ENSTA Paris

MA201 – Estimation et Identification Statistique

Localisation et Poursuite de Cible dans un Plan

Kateryna Tarelkina
Haïriya Guidado Aïssatou
Sofiia Chorna

November 6, 2023

Contents

1 Exercice 1

On dispose de deux mesures x_k et y_k correspondant à la position à l'instant t_k de la cible dans le plan X, Y.

1.1

Objectif : On considère dans un premier temps que la position courante de la cible est fonction de ses position, vitesse et accélération initiales (supposées constantes) à l'instant ($t_0 = 0$). Les mesures s'expriment en fonction des valeurs initiales sous la forme :

$$x_k = x_0 + v_{0x}t_k + a_{0x}t_k^2/2 + b_x,$$
$$y_k = y_0 + v_{0y}t_k + a_{0y}t_k^2/2 + b_y,$$

où $b_x \sim \mathcal{N}(0, \sigma_x^2)$, $b_y \sim \mathcal{N}(0, \sigma_y^2)$, avec $\sigma_x^2 = \sigma_y^2 = 2$. Montrer que le problème peut se mettre sous la forme d'un problème linéaire gaussien.

Développement : Montrons que le problème peut se mettre sous la forme d'un problème linéaire gaussien :

$$Z_k = H_k\theta + B_k$$

On trouve que :

$$\begin{pmatrix} x_k \\ y_k \end{pmatrix} = \begin{pmatrix} 1 & t_k & t_k^2/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & t_k & t_k^2/2 \end{pmatrix} \begin{pmatrix} x_0 \\ v_{0x} \\ a_{0x} \\ y_0 \\ v_{0y} \\ a_{0y} \end{pmatrix} + \begin{pmatrix} b_{x_k} \\ b_{y_k} \end{pmatrix},$$

où

$$Z_k = \begin{pmatrix} x_k \\ y_k \end{pmatrix}, H_k = \begin{pmatrix} 1 & t_k & t_k^2/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & t_k & t_k^2/2 \end{pmatrix}, \theta = \begin{pmatrix} x_0 \\ v_{0x} \\ a_{0x} \\ y_0 \\ v_{0y} \\ a_{0y} \end{pmatrix},$$

$$B_k = \begin{pmatrix} b_{x_k} \\ b_{y_k} \end{pmatrix} \sim \mathcal{N}(0, R_k), R_k = \begin{pmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{pmatrix}.$$

1.2

Objectif : Déterminer l'estimateur du maximum de vraisemblance associé. Quelles sont les propriétés en termes de biais et de convergence de cet estimateur?

Développement :

La variable Z suit une loi Gaussienne avec la moyenne $H_k\theta$ et la variance $R_k = \begin{pmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{pmatrix}$. La log-vraisemblance s'écrit donc :

$$\begin{aligned} \ln L(\theta; Z_1, \ldots, Z_N) = \ln f(Z_1, \ldots, Z_N; \theta) &= \sum_{k=1}^{N} \ln f_k(Z_k, \theta) \\ &= \sum_{k=1}^{N} -\ln(\sqrt{(2\pi)^{n_z} det(R_k)}) - \ln(n_z \det(R_k)) \\ &\quad -\frac{1}{2}(Z_k - H_k\theta - m_k)^T R_k^{-1}(Z_k - H_k\theta - m_k) \end{aligned}$$

La recherche du maximum amène à considérer l'égalité suivante :

$$\frac{\partial L(\theta; Z_1, \ldots, Z_N)}{\partial \theta} = 0$$

ce qui implique :

$$\sum_{k=1}^{N} \frac{\partial}{\partial \theta} \left((Z_k - H_k\theta - m_k)^T R_k^{-1}(Z_k - H_k\theta - m_k) \right) = 0,$$

où $m_k = (0, 0)$ – la moyenne du bruit (le bruit est centré).
L'objectif de l'estimation du maximum de vraisemblance revient à la résolution d'un problème linéaire :

$$\hat{\theta_N} = (H_{1:N}^T R_{1:N}^{-1} H_{1:N})^{-1} H_{1:N}^T R_{1:N}^{-1} (Z_{1:N} - m_{1:N})$$

On utilise le Théorème de Gauss-Markov parce que toutes les mesures sont disponibles au moment de l'estimation, et la matrice $H_{1:N}^T R_{1:N}^{-1} H_{1:N}$ est inversible, donc on résout directement l'équation normale par les méthodes d'inversions classiques.
Selon le théorème, l'estimateur trouvé est un estimateur linéaire sans biais à minimum de variance, et consistant, et c'est le seul.

1.3

Objectif : Déterminer l'estimé au sens du maximum de vraisemblance de valeurs initiales (position, vitesse, accélération) de la cible. Calculer la matrice de covariance résultante et la comparer avec l'inverse de la matrice d'information de Fisher correspondant que l'on calculera (on rappelera l'expression de cette matrice dans le cas considéré)

Développement :
On trouve l'estimé θ (la solution en matlab se trouve dans l'annexe) :

$$\theta = \begin{pmatrix} x_0 \\ v_{0x} \\ a_{0x} \\ y_0 \\ v_{0y} \\ a_{0y} \end{pmatrix} = \begin{pmatrix} 21.3855 \\ 69.5969 \\ 33.2930 \\ 22.3669 \\ 148.1006 \\ 18.6696 \end{pmatrix}$$

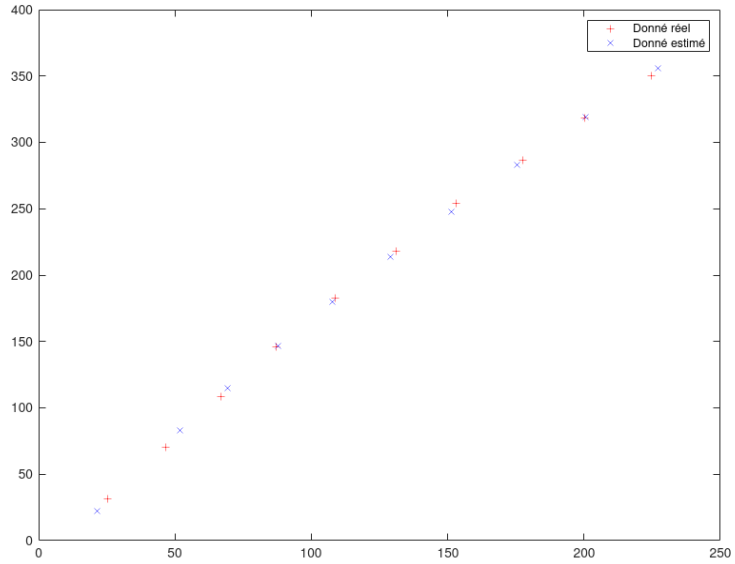


Figure 1: Donnés réels et estimés

La formule de l'estimateur de Gauss-Markov est :

$$\hat{\theta}^N = \left(H_{1:N}^T R_{1:N}^{-1} H_{1:N}\right)^{-1} H_{1:N}^T R_{1:N}^{-1} \left(Z_{1:N} - m_{1:N}\right)$$

La matrice de covariance de cet estimateur donne donc :

$$\mathrm{cov}(\hat{\theta}^N) = \left(H_{1:N}^T R_{1:N}^{-1} H_{1:N}\right)^{-1}$$

La log-vraisemblance s'écrit :

$$\begin{aligned}
\ln L(\theta; Z_1, \ldots, Z_N) &= \ln f(Z_1, \ldots, Z_N; \theta) \\
&= \sum_{k=1}^{N} \ln f_k(Z_k; \theta) \\
&= \sum_{k=1}^{N} \left(- \ln \left(\sqrt{(2\pi)^{n_z} \det(R_k)} \right) - \frac{1}{2} \left(Z_k - H_k\theta - m_k\right)^T R_k^{-1} \left(Z_k - H_k\theta - m_k\right) \right)
\end{aligned} \tag{1}$$

La formule de la matrice d'information de fisher est :

$$M_f(\theta) = \mathrm{Cov}\left(\frac{\partial \ln L_n(\theta; X_1, \ldots, X_n)}{\partial \theta} \right) = \mathbb{E}\left[\frac{\partial \ln L_n(\theta; X_1, \ldots, X_n)}{\partial \theta} \frac{\partial \ln L_n(\theta; X_1, \ldots, X_n)}{\partial \theta}^T \right]$$

Ainsi, on obtient :

$$M_f(\theta) = -\mathbb{E}\left[\frac{\partial^2}{\partial \theta^2} \sum_{k=1}^{N} (Z_k - H_k\theta - m_k)^T R_k^{-1} (Z_k - H_k\theta - m_k) \right]$$

Donc,

$$M_f(\theta) = -\mathbb{E}\left[\frac{\partial}{\partial \theta} \sum_{k=1}^{N} (H_k^T R_k^{-1} H_k)\theta - \sum_{k=1}^{N} H_k^T R_k^{-1} (Z_k - m_k) \right]$$

$$M_f(\theta) = H_k^T R_k^{-1} H_k$$

On constate donc que :

$$\mathrm{Cov}(\hat{\theta}^n) = (M_f(\theta))^{-1}$$

On peut donc conclure que c'est un estimateur efficace.

1.4

Objectif : On reprend les hypothèses identiques mais en utilisant le fichier xx2.mat. Calculer l'estimé au sens du maximum de vraisemblance obtenu pour ce jeu de données. Que constate-t-on ? Pour améliorer les résultats, on considère qu'on dispose d'une information a priori sur les valeurs initiales qui est qu'elles sont des variables gaussiennes de moyenne $x_0, y_0, v_{0x}, v_{0y}, a_{0x}, a_{0y}$, de variances a priori $\sigma_x, \sigma_y, \sigma_v, \sigma_a$. Ecrire l'expression de l'estimé au sens du maximum a posteriori de ces valeurs. Comparer les résultats obtenus avec cette nouvelle hypothèse en termes de covariance d'estimation pour différentes valeurs des moyennes et des variances a priori. Expliquer les différences résultantes.

Développement : On trouve l'estimé θ en utilisant le fichier xx2.mat (la solution en matlab se trouve dans l'annexe) :

$$\theta = \begin{pmatrix} x_0 \\ v_{0x} \\ a_{0x} \\ y_0 \\ v_{0y} \\ a_{0y} \end{pmatrix} = \begin{pmatrix} 32.6763 \\ 93.0111 \\ 11.4821 \\ -107.7054 \\ 223.2034 \\ -26.6909 \end{pmatrix}$$

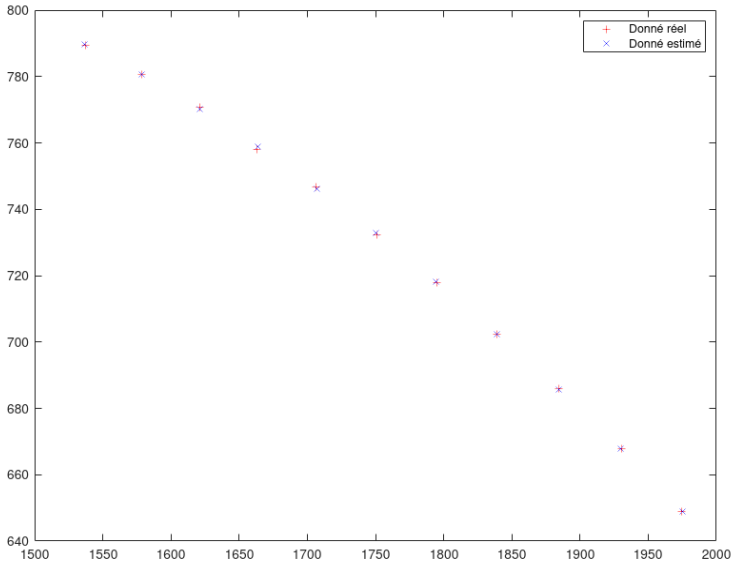


Figure 2: Donnés réels et estimés

L'estimé trouvé à l'aide du fichier xx1.mat est different de celui obtenu avec les donnés du fichier xx2.mat(l'estimation est plus précise dans ce cas). On constate donc que les valeurs de θ dépendent totalement des valeurs des observations. En conclusion, Il nous faut plus d'informations pour estimer θ de manière plus précise.
On considère qu'on dispose d'une information a priori sur les valeurs initiales qui est qu'elles sont des variables gaussiennes de moyenne $x_0, y_0, v_{0x}, v_{0y}, a_{0x}, a_{0y}$, de variances a priori x, y, v, a.
Trouvons l'expression de l'estimé au sens du maximum a posteriori de ces valeurs.
Nous supposons que θ est un vecteur aléatoire de loi a priori $\pi(\theta)$ normale :

$$\theta \sim \mathcal{N}(m_\theta, R_\theta).$$

De plus, pour tout $k \in 1, \ldots, N$, la loi conditionnelle de Z_k sachant θ, est donnée par :

$$Z_{k|\theta} \in \mathcal{N}(H_k\theta + m_k, R_k)$$

La log-vraisemblance A Posteriori s'écrit :

$$\begin{aligned}
\ln f(\theta|Z_1, \ldots, Z_N) &= \ln\left[\frac{f(Z_1, \ldots, Z_N|\theta)\pi(\theta)}{f(Z_1, \ldots, Z_N)}\right] \\
&= \sum_{k=1}^{N} \ln f_k(Z_k;\theta) + \ln \pi(\theta) - \ln f(Z_1, \ldots, Z_N) \\
&= \sum_{k=1}^{N} -\ln\left(\sqrt{(2\pi)^{n_z}\det(R_k)}\right) \\
&- \frac{1}{2}(Z_k - H_k\theta - m_k)^T R_k^{-1}(Z_k - H_k\theta - m_k) \\
&- \ln\left(\sqrt{(2\pi)^{n_z}\det(R_k)}\right) - \frac{1}{2}(\theta - m_\theta)^T R_\theta^{-1}(\theta - m_\theta) \\
&- \ln f(Z_1, \ldots, Z_N)
\end{aligned}$$

