

DM - filtrage

KATHIR Ranujan - Nathan Graveleau - Antoine Chapron

13/02/2023

Position du problème et modèle

Dans un monde situé dans un univers parallèle, unidimensionnel et où le temps s'écoule de façon discrète, le pays A déclare la guerre au pays B, notamment en visant ses installations stratégiques avec des missiles guidés. De fait, le pays B décide de mettre en place des moyens techniques afin de "tracker" les missiles du pays A. Pour ce faire, pays B dispose de systèmes de radars dont la mesure de la position du missile, est, malheureusement entachée d'erreur. Les fabricants de missiles du pays A laissent une certaine latitude à leurs missiles guidés : ces derniers sont autorisés à modifier leur vitesse en choisissant aléatoirement d'accélérer ou de décélérer suivant une loi gaussienne dont les paramètres sont connus du pays B. \

On se place du point de vue d'opérateurs du pays B, cherchant à estimer la position des missiles. Dans le modèle ci-dessous, la suite $(X_n)_n$ désigne la position du missile guidé au cours du temps, l'origine du temps et de l'espace sont initialisés à l'instant de lancement du missile et à son site, respectivement; et la suite $(Y_n)_n$ désigne les mesures faites par le radar depuis le moment où le missile quitte les installations. \

Les suites $(W_n)_n$ et $(V_n)_n$ sont deux suites de v.a. i.i.d., indépendantes, telles que pour tout n dans \mathbb{N} , $W_n \sim \mathcal{N}(0, \sigma^2)$ et $V_n \sim \mathcal{N}(0, \tau^2)$ (la première représente la latitude accordée aux missiles à chaque instant, la seconde représente l'erreur de mesure du radar). On considère, pour $n \geq 0$, $a \in (0, 1)$,

$$\begin{cases} X_{n+1} = aX_n + W_n, & X_0 = W_0 \\ Y_n = X_n + V_n \end{cases}$$

Votre but consiste, de fait, à estimer la position du missile à chaque instant n , i.e. X_n , au vu des observations faites par le radar jusqu'à l'instant n , i.e. Y_0, Y_1, \dots, Y_n . On pourrait, éventuellement, aussi s'intéresser (à des fins de préventions ou de contremesure) à la position future, ou prédite du missile, i.e. X_{n+1} , au vu des observations faites par le radar jusqu'à l'instant n , i.e. Y_0, Y_1, \dots, Y_n . Dans les deux cas, le candidat naturel est l'espérance conditionnelle, respectivement $\mathbb{E}[X_{n+1}|Y_0, \dots, Y_n]$ et $\mathbb{E}[X_n|Y_0, \dots, Y_{n-1}]$. Pour ce faire, vous procédez par méthode de filtrage, en l'état, un filtre de Kalman.

Filtrage

On pose, pour tout $n \geq 0$, \

$$\hat{X}_n := \mathbb{E}[X_n|Y_0, \dots, Y_n], P_n := \mathbb{E}[(X_n - \hat{X}_n)^2].$$

et pour $n \geq 1$, \

$$\hat{X}_n^- := \mathbb{E}[X_n|Y_0, \dots, Y_{n-1}], P_n^- := \mathbb{E}[(X_n - \hat{X}_n^-)^2].$$

En vous aidant des résultats fournis en Annexe, établir les différentes étapes du filtre de Kalman:

1. **Initialisation du filtre :** Déterminer la Loi($X_0|Y_0$). En déduire \hat{X}_0 et P_0 .

On a $X_0 = W_0 \sim \mathcal{N}(0, \sigma^2)$ et $Y_0 = X_0 + V_0$ avec $V_0 \sim \mathcal{N}(0, \tau^2)$. Donc, X_0 est gaussien et Y_0 l'est aussi en tant que somme de variable gaussienne. On conclut que, $(X_0|Y_0)$ suit une loi gaussienne d'espérance \hat{X}_0 et de variance P_0 . Alors, on a :

$$\hat{X}_0 := \mathbb{E}[X_0|Y_0] = \mathbb{E}[Y_0 - V_0|Y_0] = \mathbb{E}[Y_0|Y_0] = Y_0$$

