Exemples de mise en œuvre du filtre de Kalman Etendu

SDIA 2020-2021

1 Mise en œuvre du filtre de Kalman étendu

Dans cette application on cherche à estimer un signal aléatoire sinusoïdal. C'est à dire que l'on cherche à estimer l'amplitude a_k du signal et la phase ϕ_k du signal de la forme :

$$y(k) = a_k \sin(2\pi\nu_0 k T_e + \phi_k)$$

Pour cela on considère l'équation d'état suivante :

$$\left\{ \begin{array}{l} x_{k+1} = f_k\left(x_k\right) + v_k \\ y_k = h\left(x_k\right) + w_k \end{array} \right.$$

Le vecteur d'état est :

$$x_k = \begin{pmatrix} a_k \\ \phi_k \end{pmatrix}$$

Avec :

- a_k : l'amplitude.
- ϕ_k : la phase.

et

- v_k : le bruit d'état.
- w_k : le bruit de mesure.
- $h\left(x_k\right) = a_k \sin(2\pi\nu_0 k T_e + \phi_k)$.
- T_e : la période d'échantillonnage.

avec v_k le bruit d'état tel que $v_k \sim \mathcal{N}\left(\mathbf{0}, \mathbf{Q}_{k+1}\right)$ et w_k le bruit de mesure tel que $w_k \sim \mathcal{N}\left(\mathbf{0}, \mathbf{R}_{k+1}\right)$

La mise en oeuvre du filtre de Kalman étendu nécessite le calcul de :

$$F_k = \left.\frac{\partial f_k}{\partial x}\right|_{x=\hat{x}_{k|k}}$$

$$H_{k+1} = \left.\frac{\partial h_{k+1}}{\partial x}\right|_{x=\hat{x}_{k+1|k}}$$

Entrée: Un tableau de mesure z de taille N
Sortie : Un tableau $\hat{x}$ contenant l'évolution de l'estimation du vecteur d'état

Initialisation du vecteur d'état;
$\hat{x}(0|0) \leftarrow m_0$;
Initialisation de la matrice d'autocorrélation;
$\hat{P}(0|0) \leftarrow \Lambda_0$;
$k \leftarrow 0$;
while $k < N$ **do**
  $\hat{x}(k+1|k) \leftarrow f(\hat{x}(k|k)$;
  $P(k+1|k) \leftarrow F(k)P(k|k)F(k)^T + Q(k)$;
  $k \leftarrow k+1$;
  $S(k) \leftarrow H(k)P(k|k-1)H(k)^T + R(k)$;
  $K(k) \leftarrow P(k|k-1)H(k)^T S(k)^{-1}$;
  $\epsilon(k) \leftarrow z(k) - h\left(\hat{x}(k|k-1)\right)$;
  $\hat{x}\left(k|k\right) \leftarrow \hat{x}\left(k|k-1\right) + K(k)\epsilon(k)$;
  $P\left(k|k\right) \leftarrow P\left(k|k-1\right) - K(k)H(k)P(k|k-1)$;
end

Algorithm 1: filtre de Kalman étendu

2 Simulations

Etablir l'équation d'état, en vérifiant que :

$$\begin{cases} x_{k+1} = x_k + v_k \\ y_k = h\left(x_k\right) + w_k \end{cases}$$

- v_k : le bruit d'état.
- w_k : le bruit de mesure.
- $h\left(x_k\right) = a_k \sin(2\pi\nu_0 k T_e + \phi_k)$.
- T_e : la période d'échantillonnage.

