a.a. 2020/21

Fabs :)

Elaborazione di daTI   
e segnali biomedici - matlab

Silvia Francesca Storti

# NOTA

Questi appunti/sbobinatura/versione “discorsiva” delle slides sono per mia utilità personale,  
 quindi pur avendole revisionate potrebbero essere ancora presenti typos, commenti/aggiunte personali (che anzi, lascio di proposito) e nel caso peggiore qualche inesattezza!

Comunque spero siano utili! 🌸✨

**Questa sbobina fa parte della mia collezione di sbobinature,   
che è disponibile (e modificabile!) insieme ad altre in questa repo:**<https://github.com/fabfabretti/sbobinamento-seriale-uniVR>

Table of Contents

[NOTA 1](#_Toc107249843)

[Trasformata di Fourier (DFT) 3](#_Toc107249844)

[Lecture #6 : Funzioni relative a sistemi LTI e filtri (lecture 6) 4](#_Toc107249845)

[Lecture #6: Utilizzo e sintesi di filtri 5](#_Toc107249846)

[Lab #3: EEGlab – Analisi completa via GUI 12](#_Toc107249847)

[Lab #4: EEGlab – Analisi completa via scripting 17](#_Toc107249848)

[Lab #5: FSL e FSLeyes – elaborazione di mappe statistiche via GLM 19](#_Toc107249849)

# Trasformata di Fourier (DFT)

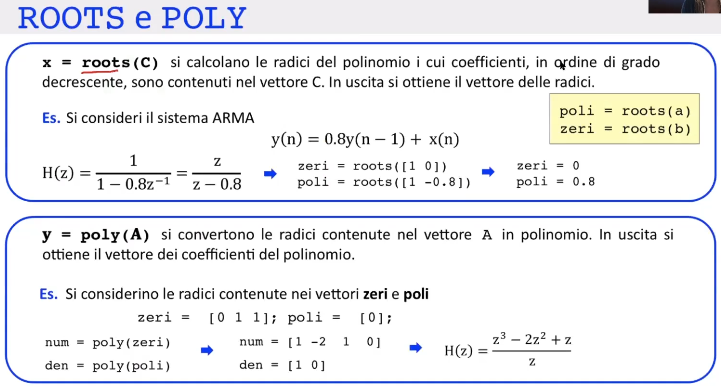
L’intervallo che distanzia due campionamenti è l’intervallo di campionamento, e il suo reciproco la frequenza di campionamento.

In matlab esiste solo la trasformata discreta (obv non possiamo lavorare su sequenze infinite in un calcolatore!)

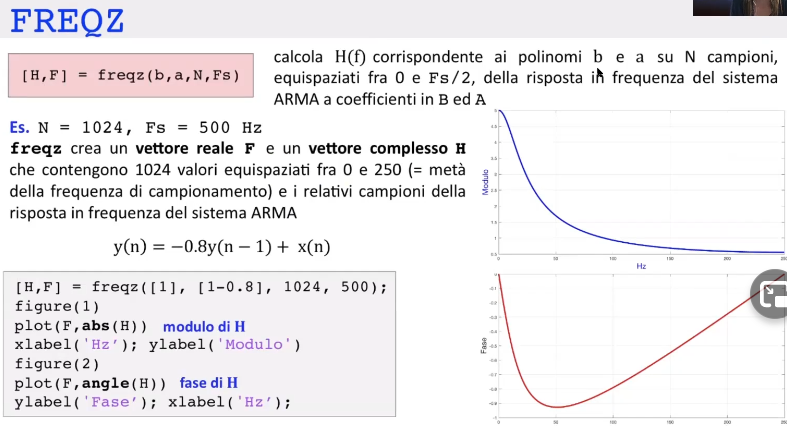
|  |  |
| --- | --- |
|  | Poi tradurremo questo segnale in un segnale in frequenza, ovvero una figura che rappresenta a trasformata di fourier e quindi il contenuto spettrale.  N = length(signal); -> numero di campioni  Fs = 256; -> frequenza di campionamento  L = N/Fs; -> finestra di osservazione (in secondi)  T = (0 : 1/Fs : L-1/Fs);  FTx=fft(signal,N) -> prende il segnale e il numero di campioni  S=(abs(FTx).^2)/N -> formula della potenza spettrale  Anche qui non vogliamo plottare “da solo” ma mettere su x il vettore delle frequenze f\_FT = (0:Fs/N:Fs-Fs/N) Non è necessario plottarlo tutto: lo spettro è a specchio, quindi posso prelevare solo la metà dei campioni  S=s(1:N/2) f\_ft=f\_ft(1:N/2) plot(f\_FT,s) |

