

TD2. Signaux déterministes. Echantillonnage. Convolution.

Exercice 1 : représentation de signaux sinusoïdaux

1 - Signal sinus en temps continu

La sinusoïde est la fonction de base pour l'analyse des signaux et systèmes, qu'il faut bien connaître

$$f(t) = 500 \cos(2000\pi t - \pi/3)$$

1. Quelle est la fréquence et la phase du signal $f(t)$ ci-dessous où t est le temps ?
2. Donnez l'expression temporelle du signal électrique sinusoïdal d'amplitude 0,5V, de fréquence 1kHz, et nul à l'instant $t=0$. Représentez-le sur une période.

2 - Signal discret (ou en temps discret)

Un signal discret est un ensemble de valeurs ordonnées ou d'échantillons prélevés sur un signal continu, on note $s(n)$ ou $s(nT_e)$ la n ème valeur du signal. Les échantillons sont régulièrement espacés de la valeur T_e qui est la période d'échantillonnage. Le signal lui-même est représenté par $\{s(nT_e)\}$, bien que par abus de notation, on le représente également par $s(nT_e)$ ou $s(n)$.

Représentez l'allure des signaux discrets suivants :

1. $\{s(n)\}$, obtenu en échantillonnant le signal ci-dessus à la fréquence $F_e=8\text{kHz}$ à partir de l'instant $t=0$. Représenter $\{s(n)\}$ sur une période de s .
2. $\{k(n)\}$ nulle pour tout n sauf pour $n=0$ qu'on nommera impulsion
3. $\{u(n)\}$ nulle pour $n<0$ et égal à 1 autrement, qu'on nommera échelon unité.

3- Programmation

Soit la ligne de programme MATLAB suivante : `a=sin((2*pi/8)*(0:0.01:8));`

1. Donnez la fréquence de ce signal, le pas d'échantillonnage et la fréquence d'échantillonnage;
2. Le nombre d'échantillons calculés, la durée du signal et le nombre de périodes générées (l'unité est la seconde);
3. La valeur moyenne du signal sur la durée générée.

Exercice 2: filtrage et produit de convolution

Pour un système comme celui décrit dans l'exercice 1, il est nécessaire de pouvoir traiter l'information mesurée. C'est un des rôles joués par le microprocesseur qui permet l'implantation d'algorithmes de traitement du signal comme notamment :

- le filtrage pour lisser les données souvent bruitées et donc fluctuantes,

- la détection de sauts ou de discontinuités dans les données mesurées.

Ces algorithmes reposent sur le produit de convolution qui fait l'objet de cet exercice.

1. Donnez l'équation de convolution pour des signaux continus et discrets.
2. Calculez à la main le produit de convolution des signaux discrets $\{x(kT_e)\}$ et $\{y(kT_e)\}$ suivants :
 $\{x(kT_e)\}$: 0 0 1 1 1 1 2 2 2 2 1 1 1 1 0 0
 $\{y(kT_e)\}$: 1 1 1
3. Divisez la valeur des échantillons résultats par la somme des coefficients de $\{y(kT_e)\}$. Représentez les deux signaux. Que constatez-vous ?
4. Ecrire la partie du programme en C qui concerne les deux boucles nécessaires au calcul du produit de convolution. Vous considérerez que les échantillons des signaux sont stockés dans deux tableaux tabx et taby. Les longueurs respectives sont lx et ly. Le signal résultat sera appelé tabz de longueur lz=lx+ly-1. Pour simplifier la gestion des bords, vous utiliserez un tableau temporaire tabtempx où aura été préalablement stocké tabx, et complété de (lx-1) zeros de part et d'autre du tableau.
5. Représentez graphiquement les fonctions $x(t)=\text{rect}((t-2)/4)$ et $y(t)=\text{rect}(t-1/2)$
6. En utilisant l'expression continue du produit de convolution, calculez le produit de convolution de x avec y. Représentez le.