La mise en œuvre du filtre de Kalman étendu nécessite le calcul de :

$$H_k = \frac{\partial h(x_k)}{\partial x_k}$$

Dans notre cas :

$$\frac{\partial h(x_k)}{\partial a_k} = \sin(2\pi\nu_0 kT_e + \phi_k)$$

$$\frac{\partial h(x_k)}{\partial \phi_k} = a_k \cos(2\pi\nu_0 kT_e + \phi_k)$$

Et donc :

$$H_k = [\sin(2\pi\nu_0 kT_e + \phi_k)\ a_k \cos(2\pi\nu_0 kT_e + \phi_k)]$$

Pour la simulation on pourra prendre les valeurs suivantes :

Les données ci-dessous sont utilisées pour simuler des valeurs de $y(k)$ et de $y(k)$ bruitée. ces données sont stockée dans le fichier **donnee.xlsx** (table 1).

Temps	signalReel	signalBruite
$k \times T_e$	$y(k)$	$y(k) + b(k)$
0	0	-0,503935968
0,005173841	1,90139724	4,727026464
0,010347682	3,517096748	5,284031238
0,015521523	4,604328801	3,413861059
0,020695364	4,999729466	8,070116733
0,025869205	4,643887131	1,597133008
0,031043046	3,590269503	2,944079463
$\vdots$	$\vdots$	$\vdots$

Table 1 – Les données du fichier **donnee.xlsx** ($b(k) \sim \mathcal{N}(0,3)$)

Les données du fichier **donnee.xlsx** seront utilisées pour estimer :

$$x_k = \begin{pmatrix} a_k \\ \phi_k \end{pmatrix}$$

- $a_k = 5$, amplitude de la sinusoïde.
- $\phi_k = 0$, valeur de la phase de la sinusoïde.
- ν_0, valeur de la fréquence de la sinusoïde .
- $\nu_e = 129.28$ Hz, fréquence d'échantillonnage.

Matrice de covariance du bruit d'état :

$$\mathbf{Q} = \begin{pmatrix} 2\ 10^{-5} & 0 \\ 0 & 2\ 10^{-1} \end{pmatrix}$$

Matrice de covariance du bruit de mesure :

$$\mathbf{R} = 3$$

Matrice décrivant la dynamique du système :

$$\mathbf{A} = \begin{pmatrix} 1.0 & 0 \\ 0 & 1.0 \end{pmatrix}$$

$$\mathbf{Q} = \begin{pmatrix} 2\ 10^{-5} & 0 \\ 0 & 2\ 10^{-1} \end{pmatrix} \quad \text{et} \quad \mathbf{R} = 3$$

2.1 Exemple de résultats

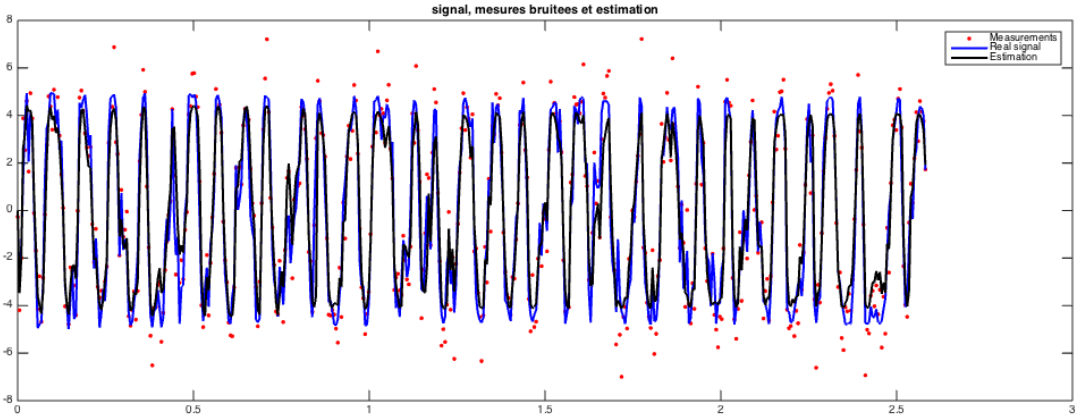


FIGURE 1: Mise en œuvre du filtre de Kalman étendu sur des données simulées.