La recherche du maximum amène à considérer l'égalité suivante :

$$\frac{\partial}{\partial\theta} L\left(\theta; Z_1, \ldots, Z_N\right) = 0.$$

ce qui implique :

$$\sum_{k=1}^{N} \frac{\partial}{\partial\theta}\left((Z_k - H_k\theta - m_k)^T R_k^{-1}(Z_k - H_k\theta - m_k) + (\theta - m_k)^T R_k^{-1}(\theta - m_\theta)\right) = 0.$$

Ainsi, l'équation donne :

$$\left(R_\theta^{-1} + \sum_{k=1}^{N}(H_k^T R_k^{-1} H_k)\right)\hat{\theta}_N = R_\theta^{-1} m_\theta + \sum_{k=1}^{N} H_k^T R_k^{-1}(Z_k - m_k)$$

ou, sous une forme plus condensée, nous obtenons l'équation normale associée au problème d'estimation linéaire gaussien :

$$\left(R_\theta^{-1} + H_{1:N}^T R_{1:N}^{-1} H_{1:N}\right)\hat{\theta}_N = R_\theta^{-1} m_\theta + H_{1:N}^T R_{1:N}^{-1}(Z_{1:N} - m_{1:N})$$

avec $Z_{1:N} = \begin{bmatrix} Z_1 \\ \vdots \\ Z_N \end{bmatrix}, H_{1:N} = \begin{bmatrix} H_1 \\ \vdots \\ H_N \end{bmatrix}, m_{1:N} = \begin{bmatrix} m_1 \\ \vdots \\ m_N \end{bmatrix}, R_{1:N} = \begin{bmatrix} R_1 & 0 & \ldots & 0 \\ 0 & R_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & R_N \end{bmatrix},$

$H_k = \begin{pmatrix} 1 & t_k & t_k^2/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & t_k & t_k^2/2 \end{pmatrix}.$

Comme la matrice normale est inversible, on résout l'équation directement :

$$\hat{\theta}_N = \left(R_\theta^{-1} + H_{1:N}^T R_{1:N}^{-1} H_{1:N}\right)^{-1}\left(R_\theta^{-1} m_\theta + H_{1:N}^T R_{1:N}^{-1}(Z_{1:N} - m_{1:N})\right)$$

L'implementation de l'algorithme en matlab se trouve dans l'annexe.

On a les résultats suivants:
Avec $\sigma_x^2 = \sigma_y^2 = \sigma_v^2 = \sigma_a^2 = 2$ et $m_\theta = \hat{\theta}_N$ (on prend pour la valeur moyenne de θ l'estimé trouvé dans l'exercice précedent) :

$$\theta = \begin{pmatrix} x_0 \\ v_{0x} \\ a_{0x} \\ y_0 \\ v_{0y} \\ a_{0y} \end{pmatrix} = \begin{pmatrix} 21.2926 \\ 69.7170 \\ 33.2215 \\ 22.3669 \\ 148.1006 \\ 18.6696 \end{pmatrix}$$

Avec $\sigma_x^2 = \sigma_y^2 = \sigma_v^2 = \sigma_a^2 = 10$ et $m_\theta = \text{Int}\left(\hat{\theta}_N\right)$ (la valeur moyenne est la partie entière de l'estimé trouvé dans l'exercice précedent) :

$$\theta = \begin{pmatrix} x_0 \\ v_{0x} \\ a_{0x} \\ y_0 \\ v_{0y} \\ a_{0y} \end{pmatrix} = \begin{pmatrix} 21.1874 \\ 69.9595 \\ 33.0100 \\ 22.3686 \\ 148.0288 \\ 18.7587 \end{pmatrix}$$

Estimateur du Maximum A Posteriori

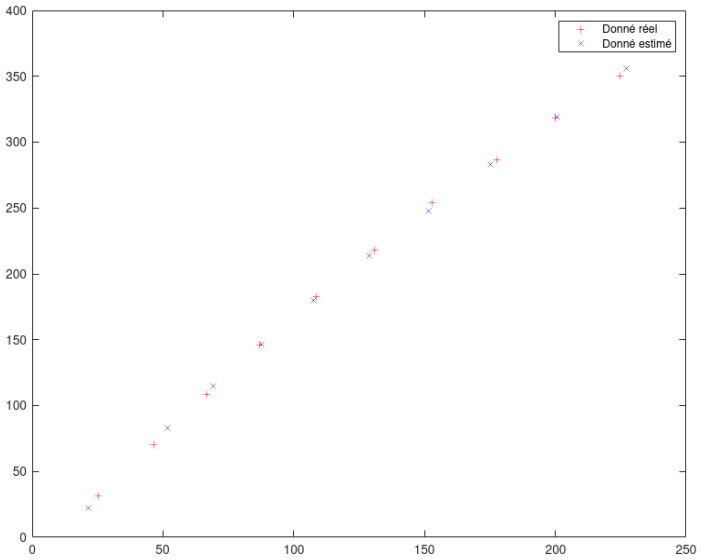


Figure 3: Donnés réels et estimés. $\sigma_x^2 = \sigma_y^2 = \sigma_v^2 = \sigma_a^2 = 2$, $m_\theta = \hat{\theta}_N$

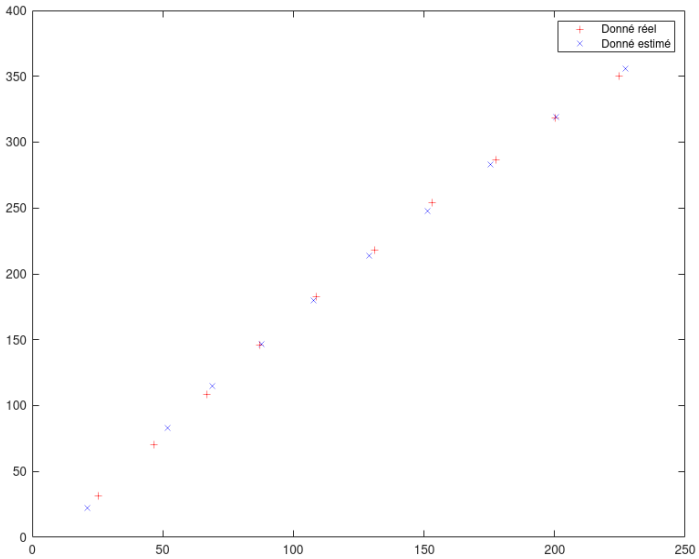


Figure 4: Donnés réels et estimés. $\sigma_x^2 = \sigma_y^2 = \sigma_v^2 = \sigma_a^2 = 10$, $m_\theta = \text{Int}\left(\hat{\theta}_N\right)$

On constate que dans tous les cas les valeurs de l'estimé trouvé sont très proches. En comparant l'erreur quadratique (on la calcule pour chaque mesure comme $(x - x_\theta)^2 + (y - y_\theta)^2$, où (x, y) – vecteur de mesures, (x_θ, y_θ) – vecteur de la position estimé), on peut dire que l'estimateur du maximum a posteriori donne un meilleur résultat, mais l'erreur reste toujours assez importante.

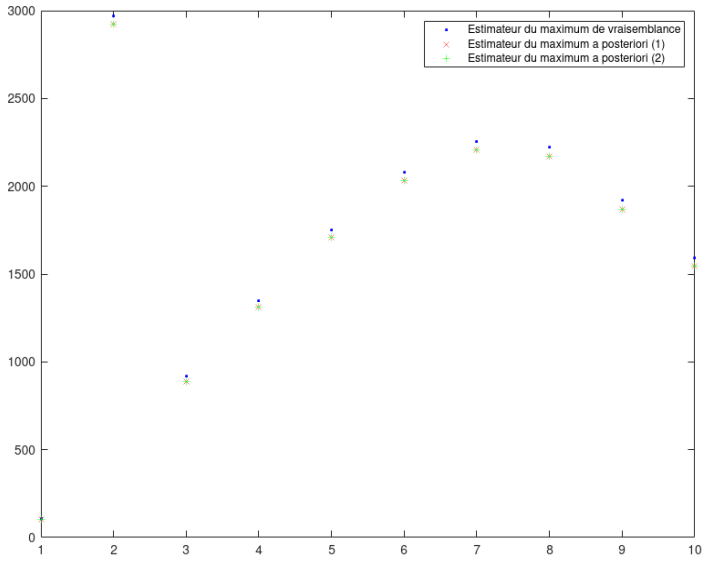


Figure 5: Erreur quadratique. Estimateurs du maximum a posteriori : (1) $\sigma_x^2 = \sigma_y^2 = \sigma_v^2 = \sigma_a^2 = 2$, $m_\theta = \hat{\theta}_N$, (2) $\sigma_x^2 = \sigma_y^2 = \sigma_v^2 = \sigma_a^2 = 10$, $m_\theta = \text{Int}\left(\hat{\theta}_N\right)$

2 Exercice 2

Objectif : Un dispositif de poursuite (radar, sonar etc...), dont la position est fixe et connue, mesure à des instants discrets $t = kT$, T période de mesure connue également, la distance qui le sépare de la cible ainsi que l'angle (site) que fait la direction de la droite cible - radar

avec l'horizontale locale. Ces mesures, notées par la suite D_k et α_k, sont entachées d'erreurs additives modélisées par des bruits blancs gaussiens centrés $n_D(k)$ et $n_s(k)$ de variances $\sigma_D^2(k)$ et $\sigma_s^2(k)$ à l'instant kT. Ces bruits sont supposés indépendants.
Le problème est d'estimer à chaque instant la position $(x(k), y(k))$ de la cible à l'aide de ces mesures. On se place dans un repère (O, x, y) du plan, le dispositif de poursuite est fixe en $(x_0, y_0) = (0, 0)$ et la cible se déplace dans le plan x, y. Sa position dans le plan est donc entièrement définie à chaque instant par le couple $x(k), y(k)$. Nous allons dans un premier temps établir les équations dynamiques de l'état de la cible qui seront utilisées dans les filtres. On considère qu'entre deux temps de mesure consécutifs d'une durée ΔT, l'accélération de la cible peut être considérée comme constante. Sous cette hypothèse, l'évolution des positions, vitesses et accélérations suivant x et y, s'écrit :

$$\begin{aligned}
x(k+1) &= x(k) + v_x(k)\Delta T + a_x(k)\frac{\Delta T^2}{2} \\
y(k+1) &= y(k) + v_y(k)\Delta T + a_y(k)\frac{\Delta T^2}{2} \\
v_x(k+1) &= v_x(k) + a_x(k)\Delta T \\
v_y(k+1) &= v_y(k) + a_y(k)\Delta T \\
a_x(k+1) &= a_x(k) \\
a_y(k+1) &= a_y(k)
\end{aligned}$$

Pour permettre de tenir compte d'une évolution de l'accélération de la cible $a(k) = (ax(k), ay(k))$, on considère que l'évolution s'exprime sous la forme :

$$\begin{aligned}
a_x(k+1) &= a_x(k) + w_x(k) \\
a_y(k+1) &= a_y(k) + w_y(k)
\end{aligned}$$

Les deux composantes $w_x(k)$ suivant x et $w_y(k)$ suivant y sont supposées être des bruits gaussiens indépendants de moyenne nulle et de même variance σ_a^2.

2.1

Objectif : Déduire l'équation de la dynamique de l'état défini par

$$X(k) = [x(k), y(k), v_x(k), v_y(k), a_x(k), a_y(k)]^T$$

sous la forme : $X(k+1) = F_k X(k) + G_k w(k)$. On explicitera les matrices F_k, G_k et les caractéristiques du bruit d'état $w(k)$.

Développement :

$$X(k+1) = F_k X(k) + G_k w(k)$$

$$\begin{pmatrix} x(k+1) \\ y(k+1) \\ v_x(k+1) \\ v_y(k+1) \\ a_x(k+1) \\ a_y(k+1) \end{pmatrix} = \begin{pmatrix} 1 & 0 & T & 0 & T^2/2 & 0 \\ 0 & 1 & 0 & T & 0 & T^2/2 \\ 0 & 0 & 1 & 0 & T & 0 \\ 0 & 0 & 0 & 1 & 0 & T \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x(k) \\ y(k) \\ v_x(k) \\ v_y(k) \\ a_x(k) \\ a_y(k) \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} \omega(k),$$

$$\omega(k) \in \mathcal{N}(0, W), W = \begin{pmatrix} \sigma_a^2 & 0 \\ 0 & \sigma_a^2 \end{pmatrix}$$

2.2

Objectif : Le vecteur de mesures (D_k, α_k) à l'instant t, distance entre la cible et le radar et angle entre la droite cible-radar et l'horizontale, peut s'exprimer en fonction des positions $(x(t), y(t))$ et (x_0, y_0) de la cible et du dispositif d'observation. Écrire l'équation d'observation associée.