# Lecture #6 : Funzioni relative a sistemi LTI e filtri (lecture 6)

|  |  |
| --- | --- |
|  | Usiamo la funzione filter per creare e applicare un filtro. Dato in input vettori coi coefficienti + segnale di input, butta fuori il segnale filtrato.  Usiamo Filter anziché Conv in quanto Conv. |



La funzione root estrude le soluzioni dai coefficienti ak e bk.Attenzione a metterli in ordine decrescente!



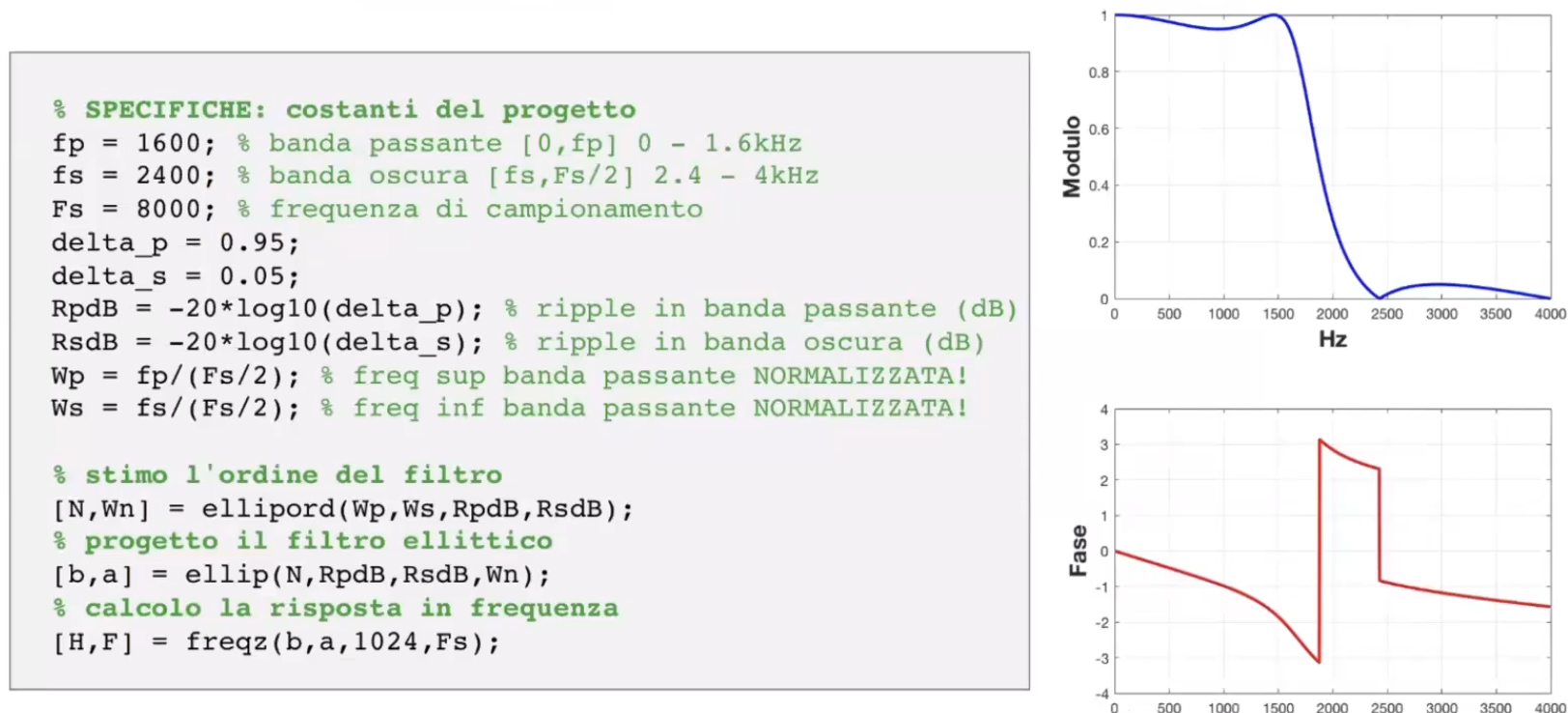
La funzione freqz permette di calcolare la risposta in frequenza dell’ARMA dando campioni e coefficienti. Restituisce un dato H che permette di estrarre modulo e fase della risposta in frequenza, usando rispettivamente **abs** e **angle**.