Références

[1] Y. Bar-Shalom, X. Rong Li, and T. Kirubarayan. *Estimation with Applications to Tracking and Navigation. Theory, Algorithms and Software.* John Wiley and Sons, New York Chichester Weinheim Brisbane Singapore Toronto, 2001.

[2] F. Caron. *Inférence bayésienne pour la détermination et la sélection de modèles stochastiques.* PhD thesis, Ecole Centrale de Lille & Université des Sciences et Technologies de Lille, 2006. `http://tel.archives-ouvertes.fr/index.php?halsid=dc20mttsgbavbqs1spk0gdmlh5&view_this_doc=tel-00140088&version=1`.

[3] J. Hartikainen and S. Särkkä. Optimal filtering with kalman filters and smoothers – a manual for matlab toolbox ekf/ukf. `http://www.lce.hut.fi/research/mm/ekfukf/`, Department of Biomedical Engineering and Computational Science, Helsinki University of Technology, 2008.

[4] B. Ristic, S. Arulampalam, and N. Gordon. *Beyond the Kalman Filter Particle Filter for Tracking Applications.* Artech House, Boston London, 2004.

Annexe

December 11, 2019

0.0.1 Extended Kalman Filter

```
In [1]: import numpy as np
        import matplotlib.pyplot as plt
        from scipy.stats import norm
```

0.0.2 Pour la simulation

$a_k = 5$, amplitude de la sinusoïde
$\phi_k = 0$, valeur de la phase de la sinusoïde
ν_0, valeur de la fréquence de la sinusoïde
$\nu_e = 129.28$ Hz, fréquence d'échantillonnage
Matrice de covariance du bruit d'état :

$$\mathbf{Q} = \begin{pmatrix} 2\,10^{-5} & 0 \\ 0 & 2\,10^{-1} \end{pmatrix}$$

Matrice de covariance du bruit de mesure:

$$\mathbf{R} = 3$$

Matrice décrivant la dynamique du système:

$$\mathbf{A} = \begin{pmatrix} 1.0 & 0 \\ 0 & 1.0 \end{pmatrix}$$

```
In [2]: Amplitude = 5
        Phi = 0
        Nu_e = 193.28
        Nu_0 = 12
        Te = 1 / Nu_e
        nbEchantillon = 20000
        t = np.arange(0, nbEchantillon , 1) * Te
```

```
In [3]: A = np.array([[1.0, 0.0],[0.0, 1.0]])
        Q = np.array([[2e-5, 0.0],[0.0, 2e-1]])
```

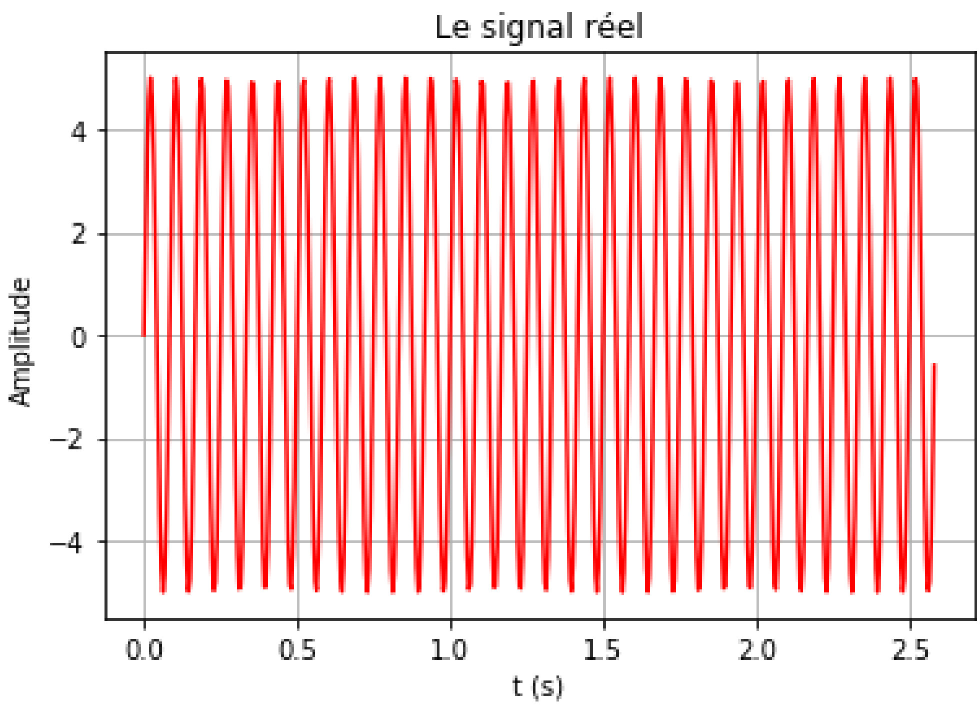
```
In [4]: # X l'état du système
        X = np.zeros(shape = (2, nbEchantillon))
        Xhat = np.transpose([Amplitude, Phi])
```

```
X[:,0] = Xhat
# On applique l'équation qui caractérise la dynamique du système
for i in range(1, nbEchantillon, 1):
    X[:,i] = np.dot(A, X[:,i-1])
    + np.transpose(np.random.multivariate_normal(np.transpose([0, 0]), Q))
```