\

$$P_0 := \mathbb{E}[(X_0 - \hat{X}_0)^2] = \mathbb{E}[W_0^2] - 2\mathbb{E}[X_0 Y_0] + \mathbb{E}[Y_0^2] = \sigma^2 - 2\mathbb{E}[X_0 Y_0] + \mathbb{E}[Y_0^2]$$

Par indépendance des v.a. W_0 et V_0 , on obtient $\mathbb{E}[X_0 Y_0] = \mathbb{E}[W_0(W_0 + V_0)] = \mathbb{E}[W_0^2] + \mathbb{E}[V_0 W_0] = \sigma^2$. On s'intéresse au calcul de $\mathbb{E}[Y_0^2]$, pour cela, on peut développer en sachant que $Y_0 = X_0 + V_0$. \

$$\mathbb{E}[Y_0^2] = \mathbb{E}[(X_0 + V_0)^2] = \mathbb{E}[W_0^2] + 2\mathbb{E}[W_0]\mathbb{E}[V_0] + \mathbb{E}[V_0^2] = \sigma^2 + \tau^2$$

Donc, $P_0 = \sigma^2 - 2\sigma^2 + \sigma^2 + \tau^2 = \tau^2$.

2. **Étape de prédiction :** pour $n \geq 1$, déterminer $\text{Loi}(X_n|Y_0, \dots, Y_{n-1})$ en fonction de X_{n-1}^\wedge , P_{n-1} et des paramètres du modèle. En déduire \hat{X}_n^- et P_n^- . En déduire \hat{X}_0 et P_0 .

Pour n supérieur ou égal à 1, nous pouvons déterminer la loi de X_n sachant Y_0, \dots, Y_{n-1} en utilisant les équations de Kalman.

Comme $X_n = aX_{n-1} + W_{n-1}$, nous avons :

$$\hat{X}_n = \mathbb{E}(X_n|Y_0, \dots, Y_n) = aX_{n-1}^\wedge + \mathbb{E}(W_{n-1})$$

Comme W_{n-1} suit une loi normale d'espérance 0, nous avons $\mathbb{E}(W_{n-1}) = 0$

$$\hat{X}_n = aX_{n-1}^\wedge$$

$$P_n = \mathbb{E}((X_n - \hat{X}_n)^2) = \mathbb{E}((aX_{n-1} + W_{n-1} - aX_{n-1}^\wedge)^2)$$

$$P_n = a^2\mathbb{E}((X_{n-1} - X_{n-1}^\wedge)^2) + \mathbb{E}(W_{n-1}^2)$$

Comme W_{n-1} suit une loi normale de variance σ^2 , nous avons $\mathbb{E}(W_{n-1}^2) = \sigma^2$

$$P_n = a^2P_{n-1} + \sigma^2$$

En utilisant ces relations, nous pouvons déterminer \hat{X}_n^- et P_n^- en utilisant les équations suivantes :

$$\hat{X}_n^- = \mathbb{E}(X_n|Y_0, \dots, Y_{n-1}) = aX_{n-1}^\wedge$$

$$P_n^- = \mathbb{E}((X_n - \hat{X}_n^-)^2) = a^2P_{n-1} + \sigma^2$$

Ainsi, nous avons déterminé la loi de X_n sachant Y_0, \dots, Y_{n-1} , qui est une loi normale d'espérance \hat{X}_n^- et de variance P_n^- .

3. **Étape de mise à jour / filtrage** : pour $n \geq 1$, montrer que

$$\text{Loi}(X_n|Y_0, \dots, Y_n) = \mathcal{N}(\hat{X}_n, P_n),$$

où, \

$$\hat{X}_n = \hat{X}_n^- + \frac{P_n}{\tau^2}(Y_n - \hat{X}_n^-), P_n = \tau^2 \frac{P_n^-}{P_n^- + \tau^2}$$

L'équation $\hat{X}_n = \hat{X}_n^- + \frac{P_n}{\tau^2}(Y_n - \hat{X}_n^-)$ est une formule de mise à jour des prévisions dans le filtre de Kalman. Elle exprime comment la prévision précédente \hat{X}_n^- de la variable X_n est mise à jour en utilisant l'observation récente Y_n .

Pour comprendre la signification de cette équation, nous pouvons considérer un modèle simple où X_n est la position d'un objet en mouvement et Y_n est une observation de sa position. Dans ce cas, \hat{X}_n^- est la prévision précédente de la position de l'objet basée sur les observations antérieures, et P_n est la variance associée à cette prévision.

