Esercitazione 5

Medie di insieme per processing. Nel file "cg_o1t.asc" sono contenuti dei segnali di potenziali evocati da stimolo visivo (VEP) ottenuti su diversi soggetti. Ogni file contiene diversi intervalli di 512 campioni privi di artefatti (256 pre- e 256 post- stimolazione) acquisiti a 250 Hz. Tutti gli intervalli sono memorizzati in maniera consecutiva su un file di una singola colonna. I dati hanno già subito un primo pre-processing con un filtraggio passabanda nel range 0.1-70Hz.

Utilizzare i dati forniti per valutare il potenziale evocato medio, plottando il fascio di realizzazioni che entrano nella media sincronizzata e, sovrapposto, il segnale medio ottenuto (in rosso e con uno spessore maggiore).

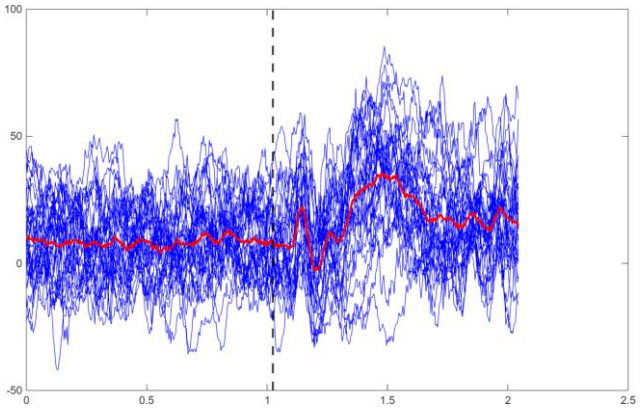
```
load cg_o1t.asc -ascii
Fs=250;
epoch_len=512;

chk=rem(length(cg_o1t), epoch_len); % resto della divisione intera
if chk~=0
    cg_o1t(end-chk+1:end)=[];
end
num_epochs=length(cg_o1t)/epoch_len;

epochs=zeros(30,epoch_len);
for i=1:num_epochs
    epochs(i,:)=cg_o1t(1+(i-1)*epoch_len:i*epoch_len);
end

% oppure
epochs_r=reshape(cg_o1t,[epoch_len, num_epochs])';

figure(1)
sync_avg=mean(epochs_r);
t=(0:epoch_len-1)/Fs;
plot(t,epochs_r','b')
hold on
plot(t,sync_avg,'LineWidth',2,'Color','r')
xline((epoch_len/2)/Fs,'Color','k','LineStyle','--','LineWidth',2)
```



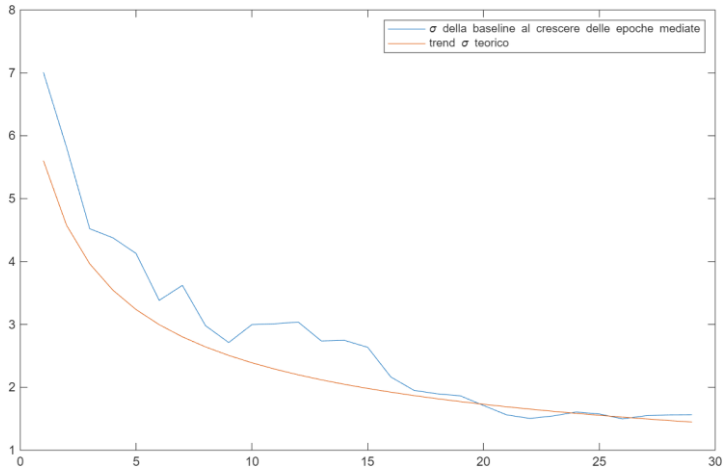
Come vi spiegate il comportamento?

Il rumore può essere assunto come un rumore di fondo casuale che avrà una media (nulla o meno è irrilevante). Se il rumore non è sincrono con il segnale, nelle diverse epoche la media di insieme tende alla media del processo, ovvero a un valore costante. Invece il potenziale evocato si ripete in modo simile tra una realizzazione e l'altra, e quindi la media su diverse epoche tende a coincidere con il potenziale evocato. Pertanto, il segnale rosso è il potenziale evocato + la media del processo causale che rappresenta il rumore di fondo.

Valutare come varia il livello del rumore al crescere del numero di realizzazioni in media sincronizzata.

```
for i=2:num_epochs
    sync_avg(i,:)=mean(epochs_r(1:i,:));
    sigma(i)=std(sync_avg(i,1:(epoch_len/2)));
end
sigma_orig=std(epochs_r(1,1:(epoch_len/2)));

figure
plot(sigma(2:end))
hold on
decr=sigma_orig./sqrt(2:num_epochs);
plot(decr)
legend('\sigma della baseline al crescere delle epoche mediate','trend
\sigma teorico')
```



Esercitazione 6

Scaricare il tool Pink, Red, Blue and Violet Noise Generation with Matlab Versione 2.0.0.0 (7,12 KB) di Hristo Zhivomirov da File Exchange di Matlab.