Exercice 3 : programmation, filtrage, convolution

Soit le programme matlab suivant. Précisez son rôle. Quelle est la réponse impulsionnelle du système? A quoi sert la boucle d'indice k.

```
[x,FS,NBITS]=wavread('speech_dft.wav');

NT=size(x);

N=NT(1);

bruit=0.1*rand(N,1);

xb=x+bruit;

xbs=xb;

a0=1/2;

y(1)=a0*xb(1);

for k=1:10

    for i=2:N

        y(i)=a0*xb(i)+a0*xb(i-1);

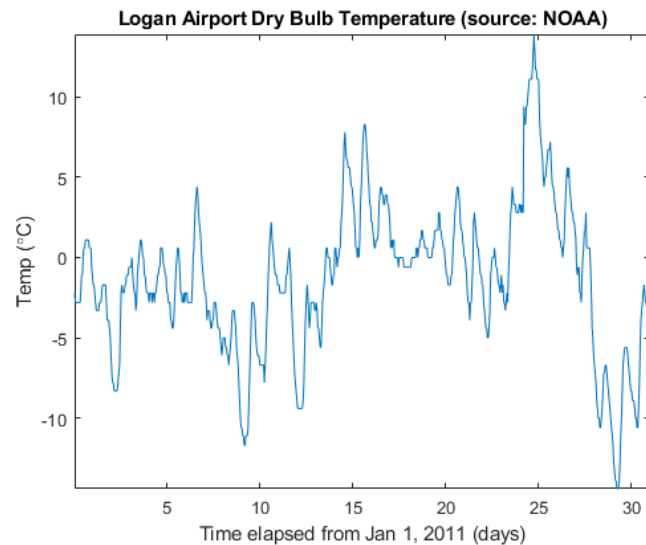
    end

    xb=y;

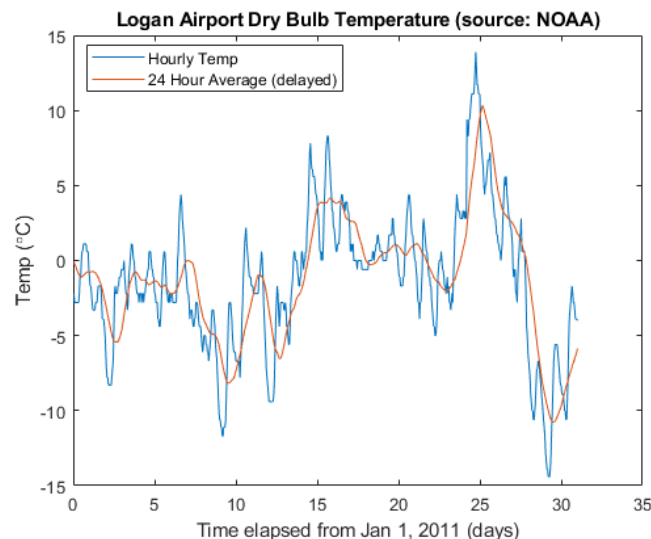
end
```

Exercice 4 - application 1 : lissage et moyenne mobile.

A partir des figures et des programmes associés, identifiez les traitements effectués et leur objectif

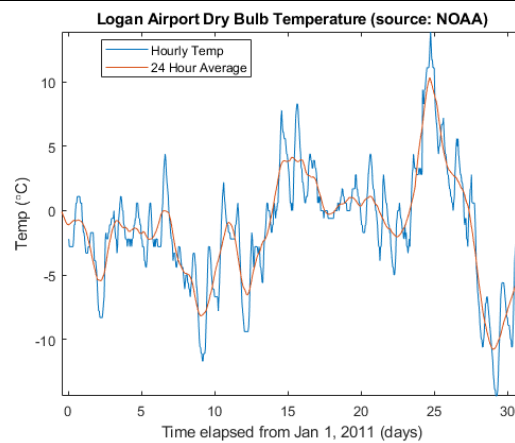


```
load bostemp
days = (1:31*24)/24;
plot(days, tempC)
axis tight
ylabel('Temp (\text{°C})')
xlabel('Time elapsed from Jan 1, 2011 (days)')
title('Logan Airport Dry Bulb Temperature (source: NOAA)')
```