Lorsqu'une nouvelle observation Y_n est disponible, nous pouvons utiliser l'équation $\hat{X}_n = \hat{X}_n^- + \frac{P_n}{\tau^2}(Y_n - \hat{X}_n^-)$ pour mettre à jour notre prévision de la position de l'objet. Le terme $\frac{P_n}{\tau^2}(Y_n - \hat{X}_n^-)$ mesure la correction apportée à la prévision précédente en fonction de la différence entre la prévision et l'observation. La quantité $\frac{P_n}{\tau^2}$ est appelée gain de Kalman et permet d'ajuster la correction en fonction de la qualité de la prévision précédente (mesurée par la variance P_n) et de la qualité de l'observation (mesurée par la variance τ^2).

En résumé, l'équation $\hat{X}_n = \hat{X}_n^- + \frac{P_n}{\tau^2}(Y_n - \hat{X}_n^-)$ permet de combiner les informations fournies par les observations antérieures et la nouvelle observation pour produire une prévision mise à jour de la variable d'intérêt.

Pour prouver que $X_n|Y_0, \dots, Y_n$ suit une loi normale, il est nécessaire de déterminer son espérance et sa variance.

L'espérance de $X_n|Y_0, \dots, Y_n$ peut être trouvée en utilisant la formule $\mathbb{E}(X_n|Y_0, \dots, Y_n) = \mathbb{E}(\mathbb{E}(X_n|\hat{X}_n))$. Comme X_n et \hat{X}_n sont liés par la relation $\hat{X}_n = \hat{X}_n^- + \frac{P_n}{\tau^2}(Y_n - \hat{X}_n^-)$, nous avons $\mathbb{E}(X_n|\hat{X}_n) = \hat{X}_n$. Par conséquent, $\mathbb{E}(X_n|Y_0, \dots, Y_n) = \hat{X}_n$.

La variance de $X_n|Y_0, \dots, Y_n$ peut être trouvée en utilisant la formule $\text{Var}(X_n|Y_0, \dots, Y_n) = \text{Var}(\mathbb{E}(X_n|\hat{X}_n)) + \mathbb{E}(\text{Var}(X_n|\hat{X}_n))$. Comme $\mathbb{E}(X_n|\hat{X}_n) = \hat{X}_n$, nous avons $\text{Var}(\mathbb{E}(X_n|\hat{X}_n)) = 0$. De plus, $\text{Var}(X_n|\hat{X}_n) = P_n$, donc $\mathbb{E}(\text{Var}(X_n|\hat{X}_n)) = P_n$. Par conséquent, $\text{Var}(X_n|Y_0, \dots, Y_n) = P_n$.

Ainsi, nous pouvons conclure que $X_n|Y_0, \dots, Y_n$ suit une loi normale d'espérance \hat{X}_n et de variance P_n .

Mise en oeuvre du filtre

Pour cette partie, nous allons modifier légèrement le problème ci-dessus en supposant la poursuite d'une seule cible (un missile) à partir des données angulaires de deux capteurs (radars). L'état de la cible à l'instant n consiste à la position (X_n, Z_n) et à sa vitesse en 2S (\dot{X}_n, \dot{Z}_n) . La dynamique du vecteur d'état $\mathbb{X}_n = (X_n, Z_n, \dot{X}_n, \dot{Z}_n)^T$ est modélisée par le modèle discrétisé de Wiener aussi connu sous le nom de "Nearly Constant Velocity (NCV) model" et est donné comme suit :

$$\begin{pmatrix} X_n \\ Z_n \\ \dot{X}_n \\ \dot{Z}_n \end{pmatrix} = \begin{pmatrix} 1 & 0 & \Delta & 0 \\ 0 & 1 & 0 & \Delta \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X_{n-1} \\ Z_{n-1} \\ \dot{X}_{n-1} \\ \dot{Z}_{n-1} \end{pmatrix} + \mathbb{W}_n, n = 1, \dots, 500$$

où \mathbb{W}_n est un processus multi-dimensionnel centré de loi Gaussienne et de matrice de covariance donnée par

$$Q = \begin{pmatrix} q_1^c \frac{\Delta^3}{3} & 0 & q_1^c \frac{\Delta^2}{2} & 0 \\ 0 & q_2^c \frac{\Delta^3}{3} & 0 & q_2^c \frac{\Delta^2}{2} \\ q_1^c \frac{\Delta^2}{2} & 0 & q_1^c \Delta & 0 \\ 0 & q_2^c \frac{\Delta^2}{2} & 0 & q_2^c \Delta \end{pmatrix}$$

