traitement de signal

tp3

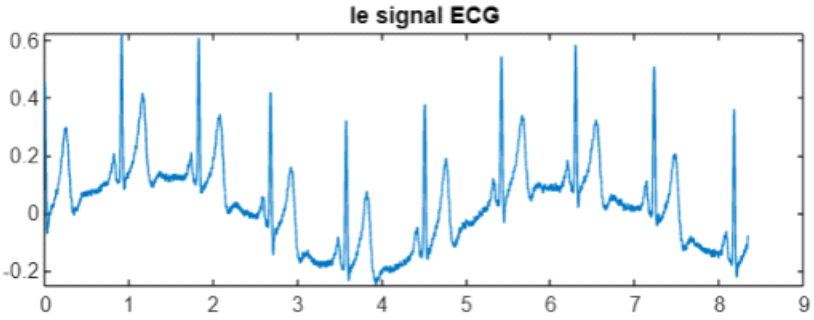
Réalisé par: Ayoub El Anguoud

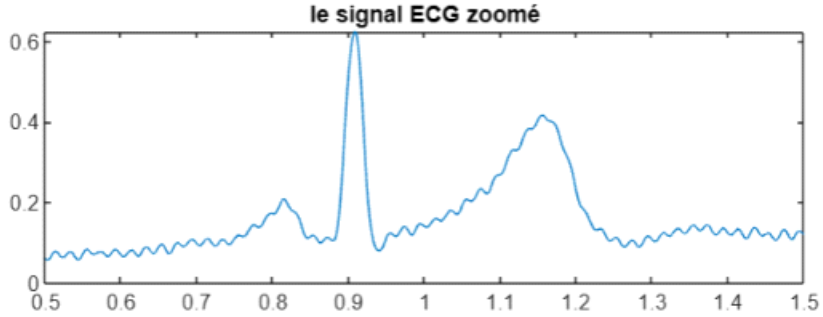
1- on Sauvegarde le signal ECG sur le répertoire, puis on le charge à l'aide de la commande load.

```
load("ecg.mat");
x=ecg;
```

2- le signal a été échantillonné avec une fréquence de 500Hz. on le Trace en fonction du temps.

```
fs=500;
N=length(x);
ts=1/fs;

t=(0:N-1)*ts;
subplot(2,1,1)
plot(t,x);
title("le signal ECG");

%tracer ECG zoomé
subplot(2,1,2)
plot(t,x);
xlim([0.5, 1.5])
title("le signal ECG zoomé");
```





3- afin de supprimer les bruits à très basse fréquence dues aux mouvements du corps,on utilisera un filtre idéal passe-haut.

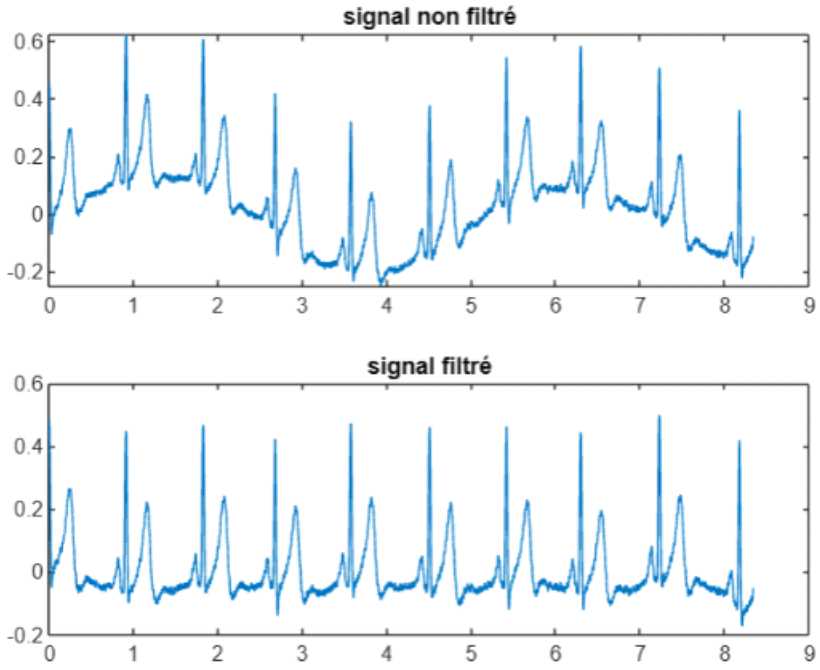
```
 y = fft(x);
 f = (0:N-1)*(fs/N);
 fshift = (-N/2:N/2-1)*(fs/N);

plot(fshift,fftshift(abs(y)))
title("spectre Amplitude")

h = ones(size(x));
fh = 0.5;
index_h = ceil(fh*N/fs);
h(1:index_h)=0;
h(N-index_h+1:N)=0;
ecg1_freq = h.*y;
ecg1 =ifft(ecg1_freq,"symmetric");
```

