Introduction à la robotique mobile
Solutionnaire du cahier d'exercices version 0.52

Philippe Giguère

19 septembre 2017

2 Révision calcul matriciel

2.1 Équations linéaire

a)

$$\begin{bmatrix} 2 & -5 & 4 \\ 3 & 0 & 9 \\ 0 & -1 & 17 \end{bmatrix} \mathbf{x} = \mathbf{y}$$

b)

matlab

```
A = [2 -5 4; 3 0 9; 0 -1 17];
x = [3 4 5]';
y = A*x
```

Python

```
A = np.matrix([[2, -5, 4], [3, 0, 9],[0, -1, 17]])
x = np.matrix([3, 4, 5])
x = x.transpose()
y = A*x
```

2.2

Nous avons

$$\vec{c} = \begin{bmatrix} 4 \\ 2 \\ -3 \end{bmatrix} \text{ et } \vec{d} = \begin{bmatrix} -1 \\ 7 \\ 3 \end{bmatrix}$$

Nous utiliserons la relation

$$\vec{c} \cdot \vec{d} = \|\vec{c}\| \|\vec{d}\| \cos\theta$$

$$\theta = \cos^{-1}\left(\frac{\vec{c} \cdot \vec{d}}{\|\vec{c}\| \|\vec{d}\|}\right)$$

avec

$$\vec{c} \cdot \vec{d} = 4 \times -1 + 2 \times 7 - 3 \times 3 = 1$$

$$\|\vec{c}\| = 5.3852, \text{ et } \|\vec{d}\| = 7.6811$$

$$\theta = \cos^{-1}\left(\frac{\vec{c} \cdot \vec{d}}{\|\vec{c}\| \|\vec{d}\|}\right)$$

$$\theta = \cos^{-1}\left(\frac{1}{5.3852 \times 7.6811}\right) = 1.4566 \text{ rad}$$

2.3

$\mathbf{Q}^{-1} = \mathbf{Q}^T$

Ce que nous verrons, c'est que pour faire une rotation en sens inverse, il suffit simplement d'utiliser la transpose de la matrice de rotation. Simple comme bonjour !

3 Produit de deux fonctions Gaussiennes

3.1

La moyenne est μ_f et l'écart-type = σ_f. Oui je sais c'est simpliste, mais j'ai vu des cas où les personnes ne comprenaient pas bien que les paramètres d'une distribution Gaussiennes encodent directement ces statistiques.

3.2

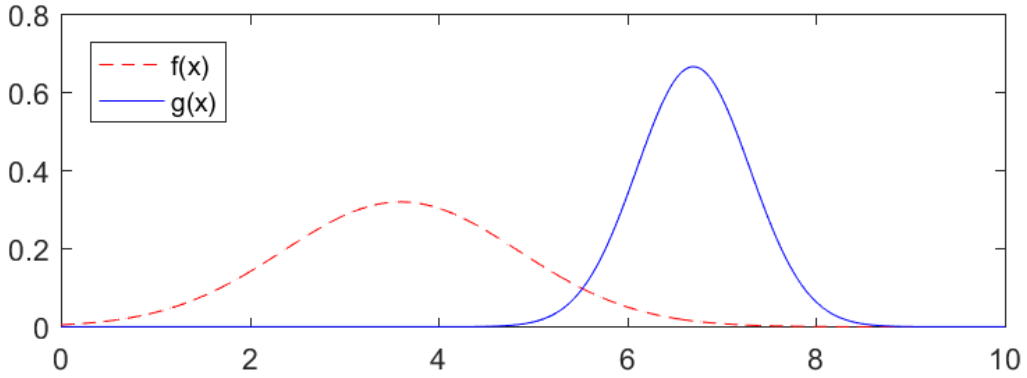


FIGURE 1 – Solution

3.3

On voit bien ici que les deux fonctions ont la même forme, à un facteur d'échelle près. Pour mieux illustrer, j'ai donné un facteur de 50 pour le produit $f(x)g(x)$ sur le tracé ici-bas.