Dans ce scénario les coefficients de diffusion sont $q_1^c = q_2^c = 0.1$ et la période d'échantillonnage entre deux observations est $\Delta = 0.01$. L'équation de mesure pour chaque capteur $i \in 1, 2$ est donnée par :

$$\theta_n^i = \tan^{-1} \left(\frac{Z_n - s_z^i}{X_n - s_x^i} \right) + V_n$$

où (s_z^i, s_x^i) est la position du capteur i et $V_n \sim \mathcal{N}(0, \sigma^2)$ correspond au bruit de mesure centré et de standard déviation σ égale à 0.05.

But : Estimer la trajectoire de la cible $X_{1:T}$ à partir des observations bruitées issues des radars $\theta_{1:T}^i, i = 1, 2$.

Pour ce faire, nous travaillerons sur données simulées et supposons pour simplifier le problème qu'un seul capteur $i=1$ est disponible et tel que $(s_x^1, s_z^1) = (-1.5, 0.5)$.

- 1) Simuler les données $\theta_{1:T}^1$ selon le modèle (1) avec les paramètres définis ci-dessus et $X_1 = (0, 0, 1, 0)$.

Librarie

```
library(MASS)
library(mvtnorm)
set.seed(123)
```

```
## Loading required package: Matrix
##
## Attaching package: 'expm'
## The following object is masked from 'package:Matrix':
##
##      expm
```

initialisation des paramètres

```
n <- 500
delta <- 0.01
q_1 = 0.1
q_2 = 0.1

C <- matrix(c(1,0,delta,0,0,1,0,delta,0,0,1,0,0,0,0,1), ncol=4, byrow = TRUE)
Q <- matrix(c(q_1*(delta^3)/3,0,q_1*(delta^2)/2,0,0,q_2*(delta^3)/3,0,q_2*(delta^2)/2,q_1*(delta^2)/2,0,0,q_2*(delta^2)/2,0,0,q_1*delta,q_2*delta), ncol=4, byrow = TRUE)

# création des vecteurs d'états par l'observation (chaque colonne représente ainsi la composante)
X <- matrix(0, ncol=4, nrow=n)
X[1,] <- c(0,0,1,0)

# création du processus multi-dimensionnel centré de loi gaussienne et de matrice de covariance Q
sigma <- 0.05
W <- mvrnorm(n, mu = c(0,0,0,0), Sigma = Q)
V <- rnorm(n,0,sigma)
```

```

# création de la boucle : trouver les vecteurs d'états pour chaque états
for (i in 2:n){
  X[i,] <- C %*% X[i-1,] + W[i,]
}

# simulation des données theta_1
theta_11 <- sample(n) # création de l'échantillon

s_x1 <- -1.5
s_z1 <- 0.5

for (i in 1:n){
  theta_11[i] <- atan((X[i,2]-s_z1)/(X[i,1]-s_x1)) + V[i]
}

plot(theta_11, type="l", xlab = "temps (t)", ylab = "mesure")

```

DM--Filtrage_files/figure-latex/unnamed-chunk-3-1.pdf

- 2) Pour ce problème, nous ne souhaitons pas utiliser un filtre particulière. Dans ce cas, que proposeriez-vous pour estimer la trajectoire du missile ? Justifier votre choix.

On utilise un filtre de Kalman étendu pour estimer la trajectoire. Toutefois, il faut linéariser la fonction. Pour cela, on va utiliser le DL de la fonction arctan à l'ordre 1 pour chacune des composantes du vecteur X sur la fonction θ_n^i et on obtient un vecteur de taille 4 avec deux composantes nulles.

On réécrit l'équation de mesure pour mieux appréhender la linéarisation: \

$$\theta_n^i = f(\mathbb{X}_n, V_n)$$

où f est la fonction définie par $f(\mathbb{X}_n, V_n) = \tan^{-1}\left(\frac{Z_n - s_z^i}{X_n - s_x^i}\right) + V_n$.

