pour le SNR quand R est dans l'ordre de 10^4. Cela prouve que le filtre Kalman est très utile pour des observations bruitées.

1.1.3 Réstauration

On veut montrer que même si l'observation Y n'est pas complète, le filtrage fonctionne toujours. Cela vient du fait que :

$$p(x_{k+1}|x_k) = \int p(x_{k+1}|x_k)p(x_k|y_{1:k})dx_k$$

On met une partie de y en 0 :

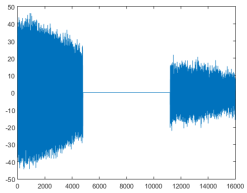

Figure 3: Observation n'est pas complète

on applique le filtre de Kalman :

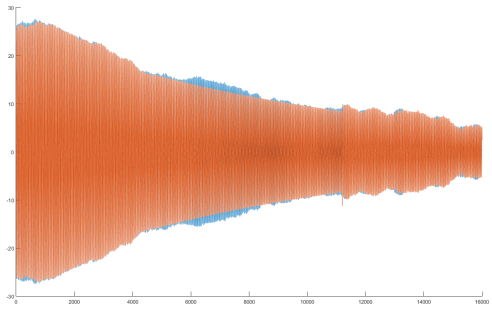

Figure 4: Vrai etat en bleu et estimé en rouge

Donc on voit qu'en utilisant la prédiction, l'absence de B n'est pas un problème pour restaurer la partie perdue.

1.2 Exercice 2. Pitch tracking monophonique avec Filtre Particulaire

1.2.1 Génération des données

On prend toujours le même modèle pour générer le sinus, on rajoute encore un état caché pour la fréquence. Donc on a M pitch à utiliser. La matrice de transition pour aller d'un pitch à un autre est (ici on prend $M = 3$) :

$$p(R_t|R_{t-1}) = \begin{pmatrix} 1-\nu & \frac{\nu}{M-1} & \frac{\nu}{M-1} \\ \frac{\nu}{M-1} & 1-\nu & \frac{\nu}{M-1} \\ \frac{\nu}{M-1} & \frac{\nu}{M-1} & 1-\nu \end{pmatrix}$$

On génère une mélodie et on l'écoute :

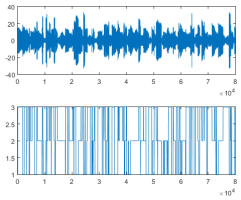

Figure 5: Génération des données

1.2.2 Pitch tracking

On implémente l'algorithme du SIS (Sequential Importance Sampling). Ici on utilise le Bootstrap Filter c'est à dire que notre loi de proposition :

$$\pi(X_k|X_{0:k-1}, y_{1:k}) = p(X_k|X_{k-1})$$

À chaque instant, on tire N échantillons qui suivent la loi de proposition π. Comme c'est un Bootstrap filtre, on fait evoluer le poid W avec la loi d'observation $p(Y_k|X_k)$ et on note $W^* = Wp(Y_k|X_k)$. Quand on finit de tirer les particules, on pose $W = \frac{W^*}{\sum W^*}$. Ici ce qu'on veut estimer est l'indice du pitch r, à la fin de chaque tirage, on prend le maximum du poid W pour voir quelle est la particule la plus importante. On fait itérer cet algorithme et on a le résultat :

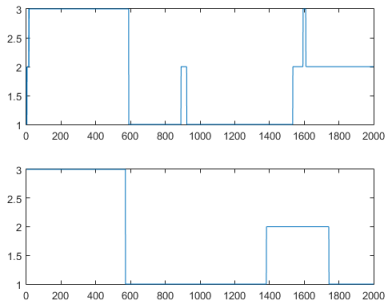

Figure 6: etat estimé et vrai etat

2 Questions écrites

2.1 Exercice 3. Filtre de Kalman étendu

2.1.1 Calcul des différentielles

On a notre modèle HMM :

$$\begin{cases} \begin{pmatrix} X_{1,k+1} \\ X_{2,k+1} \end{pmatrix} &= \begin{pmatrix} X_{1,k+1} + hX_{2,k+1} \\ X_{2,k+1} - \gamma h \sin(X_{1,k+1}) \end{pmatrix} + \begin{pmatrix} \epsilon_{1,k+1} \\ \epsilon_{2,k+1} \end{pmatrix} \\ Y_{k+1} &= \sin(X_{1,k+1}) + \eta_{k+1} \end{cases}$$