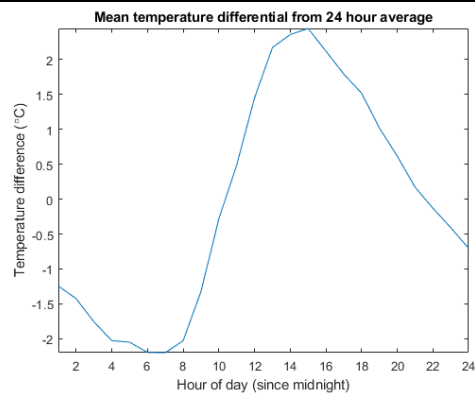


```
hoursPerDay = 24;
coeff24hMA = ones(1, hoursPerDay)/hoursPerDay;

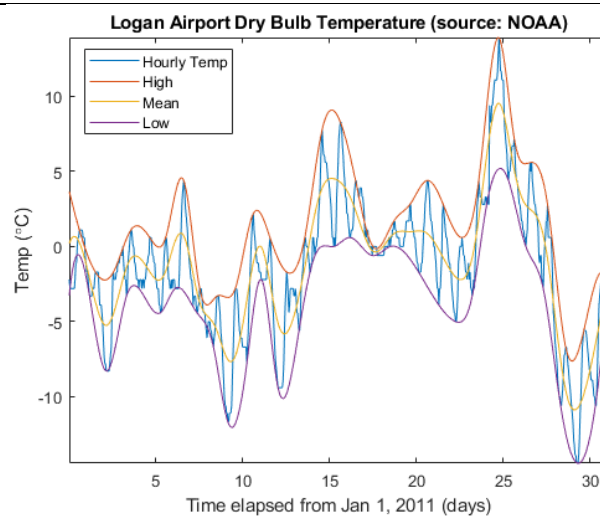
avg24hTempC = filter(coeff24hMA, 1, tempC);
plot(days,[tempC avg24hTempC])
legend('Hourly Temp','24 Hour Average (delayed)','location','best')
```



```
fDelay = (length(coeff24hMA)-1)/2;
plot(days,tempC, ...
      days-fDelay/24,avg24hTempC)
```

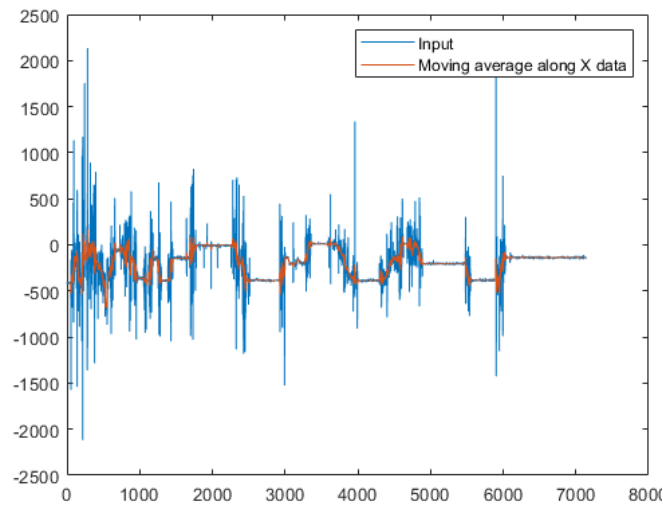


```
figure
deltaTempC = tempC - avg24hTempC;
deltaTempC = reshape(deltaTempC, 24, 31).';
plot(1:24, mean(deltaTempC))
```



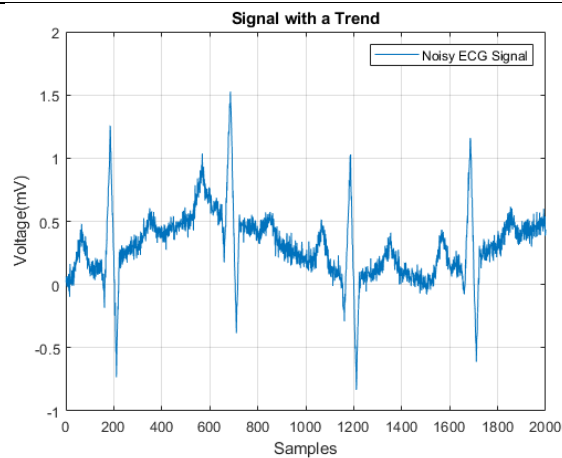
```
[envHigh, envLow] = envelope(tempC,16,'peak');
```

```
envMean = (envHigh+envLow)/2;
plot(days,tempC, days,envHigh,days,envMean,days,envLow)
```