Caricare il segnale sintetico ecg_sim fornito, che consta di 7 ripetizioni di un segnale pseudoperiodico a 500Hz. Creare un segnale più lungo replicando fino a 10 volte lo spezzone.

```
load ecg_sim.dat -ascii
fs=500; % Hz

% Caricare il segnale sintetico ecg_sim fornito, che consta di 7
% ripetizioni di un segnale pseudoperiodico a 500Hz. Creare un segnale
% più lungo replicando fino a 10 volte lo spezzone.

N=10;
s=repmat(ecg_sim,1,N);
```

Standardizziamo il segnale per renderlo a varianza unitaria.

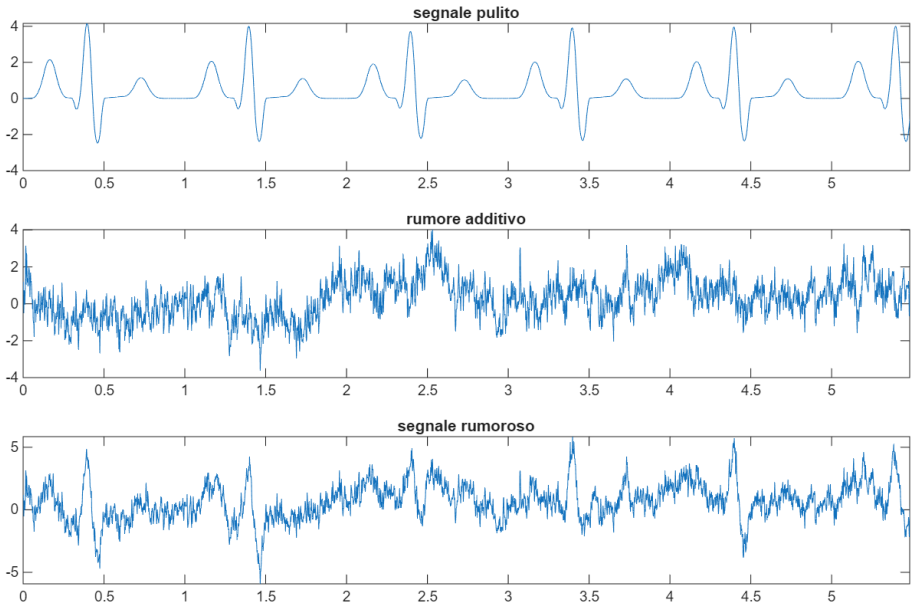
```
s=s/std(s);
```

Iniziamo con il rumore rosa.

```
scale=1;
n=pinknoise(size(s,1),size(s,2))*scale;
x=s+n;

figure
t=(0:length(n)-1)/fs;
pl1a=subplot(3,1,1);
plot(pl1a,t,s)
title('segnale pulito')
pl2a=subplot(3,1,2);
plot(pl2a,t,n)
title('rumore additivo')
pl3a=subplot(3,1,3);
plot(pl3a,t,x)
title('segnale rumoroso')
linkaxes([pl1a pl2a pl3a],'x')
```

L'immagine seguente ci mostra come il rumore rosa, il quale presenta una componente ad alta frequenza ma anche una significativa componente a bassa frequenza, comporti in alcuni punti una riduzione significativa dell'ampiezza dei complessi QRS. Questo dovrebbe focalizzare la nostra attenzione sul fatto che, quando si stima l'SNR di un segnale del genere, è importante non considerare l'ampiezza puntuale del singolo complesso, perché potrebbe essere affetta eccessivamente dal rumore e quindi non rappresentare una stima corretta del livello del segnale.

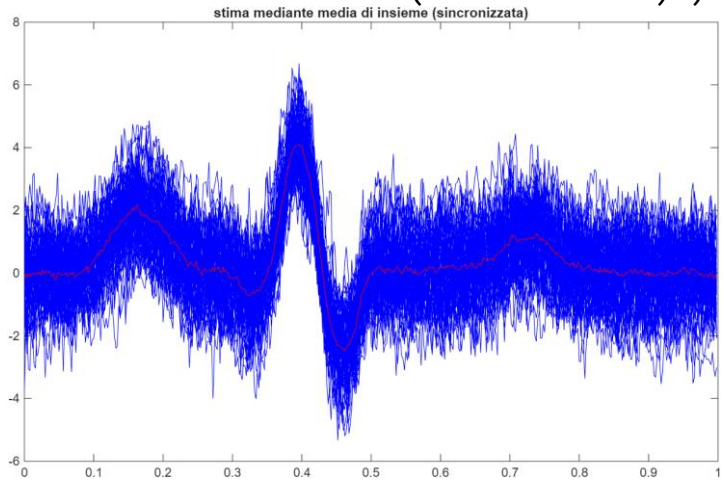


Linkaxes ci permette di collegare gli assi dei 3 subplot in modo che lo zoom sia sempre applicato simultaneamente a tutti i segnali. Si possono linkare solo gli assi x, solo gli assi y, o entrambi. L'opzione più interessante è quella di linkare gli assi x, che rappresentano la variabile indipendente.