Développement :

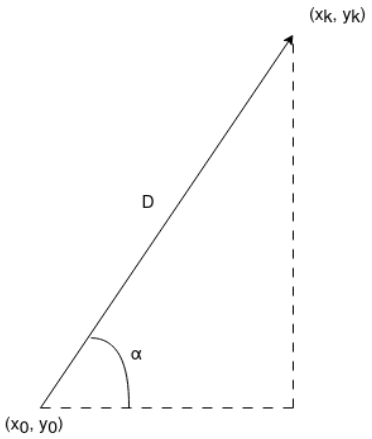


Figure 6: Dépendence entre les coordonnés, la distance cible-radar et l'angle

L'équation d'observation s'écrit :

$$Z(k) = h_k(X(k)) + v(k)$$

avec

$$Z(k) = (D_k, \alpha_k), X(k) = (x(k), y(k), v_x(k), v_y(k), \alpha_x(k), \alpha_y(k))^T,$$
$$h(X(k)) = \begin{pmatrix} \sqrt{(x(k) - x_0)^2 + (y(k) - y_0)^2} \\ \arctan(y(k)/x(k)) \end{pmatrix},$$
$$v(k) \sim \mathcal{N}(0, V(k)), V(k) = \begin{pmatrix} \sigma_D^2(k) & 0 \\ 0 & \sigma_s^2(k) \end{pmatrix}$$

2.3

Objectif : Construction des pseudos mesures : On considère ici une transformation des mesures permettant de les transformer sous une forme plus proche de l'état que l'on cherche à estimer. Soient $\tilde{x}(k) = D_k \cos \alpha_k$ et $\tilde{y}(k) = D_k \sin \alpha_k$ où D_k et α_k sont les mesures bruitées à l'instant k. Montrer que les hypothèses de bruit gaussien sur les mesures D_k et α_k ne permettent pas d'assurer que les pseudo-mesures peuvent être modélisées sous la forme d'une somme d'une fonction de l'état et de bruits gaussiens.

Développement : L'équation d'observation s'écrit :

$$Z(k) = HX(k) + v(k)$$

avec

$$Z(k) = \begin{pmatrix} \tilde{x}(k) \\ \tilde{y}(k) \end{pmatrix} = \begin{pmatrix} D_k \cos \alpha_k \\ D_k \sin \alpha_k \end{pmatrix},$$

$$X(k) = (x(k), y(k), v_x(k), v_y(k), a_x(k), a_y(k))^T,$$
$$H = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix},$$
$$v(k) \sim \mathcal{N}(0, V(k)), V(k) = \begin{pmatrix} \sigma_x^2(k) & 0 \\ 0 & \sigma_y^2(k) \end{pmatrix}$$

Considérant la cible et le radar comme deux points sur un plan cartésien (la figure de l'exercice précedent), la distance entre eux (D_k) et l'angle entre la droite cible-radar et l'horizontale (α_k) s'expriment comme :

$$D_k = \sqrt{(x_k - x_0)^2 + (y_k - y_0)^2} + n_D(k)$$
$$\alpha_k = \arctan\left(\frac{y_k - y_0}{x_k - x_0}\right) + n_s(k)$$

Si D_k et α_k sont bruités, alors à partir de ces équations on peut écrire :

$$D_k = \sqrt{x_k^2 + y_k^2} + B_d, \quad \alpha_k = \arctan\left(\frac{y_k}{x_k}\right) + B_\alpha$$

où B_d et B_α sont les bruits associés aux variables D_k et α_k, respectivement. Ensuite, on a:

$$\tilde{x} = D_k \cos(\alpha_k), \quad \tilde{y} = D_k \sin(\alpha_k)$$

Si on développe, on obtient:

$$\tilde{x} = x_k \cos(B_\alpha) - y_k \sin(B_\alpha) + \frac{B_d}{\sqrt{x_k^2 + y_k^2}} (x_k \cos(B_\alpha) - y_k \sin(B_\alpha))$$
$$\tilde{y} = y_k \cos(B_\alpha) + x_k \sin(B_\alpha) + \frac{B_d}{\sqrt{x_k^2 + y_k^2}} (y_k \cos(B_\alpha) + x_k \sin(B_\alpha))$$

Enfin, on peut voir que les équations représentent un système non linéaire qui dépend des bruits de distance et du sinus ou cosinus du bruit de l'angle, donc les pseudo mesures ne peuvent pas être modélisées sous la forme d'une somme d'une fonction et d'un bruit gaussien.

3 Exercise 3

3.1

Objectif : Ecrire les équations du filtre de Kalman simple correspondant aux équations dynamiques de l'état et l'équation d'observation correspondant aux pseudomesures.

Développement : L'équation de la dynamique :

$$X(k+1) = F_k X(k) + G_k w(k)$$

si l'on développe, on obtient :

$$\begin{pmatrix} x(k+1) \\ y(k+1) \\ v_x(k+1) \\ v_y(k+1) \\ a_x(k+1) \\ a_y(k+1) \end{pmatrix} = \begin{pmatrix} 1 & 0 & T & 0 & T^2/2 & 0 \\ 0 & 1 & 0 & T & 0 & T^2/2 \\ 0 & 0 & 1 & 0 & T & 0 \\ 0 & 0 & 0 & 1 & 0 & T \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x(k) \\ y(k) \\ v_x(k) \\ v_y(k) \\ a_x(k) \\ a_y(k) \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} \omega(k),$$

$$\omega(k) \in \mathcal{N}(0, W), W = \begin{pmatrix} \sigma_a^2 & 0 \\ 0 & \sigma_a^2 \end{pmatrix}$$

L'équation de mesures :

$$Z(k) = HX(k) + v(k)$$

ce qui donne :

$$Z(k) = \begin{pmatrix} \tilde{x}(k) \\ \tilde{y}(k) \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} x(k) \\ y(k) \\ v_x(k) \\ v_y(k) \\ a_x(k) \\ a_y(k) \end{pmatrix} + v(k),$$

$$v(k) \in \mathcal{N}(0, V(k))$$

On trouvera la matrice de covariance $V(k)$ dans l'exercice suivant.

3.2

Objectif : Suggérer un moyen de déterminer les matrices de covariances associées aux pseudo-mesures

Développement :
La transformation des mesures en pseudo-mesures est non-lineaire et peut être écrite sous la forme :

$$\tilde{x}(k) = D_k \cos(\alpha_k),$$
$$\tilde{y}(k) = D_k \sin(\alpha_k)$$

Pour trouver la matrice de covariance des pseudo-mesures, on utilise la matrice jacobienne :

$$J_k = \begin{pmatrix} \frac{\partial \tilde{x}}{\partial D_k} & \frac{\partial \tilde{x}}{\partial \alpha_k} \\ \frac{\partial \tilde{y}}{\partial D_k} & \frac{\partial \tilde{y}}{\partial \alpha_k} \end{pmatrix} = \begin{pmatrix} \cos(\alpha_k) & -D_k \sin(\alpha_k) \\ \sin(\alpha_k) & D_k \cos(\alpha_k) \end{pmatrix}$$

Ainsi, la covariance des pseudo-mesures s'écrit comme :

$$V(k) = J_k \begin{pmatrix} \sigma_D^2(k) & 0 \\ 0 & \sigma_\alpha^2(k) \end{pmatrix} J_k^T$$

3.3

Objectif : Implémenter l'algorithme résultant sous matlab.

Développement :
Le code est dans l'annexe.
Rappel : on initialize $P = \lambda I$ avec λ arbitrairement grand. On obtient des résultats differents en changeant les paramètres λ et σ_a^2 (la variance de l'accelera-tion). On considère les variances de l'angle σ_α^2 et de la distance σ_D^2 constantes et $\sigma_\alpha^2 = 0.01$, $\sigma_D^2 = 100$ (selon le fichier mesurestrajKalm2.mat). On trouve les résultats suivants.

Filtre de Kalman simple

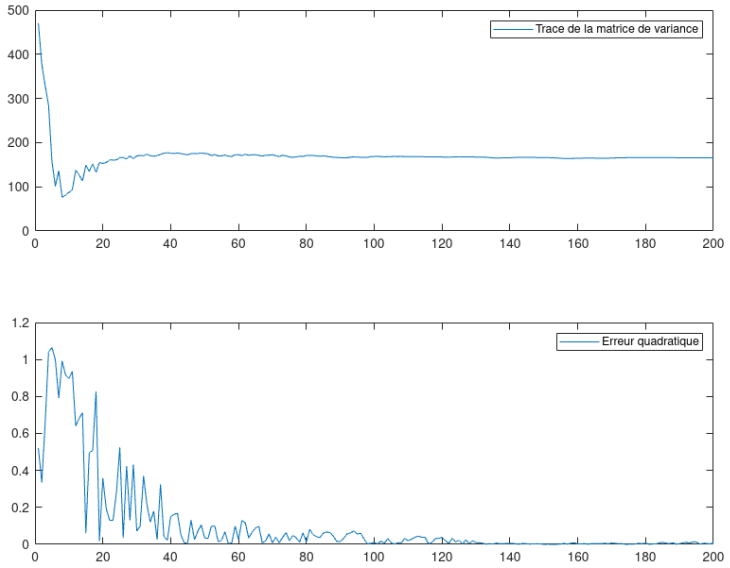


Figure 7: $\sigma_a^2 = 10$, $\lambda = 100$

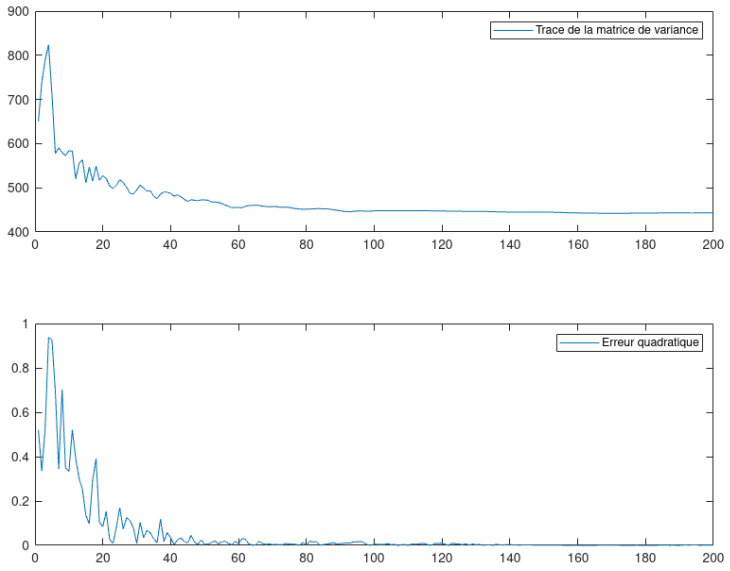


Figure 8: $\sigma_a^2 = 100$, $\lambda = 100$

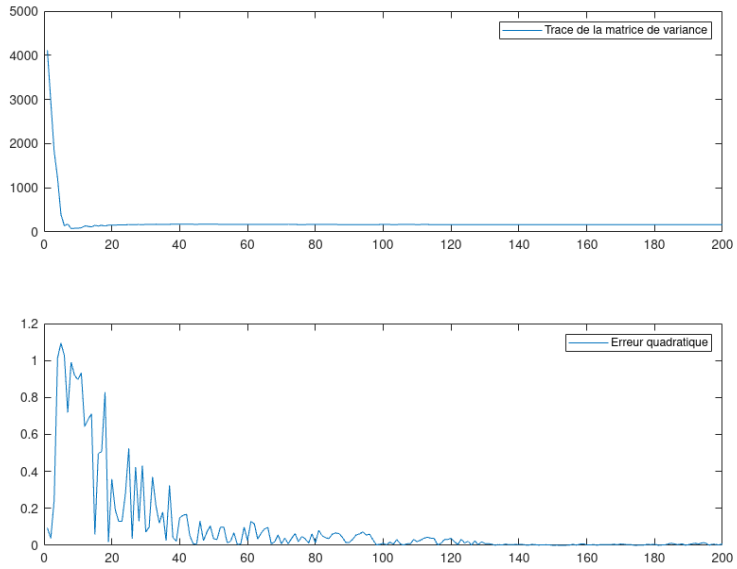


Figure 9: $\sigma_a^2 = 10$, $\lambda = 1000$

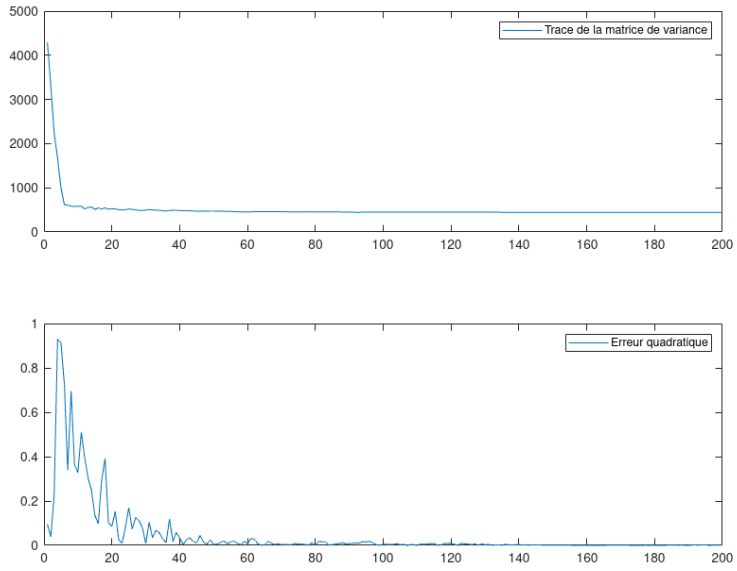


Figure 10: $\sigma_a^2 = 100$, $\lambda = 1000$

On constate que l'erreur quadratique diminue plus rapidement quand la variance de l'acceleration est plus grande. Par contre, dans ce cas la trace de la matrice de variance devient presque constante et ne tends pas vers zéro après un certain nombre d'iterations.

4 Exercice 4

4.1

Objectif : Écrire l'équation d'observation linéarisée autour de $\hat{X}(k|k-1)$ sous la forme $z_k = H_k X(k) + b(k)$ en exprimant H_k et $b(k)$ à l'aide de $\hat{X}(k|k-1)$, des bruits de mesure $n_D(k)$ et $n_s i(k)$, $b(k)$ est-il centré ?