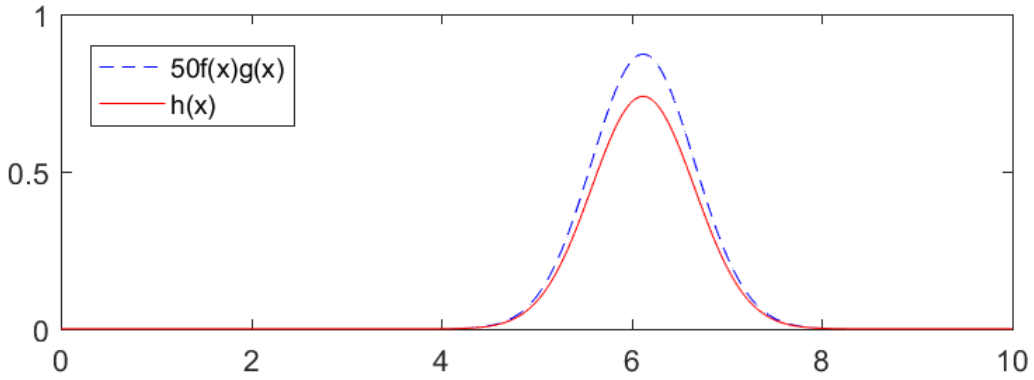


FIGURE 2 – Solution

3.4

Solution de D. Landry

Si $f(x)$ et $g(x)$ sont des gaussiennes, alors leur produit peut se développer ainsi.

$$\begin{aligned} f(x)g(x) &= ae^{-\frac{1}{2}(\frac{x-\mu_f}{\sigma_f})^2} be^{-\frac{1}{2}(\frac{x-\mu_g}{\sigma_g})^2} \\ &= abe^{-\frac{1}{2}((\frac{x-\mu_f}{\sigma_f})^2+(\frac{x-\mu_g}{\sigma_g})^2)} \end{aligned}$$

Intéressons-nous à l'exposant seulement. Nous voulons le ramener sous la forme $q(x) = ax^2 + bx + c$.

$$\begin{aligned} &-\frac{1}{2}((\frac{x-\mu_f}{\sigma_f})^2 + (\frac{x-\mu_g}{\sigma_g})^2) \\ &\qquad = -\frac{1}{2}((\frac{x^2 - 2\mu_f + \mu_f^2}{\sigma_f^2}) + (\frac{x^2 - 2\mu_g + \mu_g^2}{\sigma_g^2})) \\ &\qquad = -\frac{1}{2}(\sigma_g^2(\frac{x^2 - 2\mu_f + \mu_f^2}{\sigma_f^2\sigma_g^2}) + \sigma_f^2(\frac{x^2 - 2\mu_g + \mu_g^2}{\sigma_g^2\sigma_f^2})) \\ &\qquad = -\frac{1}{2}(\frac{(\sigma_f^2 + \sigma_g^2)x^2 - 2(\sigma_g^2\mu_f + \sigma_g^2\mu_g)x + \mu_f^2\mu_g^2\sigma_f^2\sigma_g^2}{\sigma_f^2\sigma_g^2} \\ &\qquad = -(\frac{1}{2\sigma_g^2} + \frac{1}{2\sigma_f^2})x^2 + (\frac{\mu_f}{\sigma_f^2} + \frac{\mu_g}{\sigma_g^2})x + -\frac{\mu_f^2\mu_g^2}{2} \end{aligned}$$

On a donc obtenu une quadratique, ce qui montre que le produit de deux gaussiennes est toujours une gaussienne.