Effettuare la media sincronizzata e valutare come varia al variare del numero di battiti messi in media sincronizzata il rapporto segnale-rumore.

Per fare questo, trasformiamo il segnale in una matrice che riporta un battito in ogni riga, ed effettuiamo la media di insieme. In questo caso, ogni battito è lungo 500 campioni esatti (non c'è variabilità) quindi il passaggio è molto facile. Dapprima vediamo la media di insieme complessiva:

```
xm=reshape(x,500,length(x)/500)';
figure
t=(0:500-1)/fs;
x_sync=mean(xm);
plot(t,xm','b')
hold on
plot(t,x_sync,'r')
title('stima mediante media di insieme (sincronizzata)')
```

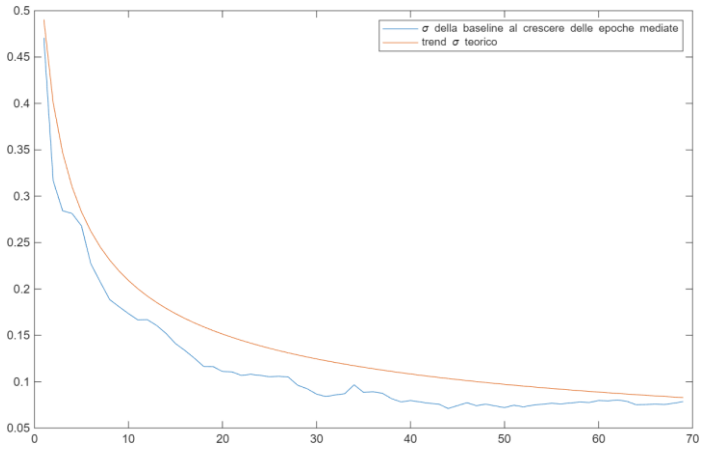


A questo punto, per vedere cosa succede al variare del numero di battiti messi in media, dobbiamo itearare con diverse medie sincronizzate. Salviamo la singola media sincronizzata ma soprattutto la deviazione standard del rumore. Per stimarla, dalla figura sopra vediamo che l'ultimo 10% del segnale è di fatto un'isoelettrica, quindi possiamo catturare gli ultimi 50 campioni della media di insieme effettuata ogni volta con M epoche (M variabile):

```
M=size(xm,1);
for i=2:M
    sync_avg(i,:)=mean(xm(1:i,:));
    sigma(i)=std(sync_avg(i,end-50:end));
end
sigma_orig=std(xm(1,end-50:end));
```

A questo punto, possiamo plottare come varia la sigma al variare di M, in relazione a come dovrebbe variare teoricamente:

```
figure
plot(sigma(2:end))
hold on
decr=sigma_orig./sqrt(2:M);
plot(decr)
legend('\sigma della baseline al crescere delle epoche mediate','trend
\sigma teorico')
```



L'andamento è quello previsto, con la variabilità imposta dal fatto che si tratta di un processo casuale reale (il rumore) e quindi l'andamento teorico è, appunto, solo teorico e viene rispettato in parte.

Valutare l'SNR del segnale usando il rumore noto per valutare la sua potenza, oppure il terzo centrale fra i primi due picchi R del segnale rumoroso. Per trovare i picchi, usare la funzione findpeaks, mentre per trovare l'ampiezza picco-picco calcolare il minimo intorno al picco R in una finestra di 200ms. App sia la media dell'ampiezza dell'R – la media dell'ampiezza dell'S.

```
[p,loc]=findpeaks(s,fs,'MinPeakDistance',0.8);
lp=nan(size(loc));
for i=1:numel(loc)
```

```
    lp(i)=min(s(loc(i)*fs-round(0.1*fs):loc(i)*fs+round(0.1*fs)));
end
App=mean(p)-mean(lp);

disp('SNR reale del segnale rumoroso')
SNR=10*log10((App^2)/(8*var(n)))

disp('SNR stimato del segnale rumoroso')
[p,loc]=findpeaks(x,fs,'MinPeakDistance',0.8);
RR1=(loc(2)-loc(1))*fs;
v=var(x(loc(1)*fs+round(RR1/3):loc(2)*fs-round(RR1/3)));
SNR=10*log10((App^2)/(8*v))

>>
SNR reale del segnale rumoroso
SNR = 6.9885
SNR stimato del segnale rumoroso
SNR = 8.5090
```