Développement :
On doit lineairiser z autour de $x_{k|k-1}$, $y_{k|k-1}$. En utilisant l'équation de la tangente $y_0 = f'(x_0)(x - x_0) + f(x_0)$ on peut écrire :

$$z_{k|k-1} = h'(X_{k|k-1})(X - X_{k|k-1}) + h(X_{k|k-1}),$$

où

$$h : \begin{pmatrix} x \\ y \end{pmatrix} \to \begin{pmatrix} \sqrt{x^2+y^2} \\ \arctan(y/x) \end{pmatrix} = \begin{pmatrix} D(x,y) \\ \alpha(x,y) \end{pmatrix}$$

La jacobienne de h:

$$J(h) = \begin{pmatrix} \frac{\partial D}{\partial x} & \frac{\partial D}{\partial y} \\ \frac{\partial \alpha}{\partial x} & \frac{\partial \alpha}{\partial y} \end{pmatrix} = \begin{pmatrix} \frac{x}{\sqrt{x^2+y^2}} & \frac{y}{\sqrt{x^2+y^2}} \\ \frac{-y}{x^2+y^2} & \frac{x}{x^2+y^2} \end{pmatrix}$$

Cela nous donne l'équation suivante (on note $x_{k|k-1} = x_0$ et $y_{k|k-1} = y_0$) :

$$z_0 = \begin{pmatrix} \frac{x_0}{\sqrt{x_0^2+y_0^2}} & \frac{y_0}{\sqrt{x_0^2+y_0^2}} \\ \frac{-y_0}{x_0^2+y_0^2} & \frac{x}{x_0^2+y_0^2} \end{pmatrix} \left[\begin{pmatrix} x \\ y \end{pmatrix} - \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} \right] + \begin{pmatrix} \sqrt{x_0^2+y_0^2} \\ \arctan(y_0/x_0) \end{pmatrix} + \begin{pmatrix} n_D(k) \\ n_{si}(k) \end{pmatrix}$$

On voit que le bruit n'est pas centré, car sa moyenne est non-nulle :

$$b(k) \sim \mathcal{N}(\mu(k), V(k)), V(k) = \begin{pmatrix} \sigma_D^2(k) & 0 \\ 0 & \sigma_{si}^2(k) \end{pmatrix},$$

$$\mu(k) = -\begin{pmatrix} \frac{x_0}{\sqrt{x_0^2+y_0^2}} & \frac{y_0}{\sqrt{x_0^2+y_0^2}} \\ \frac{-y_0}{x_0^2+y_0^2} & \frac{x}{x_0^2+y_0^2} \end{pmatrix} \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} + \begin{pmatrix} \sqrt{x_0^2+y_0^2} \\ \arctan(y_0/x_0) \end{pmatrix}$$

4.2

Objectif : Trouver le moyen de modifier les équations du filtre de Kalman pour tenir compte d'un bruit de mesure non centré.

Développement :
Dans l'exercice précedent on a trouvé que :

$$z_k = H_k X - H_k X_{k|k-1} + \begin{pmatrix} \sqrt{x_{k|k-1}^2 + y_{k|k-1}^2} \\ \arctan(y_{k|k-1}/x_{k|k-1}) \end{pmatrix} + \begin{pmatrix} n_D(k) \\ n_{si}(k) \end{pmatrix}$$

Notons

$$M_k = -H_k X_{k|k-1} + \begin{pmatrix} \sqrt{x_{k|k-1}^2 + y_{k|k-1}^2} \\ \arctan(y_{k|k-1}/x_{k|k-1}) \end{pmatrix}$$

On peut donc écrire :

$$z_k = \begin{pmatrix} H_k & I_2 \end{pmatrix} \begin{pmatrix} X \\ M_k \end{pmatrix} + \begin{pmatrix} n_D(k) \\ n_{si}(k) \end{pmatrix},$$

ou sous une forme plus condensée :

$$z_k = \tilde{H}_k \tilde{X} + \begin{pmatrix} n_D(k) \\ n_{si}(k) \end{pmatrix},$$

où $\tilde{H}_k = \begin{pmatrix} H_k & I_2 \end{pmatrix}$, $\tilde{X} = \begin{pmatrix} X \\ M_k \end{pmatrix}$ – un nouveaux vecteur de mesures, $\begin{pmatrix} n_D(k) \\ n_{si}(k) \end{pmatrix}$ – un bruit centré.

4.3

Objectif : En déduire les équations du filtre de Kalman correspondant à ce modèle linéarisé, en faisant apparaître les paramètres nécessaires à l'initialisation et au fonctionnement du filtre d'une récurrence à l'autre.

Développement :
Le filtre de Kalman contient toujours deux étapes. Cependent, il faut tenir compte que le nouveaux vecteur de mesures a été modifié pour que l'on puisse traiter un bruit non-centré : $\tilde{X} = \begin{pmatrix} X \\ M_k \end{pmatrix}$. On réécrit donc les équations de Kalman en ajoutant deux paramètres au vecteur de mesures.

Prédiction :

$$x_{k|k-1} = F_k x_k + B_k$$
$$P_{k|k-1} = F_k P_k F_k^T + W_k$$

où

$$B_k = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ M_k \end{pmatrix}$$

$$M_k = -H_k X_{k|k-1} + \begin{pmatrix} \sqrt{x_{k|k-1}^2 + y_{k|k-1}^2} \\ \arctan(y_{k|k-1}/x_{k|k-1}) \end{pmatrix}$$

$$F_k = \begin{pmatrix} 1 & 0 & T & 0 & T^2/2 & 0 & 0 & 0 \\ 0 & 1 & 0 & T & 0 & T^2/2 & 0 & 0 \\ 0 & 0 & 1 & 0 & T & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & T & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$W_k = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_a^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_a^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Correction – Calcul du gain de Kalman :

$$K_k = P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T + V_k)^{-1}$$

où

$$V_k = \begin{pmatrix} \sigma_D^2 & 0 \\ 0 & \sigma_\alpha^2 \end{pmatrix}$$

Correction – Mise à jour :

$$x_k = x_{k|k-1} + K_k(z_k - H_k x_{k|k-1})$$

$$P_k = (I - K_k H_k)\, P_{k|k-1}$$

où

$$H_k = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

4.4

Objectif :
En pratique, il est très courant d'initialiser un filtre de Kalman avec $x_0 = 0$ et $P_0 = \lambda I$, avec λ grand de manière à traduire notre incertitude sur cet état initial. On démarre le filtre et au bout de quelques récurrences, grâce aux observations, la covariance diminue considérablement traduisant la plus grande confiance que l'on peut accorder à l'estimation de l'état. Que se passe-t-il dans le cas du filtre étendu considéré ? Expliquer ce phénomène.

Développement :
On obtient des résultats differents en changeant les paramètres λ et σ_a^2 (la variance de l'acceleration). On considère les variances de l'angle σ_α^2 et de la distance σ_D^2 constantes et $\sigma_\alpha^2 = 0.01$, $\sigma_D^2 = 100$ (selon le fichier mesurestrajKalm2.mat). On trouve les résultats suivants.
Le vecteur x_0 a été initialisé avec la première paire de données. Pour $x_0 = 0$ le filtre ne converge pas, ce qui peut être lié à la linéarisation de la matrice H. Par contre, si l'on initialise le filtre avec les donnés réels, il converge plus rapidement que le filtre de Kalman simple avec les mêmes paramètres.

Filtre de Kalman étendu

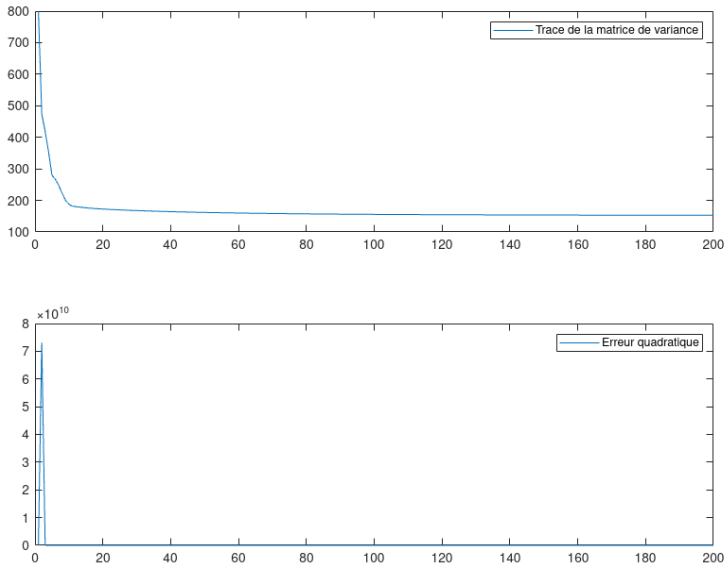


Figure 11: $\sigma_a^2 = 10$, $\lambda = 100$

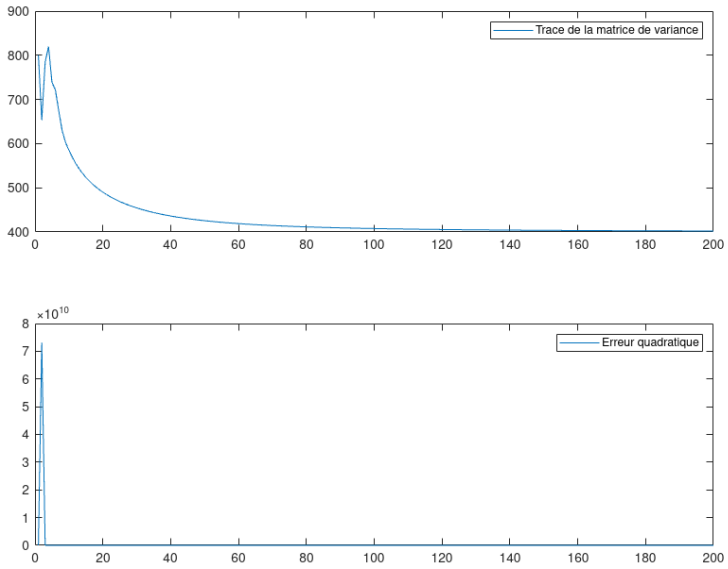


Figure 12: $\sigma_a^2 = 100$, $\lambda = 100$

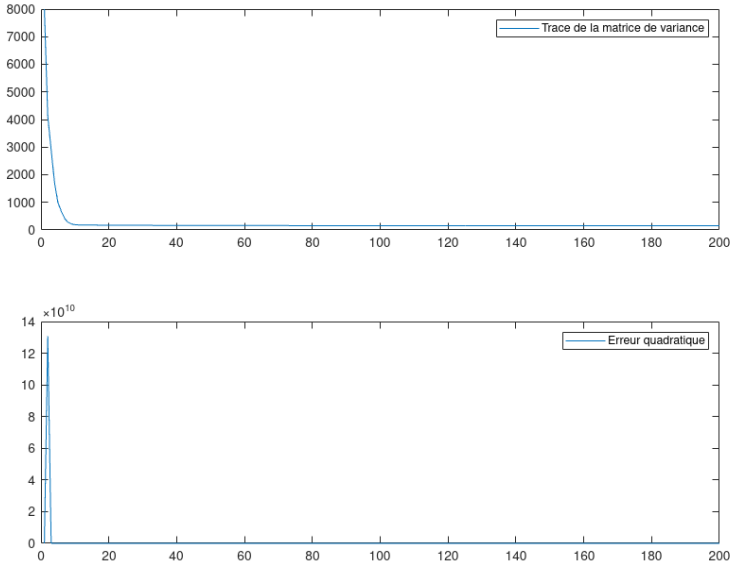


Figure 13: $\sigma_a^2 = 10$, $\lambda = 1000$

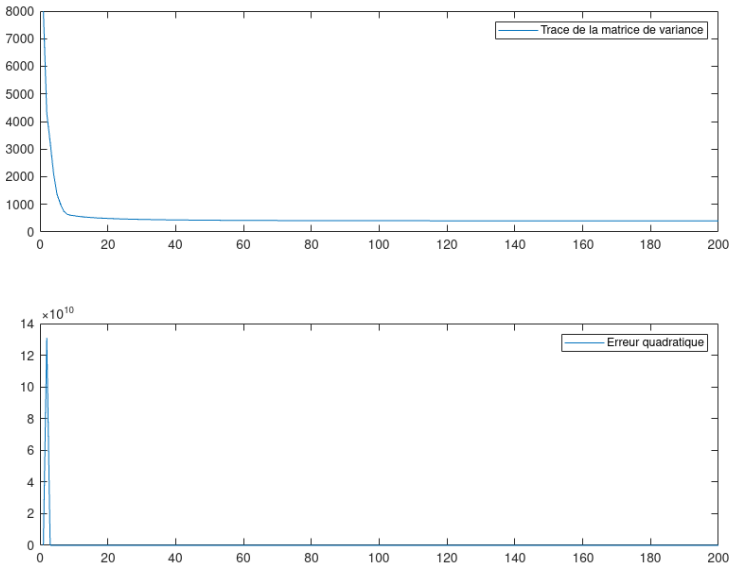


Figure 14: $\sigma_a^2 = 100$, $\lambda = 1000$

4.5

Objectif : Implémenter l'algorithme résultant sous matlab.

Développement : Le code se trouve dans l'annexe. Les résultats sont expliqués dans l'exercice précedent.

5 Exercice 5

5.1

Objectif : Rappeler l'expression de la prédiction d'observation $y(k+1/k)$ et de la matrice de covariance $S_{k+1/k}$ associée.