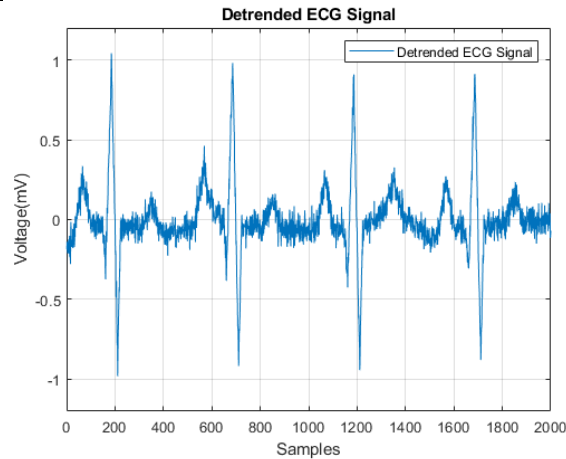


```
winLen = 10;
accel = load('LSM9DS1accelData73.mat');
movAvg = movmean(accel.data,winLen,'Endpoints','fill');
plot([accel.data(:,1),movAvg(:,1)]);
legend('Input','Moving average along X data');
```

Exercice 4 – application 2 : analyse d'un signal ECG



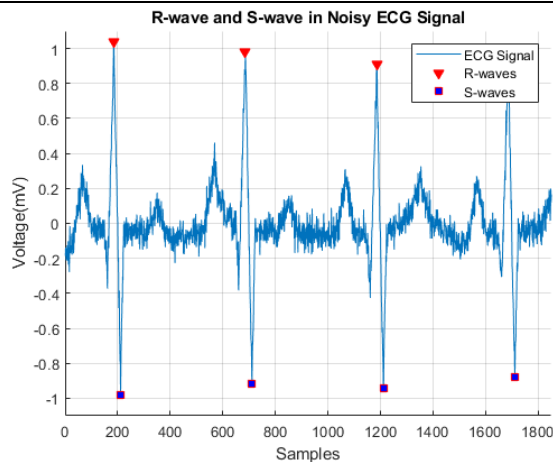
```
load noisyeeg.mat
t = 1:length(noisyECG_withTrend);
figure
plot(t,noisyECG_withTrend)
title('Signal with a Trend')
xlabel('Samples');
ylabel('Voltage(mV)')
legend('Noisy ECG Signal')
grid on
```



```
[p,s,mu] = polyfit((1:numel(noisyECG_withTrend))',noisyECG_withTrend,6);
f_y = polyval(p,(1:numel(noisyECG_withTrend))',[],mu);
```

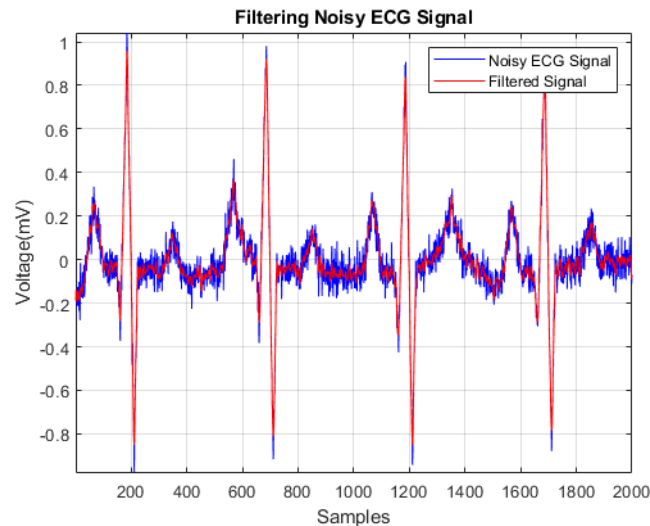
```
ECG_data = noisyECG_withTrend - f_y;           % Detrend data
```

```
Figure plot(t,ECG_data)
grid on
ax = axis;
axis([ax(1:2) -1.2 1.2])
title('Detrended ECG Signal')
xlabel('Samples')
ylabel('Voltage(mV)')
legend('Detrended ECG Signal')
```



```
[~,locs_Rwave] = findpeaks(ECG_data,'MinPeakHeight',0.5,...
                             'MinPeakDistance',200);
ECG_inverted = -ECG_data;
[~,locs_Swave] = findpeaks(ECG_inverted,'MinPeakHeight',0.5,...
                             'MinPeakDistance',200);
```