Les dérivées obtenues selon les composantes de \mathbb{X}_n sont les suivantes: \

$$\frac{\partial \theta_n^1}{\partial X_n} = -\frac{Z_n - s_z^1}{(X_n - s_x^1)^2 + (Z_n - s_z^1)^2}, \frac{\partial \theta_n^1}{\partial Z_n} = \frac{X_n - s_x^1}{(X_n - s_x^1)^2 + (Z_n - s_z^1)^2}, \frac{\partial \theta_n^1}{\partial \dot{X}_n} = 0, \frac{\partial \theta_n^1}{\partial \dot{Z}_n} = 0.$$

- 3) Implémenter l'approche que vous proposerez pour ce problème. #### Linéarisation

```

# On utilise un filtre de Kalman pour estimer la trajectoire.
# Toutefois, il faut linéariser la fonction.
# Pour cela, on va utiliser le DL de la fonction arctan à l'ordre 1
jacobien <- matrix(0, ncol=4, nrow=n) # création des vecteurs

# Pour cela, on fixe Xn, et on effectue le DL selon la variable Zn
for (j in 1:n){
  jacobien[j,1] <- -((X[j,2]-s_z1)/((X[j,1]-s_x1)^2+(X[j,2]-s_z1)^2)
  jacobien[j,2] <- ((X[j,1]-s_x1)/((X[j,1]-s_x1)^2+(X[j,2]-s_z1)^2)
  jacobien[j,3] <- 0
  jacobien[j,4] <- 0
}

```

Filtrage

```
# Extended Kalman filter application in order to estimate the vector X from the measurements Y
n = length(theta_11)
d = dim(C)[1]

P.minus = array(0,c(d,d,n))
P.minus[, ,1] = Q
X.minus = matrix(0,ncol = d,nrow = n)
X.minus[1,] = X[1,]
P = array(0,c(d,d,n))
X.hat = matrix(0,ncol = d,nrow = n)
K = matrix(0,ncol = d,nrow = n)
P0=matrix(0,ncol = d,nrow = d)
P[, ,1] = P0
X.hat[1,] = X[1,]
theta.minus1 <- sample(n)
for (k in 2:n){

  # Prediction step
  X.minus[k,] = C%*%X.hat[k-1,]
  P.minus[, ,k] = C %*% P[, ,k-1] %*% t(C) + Q
  theta.minus1[k] = atan((X.minus[k,2]-s_z1)/(X.minus[k,1]-s_x1))
  # Update step
  K[k,] = (P.minus[, ,k] %*% jacobien[k,]) / (jacobien[k,] %*% P.minus[, ,k] %*% jacobien[k,] + sigma^2)

  X.hat[k,] = X.minus[k,] + K[k,] * (theta_11[k]-theta.minus1[k])
  res = (1 - K[k,]%*%jacobien[k,])
  P[, ,k] = res[1,1]*P.minus[, ,k]
}

theta.hat <- sample(n)
for (k in 2:n){
  theta.hat[k] <- atan((X.hat[i,2]-s_z1)/(X.hat[i,1]-s_x1)) + V[i]
}
```

Résultats

Visualisation des composants du vecteur X

```
par(mfrow = c(2,2))
plot(X[,1], type = "l", col = "blue")
lines(X.hat[,1], type = "l", col = "red")

plot(X[,2], type = "l", col = "blue")
lines(X.hat[,2], type = "l", col = "red")

plot(X[,3], type = "l", col = "blue")
lines(X.hat[,3], type = "l", col = "red")

plot(X[,4], type = "l", col = "blue")
lines(X.hat[,4], type = "l", col = "red")
```

DM--Filtrage_files/figure-latex/unnamed-chunk-6-1.pdf

Nous n'obtenons pas de bons résultats. En effet, les courbes ne cessent d'augmenter, on peut penser à une erreur de calcul lors de la linéarisation.

- 4) Auriez-vous pu considérer une autre approche pour cette estimation qui donnerait de meilleurs résultats. Justifier pourquoi. (**Pour cette question, il n'est pas demandé d'implémenter une approche alternative.**)

Compte-tenu du fait qu'on ait des données bruyantes, un autre moyen d'estimer la trajectoire d'un missile est d'utiliser des filtres nécessitant des méthodes de Monte-Carlo séquentielles en utilisant une taille d'échantillon suffisamment large. Il permet d'être plus précis que le filtre de Kalman étendu. Un exemple de filtre qu'on peut utiliser est le filtre SIS.