Développement :

Prédiction : $\hat{y}_{k+1|k} = F_k \hat{y}_k, \quad S_{k+1|k} = F_k P_k F_k^T + G_k u_k$

Correction : Calcul du gain de Kalman : $K_{k+1} = S_{k+1|k} H_{k+1}^T (H_{k+1} S_{k+1|k} H_{k+1}^T + b_{k+1})^{-1}$

Mise à jour : $\hat{y}_{k+1} = \hat{y}_{k+1|k} + K_{k+1}(Z_{k+1} - H_{k+1}\hat{y}_{k+1|k}), \quad S_{k+1} = (I - K_{k+1}H_{k+1})S_{k+1|k}$

5.2

Objectif : Sous l'hypothèse que la mesure issue de la cible à l'instant $k+1$, conditionnée par $y(k)$, admet une probabilité gaussienne, on définit un ellipsoïde V_{k+1} par

$$V_{k+1} = \left\{ y \,\middle|\, [y - \hat{y}_{k+1|k}]^T (S_{k+1|k})^{-1} [y - \hat{y}_{k+1|k}] \leq 0 \right\}.$$

Donner une interprétation de cet ellipsoïde (ou domaine de validation).

Développement :

L'éllipsoïde V_{k+1} défini comme suit :

$$V_{k+1} = \left\{ y \,\middle|\, [y - \hat{y}_{k+1|k}]^T (S_{k+1|k})^{-1} [y - \hat{y}_{k+1|k}] \leq 0 \right\}.$$

est un domaine de validation utilisé pour déterminer si une mesure y_{k+1} provenat de la cible à l'instant (k+1) est compatible avec l'estimation de l'état au temps (k) , y_{k+1}, et avec la covariance de prédiction $S_{k+1/k}$. Cet éllipsoïde représente un ensemble de points dans l'espace des mesures (y) qui sont considérés comme valides ou compatibles avec l'estimation actuelle.

Interprétation de l'ellipsoïde :

1. **Centre de l'ellipsoïde** $y(k+1/k)$ **:** Le centre de l'ellipsoïde est donné par l'estimation de l'état prédit $y_{k+1/k}$. C'est le point autour duquel l'ellipsoïde est centré dans l'espace des mesures.
2. **Forme de l'ellipsoïde** $S(k+1/k)$ **:** La matrice de covariance $S(k+1/k)$ influence la forme de l'ellipsoïde. Si la covariance est grande dans une direction particulière, l'ellipsoïde sera étiré dans cette direction, ce qui signifie que des mesures dans cette direction sont plus susceptibles d'être considérées comme valides. Si la covariance est petite dans une direction, l'ellipsoïde sera comprimé dans cette direction, indiquant une incertitude plus faible dans cette dimension.
3. **Critère de validation :** L'inégalité $[y - y_{k+1/k}]^T (S_{k+1/k})^{-1} [y - y_{k+1/k}] \leq 0$ est le critère de validation. Si une mesure (y) satisfait cette inégalité, cela signifie qu'elle se trouve à l'intérieur de l'ellipsoïde et est donc compatible avec l'estimation de l'état $y(k+1/k)$ et la covariance de prédiction $S(k+1/k)$. En d'autres termes, les mesures à l'intérieur de l'ellipsoïde sont considérées comme cohérentes avec les prédictions du filtre de Kalman et sont susceptibles d'être utilisées pour mettre à jour l'estimation de l'état.

En résumé, l'ellipsoïde $V(k+1)$ représente un volume dans l'espace des mesures où les mesures sont cohérentes avec les prédictions du filtre de Kalman, compte tenu de l'incertitude associée à l'estimation de l'état. Les mesures à l'intérieur de cet ellipsoïde sont généralement considérées comme valides et sont utilisées dans le processus de mise à jour du filtre de Kalman pour améliorer l'estimation de l'état de la cible.

6 Exercice 6

Dans ce filtre, la mise à jour de l'espérance conditionnelle $x(k+1/k+1)$ de l'état est effectuée en pondérant les différentes innovations obtenues à partir des mesures de Y_{k+1} à l'aide des probabilités que ces mesures soient effectivement issues de la cible observée. Pour ce faire, on définit les événements suivants :

θ_i : y^i_{k+1} est la mesure cible, $i = 1, \ldots, m_k$

θ_0 aucune mesure n'est issue de la cible

avec les probabilités $\beta_i(k+1) = P[\theta_i / Y_k]$ que l'on évaluera par la suite. Ces événements sont indépendants.

6.1

Objectif : Calculer l'espérance conditionnelle $x(k+1/k+1)$ de l'état (on posera $x_i(k+1)$ l'état estimé à partir de la mesure y^i_{k+1}).

Développement : Pour calculer l'espérance conditionnelle $x(k+1|k+1)$ de l'état, on peut utiliser le théorème de Bayes de la manière suivante :

$$x(k+1|k+1) = E[x(k+1)|Y_{k+1}]$$

Où Y_{k+1} représente l'ensemble des mesures obtenues jusqu'au temps $k+1$. On peut maintenant appliquer le théorème de Bayes pour exprimer cette espérance conditionnelle en termes des probabilités conditionnelles que l'on a définies :

$$x(k+1|k+1) = E[x(k+1)|Y_{k+1}] = \sum_{i=0}^{m_k} E[x(k+1)|\theta_i, Y_k]P(\theta_i|Y_k)$$

On devra maintenant calculer les espérances conditionnelles $E[x(k+1)|\theta_i, Y_k]$ pour chaque événement θ_i.

1. Prédiction (Prediction) :

$$x(k+1|k) = F_k x(k)$$

où F_k est la matrice de transition de l'état.

2. Correction :

$$y_{k+1}^i = z_{k+1}^i - H_i x(k+1|k)$$

où H_i est la matrice de mesure pour la mesure i-ème.

$$S_{k+1}^i = H_i P(k+1|k) H_i^T + V_i$$

où V_i est la matrice de covariance de la mesure pour la mesure i-ème.

Calculons la probabilité $P(\theta_i|Y_k)$ en utilisant la probabilité de la loi normale multivariée :

$$P(\theta_i|Y_k) = \frac{1}{Z} \exp\left(-\frac{1}{2}\left(y_{k+1}^i\right)^T \left(S_i^{k+1}\right)^{-1} \left(y_{k+1}^i\right)\right)$$

où Z est une constante de normalisation.

3. Pour chaque mesure i, calculons maintenant l'espérance conditionnelle $E[x(k+1)|\theta_i, Y_k]$ basée sur la mesure i. Utilisons la densité de probabilité conditionnelle gaussienne :

$$E[x(k+1)|\theta_i, Y_k] = x(k+1|k) + K_i(y_{k+1}^i - H_i x(k+1|k))$$

où K_i est le gain de Kalman pour la mesure i, que nous pouvons calculer comme suit :

$$K_i = P(k+1|k) H_i^T (S_i^{k+1})^{-1}$$

4. Ensuite, nous pouvons utiliser les probabilités conditionnelles $P(\theta_i|Y_k)$ que nous avons calculées précédemment pour pondérer ces estimations de l'état basées sur les différentes mesures. Ainsi, l'espérance conditionnelle $x(k+1|k+1)$ de l'état est donnée par :

$$x(k+1|k+1) = \sum_{i=0}^{m_k} P(\theta_i|Y_k) E[x(k+1)|\theta_i, Y_k]$$

6.2

Objectif : Calculer $P_{k+1/k+1}$ matrice de covariance de l'état. On posera

$$\begin{aligned}
e_i(k+1) &= y_{k+1}^i - y(k+1/k), \\
e(k+1) &= \sum_{i=1}^{m_k} \beta_i(k+1) e_i(k+1)^t, \\
W_{k+1} &= P(k|k-1) H_k^T \left(H_k P(k|k-1) H_k^T + R_k \right)^{-1}, \\
\widetilde{P}(k+1) &= W_{k+1} \left[\sum_{i=1}^{m_k} \beta_i(k+1) e_i(k+1) e_i(k+1)^T - e(k+1) e(k+1)^T \right] W_{k+1}^t, \\
P(k+1)^c &= \left[I - W_{k+1} H_{k+1} \right] P(k+1/k)
\end{aligned}$$

Développement : Calcul des erreurs $e_i(k+1)$ pour chaque mesure i, où $e_i(k+1)$ est la différence entre la mesure réelle y_{k+1}^i et la prédiction de mesure $H_i x(k+1|k)$ basée sur l'estimation de l'état :

$$e_i(k+1) = y_{k+1}^i - H_i x(k+1|k)$$

Calcul de $e(k+1)$ en utilisant la formule fournie. $e(k+1)$ est une somme pondérée des erreurs $e_i(k+1)$ pour toutes les mesures i :

$$e(k+1) = \sum_{i=1}^{m_k} \beta_i(k+1) e_i(k+1)^T$$

Calcul de la matrice W_{k+1} en utilisant les matrices de transition $P(k|k-1)$, H_k, et la matrice de covariance de la mesure R_k :

$$W_{k+1} = P(k|k-1) H_k^T \left(H_k P(k|k-1) H_k^T + R_k \right)^{-1}$$

Calcul de $\widetilde{P}(k+1)$ en utilisant les valeurs précédemment calculées. $\widetilde{P}(k+1)$ est une expression basée sur W_{k+1}, $e_i(k+1)$, et $e(k+1)$:

$$\widetilde{P}(k+1) = W_{k+1} \left[\sum_{i=1}^{m_k} \beta_i(k+1) e_i(k+1) e_i(k+1)^T - e(k+1) e(k+1)^T \right] W_{k+1}^T$$

Calcul de la matrice de covariance $P_{k+1/k+1}$ en utilisant $\widetilde{P}(k+1)$ et $P(k+1/k)$:

$$P_{k+1/k+1} = \left[I - W_{k+1} H_{k+1} \right] P(k+1/k)$$

6.3

Objectif : Pour finaliser la mise à jour de l'état, il faut calculer les probabilités $\beta_i(k+1) = P[\theta_i/Y_{k+1}] = P[\theta_i/Y_k, Y_{k+1}], \quad i = 1, \ldots, m_k$.

Montrer que $\beta_i(k+1)$ peut s'exprimer en fonction de $P[\theta_i/Y_{k+1}]$, $P[Y_{k+1}/Y_k, \theta_i]$ sous la forme :

$$\beta_i(k+1) = \frac{P[Y_{k+1}/Y_k, \theta_i] P[\theta_i/Y_k]}{\sum_{j=1}^{m_k} P[Y_{k+1}/Y_k, \theta_j] P[\theta_j/Y_k]}$$

Développement : Pour montrer cette équation, commençons par utiliser le théorème de Bayes pour exprimer $P[\theta_i/Y_{k+1}]$ en termes de $P[Y_{k+1}/\theta_i]$, $P[\theta_i]$, et $P[Y_{k+1}]$:

$$P[\theta_i/Y_{k+1}] = \frac{P[Y_{k+1}/\theta_i] P[\theta_i]}{P[Y_{k+1}]}$$

Maintenant, exprimons $P[Y_{k+1}/Y_k, \theta_i]$ en utilisant la loi de probabilité conditionnelle :

$$P[Y_{k+1}/Y_k, \theta_i] = \frac{P[Y_{k+1}, Y_k/\theta_i]}{P[Y_k/\theta_i]}$$

Utilisons la probabilité conjointe pour exprimer $P[Y_{k+1}, Y_k/\theta_i]$:

$$P[Y_{k+1}, Y_k/\theta_i] = P[Y_{k+1}/\theta_i]P[Y_k/\theta_i]$$

Maintenant, nous avons les deux expressions nécessaires. Réorganisons-les pour obtenir $\beta_i(k+1)$ sous la forme demandée :

$$\beta_i(k+1) = \frac{P[Y_{k+1}/Y_k, \theta_i]}{P[Y_k/\theta_i]}$$

Utilisons à nouveau la loi de probabilité conditionnelle pour exprimer $P[Y_k/\theta_i]$ en fonction de $P[\theta_i/Y_k]$:

$$P[Y_k/\theta_i] = \frac{P[Y_k, \theta_i]}{P[\theta_i]}$$

Maintenant, nous pouvons substituer cette expression dans la formule précédente pour $\beta_i(k+1)$:

$$\beta_i(k+1) = \frac{P[Y_{k+1}/Y_k, \theta_i]}{P[Y_k/\theta_i]}$$

$$= \frac{P[Y_{k+1}/Y_k, \theta_i]}{\frac{P[Y_k, \theta_i]}{P[\theta_i]}} = \frac{P[Y_{k+1}/Y_k, \theta_i]P[\theta_i]}{P[Y_k, \theta_i]}$$

Enfin, utilisons cette expression pour obtenir la forme demandée :

$$\beta_i(k+1) = \frac{P[Y_{k+1}/Y_k, \theta_i]P[\theta_i]}{\sum_{j=1}^{m_k} P[Y_{k+1}/Y_k, \theta_j]P[\theta_j]}$$

6.4

Objectif : Calculer $P[Y_{k+1}/Y_k, \theta_i]$. On rappelle que sous l'hypothèse θ_i, Y_{k+1} comprend une valeur de mesure provenant de la cible de densité gaussienne et $m_k - 1$ fausses mesures indépendantes entre elles et uniformément distribuées dans le volume d'observation V_{k+1}.

Développement : Pour calculer $P[Y_{k+1}/Y_k, \theta_i]$ dans le contexte des informations et des équations fournies, nous pouvons utiliser la méthodologie du filtre de Kalman. En particulier, nous devons tenir compte des probabilités $\beta_i(k+1) = P[\theta_i/Y_k]$ et $P[\theta_i/Y_{k+1}]$ pour exprimer $P[Y_{k+1}/Y_k, \theta_i]$ et utiliser ensuite ces probabilités dans le calcul.