On réecrit le modèle avec les fonctions f, g sous la forme ci-dessous :

$$\begin{cases} X_{k+1} &= f(X_k) + \epsilon_k \\ Y_{k+1} &= g(X_{k+1}) + \eta_{k+1} \end{cases}$$

avec :

$$\left\{ \begin{array}{r} f(X_{1,k}, X_{2,k}) = \begin{pmatrix} X_{1,k+1} + hX_{2,k+1} \\ X_{2,k+1} - \gamma h \sin(X_{1,k+1}) \end{pmatrix} \\ g(X_{1,k+1}, X_{2,k+1}) = \sin(X_{1,k+1}) \end{array} \right.$$

Les fonctions f, g sont non-linéaires, on veut donc effectuer une approximation gaussienne pour linéariser ce modèle. On calcule la différentielle de f au X_k et celle de g au X_{k+1} :

$$\left\{ \begin{array}{rcl} F_x & = & \begin{pmatrix} 1 & h \\ -\gamma h \cos(X_{1,k}) & 1 \end{pmatrix} \\ G_x & = & \begin{pmatrix} \cos(X_{1,k}) & 0 \end{pmatrix} \end{array} \right.$$

2.1.2 Étape de prédiction

On suppose que $(X_{1,k-1}|y_{1:k-1}) \sim \mathcal{N}(m_{k-1}, P_{k-1})$ On en déduit la loi jointe $\begin{pmatrix} X_{k-1} \\ X_k \end{pmatrix} |y_{1:k-1}\Big)$ en fonction de m_{k-1}, P_{k-1}. On a donc :

$$\mathcal{L}\left(\begin{array}{c} X_{k-1} \\ X_k \end{array} |y_{1:k-1}\right) = \mathcal{L}\left(\begin{array}{c} X_{k-1} \\ f(X_{k-1}) + \epsilon_{k-1} \end{array} |y_{1:k-1}\right) = \mathcal{L}\begin{pmatrix} W_{k-1} \\ f(W_{k-1}) + \epsilon_{k-1} \end{pmatrix}$$

D'où W_k est un vecteur aléatoire $W_k \sim \mathcal{N}(m_{k-1}, P_{k-1})$ Ensuite, on effectue l'approximation linéaire :

$$\begin{pmatrix} m_{k-1} + \delta W_{k-1} \\ f(m_{k-1}) + A\delta W_{k-1} + \epsilon_{k-1} \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m_{k-1} \\ m_k^- \end{pmatrix}, \begin{pmatrix} P_{k-1} & P_{k-1}A^T \\ AP_{k-1} & P_k^- \end{pmatrix} \right)$$

Avec : A la différentielle $A = F_x(m_{k-1})$, $m_k^- = f(m_{1,k-1}, m_{2,k-1})$ et $P_k^- = AP_{k-1}A^T + Q_{k-1}$. On peut donc en déduire l'approximation de la loi marginale $\mathcal{L}(X_k|y_{1:k-1}) \approx \mathcal{N}(m_k^-, P_k^-)$

2.1.3 Étape de mise à jours

Pareil que l'étape de prédiction, on fait l'approximation linéaire pour la loi jointe $\begin{pmatrix} X_k \\ Y_k \end{pmatrix} |y_{1:k-1}\Big)$ pour effectuer l'étape de mise à jours. On a donc :

$$\mathcal{L}\left(\begin{array}{c} X_k \\ Y_k \end{array} |y_{1:k-1}\right) = \mathcal{L}\left(\begin{array}{c} X_k \\ g(X_k) + \eta_k \end{array} |y_{1:k-1}\right) = \mathcal{L}\begin{pmatrix} W_k \\ g(W_k) + \eta_k \end{pmatrix}$$

D'où W_k est un vecteur aléatoire $W_k \sim \mathcal{N}(m_k^-, P_k^-)$. Ensuite, on exprime $\mathcal{L}\begin{pmatrix} W_k \\ g(W_k) + \eta_k \end{pmatrix}$ en fonction de m_k^-, P_k^- :

$$\begin{pmatrix} m_k^- + \delta W_k \\ f(m_k^-) + B\delta W_k + \eta_k \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m_k^- \\ g(m_k^-) \end{pmatrix}, \begin{pmatrix} P_k^- & P_k^- B^T \\ BP_k^- & S \end{pmatrix} \right)$$

Avec : B la différentielle $B = G_x(m_k^-)$ et $S = BP_k^- B^T + R_k$.