Attenzione che i risultati di sopra sono puramente casuali. Si noti come la stima su un singolo battito dell'ampiezza picco-picco e su un singolo intervallo della potenza del rumore possa essere molto fuorviante: in questo caso, per esempio, sembrerebbe che il rapporto segnale-rumore sia maggiore di quello effettivo, esclusivamente perché, per puro caso, il rumore locale sommato al segnale porta ad un QRS apparentemente più ampio (attenzione, i dati sono relativi a simulazioni diverse, quindi non considerarli perfettamente associati alle figure precedenti, essendo il rumore casuale da un run all'altro).

Provare a cambiare metodo per la stima dell'SNR.

```
SNR1=20*log10(App/(4*std(n)))
SNR2=20*log10(std(s)/std(n))
SNR1 =
    3.9782
SNR2 =
  -6.8467e-14
```

La prima è giustamente minore della precedente, perché il denominatore è molto più grande (prima era $2\sqrt{2} = 2{,}4$ se riferito alla deviazione standard), la seconda ci darà 0dB avendo standardizzato il segnale ed essendo il rumore a varianza unitaria (anche qui, non aspettiamoci che, numericamente, ci venga realmente zero).

Valutare l'SNR prima e dopo l'elaborazione. Lo abbiamo già valutato a monte, ora vediamo a valle del processing:

```
disp('SNR stimato del segnale stimato')
[p,loc]=findpeaks(x_sync,fs,'MinPeakDistance',0.8);
lp=min(x_sync(loc(1)*fs-round(0.1*fs):loc(1)*fs+round(0.1*fs)));
App=p(1)-lp;
Pnoise=var(x_sync(1,end-50:end));
```

```
SNR=10*log10((App^2)/(8*v))
SNR stimato del segnale stimato
SNR =
    8.7324
```

L'SNR stimato è maggiore di quello del segnale rumoroso originale, ma non dissimile da quello stiamo male su un solo battito (ecco perché è importante fare attenzione).

Valutare l'effetto in presenza di rumore AWGN e Brown.

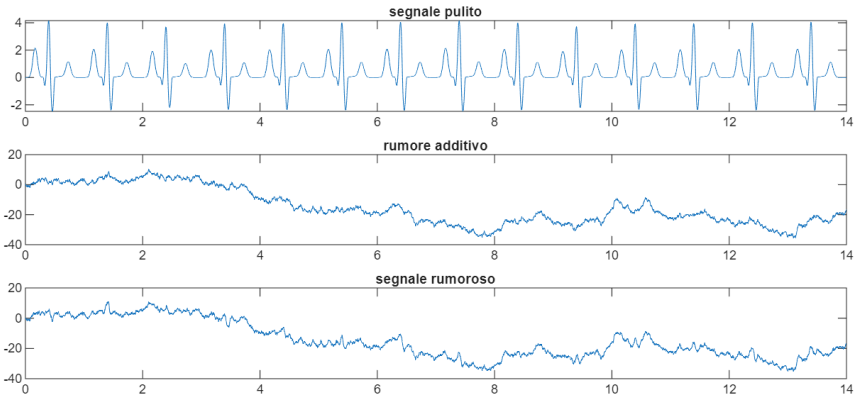
```
pab=input('Inserire il rumore desiderato (1:pink, 2: AWGN, 3: Brown): ');
if pab<1 || pab>3
    error('Rumore inesistente')
end
switch pab
    case 1
        n=pinknoise(size(s,1),size(s,2))*scale;
    case 2
        sigma=(App/20);
        n=wgn(size(s,1),size(s,2),sigma.^2,'linear');
    case 3
        sigma=(App/20);
        n=wgn(size(s,1),size(s,2),sigma.^2,'linear');
        n=cumsum(n);
end

x=s+n;

figure
t=(0:length(n)-1)/fs;
pl1a=subplot(3,1,1);
plot(pl1a,t,s)
title('segnale pulito')
pl2a=subplot(3,1,2);
plot(pl2a,t,n)
title('rumore additivo')
pl3a=subplot(3,1,3);
plot(pl3a,t,x)
title('segnale rumoroso')
linkaxes([pl1a pl2a pl3a],'x')
```

Poi ovviamente si dovrà ripetere il processo già visto in precedenza.

Si può osservare come in presenza di rumore significativo in bassa frequenza (Brown) la stima sia scorretta. Questo dipende dal fatto che il processo non è localmente a media nulla e che il rumore a bassa frequenza rende molto diverse differenti realizzazioni del segnale.


segnale pulito
rumore additivo
segnale rumoroso