4 Distribution Gaussienne multivariée

Code matlab pour tracer toutes les réponses. Notez particulièrement l'utilisation de la racine carrée `sqrt` lors du scaling `l1` et `l1` des vecteurs propre, car c'est une matrice de covariance et non pas d'écart-types.

```
n = 1000;

% Matrice de covariance
S = [7 3; 3 2];
% On genere les points
pts = mvnrnd([0 0],S,n);

subplot(1,2,1);
plot(pts(:,1),pts(:,2),'b.');
axis equal;
hold on;
title('a)');

% On calcule la matrice de covariance
Sest = cov(pts);

% On trace les vecteurs propres
[V,D] = eig(Sest);

v1 = V(:,1);
v2 = V(:,2);
l1 = sqrt(D(1,1));
l2 = sqrt(D(2,2));

line([0 v1(1)*l1], [0 v1(2)*l1],'LineWidth',2,'Color','r');
line([0 v2(1)*l2], [0 v2(2)*l2],'LineWidth',2,'Color','r');

subplot(1,2,2);
plot(pts(:,1),pts(:,2),'.');
axis equal;
hold on;
line([0 v1(1)*l1], [0 v1(2)*l1],'LineWidth',2,'Color','r');
line([0 v2(1)*l2], [0 v2(2)*l2],'LineWidth',2,'Color','r');
% et une ellipse maintenant
h = ellipse(l1,l2,acos(v2(2)),0,0,'r',200); set(h,'LineWidth',2);
%text(-2,3,'EXEMPLE');
title('b)');
```

Code en Python.

```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.patches import Ellipse
from plotEllipse import plotEllipse

S = np.matrix('7 3; 3 2')
pts = np.random.multivariate_normal([0,0],S,1000);

plt.plot(pts[:,0],pts[:,1], 'b.')

# estimation de la covariance d'une matrice
EstimatedCov = np.cov(np.transpose(pts));

# eigenvalues
[W, v] = np.linalg.eig(EstimatedCov)

Vec1 = v[:,0]*np.sqrt(W[0]);
Vec2 = v[:,1]*np.sqrt(W[1]);
plt.plot([-Vec1[0],Vec1[0]],[-Vec1[1],Vec1[1]], 'r')
plt.plot([-Vec2[0],Vec2[0]],[-Vec2[1],Vec2[1]], 'r')
plotEllipse([0, 0],EstimatedCov,'r')

plt.axis('equal')
plt.show()
```

5 Révision trigonométrie

Il existe plusieurs solutions possibles. Vous en retrouvez trois ici-bas.

$$\begin{aligned}
\frac{h}{\sin(\theta)} &= \frac{b}{\sin(c)} \\
\frac{h}{\sin(\theta)} &= \frac{b}{\sin(\frac{\pi}{2}-\theta)} \\
\frac{h}{\sin(\theta)} &= \frac{b}{\cos(\theta)} \\
\frac{h\cos(\theta)}{\sin(\theta)} &= b \\
h\cot(\theta) &= b \\
b &= h\cot(\theta)
\end{aligned}$$

In triangle 2 summing the angles we have that

$$\begin{aligned}
\phi + d + \frac{\pi}{2} &= \pi \\
d &= \frac{\pi}{2} - \phi
\end{aligned}$$

And applying the sine rule to triangle 2 we get

$$\begin{aligned}
\frac{h}{\sin(\phi)} &= \frac{a-b}{\sin(d)} \\
\frac{h}{\sin(\phi)} &= \frac{a-b}{\sin(\frac{\pi}{2}-\phi)} \\
\frac{h}{\sin(\phi)} &= \frac{a-b}{\cos(\phi)} \\
\frac{h\cos(\phi)}{\sin(\phi)} &= a-b \\
h\cot(\phi) &= a-b \\
h\cot(\phi) &= a - h\cot(\theta) \\
h\cot(\phi) + h\cot(\theta) &= a \\
h(\cot(\phi)+\cot(\theta)) &= a \\
h &= \frac{a}{\cot(\phi)+\cot(\theta)} = \frac{a}{\frac{1}{tan(\theta)}+\frac{1}{tan(\phi)}}
\end{aligned}$$

FIGURE 3 – Solution 1

On pose la distance BH = b
On a :

$$\frac{1}{\tan\theta} = \frac{b}{h}$$

et

$$\frac{1}{\tan\phi} = \frac{a-b}{h}$$

donc :

$$\frac{1}{\tan\theta} + \frac{1}{\tan\phi} = \frac{a}{h}$$

donc on pourait dire que

$$h = \frac{a\tan\theta\tan\phi}{\tan\theta + \tan\phi}$$

FIGURE 4 – Solution 2

Soit :

$$\frac{\boldsymbol{a}}{\boldsymbol{sin(\psi)}} = \frac{\boldsymbol{b}}{\boldsymbol{sin(\theta)}} \text{ ,où } \boldsymbol{\psi = 180 - \theta - \phi}. \tag{1}$$