Pour éliminer les bruits de basse fréquence causés par les mouvements corporels, nous avons utilisé la fonction FFT pour convertir le signal temporel en signal fréquentiel. Ensuite, nous avons créé un filtre passe-haut en mettant à zéro les fréquences inférieures à 0,5 Hz. Enfin, nous avons utilisé la fonction IFFT pour retransformer le signal filtré en signal temporel

4- on trace le nouveau signal ecg1, et noter les différences par

rapport au signal d'origine.

```
subplot(2,1,1)
plot(t,ecg);
title("signal non filtré")
subplot(2,1,2)
plot(t,ecg1);
title("signal filtré")
```



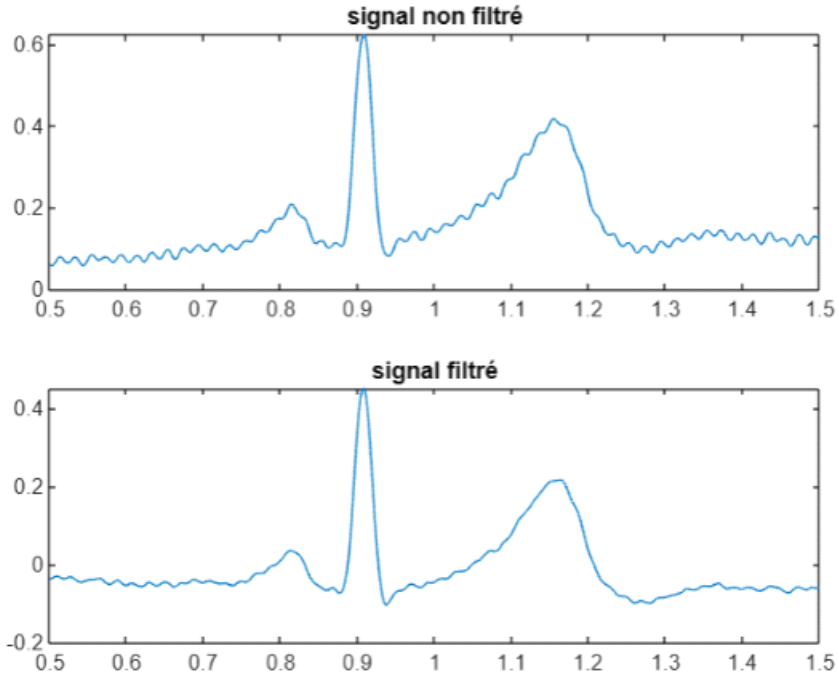
En utilisant un filtre passe-haut pour éliminer les bruits de basse fréquence dans un signal ECG, nous avons constaté une amélioration de la qualité du signal en enlevant les vibrations indésirables ou les bruits de fond.

5- On utilise un filtre Notch idéal pour supprimer une composante spécifique . Les filtres Notch sont conçus pour rejeter une fréquence particulière dans une bande de fréquences donnée

```
Notch = ones(size(x));
fcn = 50;
index_hcn = ceil(fcn*N/fs)+1;
Notch(index_hcn)=0;
Notch(index_hcn+2)=0;
ecg2_freq = Notch.*fft(ecg1);
ecg2 =ifft(ecg2_freq,"symmetric");
```

6. on Visualise le signal ecg2 après le filtrage.

```
subplot(2,1,1)
plot(t,ecg);
xlim([0.5 1.5])
title("signal non filtré")
subplot(2,1,2)
plot(t,ecg2);
title("signal filtré")
xlim([0.5 1.5])
```