# Lecture #6: Utilizzo e sintesi di filtri

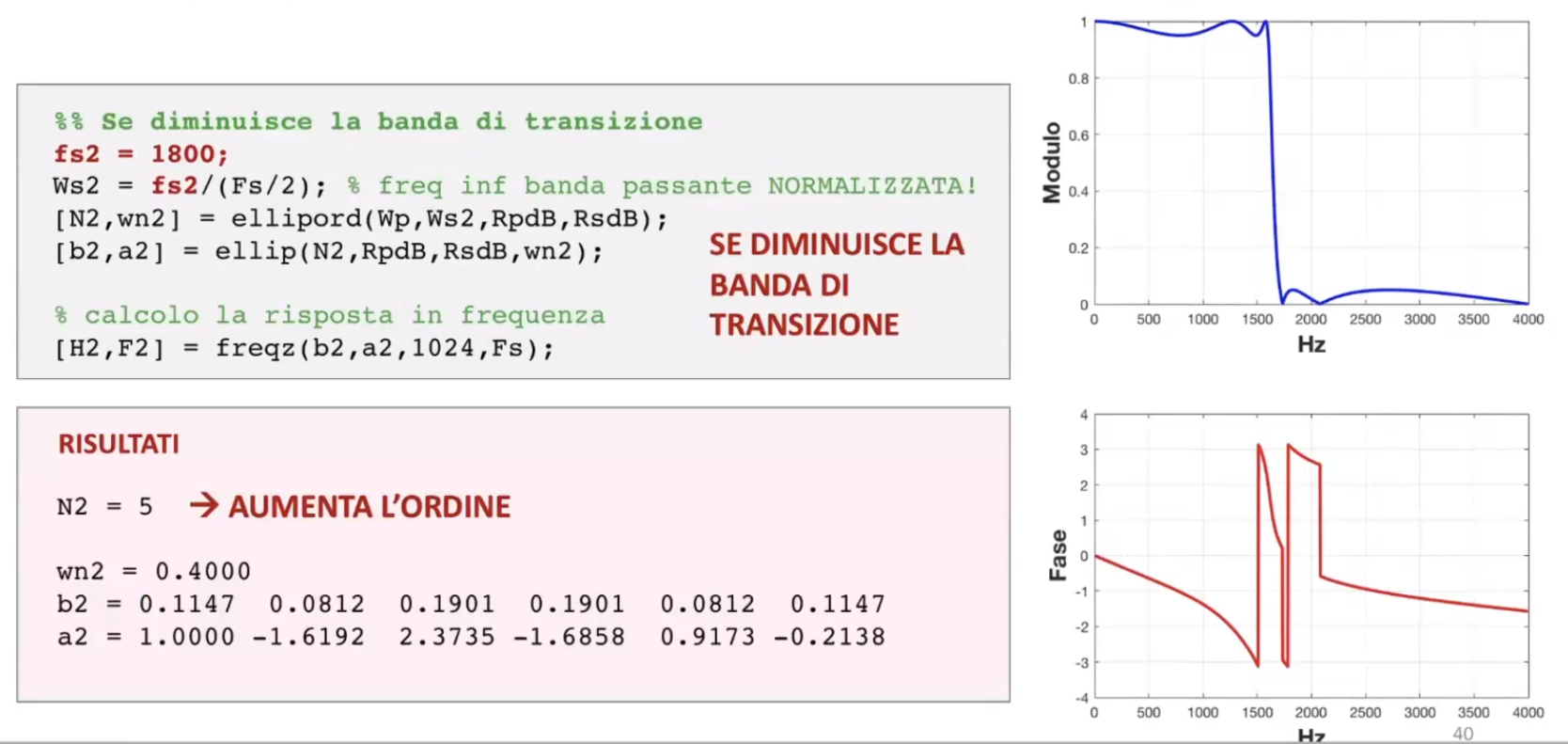
1. Sintesi filtro di butterworth
   1. Definisco le specifiche del filtro analogico*: ωp, ωs, rp, rs*
   2. Cerco l’ordine minimo *n* ideale da utilizzare nel filtro per rendere l’errore fra filtro ideale e reale, usando la funzione **buttord**
      * **Input**: banda passante, banda oscura, ripple in banda passante, ripple in banda oscura.
      * **Output**: ordine del modello, frequenza di taglio naturale (=mi fa abbassare il tutto di 3 dB)
   3. Applico la funzione **butter** per ottenere i parametri dell’equazione alle differenze.  
      