En isolant $\boldsymbol{b}$ on obtient :

$$\boldsymbol{b} = \frac{\boldsymbol{asin(\theta)}}{\boldsymbol{sin(\psi)}} \tag{2}$$

La hauteur $\boldsymbol{h}$ peut être exprimée en fonction des variables suivantes :

$$\boldsymbol{h = bsin(\phi)} \tag{3}$$

En subtituant $\boldsymbol{b}$ par sont équivalent en (2) on obtient :

$$\boldsymbol{h} = \frac{\boldsymbol{asin(\theta)sin(\phi)}}{\boldsymbol{sin(\psi)}} \tag{4}$$

Puisque $\boldsymbol{\psi = 180 - \theta - \phi}$, il est possible, grâce à l'identitée trigonométrique $\boldsymbol{sin(\pi - x) = sin(x)}$, de simplifier $\boldsymbol{sin(\psi)}$ en (4) de la façon suivante :

$$\boxed{\boldsymbol{h} = \frac{\boldsymbol{asin(\theta)sin(\phi)}}{\boldsymbol{sin(\theta + \phi)}}} \; \boldsymbol{ou} \; \frac{\boldsymbol{asin(\theta)sin(\phi)}}{\boldsymbol{sin(\pi - \theta - \phi)}} \tag{5}$$

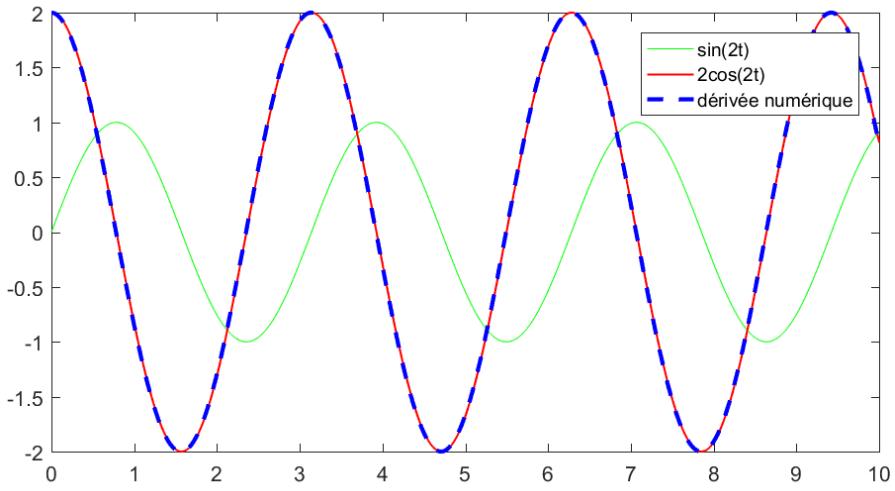
FIGURE 5 – Solution 3

6 Révision calcul différentiel et intégral

6.1 Dérivée

```
dt = 0.01
t =0:dt:10
Signal = sin(2*t)

h(1) = plot(t,Signal,'g');
hold on;
h(2) = plot(t,2*cos(2*t),'r','linewidth',1);
DiffSignal = (Signal(2:end)- Signal(1:end-1))/dt;
h(3) = plot(t(1:end-1),DiffSignal,'b--','linewidth',2);
legend(h,{'sin(2t)', '2cos(2t)','derivee numerique'})
```



6.2 Intégrale

```
dt = 0.01
t =0:dt:10
Signal = sin(2*t)

h(1) = plot(t,Signal,'g');
hold on;
h(2) = plot(t,-0.5*cos(2*t),'r','linewidth',1);
IntSignal = cumsum(Signal)*dt;
h(3) = plot(t,IntSignal,'b--','linewidth',2);
legend(h,{'sin(2t)', '-1/2cos(2t)','Integrale ordre 0'})
```