Exprimons $P[Y_{k+1}/Y_k, \theta_i]$ en utilisant les probabilités $\beta_i(k+1)$, $P[\theta_i/Y_k]$ et $P[\theta_i/Y_{k+1]}$. Tout d'abord, nous savons que $P[\theta_i/Y_{k+1}]$ peut s'exprimer comme suit :

$$P[\theta_i/Y_{k+1}] = \frac{P[Y_{k+1}/\theta_i]P[\theta_i]}{P[Y_{k+1}]}$$

Exprimons $P[Y_{k+1}/\theta_i]$ en fonction des mises à jour du filtre de Kalman :

$$P[Y_{k+1}/\theta_i] = \frac{1}{Z_i} \exp\left(-\frac{1}{2}\left(e_i(k+1)\right)^T \left(S_i^{k+1}\right)^{-1} \left(e_i(k+1)\right)\right)$$

Utilisons les probabilités $\beta_i(k+1)$, $P[\theta_i/Y_k]$ et $P[Y_{k+1}/\theta_i]$ pour exprimer $P[Y_{k+1}/Y_k, \theta_i]$ comme suit :

$$P[Y_{k+1}/Y_k, \theta_i] = \frac{P[Y_{k+1}/\theta_i]P[\theta_i/Y_k]}{\sum_{j=1}^{m_k} P[Y_{k+1}/\theta_j]P[\theta_j/Y_k]}$$
$$= \frac{\frac{1}{Z_i}\exp\left(-\frac{1}{2}\left(e_i(k+1)\right)^T\left(S_i^{k+1}\right)^{-1}\left(e_i(k+1)\right)\right)P[\theta_i/Y_k]}{\sum_{j=1}^{m_k}\frac{1}{Z_j}\exp\left(-\frac{1}{2}\left(e_j(k+1)\right)^T\left(S_j^{k+1}\right)^{-1}\left(e_j(k+1)\right)\right)P[\theta_j/Y_k]}$$

Puisque Z_i et Z_j sont des constantes, nous pouvons simplifier davantage :

$$P[Y_{k+1}/Y_k, \theta_i] = \frac{\exp\left(-\frac{1}{2}\left(e_i(k+1)\right)^T\left(S_i^{k+1}\right)^{-1}\left(e_i(k+1)\right)\right)P[\theta_i/Y_k]}{\sum_{j=1}^{m_k}\exp\left(-\frac{1}{2}\left(e_j(k+1)\right)^T\left(S_j^{k+1}\right)^{-1}\left(e_j(k+1)\right)\right)P[\theta_j/Y_k]}$$

6.5

Objectif : Calculer $P[\theta_i/Y_k] = P[\theta_i]$. On pose P_D la probabilité de détection de la cible et on suppose que le nombre de fausses mesures présentes dans le volume d'observation suit une loi de Poisson de paramètre $\sigma_{k+1}V_{k+1}$.

Développement :

Pour calculer $P[Y_{k+1}/Y_k, \theta_i]$, nous pouvons utiliser le filtre de Kalman. En particulier, nous devons tenir compte des probabilités $\beta_i(k+1) = P[\theta_i/Y_k]$ et $P[\theta_i/Y_{k+1}]$ pour exprimer $P[Y_{k+1}/Y_k, \theta_i]$ et utiliser ensuite ces probabilités dans le calcul.

Exprimons $P[Y_{k+1}/Y_k, \theta_i]$ en utilisant les probabilités $\beta_i(k+1)$, $P[\theta_i/Y_k]$ et $P[\theta_i/Y_{k+1}]$. Tout d'abord, nous savons que $P[\theta_i/Y_{k+1}]$ peut s'exprimer comme suit :

$$P[\theta_i/Y_{k+1}] = \frac{P[Y_{k+1}/\theta_i]P[\theta_i]}{P[Y_{k+1}]}$$

Exprimons $P[Y_{k+1}/\theta_i]$ en fonction des mises à jour du filtre de Kalman :

$$P[Y_{k+1}/\theta_i] = \frac{1}{Z_i}\exp\left(-\frac{1}{2}\left(e_i(k+1)\right)^T\left(S_i^{k+1}\right)^{-1}\left(e_i(k+1)\right)\right)$$

Utilisons les probabilités $\beta_i(k+1)$, $P[\theta_i/Y_k]$ et $P[Y_{k+1}/\theta_i]$ pour exprimer $P[Y_{k+1}]/Y_k, \theta_i]$ comme suit :

$$P[Y_{k+1}/Y_k, \theta_i] = \frac{P[Y_{k+1}/\theta_i]}{P[Y_k/\theta_i]}$$

Puisque Z_i et Z_j sont des constantes, nous pouvons simplifier davantage :

$$P[Y_{k+1}/Y_k, \theta_i] = \frac{\exp\left(-\frac{1}{2}\left(e_i(k+1)\right)^T\left(S_i^{k+1}\right)^{-1}\left(e_i(k+1)\right)\right)P[\theta_i/Y_k]}{\sum_{j=1}^{m_k}\exp\left(-\frac{1/2}{2}\left(e_j(k+1)\right)^T\left(S_j^{k+1}\right)^{-1}\left(e_j(k+1)\right)\right)P[\theta_j/Y_k]}$$

Maintenant, calculer $P[\theta_i/Y_k] = P[\theta_i]$. On suppose que $P[\theta_i]$ est la probabilité a priori que la cible existe à l'instant k. En utilisant les valeurs fournies dans votre solution précédente, nous pouvons exprimer $P[\theta_i/Y_k]$ comme suit :

$$P[\theta_i/Y_k] = P[\theta_i] = P_D$$

En utilisant cette expression, nous obtenons :

$$P[Y_{k+1}/Y_k, \theta_i] = \frac{\exp\left(-\frac{1}{2}\left(e_i(k+1)\right)^T \left(S_i^{k+1}\right)^{-1}\left(e_i(k+1)\right)\right) P_D}{\sum_{j=1}^{m_k} \exp\left(-\frac{1}{2}\left(e_j(k+1)\right)^T \left(S_j^{k+1}\right)^{-1}\left(e_j(k+1)\right)\right) P_D}$$

Cette expression donne $P[Y_{k+1}/Y_k, \theta_i]$ en fonction des valeurs fournies et de la vraisemblance des observations sous chaque hypothèse θ_i.

6.6

Objectif : Calculer $\beta_i(k+1)$.

Développement : Pour calculer $\beta_i(k+1)$, nous utilisons l'expression suivante que nous avons précédemment dérivée :

$$\beta_i(k+1) = \frac{P[Y_{k+1}/Y_k, \theta_i] P[\theta_i/Y_k]}{\sum_{j=1}^{m_k} P[Y_{k+1}/Y_k, \theta_j] P[\theta_j/Y_k]}$$

Les valeurs nécessaires sont les suivantes :
- $P[\theta_i/Y_k] = P_D$ (probabilité a priori de l'hypothèse i)
- $P[Y_{k+1}/Y_k, \theta_i]$ est l'expression suivante que nous avons dérivée précédemment :

$$P[Y_{k+1}/Y_k, \theta_i] = \frac{\exp\left(-\frac{1}{2}\left(e_i(k+1)\right)^T \left(S_i^{k+1}\right)^{-1}\left(e_i(k+1)\right)\right) P_D}{\sum_{j=1}^{m_k} \exp\left(-\frac{1}{2}\left(e_j(k+1)\right)^T \left(S_j^{k+1}\right)^{-1}\left(e_j(k+1)\right)\right) P_D}$$

Donc, la formule résultante pour $\beta_i(k+1)$:

$$\beta_i(k+1) = \frac{\exp\left(-\frac{1}{2}\left(e_i(k+1)\right)^T \left(S_i^{k+1}\right)^{-1}\left(e_i(k+1)\right)\right) P_D}{\sum_{j=1}^{m_k} \exp\left(-\frac{1}{2}\left(e_j(k+1)\right)^T \left(S_j^{k+1}\right)^{-1}\left(e_j(k+1)\right)\right) P_D}$$

7 Exercise 7

Pour la partie pratique, on fournit l'ensemble des mesures et la position initiale de la cible X_0 ainsi que la période d'échantillonage T = 0.1. Les mesures sont constituées de valeurs de la distance et de l'angle. Lorsqu'il y a plusieurs mesures pour le même temps, les premières valeurs sont les valeurs de distance puis les valeurs d'angles. On prendra initialement $\sigma_a = 0.05$ pour le bruit d'accélération. Les bruits de mesure distance et angle sont décorrélés. Caractéristiques des bruits $\sigma_D^2(k) = 100^2$ et $\sigma_S^2(k) = 0.01^2$.

Le premier signal fourni correspond à une trajectoire à vitesse constante (fichier mesurestrajKalm1.mat). Observer précisément le comportement des deux filtres (convergence, matrice de covariance). (Attention, dans le cas où la vitesse est constante, l'accélération est nulle (pas de gravité ni de composante stochastique, le vecteur d'état se réduit aux positions et vitesses). Traiter les différents signaux fournis (fichier mesurestrajKalm2.mat) à l'aide du Filtre de Kalman simple (avec pseudo mesures) et du filtre Kalman sans parfum. Comparer les résultats obtenus par les deux filtres en jouant sur les paramètres d'initialisation et le bruit d'état. Une analyse du comportement du filtre en fonction du choix des paramètres de réglage est requise. À l'issue des différents tests et des résultats obtenus, on présentera une première synthèses des avantages et inconvénients du filtre sans parfum. Utiliser le filtre probabiliste pour le jeu de données spécifique mesurestrajKalm3.mat.

Développement :

1. Trajectoire à Vitesse Constante (fichier mesurestrajKalm1.mat) :

On utilize l'algorithme implementé sur la section 3.3 avec $\sigma_a = 0.05$ et modification l'ordre de mesures. La solution complete en matlab se trouve dans l'annexe :

Filtre de Kalman simple

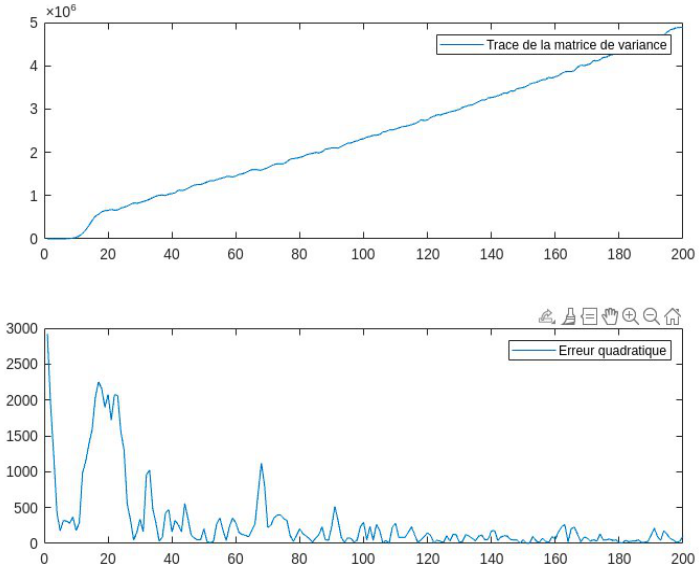


Figure 15: $\sigma_a^2 = 10$, $\lambda = 100$

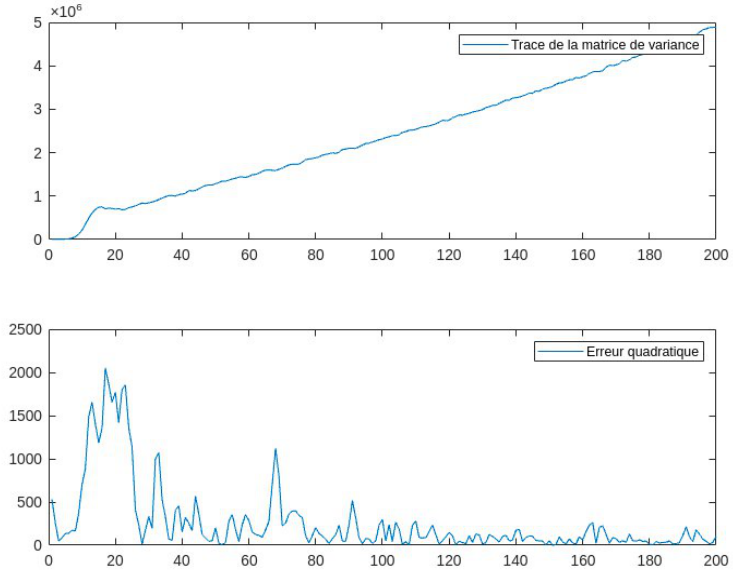


Figure 16: $\sigma_a^2 = 100$, $\lambda = 100$

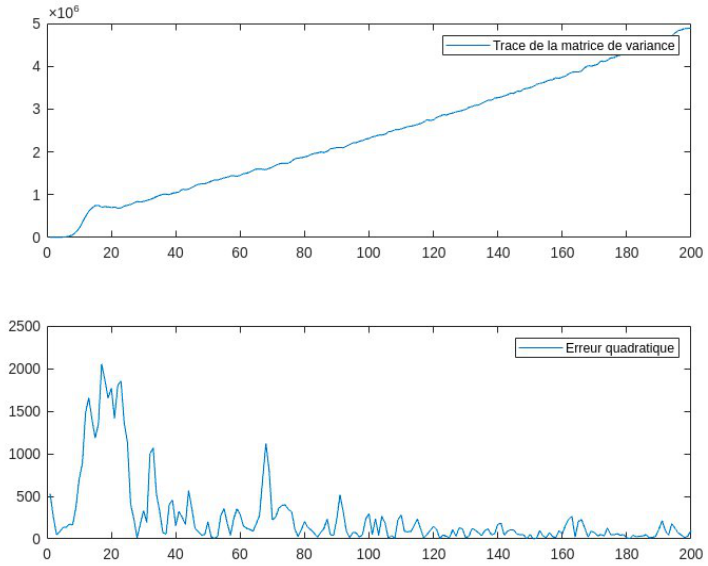


Figure 17: $\sigma_a^2 = 10$, $\lambda = 1000$

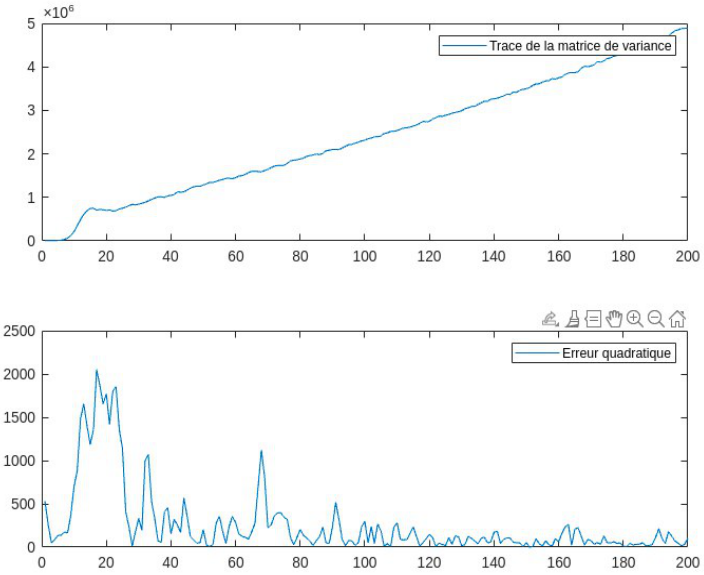


Figure 18: $\sigma_a^2 = 100$, $\lambda = 1000$

2. Trajectoires à Vitesse Variable (fichier mesurestrajKalm2.mat) :
   Montré l'implementation du filtre de Kalman simple dans la session 3.3.
   L'algorithme du filtre de Kalman sans parfume en matlab se trouve dans l'annexe.