- 5) On décide de déplacer la position du capteur $i = 1$ en posant $(s_x^1, s_z^1) = (-1, 1)$. Obtenez-vous les mêmes résultats sur l'estimation de la position du missile ?

Filtrage

```
# simulation des données theta_2
theta_12 <- sample(n) # création de l'échantillon

## 2ème cas :
s_x2 <- -1
s_z2 <- 1

# on utilise la boucle pour remplir
for (i in 1:n){
  theta_12[i] <- atan((X[i,2]-s_z2)/(X[i,1]-s_x2)) + V[i]
}

plot(theta_12, type="l")
```

DM--Filtrage_files/figure-latex/unnamed-chunk-8-1.pdf

```
# On utilise un filtre de Kalman pour estimer la trajectoire. Toutefois, il faut linéariser la fonction
jacobien2 <- matrix(0, ncol=4, nrow=n) # création des vecteurs `
# Pour cela, on fixe Xn, et on effectue le DL selon la variable Zn
for (j in 1:n){
  jacobien2[j,1] <- -(X[j,2]-s_z2)/((X[j,1]-s_x2)^2+(X[j,2]-s_z2)^2)
  jacobien2[j,2] <- (X[j,1]-s_x2)/((X[j,1]-s_x2)^2+(X[j,2]-s_z2)^2)
  jacobien2[j,3] <- 0
  jacobien2[j,4] <- 0
}

# Kalman filter application in order to estimate the vector X from the measurements Y
n = length(theta_12)
```

```

d = dim(C)[1]

P.minus2 = array(0,c(d,d,n))
X.minus2 = matrix(0,ncol = d,nrow = n)
P2 = array(0,c(d,d,n))
X.hat2 = matrix(0,ncol = d,nrow = n)
K2 = matrix(0,ncol = d,nrow = n)
I = diag(1, nrow = d, ncol = d)
P0=matrix(0,ncol = d,nrow = d)
P2[, ,1] = P0
X.hat2[1,] = X[1,]
theta.minus2 = sample(n)
theta.minus2[1] <- theta_12

## Warning in theta.minus2[1] <- theta_12: le nombre d'objets à remplacer n'est pas
## multiple de la taille du remplacement

for (k in 2:n){

  # Prediction step
  X.minus2[k,] = C%%X.hat2[k-1,]
  P.minus2[, ,k] = C %% P2[, ,k-1] %% t(C) + Q
  theta.minus2[k] = atan((X.minus2[k,2]-s_z2)/(X.minus2[k,1]-s_x2))

  # Update step
  K2[k,] = (P.minus2[, ,k] %% jacobien2[k,]) / (jacobien2[k,] %% P.minus2[, ,k] %% jacobien2[k,] + sig

  X.hat2[k,] = X.minus2[k,] + K2[k,] * (theta_12[k]-theta.minus2[k])
  res = (1 - K2[k,]%%jacobien2[k,])
  P2[, ,k] = res[1,1]*P.minus2[, ,k]
}

theta.hat2 <- sample(n)
for (k in 2:n){
  theta.hat2[k] <- atan((X.hat2[i,2]-s_z2)/(X.hat2[i,1]-s_x2)) + V[i]
}

```

Résultats

Visualisation des composants du vecteur X

```

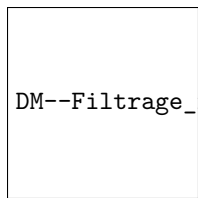
par(mfrow = c(2,2))
plot(X[,1], type = "l", col = "blue")
lines(X.hat2[,1], type = "l", col = 'red')

plot(X[,2], type = "l", col = "blue")
lines(X.hat2[,2], type = "l", col = "red")

plot(X[,3], type = "l", col = "blue")
lines(X.hat2[,3], type = "l", col = "red")

plot(X[,4], type = "l", col = "blue")
lines(X.hat2[,4], type = "l", col = "red")

```

DM--Filtrage_files/figure-latex/unnamed-chunk-9-1.pdf