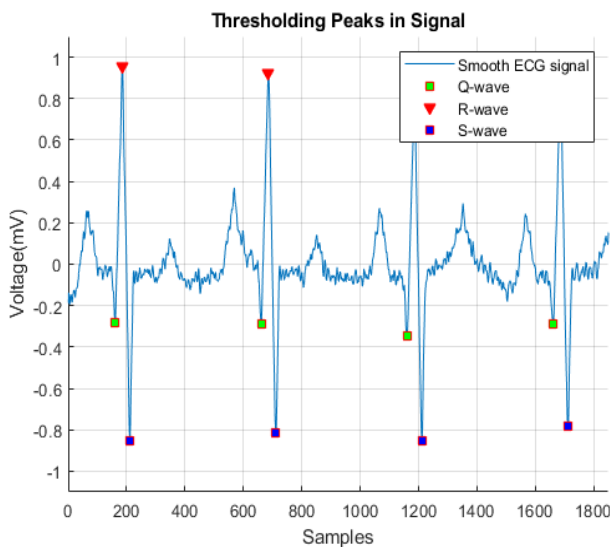
```
Figure hold on
plot(t,ECG_data)
plot(locs_Rwave,ECG_data(locs_Rwave),'rv','MarkerFaceColor','r')
plot(locs_Swave,ECG_data(locs_Swave),'rs','MarkerFaceColor','b')
axis([0 1850 -1.1 1.1]) grid on legend('ECG Signal','R-waves','S-waves')
xlabel('Samples') ylabel('Voltage(mV)')
title('R-wave and S-wave in Noisy ECG Signal')
```



```
smoothECG = sgolayfilt(ECG_data,7,21);
```

```
figure plot(t,ECG_data,'b',t,smoothECG,'r') grid on axis tight
xlabel('Samples') ylabel('Voltage(mV)')
legend('Noisy ECG Signal','Filtered Signal') title('Filtering Noisy ECG Signal')
```

% Values of the Extrema



```
[val_Qwave, val_Rwave, val_Swave] =
deal(smoothECG(locs_Qwave),
smoothECG(locs_Rwave),
smoothECG(locs_Swave));
```

```
meanError_Qwave =
mean((noisyECG_withTrend(locs_Qwave)
) - val_Qwave))
meanError_Qwave = 0.2771
```

```
meanError_Rwave =
mean((noisyECG_withTrend(locs_Rwave)
) - val_Rwave))
meanError_Rwave = 0.3476
```

```
meanError_Swave =
mean((noisyECG_withTrend(locs_Swave)
) - val_Swave))
meanError_Swave = 0.1844
```

```
[~,min_locs] = findpeaks(-smoothECG,'MinPeakDistance',40);
```

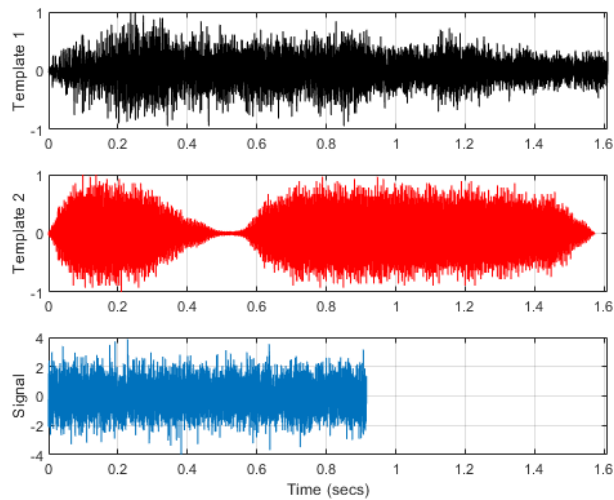
% Peaks between -0.2mV and -0.5mV

```
locs_Qwave = min_locs(smoothECG(min_locs)>-0.5 & smoothECG(min_locs)<-0.2);
```

```
figure hold on
plot(t,smoothECG);
plot(locs_Qwave,smoothECG(locs_Qwave),'rs','MarkerFaceColor','g')
plot(locs_Rwave,smoothECG(locs_Rwave),'rv','MarkerFaceColor','r')
plot(locs_Swave,smoothECG(locs_Swave),'rs','MarkerFaceColor','b')
grid on title('Thresholding Peaks in Signal')
xlabel('Samples') ylabel('Voltage(mV)') ax = axis;
axis([0 1850 -1.1 1.1]) legend('Smooth ECG signal','Q-wave','R-wave','S-wave')
```

TD3. Corrélation, échantillonnage et spectre.