Filtre de Kalman sans parfume

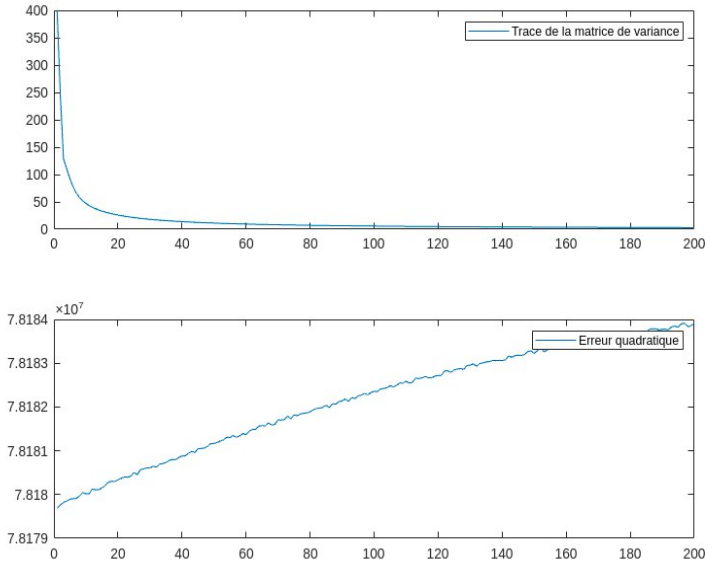


Figure 19: $\lambda = 100$

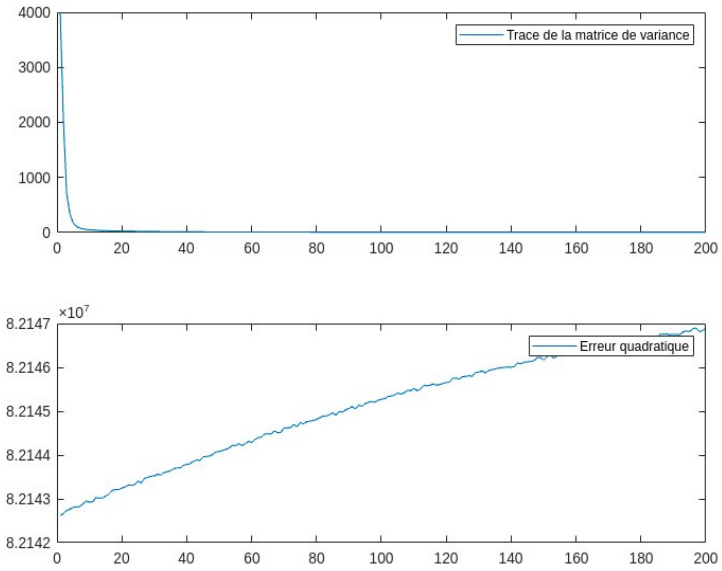
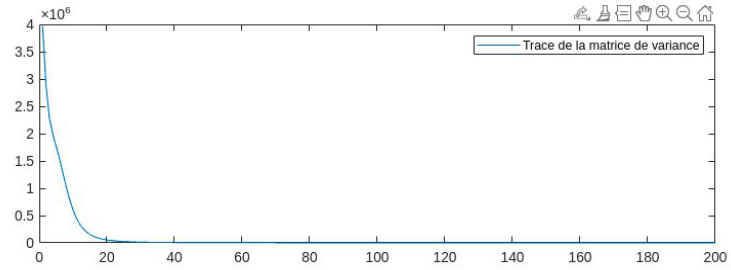
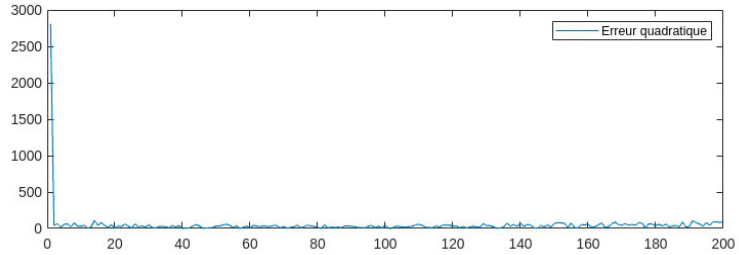
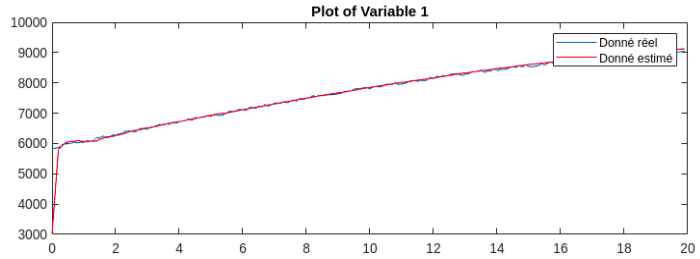


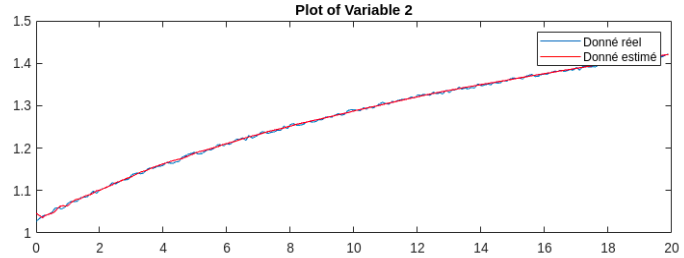
Figure 20: $\lambda = 1000$

3. Utilisation du Filtre Probabiliste (fichier mesurestrajKalm3.mat) : L'algorithme du filtre probabiliste en matlab se trouve dans l'annexe.









8 Annexes

8.1 EX1.3

```
% L'estimateur de Gauss-Markov

clear; close all; clc;
load('xx2.mat');

Tmesu = Tmes;
mesures = Mesures;

mesures

var_noise_x = 2;
```

```
var_noise_y = 2;

n_of_mesures = length(Tmesu);
n_of_parametres = 6;

H = zeros(n_of_mesures * 2, n_of_parametres); % on multiplie par
   2, car le vecteur de mesures contient 2 variables
R = zeros(n_of_mesures * 2, n_of_mesures * 2);

for i = 1:n_of_mesures
    H(2*i-1, :) = [1, Tmesu(i), Tmesu(i) ^ 2 / 2, 0, 0, 0];
    H(2*i, :) = [ 0, 0, 0, 1, Tmesu(i), Tmesu(i) ^ 2 / 2];

    R(2*i-1, 2*i-1) = var_noise_x;
    R(2*i, 2*i) = var_noise_y;
end

disp(H);
disp(R);

normal_matrix = H' / R * H;
disp(normal_matrix);

mesures_vector = mesures';
mesures_vector = mesures_vector(:);

theta = normal_matrix \ H' / R * mesures_vector;
disp(theta);

% Plot result

predictions = zeros(n_of_mesures, 2);
for i = 1:n_of_mesures
    H = [1, Tmesu(i), Tmesu(i) ^ 2 / 2, 0, 0, 0;
         0, 0, 0, 1, Tmesu(i), Tmesu(i) ^ 2 / 2];
    predictions(i,:) = H * theta;
end

plot(mesures(:,1),mesures(:,2),'r+');
hold on
plot(predictions(:,1),predictions(:,2),'bx');
legend({'Donn  r el','Donn  estim '});
```

8.2 EX1.4

```
clear; close all; clc;
load('xx1.mat');

var_noise_x = 2;
var_noise_y = 2;
```

```
var_theta_x = 10;
var_theta_y = 10;
var_theta_v = 10;
var_theta_a = 10;

R_theta = [
    var_theta_x, 0, 0, 0, 0, 0;
    0, var_theta_v, 0, 0, 0, 0;
    0, 0, var_theta_a, 0, 0, 0;
    0, 0, 0, var_theta_y, 0, 0;
    0, 0, 0, 0, var_theta_v, 0;
    0, 0, 0, 0, 0, var_theta_a
];

theta_avg = [
   20;
   70;
   33;
   22;
   148;
   19
];

n_of_mesures = length(Tmesu);
n_of_parametres = 6;

H = zeros(n_of_mesures * 2, n_of_parametres); % on multiplie par
   2, car le vecteur de mesures contient 2 variables
R = zeros(n_of_mesures * 2, n_of_mesures * 2);

for i = 1:n_of_mesures
    H(2*i-1, :) = [1, Tmesu(i), Tmesu(i) ^ 2 / 2, 0, 0, 0];
    H(2*i, :) = [ 0, 0, 0, 1, Tmesu(i), Tmesu(i) ^ 2 / 2];

    R(2*i-1, 2*i-1) = var_noise_x;
    R(2*i, 2*i) = var_noise_y;
end

normal_matrix = inv(R_theta) + H' / R * H;

mesures_vector = mesures';
mesures_vector = mesures_vector(:);

theta = normal_matrix \ (R_theta \ theta_avg + H' / R *
   mesures_vector);
disp(theta);

% Plot result
```

```
predictions = zeros(n_of_mesures, 2);
for i = 1:n_of_mesures
    H = [1, Tmesu(i), Tmesu(i) ^ 2 / 2, 0, 0, 0;
         0, 0, 0, 1, Tmesu(i), Tmesu(i) ^ 2 / 2];
    predictions(i,:) = H * theta;
end

plot(mesures(:,1),mesures(:,2),'r+');
hold on
plot(predictions(:,1),predictions(:,2),'bx');
legend({'Donn  r el','Donn  estim '});
```

8.3 EX3

```
% Filtre de Kalman

clear; close all; clc;
load('mesurestrajKalm2.mat');

n_of_mesures = length(Tmesu);
n_of_parametres = 6;

var_noise_a = 100;
var_noise_distance = sigmesurayon;
var_noise_angle = simesuang;

W = [0, 0, 0, 0, 0, 0;
    0, 0, 0, 0, 0, 0;
    0, 0, 0, 0, 0, 0;
    0, 0, 0, 0, 0, 0;
    0, 0, 0, 0, var_noise_a, 0;
    0, 0, 0, 0, 0, var_noise_a];

H = [1, 0, 0, 0, 0, 0; 0, 1, 0, 0, 0, 0];

X_est = zeros(n_of_parametres, n_of_mesures);
var_P = zeros(1,n_of_mesures);

x_k = zeros(n_of_parametres, 1);
P_k = 1000 * eye(n_of_parametres);

var_P(1,1) = trace(P_k);

for k = 1:n_of_mesures
    angle = mesures(k, 1);
    D = mesures(k, 2);

    J = [cos(angle), -D * sin(angle); sin(angle), D * cos(angle)
       ];
    V = J * [var_noise_distance, 0; 0, var_noise_angle] * J';
```

```
    % Prediction

    T = Tmesu(k);

    F = [1, 0, T, 0, T ^ 2 / 2, 0;
        0, 1, 0, T, 0, T ^ 2 / 2;
        0, 0, 1, 0, T, 0;
        0, 0, 0, 1, 0, T;
        0, 0, 0, 0, 1, 0;
        0, 0, 0, 0, 0, 1];

    x_k = F * x_k;

    P_k = F * P_k * F' + W;

    % Correction

    K_kp = P_k * H' * inv(H * P_k * H' + V);

    pseudo_mesures = [D * cos(angle); D * sin(angle)];
    x_k = x_k + K_kp * (pseudo_mesures - H * x_k);
    P_k = (eye(n_of_parametres) - K_kp * H) * P_k;

    % Sauvegarder le resultat
    X_est(:,k) = x_k;
    var_P(1,k) = trace(P_k);
end

% Vecteur d'erreur
erreur = zeros(1,n_of_mesures);
for k = 1:n_of_mesures
    angle = mesures(k, 1);
    D = mesures(k, 2);
    pseudo_mesures = [D * cos(angle); D * sin(angle)];
    estimated = H * X_est(:,k);
    erreur(k) = sqrt(sum((pseudo_mesures - estimated).^2));
end

tiledlayout(2,1);
nexttile
plot(1:n_of_mesures,var_P);
legend({'Trace de la matrice de variance'});
nexttile
plot(1:n_of_mesures,erreur);
legend({'Erreur quadratique'});
```