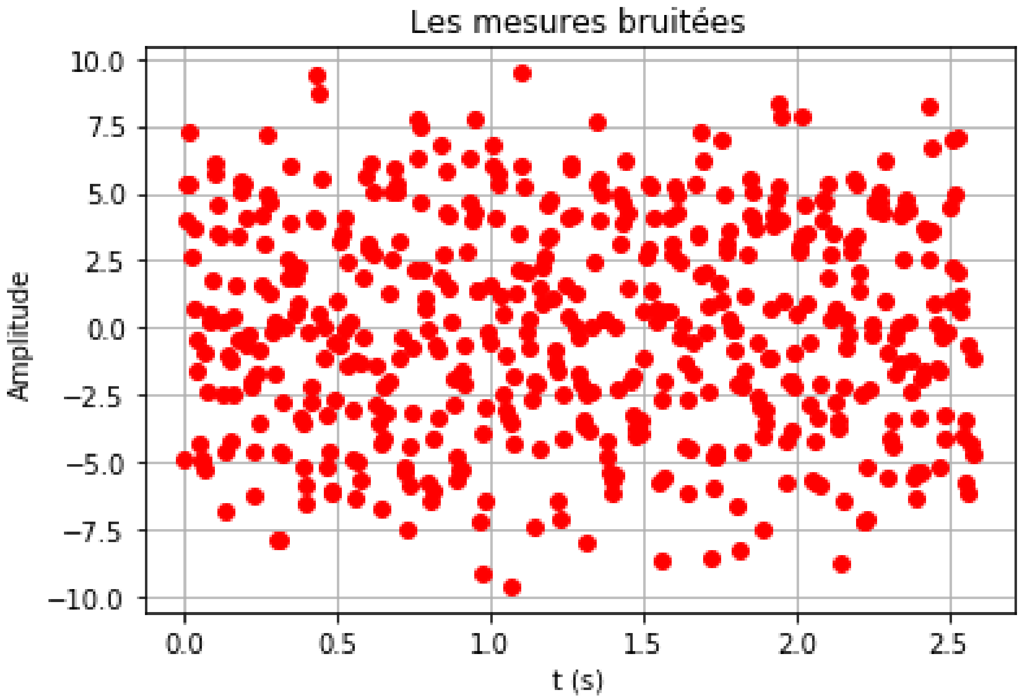
```
In [5]: def equationMesure(x, t, Nu):
            Amplitude = x[0,:]
            Phi = x[1,:]
            Resultat = Amplitude * np.sin(2 * np.pi * Nu * t + Phi )
            return Resultat
```

```
In [6]: R = 3
        sd = np.sqrt(R)
        Y = np.zeros(shape = nbEchantillon)
        # Calcul de la sortie réelle du système
        # on applique l'équation de sortie
        Y_real = equationMesure(X, t, Nu_0)
        # Bruit de mesure
        Bm = norm.rvs(0, sd, nbEchantillon)
        # Calcul de la sortie bruitée
        # Mesures dont on dispose pour faire l'estimation
        Y = Y_real + Bm
```