      * **Input**: ordine e frequenza di taglio naturale (calcolati da buttord)
      * **Output**: vettori dei coefficienti dell’eq alle differenze del filtro.
   4. Visualizzo l’approssimazione del filtro analogico
2. Sintesi filtro di chebyshev
   1. Definisco le specifiche del filtro analogico*: ωp, ωs, rp, rs*
   2. Cerco l’ordine minimo *n* ideale da utilizzare nel filtro per rendere l’errore fra filtro ideale e reale, usando la funzione **cheb1ord**
      * **Input**: banda passante, banda oscura, ripple in banda passante, ripple in banda oscura.
      * **Output**: ordine del modello, frequenza di taglio naturale, che è un polinomio
   3. Applico la funzione **cheby1** per ottenere i parametri dell’equazione alle differenze.  
      
      * **Input**: ordine e frequenza di taglio naturale (calcolati da cheb1ord), ripple passante
      * **Output**: vettori dei coefficienti dell’eq alle differenze del filtro.
   4. Visualizzo l’approssimazione del filtro analogico
3. Sintesi filtro ellittico
   1. Definisco le specifiche del filtro analogico*: ωp, ωs, rp, rs*
   2. Cerco l’ordine minimo *n* ideale da utilizzare nel filtro per rendere l’errore fra filtro ideale e reale, usando la funzione **ellipord. Posso anche non utilizzarla e fornirlo manualmente secondo le specifiche.**
      * **Input**: banda passante, banda oscura, ripple in banda passante, ripple in banda oscura.
      * **Output**: ordine del modello, frequenza di taglio naturale, che è un polinomio
   3. Applico la funzione **cheby1** per ottenere i parametri dell’equazione alle differenze.  
      
      * **Input**: ordine e frequenza di taglio naturale (calcolati da ellipord), ripple passante
      * **Output**: vettori dei coefficienti dell’eq alle differenze del filtro.
   4. Visualizzo l’approssimazione del filtro analogico

### Esempio – Esercizio 1

(delta\_p e delta\_s sono i ripple massimi; dovrò convertirli in decibel)  
(Le frequenze di taglio andranno normalizzate)  
 Otteniamo un ordine ottimo = 3; il filtro è il filtro migliore possibile con le mie richieste.