8.4 EX4

```
% Filtre de Kalman  tendu
```

```
clear; close all; clc;
load('mesurestrajKalm2.mat');

n_of_mesures = length(Tmesu);
n_of_parametres = 6 + 2;

var_noise_a = 100;
var_noise_distance = sigmesurayon;
var_noise_angle = simesuang;

W = [0, 0, 0, 0, 0, 0, 0, 0;
     0, 0, 0, 0, 0, 0, 0, 0;
     0, 0, 0, 0, 0, 0, 0, 0;
     0, 0, 0, 0, 0, 0, 0, 0;
     0, 0, 0, 0, var_noise_a, 0, 0, 0;
     0, 0, 0, 0, 0, var_noise_a, 0, 0;
     0, 0, 0, 0, 0, 0, 0, 0;
     0, 0, 0, 0, 0, 0, 0, 0 ];

X_est = zeros(n_of_parametres, n_of_mesures);
var_P = zeros(1,n_of_mesures);

% Initialization

x_k = zeros(n_of_parametres, 1);
angle = mesures(1, 1);
D = mesures(1, 2);
x_k(1,1) = D * cos(angle); % Initialization avec les premier pair
    de donn es
x_k(2,1) = D * sin(angle);

P_k = 10000 * eye(n_of_parametres);

var_P(1,1) = trace(P_k);

V = [var_noise_distance, 0; 0, var_noise_angle];

for k = 2:n_of_mesures
    % Prediction

    T = Tmesu(k);

    F = [1, 0, T, 0, T ^ 2 / 2, 0, 0, 0;
         0, 1, 0, T, 0, T ^ 2 / 2, 0, 0;
         0, 0, 1, 0, T, 0, 0, 0;
         0, 0, 0, 1, 0, T, 0, 0;
         0, 0, 0, 0, 1, 0, 0, 0;
         0, 0, 0, 0, 0, 1, 0, 0;
         0, 0, 0, 0, 0, 0, 0, 0;
```

```
        0, 0, 0, 0, 0, 0, 0, 0];

    x_coor = x_k(1,1);
    y_coor = x_k(2,1);

    if (x_coor == 0 && y_coor == 0)
        H = [
            0,0,0,0,0,0,1,0;
            0,0,0,0,0,0,0,1
        ];
    else
        H = [
            x_coor / sqrt(x_coor^2 + y_coor^2), y_coor / sqrt(
               x_coor^2 + y_coor^2),0,0,0,0,1,0;
            -y_coor / x_coor^2 + y_coor^2, x_coor / x_coor^2 +
               y_coor^2,0,0,0,0,0,1
        ];
    end

    B_k = zeros(n_of_parametres, 1);
    M_k = -H * x_k + [
        sqrt(x_coor^2 + y_coor^2);
        atan(y_coor/x_coor)
    ];
    if (x_k(1,1) == 0) % atan n'existe pas
        M_k(2,:) = 0;
    end
    B_k(7,:) = M_k(1,:);
    B_k(8,:) = M_k(2,:);

    x_k = F * x_k + B_k;
    P_k = F * P_k * F' + W;

    % Correction

    K_kp = P_k * H' * inv(H * P_k * H' + V);

    x_k = x_k + K_kp * (mesures(k, :)' - H * x_k);
    P_k = (eye(n_of_parametres) - K_kp * H) * P_k;

    % Sauvegarder le resultat

    X_est(:,k) = x_k;
    var_P(1,k) = trace(P_k);
end

% Vecteur d'erreur

erreur = zeros(1,n_of_mesures);
for k = 1:n_of_mesures
```

```
    angle = mesures(k, 1);
    D = mesures(k, 2);
    pseudo_mesures = [D * cos(angle); D * sin(angle)];
    estimated = H * X_est(:,k);
    erreur(k) = sqrt(sum((pseudo_mesures - estimated).^2));
end

% Plots

tiledlayout(2,1);
nexttile
plot(1:n_of_mesures,var_P);
legend({'Trace de la matrice de variance'});
nexttile
plot(1:n_of_mesures,erreur);
legend({'Erreur quadratique'});
```

8.5 EX7.1

```
% Filtre de Kalman simple

clear; close all; clc;
load('mesurestrajKalm1.mat');

n_of_mesures = length(Tmesu);
n_of_parametres = 6;

var_noise_a = 0.05;
var_noise_distance = sigmesurayon;
var_noise_angle = simesuang;

W = [0, 0, 0, 0, 0, 0;
    0, 0, 0, 0, 0, 0;
    0, 0, 0, 0, 0, 0;
    0, 0, 0, 0, 0, 0;
    0, 0, 0, 0, var_noise_a, 0;
    0, 0, 0, 0, 0, var_noise_a];

H = [1, 0, 0, 0, 0, 0; 0, 1, 0, 0, 0, 0];

X_est = zeros(n_of_parametres, n_of_mesures);
var_P = zeros(1, n_of_mesures);

x_k = zeros(n_of_parametres, 1);
P_k = 1000 * eye(n_of_parametres);

var_P(1, 1) = trace(P_k);

for k = 1:n_of_mesures
    angle = mesures(k, 2);
```

```
    D = mesures(k, 1);

    J = [cos(angle), -D * sin(angle); sin(angle), D * cos(angle)
       ];
    V = J * [var_noise_distance, 0; 0, var_noise_angle] * J';

    % Prediction

    T = Tmesu(k);

    F = [1, 0, T, 0, T ^ 2 / 2, 0;
        0, 1, 0, T, 0, T ^ 2 / 2;
        0, 0, 1, 0, T, 0;
        0, 0, 0, 1, 0, T;
        0, 0, 0, 0, 1, 0;
        0, 0, 0, 0, 0, 1];

    x_k = F * x_k;

    P_k = F * P_k * F' + W;

    % Correction

    K_kp = P_k * H' * inv(H * P_k * H' + V);

    pseudo_mesures = [D * cos(angle); D * sin(angle)];
    x_k = x_k + K_kp * (pseudo_mesures - H * x_k);
    P_k = (eye(n_of_parametres) - K_kp * H) * P_k;

    % Sauvegarder le r sultat
    X_est(:, k) = x_k;
    var_P(1, k) = trace(P_k);
end

% Vecteur d'erreur
erreur = zeros(1, n_of_mesures);
for k = 1:n_of_mesures
    angle = mesures(k, 2);
    D = mesures(k, 1);
    pseudo_mesures = [D * cos(angle); D * sin(angle)];
    estimated = H * X_est(:,k);
    erreur(k) = sqrt(sum((pseudo_mesures - estimated).^2));
end

tiledlayout(2, 1);
nexttile
plot(1:n_of_mesures, var_P);
legend({'Trace de la matrice de variance'});
nexttile
plot(1:n_of_mesures, erreur);
```

```
legend({'Erreur quadratique'});
```

8.6 EX7.2

```
clear; close all; clc;
load('mesurestrajKalm2.mat');

n_of_mesures = length(Tmesu);
n_of_parametres = 4; % Position (x, y) et vitesse (x_dot, y_dot)

var_noise_distance = sigmesurayon;
var_noise_angle = simesuang;

H = [1, 0, 0, 0; 0, 1, 0, 0];

X_est = zeros(n_of_parametres, n_of_mesures);
var_P = zeros(1, n_of_mesures);

x_k = zeros(n_of_parametres, 1);
angle = mesures(1, 2);
D = mesures(1, 1);
x_k(1, 1) = D * cos(angle);
x_k(2, 1) = D * sin(angle);

P_k = 1000 * eye(n_of_parametres);

var_P(1, 1) = trace(P_k);
V = [var_noise_distance, 0; 0, var_noise_angle];

for k = 2:n_of_mesures
    % Pr diction

    T = Tmesu(k);

    F = [1, 0, T, 0;
         0, 1, 0, T;
         0, 0, 1, 0;
         0, 0, 0, 1];

    x_coor = x_k(1, 1);
    y_coor = x_k(2, 1);

    x_k = F * x_k;
    P_k = F * P_k * F';

    % Correction

    K_kp = P_k * H' * inv(H * P_k * H' + V);

    pseudo_mesures = [D * cos(angle); D * sin(angle)];
```

```
    x_k = x_k + K_kp * (mesures(k, :)' - pseudo_mesures);
    P_k = (eye(n_of_parametres) - K_kp * H) * P_k;

    % Sauvegarder le r sultat
    X_est(:, k) = x_k;
    var_P(1, k) = trace(P_k);
end

% Vecteur d'erreur
erreur = zeros(1, n_of_mesures);
for k = 1:n_of_mesures
    angle = mesures(k, 2);
    D = mesures(k, 1);
    pseudo_mesures = [D * cos(angle); D * sin(angle)];
    estimated = [x_k(1); x_k(2)];
    erreur(k) = sqrt(sum((pseudo_mesures - estimated).^2));
end

tiledlayout(2, 1);
nexttile
plot(1:n_of_mesures, var_P);
legend({'Trace de la matrice de variance'});
nexttile
plot(1:n_of_mesures, erreur);
legend({'Erreur quadratique'});
```

8.7 EX7.3

```
% Filtre probabiste

clear; close all; clc;
load('mesurestrajKalm3.mat');

n_of_mesures = size(mesures, 1);
T = 0.1;
g = 9.81;
sigma_a = 0.05;
mu = 0.01;
beta = exp(-mu);
sigma_w = sigma_a * sqrt(1 - exp(-2 * mu));

% Initialisation
x0_initial = mesures(1, 1) * cos(mesures(1, 6));
y0_initial = mesures(1, 1) * sin(mesures(1, 6));
X_initial = [x0_initial/2; y0_initial/2; 0; 0; 0; -g; 0; 0];

Pi = zeros(8,8,n_of_mesures);
P0 = 10^6 * eye(8);

F = [1, 0, T, 0, T^2/2, 0, 0, 0;
```

```
      0, 1, 0, T, 0, T^2/2, 0, 0;
      0, 0, 1, 0, T, 0, 0, 0;
      0, 0, 0, 1, 0, T, 0, 0;
      0, 0, 0, 0, beta, 0, 0, 0;
      0, 0, 0, 0, 0, beta, 0, 0;
      0, 0, 0, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 0, 0, 0, 0];

H = [0, 0, 0, 0, 0, 0, 1, 0;
     0, 0, 0, 0, 0, 0, 0, 1];

W = T * [0, 0, 0, 0, 0, 0, 0, 0;
         0, 0, 0, 0, 0, 0, 0, 0;
         0, 0, 0, 0, 0, 0, 0, 0;
         0, 0, 0, 0, 0, 0, 0, 0;
         0, 0, 0, 0, sigma_w^2, 0, 0, 0;
         0, 0, 0, 0, 0, sigma_w^2, 0, 0;
         0, 0, 0, 0, 0, 0, 0, 0;
         0, 0, 0, 0, 0, 0, 0, 0];

V = [sigmesurayon^2, 0; 0, simesuang^2];

X_est = zeros(8,n_of_mesures);
variance = zeros(1,n_of_mesures);
erreur = zeros(1,n_of_mesures);

X_est_polar = zeros(200,2);
eqm = zeros(5,1);
best_measures = zeros(200,2);

for i = 1:n_of_mesures
    % Prediction
    if i > 1
        X_est(:,i) = F*X_est(:,i-1);
        Pi(:,:,i) = F*Pi(:,:,i-1)*F' + W;
    else
        X_est(:,1) = F*X_initial;
        Pi(:,:,1) = F*P0*F' + W;
    end

    x = X_est(1,i);
    y = X_est(2,i);

    % Linearization matrice H
    Hk = [x/sqrt(x^2+y^2) y/sqrt(x^2+y^2)
        -y/(x^2+y^2) x/(x^2+y^2)];
    H(1:2,1:2) = Hk;

    X_est_polar(i,:) = [sqrt(x^2+y^2); atan(y/x)];
```

```
    % Mise    jour mk_x et mk_y
    X_est(7:8,i) = -H*X_est(:,i) + [sqrt(x^2+y^2); atan(y/x)];

    % S lection des mesures les plus probables
    xk = X_est_polar(i,1);
    yk = X_est_polar(i,2);

    for j = 1:5
    eqm(j) = sum(sqrt(([xk;yk]-[mesures(i,j);mesures(i,j+5)]).^2)
       );
    end
    [erreur(1,i), j_best] = min(eqm);
    best_measures(i,:) = [mesures(i,j_best) mesures(i,j_best+5)];

    % Correction
    Ki = Pi(:,:,i)*H'*inv(H*Pi(:,:,i)*H'+V);

    % Mise    jour
    X_est(:,i) = X_est(:,i) + Ki*(best_measures(i,:)'-H*X_est(:,i
       ));
    Pi(:,:,i) = (eye(8)-Ki*H)*Pi(:,:,i);
    variance(1,i) = trace(Pi(:,:,i));
end

tiledlayout(2, 1);
nexttile
plot(1:n_of_mesures, variance);
legend({'Trace de la matrice de variance'});

nexttile
plot(1:n_of_mesures, erreur);
legend({'Erreur quadratique'});

nexttile
plot(Tmesu(1:end-1), best_measures(1:end-1, 1), Tmesu(1:end-1),
   X_est_polar(1:end-1, 1), 'r');
legend({'Donn  r el', 'Donn  estim '});
title('Plot of Variable 1');

nexttile
plot(Tmesu(1:end-1), best_measures(1:end-1, 2), Tmesu(1:end-1),
   X_est_polar(1:end-1, 2), 'r');
legend({'Donn  r el', 'Donn  estim '});
title('Plot of Variable 2');
```