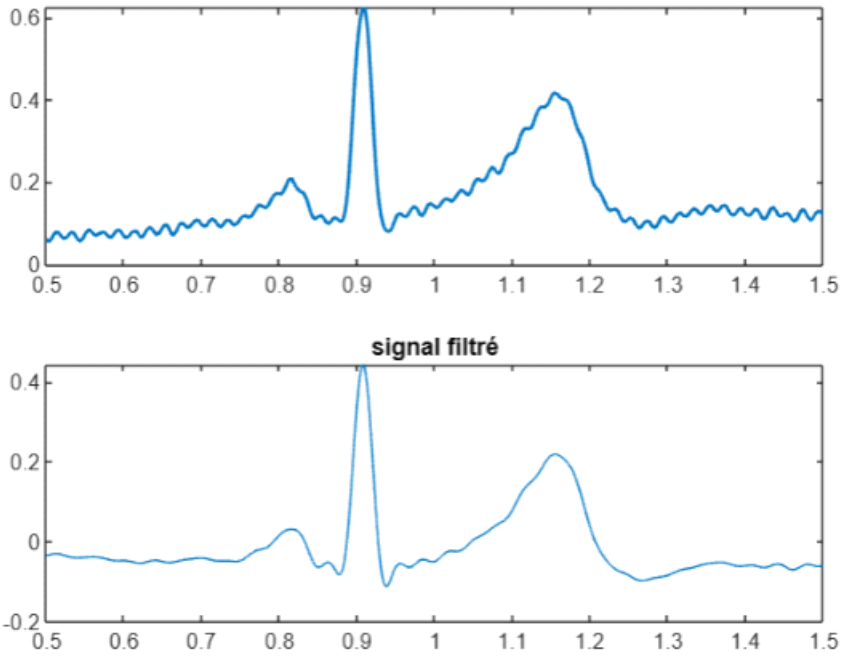


7-On a cherché un équilibre entre la fréquence de coupure pour préserver la forme du signal ECG tout en réduisant au maximum le bruit. On a testé différents choix, puis comparé et commenté les résultats obtenus.

```
pass_bas = zeros(size(x));
fcb = 30;
index_hcb = ceil(fcb*N/fs);
pass_bas(1:index_hcb)=1;
pass_bas(N-index_hcb+1:N)=1;
ecg3_freq = pass_bas.*fft(ecg2);
ecg3 =ifft(ecg3_freq,"symmetric");
```

8-Nous avons visualisé une portion du signal ECG filtré (ecg3) et identifié autant d'ondes que possible dans ce signal, comme décrit dans la section d'introduction.

```
subplot(2,1,1)
plot(t,ecg,"linewidth",1.5);
xlim([0.5 1.5])
subplot(2,1,2)
plot(t,ecg3);
title("signal filtré")
xlim([0.5 1.5])
```



9-On a écrit un programme pour calculer l'autocorrélation du signal ECG, puis pour rechercher automatiquement la fréquence cardiaque. On a utilisé ce programme sur le signal

traité (ecg3 ou ecg2) et sur le signal ECG non traité, en prenant soin de limiter la recherche de la fréquence cardiaque aux plages possibles.

```
[c,lags] = xcorr(ecg3,ecg3);


E = length(c);
Vector = [0];
for n = 1:E
    if c(n) > 10
        Vector(end+1) = c(n);
    end

    M = max(Vector);
    in = find(c == M);
    s = lags(in);
    if s <12
        Vector(Vector == M) = [];
    end
end


frequence = (s/fs)*60;
frequence_min=30;
frequence_max=160;
if frequence > frequence_min & frequence < frequence_max
  fprintf('la freqence cardiaque de ce patient est :%f .',frequence);
end
```

On a utilisé la technique d'autocorrélation pour estimer la fréquence cardiaque d'un signal ECG filtré en utilisant la fonction "xcorr" pour calculer la fonction d'autocorrélation du signal. Ensuite, nous avons utilisé une boucle pour parcourir les valeurs de la fonction d'autocorrélation, enregistrant les valeurs supérieures à 10 dans un vecteur. Nous avons ensuite trouvé la valeur maximale de ce vecteur et identifié son indice dans la fonction d'autocorrélation pour calculer la fréquence cardiaque en utilisant la fréquence d'échantillonnage du signal. Nous avons vérifié que cette fréquence cardiaque se situe dans une plage acceptable (entre 30 et 160 battements par minute) avant de l'afficher.