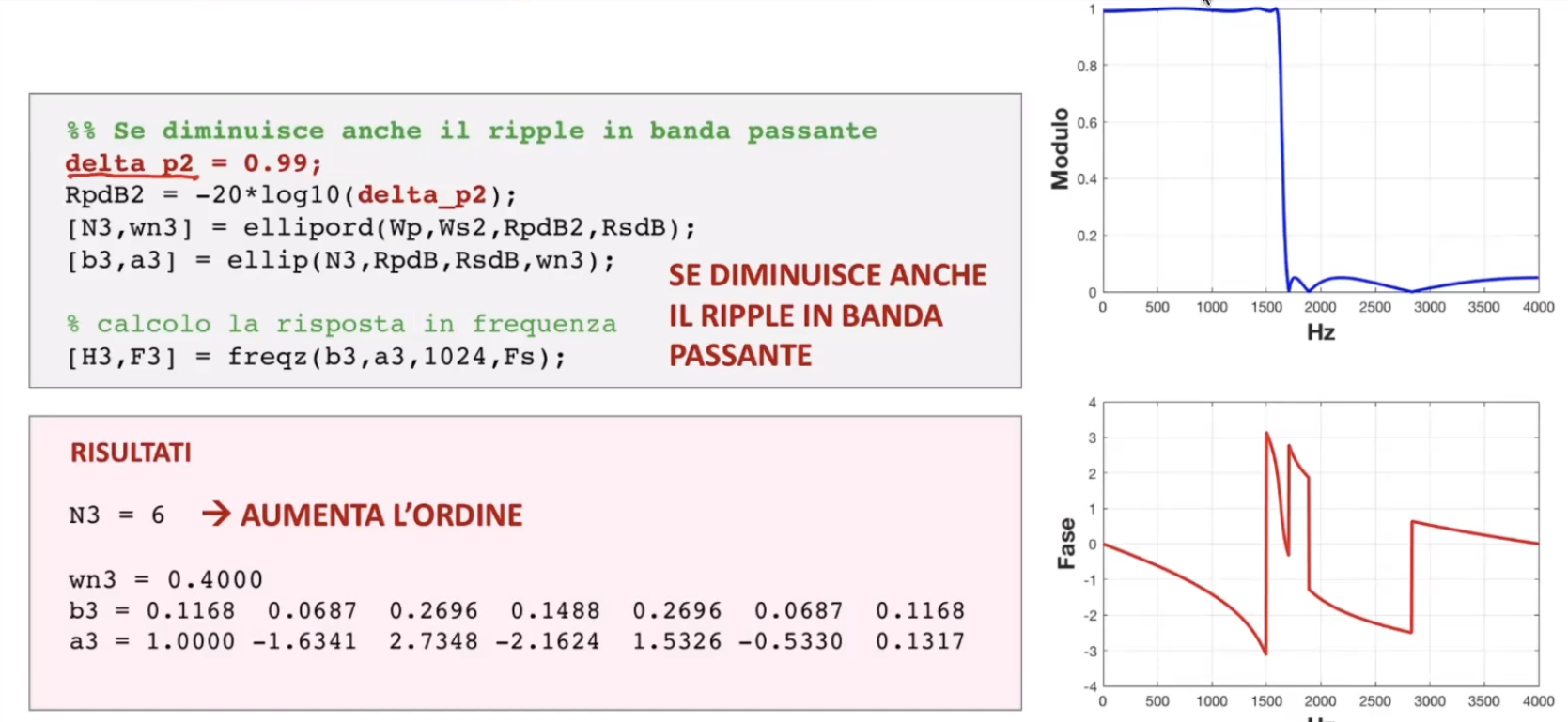


Proviamo a modificare la banda di transizione, stringendola:



L’ordine del modello aumenta, e aumentano sempre di più le oscillazioni in banda passante e oscura. **In generale se voglio diminuire la transizione devo aumentare l’ordine.**

Proviamo a ricalcolare per diminuire il ripple:

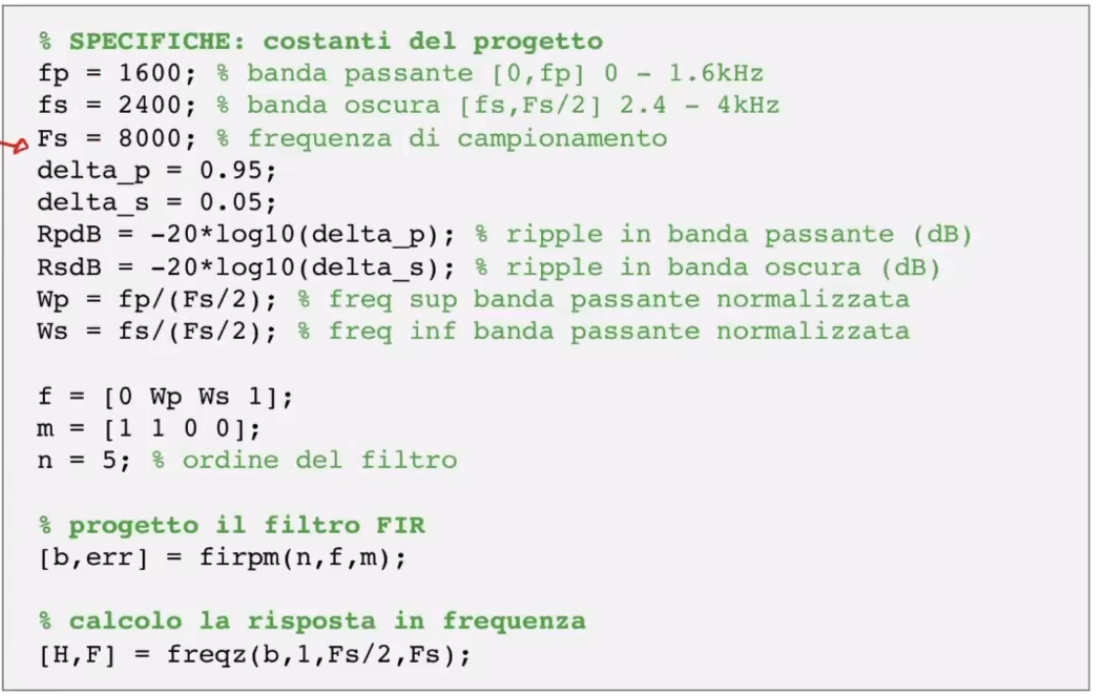


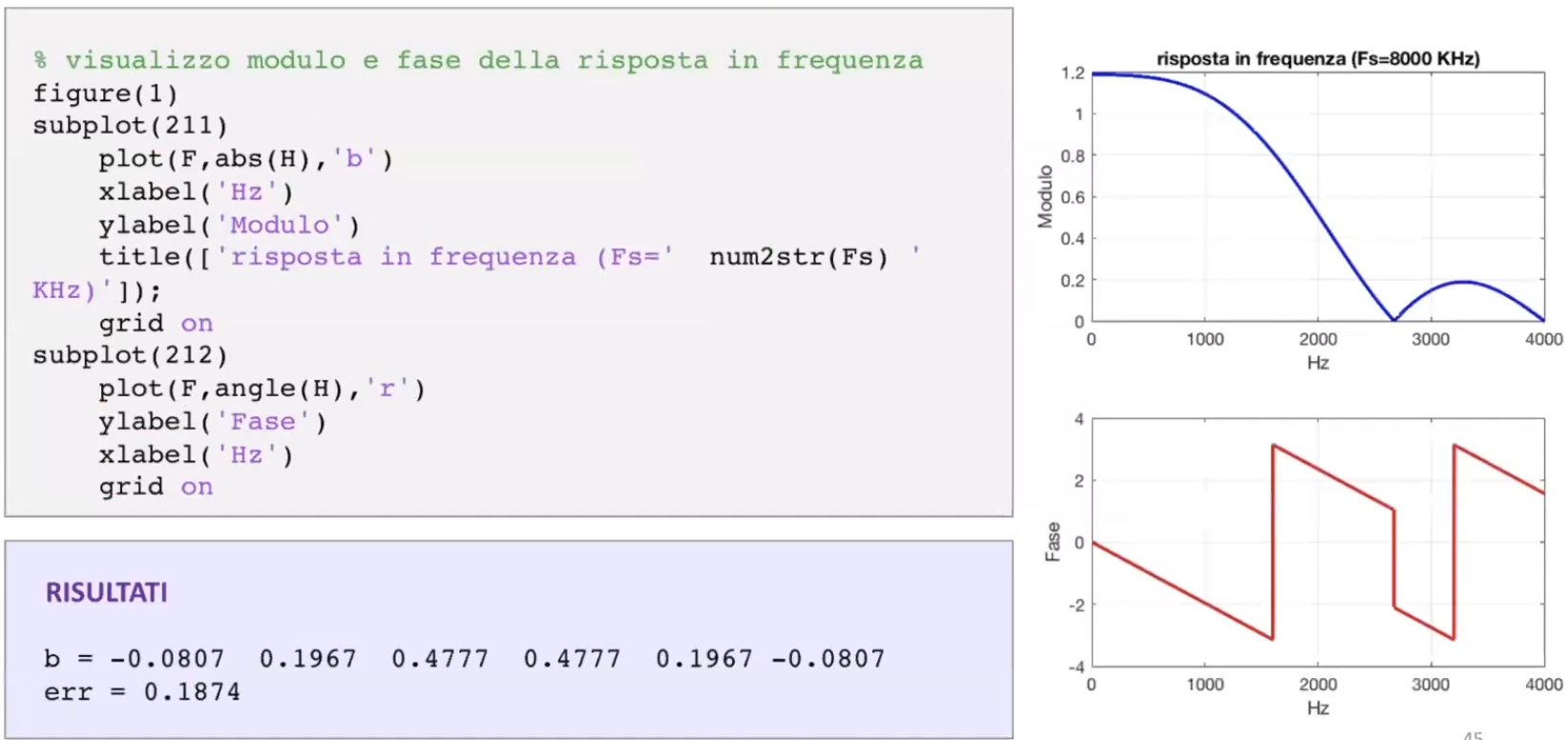
Anche qui aumenta l’ordine! Questo comporta un peggioramento della fase.

1. Sintesi filtro PArks McClellan
   1. Definisco le specifiche del filtro analogico*: ωp, ωs, rp, rs*
   2. Cerco l’ordine minimo *n* ideale da utilizzare nel filtro per rendere l’errore fra filtro ideale e reale, usando la funzione **firpmord** o imponendolo noi**.**   
      
      * **Input**: f frequenza di taglio, a ampiezze desiderate, dev = [(10^(rp/20)-1)/(10^(rp/20)+1) 10^(-rs/20)] con rp passband ripple in dB e rs stopband ripple in dB.
      * **Output**: ordine del modello, frequenza normalizzata fo, ampiezza della banda in frequenza ao, pesi .
   3. Applico la funzione **firmp** per ottenere i parametri dell’equazione alle differenze.  
        