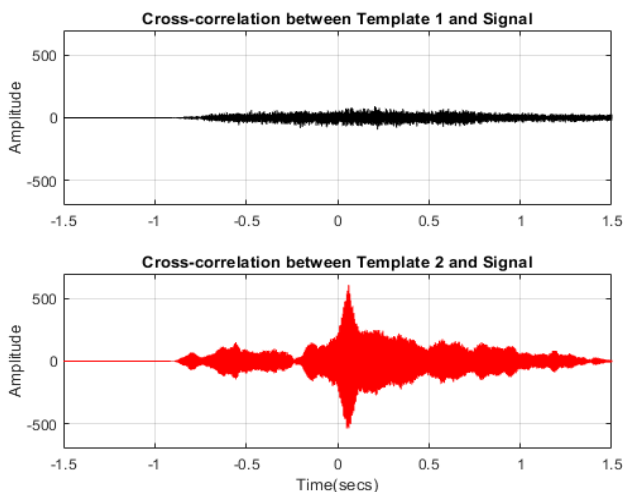
A partir des figures et des programmes associés, identifiez les traitements effectués



```
load relatedsig.mat
```

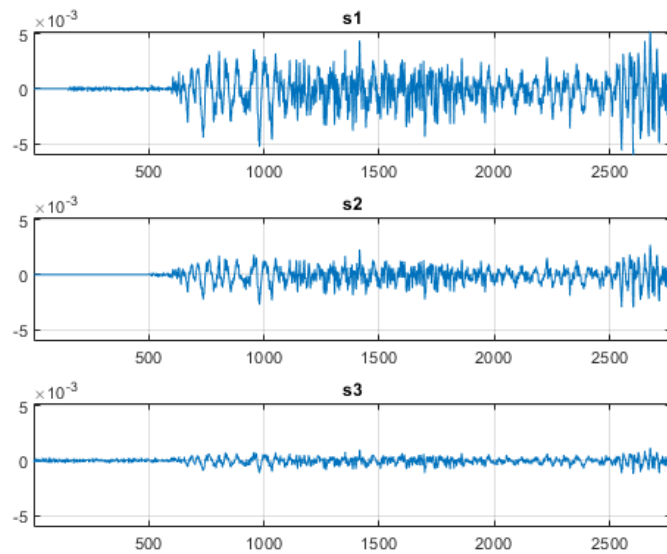
```
figure ax(1) = subplot(3,1,1); plot((0:numel(T1)-1)/Fs1,T1,'k')
ylabel('Template 1') grid on ax(2) = subplot(3,1,2); plot((0:numel(T2)-1)/Fs2,T2,'r')
ylabel('Template 2') grid on
ax(3) = subplot(3,1,3); plot((0:numel(S)-1)/Fs,S) ylabel('Signal')
grid on xlabel('Time (secs)') linkaxes(ax(1:3),'x') axis([0 1.61 -4 4])
[Fs1 Fs2 Fs] → 4096 4096 8192

[P1,Q1] = rat(Fs/Fs1); % Rational fraction approximation
[P2,Q2] = rat(Fs/Fs2); % Rational fraction approximation
T1 = resample(T1,P1,Q1); % Change sampling rate by rational factor
T2 = resample(T2,P2,Q2); % Change sampling rate by rational factor
```



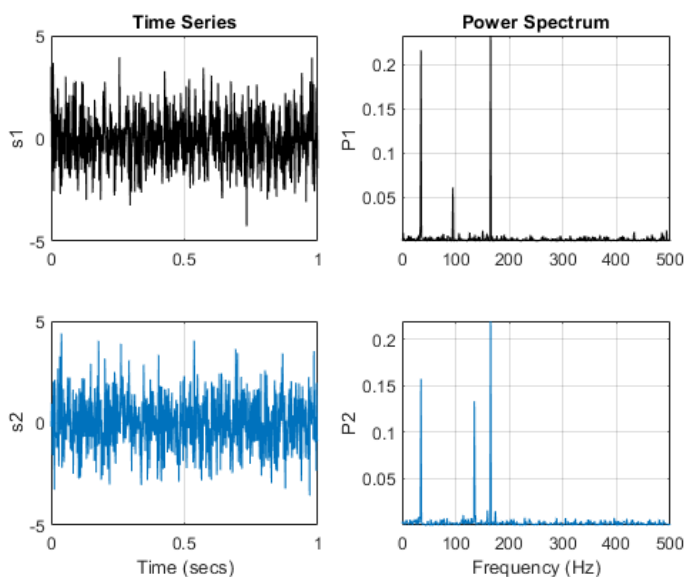
```
[~,I] = max(abs(C2));
SampleDiff = lag2(I)
SampleDiff = 499
timeDiff = SampleDiff/Fs
timeDiff = 0.0609
```

```
[C1,lag1] = xcorr(T1,S);
[C2,lag2] = xcorr(T2,S);
Figure ax(1) = subplot(2,1,1); plot(lag1/Fs,C1,'k') ylabel('Amplitude') grid on
title('Cross-correlation between Template 1 and Signal') ax(2) = subplot(2,1,2);
plot(lag2/Fs,C2,'r')
ylabel('Amplitude') grid on title('Cross-correlation between Template 2 and
Signal') xlabel('Time(secs)') axis(ax(1:2),[-1.5 1.5 -700 700 ])
```

```
s1 = alignsignals(s1,s3);
s2 = alignsignals(s2,s3);

figure
ax(1) = subplot(3,1,1);
plot(s1)
grid on title('s1') axis tight
ax(2) = subplot(3,1,2);
plot(s2)
grid on title('s2') axis tight
ax(3) = subplot(3,1,3);
plot(s3)
grid on title('s3') axis tight
linkaxes(ax,'xy')
```