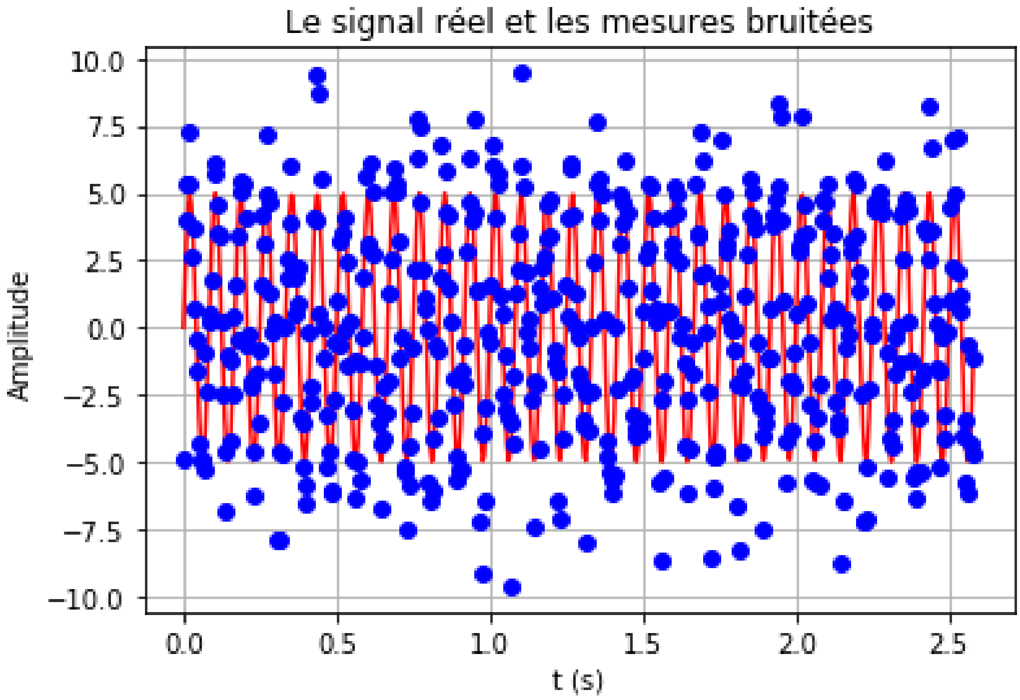
```
In [7]: limite = 500
        plt.plot(t[0:limite], Y_real[0:limite], 'r')
        plt.title('Le signal réel')
        plt.xlabel('t (s)')
        plt.ylabel('Amplitude')
        plt.grid()
        plt.show()
```



In [8]:
```
limite = 500
plt.plot(t[0:limite], Y[0:limite], 'ro')
plt.title('Les mesures bruitées')
plt.xlabel('t (s)')
plt.ylabel('Amplitude')
plt.grid()
plt.show()
```



In [9]:
```
limite = 500
plt.plot(t[0:limite], Y_real[0:limite], 'r')
plt.plot(t[0:limite], Y[0:limite], 'bo')
plt.title('Le signal réel et les mesures bruitées')
plt.xlabel('t (s)')
plt.ylabel('Amplitude')
plt.grid()
plt.show()
```


Le signal réel et les mesures bruitées
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
Amplitude
0.0
0.5
1.0
1.5
2.0
2.5
t (s)