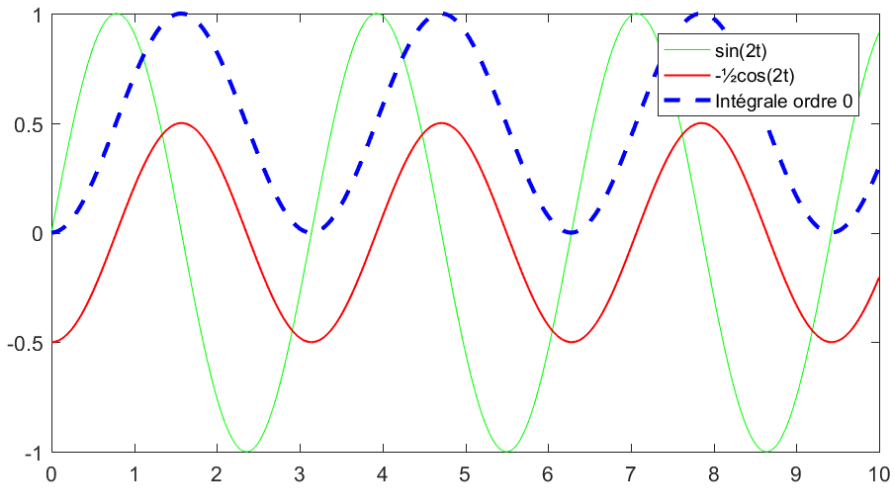


FIGURE 6 – Le décalage entre les deux solutions correspond à la fameuse constante C, qui a été choisie ici pour éviter de superposer les deux graphes.

7 Linéarisation d'une fonction de capteur non-linéaire

7.1

On trouve tout d'abord avec un logiciel en calcul symbolique ou un site web la dérivée de la fonction du capteur :

$$f'_{capteur}(x) = \frac{-120}{x^2} e^{-(\ln x - 4)^2} (2 \ln x - 7)$$

Calculons $f_{capteur}(x)$ et $f'_{capteur}(x)$ aux points voulus :

a_i	$f_{capteur}(x)$	$f'_{capteur}(x)$
$a_1 = 20$	2.1885	0.1104
$a_2 = 50$	2.3815	-0.0392
$a_3 = 140$	0.3532	-0.0073

On obtient donc :

$$f_{capteur}(x)\Big|_{a_1} = 2.1885 + 0.1104(x - 20)$$

$$f_{capteur}(x)\Big|_{a_2} = 2.3815 - 0.0392(x - 50)$$

$$f_{capteur}(x)\Big|_{a_3} = 0.3532 - 0.0073(x - 140)$$

7.2 Propagation d'une gaussienne à travers $f_{capteur}$

Il faut voir ici que la propagation de l'erreur de mesure se fera via l'inverse absolu de la pente du capteur. De façon générique, nous avons notre approximation linéaire suivante :

$$y = f(a) + f'(a)(x - a) \tag{1}$$

Comme c'est une fonction linéaire, on peut l'inverser :

$$x = \frac{y - f(a)}{f'(a)} + a \tag{2}$$

Si on admet que x et y sont des variables aléatoires, et que Δ_y est l'écart-type sur le signal, nous avons

$$Var\{x\} = Var\left\{\frac{y - f(a)}{f'(a)} + a\right\}. \tag{3}$$

Pour des variables indépendantes, la variance d'une somme est la somme des variances :

$$Var\{x\} = Var\left\{\frac{y - f(a)}{f'(a)}\right\} + Var\{a\} = Var\left\{\frac{y}{f'(a)}\right\} + Var\left\{\frac{-f(a)}{f'(a)}\right\} + Var\{a\}. \tag{4}$$

La variance d'une constante est nulle, ce qui simplifie l'équation 4 grandement :

$$Var\{x\} = Var\left\{\frac{y}{f'(a)}\right\} = \frac{1}{f'(a)^2} Var\{y\}. \tag{5}$$

Donc l'erreur (l'écart-type) sur la position x sera la racine carrée de l'équation ci-dessus :

$$\Delta_x = \frac{1}{|f'(a)|}\Delta_y \tag{6}$$

Pour $a_1 = 20$, la précision sur la position sera :

$$\Delta_x = \frac{1}{|0.1104|}0.0195 = 0.1766. \tag{7}$$

Le capteur a donc une précision d'environ 0.18 cm à $a_1 = 20$ cm. De la même manière, la précision sera de 0.50 cm à $a_2 = 50$ cm et 2.67 cm à $a_3 = 140$ cm.