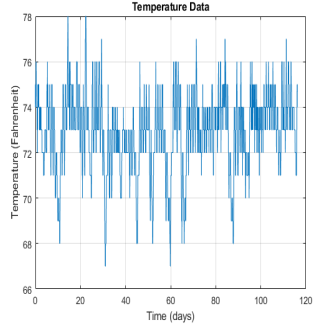
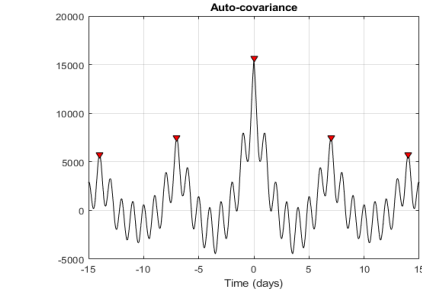
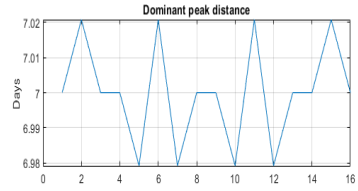
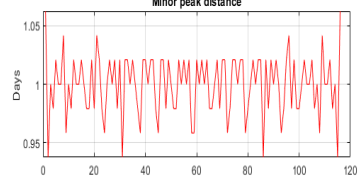
```
figure
t = (0:numel(sig1)-1)/Fs;
subplot(2,2,1)plot(t,sig1,'k')
ylabel('s1')grid on
title('Time Series')
subplot(2,2,3)plot(t,sig2)
ylabel('s2')grid on
xlabel('Time (secs)')
subplot(2,2,2) plot(f1,P1,'k')
ylabel('P1') grid on axis tight
title('Power Spectrum')
subplot(2,2,4) plot(f2,P2)
ylabel('P2') grid on
axis tight xlabel('Frequency (Hz)')
```

Fs = FsSig; % Sampling Rate

```
[P1,f1] = periodogram(sig1,[],[],Fs,'power');
[P2,f2] = periodogram(sig2,[],[],Fs,'power');
```

Sig1 et sug2 ont deux frequences en commun : 35 et 65 Hz

A partir des figures et des programmes associés, identifiez les traitements effectués

		 
<pre>load officetemp.mat Fs = 1/(60*30); % Sample rate is 1 sample every 30 minutes days = (0:length(temp)- 1)/(Fs*60*60*24); figure plot(days,temp) title('Temperature Data') xlabel('Time (days)') ylabel('Temperature (Fahrenheit)') grid on</pre>	<pre>maxlags = numel(temp)*0.5; [xc,lag] = xcov(temp,maxlags); [~,df] = findpeaks(xc,'MinPeakDistance', 5*2*24); [~,mf] = findpeaks(xc); figure plot(lag/(2*24),xc,'k',... lag(df)/(2*24),xc(df),'kv','Mar kerFaceColor','r') grid on xlim([-15 15]) xlabel('Time (days)') title('Auto-covariance')</pre>	<pre>cycle1 = diff(df)/(2*24); cycle2 = diff(mf)/(2*24); subplot(2,1,1) plot(cycle1) ylabel('Days') grid on title('Dominant peak distance') subplot(2,1,2) plot(cycle2,'r') ylabel('Days') grid on title('Minor peak distance')</pre>