2.1.4 Filtrage

On en déduit donc l'approximation de la loi marginale de filtrage: $\mathcal{L}(X_k|y_{1:k}) \approx \mathcal{N}(m_k, P_k)$ avec :

$$\begin{cases} m_k &= f(m_{1,k-1}, m_{2,k-1}) + K(y_k - g(f(m_{1,k-1}, m_{2,k-1})) \\ P_k &= (I - KB)(AP_{k-1}A^T + Q_{k-1}) \\ K &= (AP_{k-1}A^T + Q_{k-1})B^T S^{-1} \\ S &= BP_k^- B^T + R_k \end{cases}$$

2.1.5 Pseudo-code

Entrées Y, f, F_x, g, G_x, Q, R

Initialisation

$$\begin{aligned} m_0 &= zeros(1, dim(X)) \\ P_0 &= 100I \end{aligned}$$

Algorithme Filtrage :
For $k \geq 1$
$A = F_x(m_{k-1})$;
$m_k^- = f(m_{k-1})$;
$B = G_x(m_k^-)$;
$P_k^- = AP_{k-1}A^T + Q$;
$S = BP_k^- B^T + R$;
$K = P_k^- B^T S^{-1}$;
$m_k = m_k^- + K(y_k - g(m_k^-))$;
$P_k = (I - KB)(P_k^-)$

Envoyer m_k, P_k

2.2 Exercice 4. Échantillonage d'importance

2.2.1 Algorithme d'échantillonage d'importance

On a la densité de probabilité $p(x) = \lambda p_0(x)$ avec $p_0(x) = \frac{\sin(\sqrt{x})}{x^{3.1}} \mathbb{1}_{(x>1)}$. On veut approcher $\mathbb{E}_p(f(x)) = \int f(x)p(x)dx$ par $P_N = \Sigma_{i=1}^N W_i f(X_i)$. Le principe d'échantillonage d'importance est le changement de mesure c'est à dire qu'on

approche avec une loi de proposition plus facile. On a :

$$\begin{aligned}
\mathbb{E}_p(f(x)) &= \int f(x)\lambda p_0(x)dx \\
&= \frac{\int f(x)p(x)dx}{\int p_0(t)dt} \\
&= \frac{\int f(x)\frac{p_0(x)}{N\pi(x)}\pi(x)dx}{\int \frac{p_0(x)}{N\pi(x)}\pi(x)dx} \\
&= \frac{\int f(x)w_x^*\pi(x)dx}{\int w_x^*\pi(x)dx} \\
&\approx \sum_i^n \frac{w_i^*}{\sum_j w_j^*} f(X_i)
\end{aligned}$$

avec : $w_i^* = \frac{p_0(X_i)}{\pi(X_i)}$. On montre cet estimateur est non-biaisé, on note $I_N(f) = \sum_i^n \frac{w_i^*}{\sum_j w_j^*} f(X_i)$ On a donc :

$$\begin{aligned}
\mathbb{E}_\pi(I_N(f)) = & \quad \mathbb{E}_\pi(\sum_i^n \frac{w_i^*}{\sum_j w_j^*} f(X_i)) \\
= & \quad \frac{1}{N}\sum_{i=1}^N \int_\pi \frac{\lambda p_0(x)f(x)}{\pi(x)} f(x)
\end{aligned}$$

Comme π est une mesure qui inclut f, p

$$\begin{aligned}
= & \quad \frac{1}{N}\sum_{i=1}^N \int_{f,p} \frac{\lambda p_0(x)f(x)}{\pi(x)} f(x) \\
= & \quad \mathbb{E}_p(f(x))
\end{aligned}$$

Donc l'estimateur de l'échantillonage d'importance n'est pas biaisé.

2.2.2 Étude sur le biais

On applique l'échantillonage d'importance standard.

$$\begin{aligned}
\mathbb{E}_p(f(x)) = & \quad \int_{f,p} f(x)p(x)dx \\
= & \quad \int_\pi \frac{f(x)p(x)}{\pi(x)}\pi(x)dx \\
= & \quad \mathbb{E}_\pi(\frac{f(x)p(x)}{\pi(x)})
\end{aligned}$$

avec la loi de proposition $\pi \sim \mathcal{U}_{x\in(0\frac{1}{2})}$. Si on veut que notre approximation ne soit pas biasié, il y a une condition necéssaire $[f, p] \subseteq \pi$. Si f n'est pas dans

l'intervalle de π, on aura un estimateur biaisé. Par exemple, si on prend $f(x)$ où x est défini dans l'intervalle $(\frac{1}{4}), \frac{3}{4})$, on a :

$$\begin{aligned}
\mathbb{E}_\pi(\frac{f(x)p(x)}{\pi(x)}) = & \int_0^{\frac{1}{2}} \frac{f(x)p(x)}{\pi(x)}\pi(x)dx \\
= & \int_0^{\frac{1}{2}} f(x)p(x)dx \\
\neq & \int_{\frac{1}{4}}^{\frac{3}{4}} f(x)p(x)dx
\end{aligned}$$

Pour conclure, toutes les fonction f qui ne sont pas définies dans une intervalle inclut dans $(0, \frac{1}{2})$ ne possède pas un estimateur non-biaisé pour la loi de proposition π.