      Ha diverse opzioni possibili, fra cui quella di restituire un parametro err che fornisce l’errore massimo.
      * **Input**: ordine calcolato da firpmord, un vettore f che contiene le frequenze[ 0 *ωp, ωs, pi]*, vettore a che contiene le ampiezze [1 1 0 0 ] in
      * **Output**: vettore dei coefficienti dell’eq alle differenze del filtro.
   4. Visualizzo l’approssimazione del filtro analogico

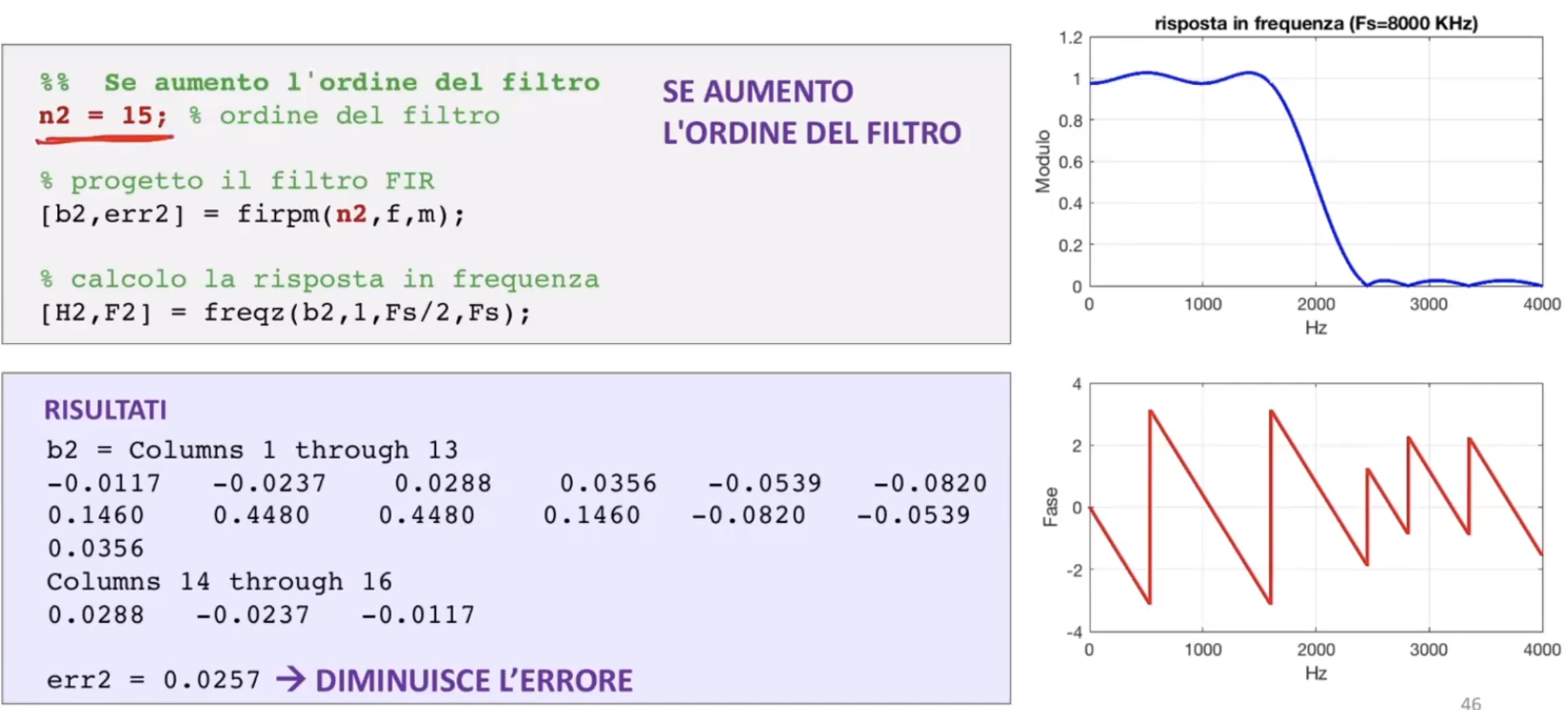
### Esempio – Esercizio 2

In questo caso abbiamo scento manualmente l’ordine del modello.

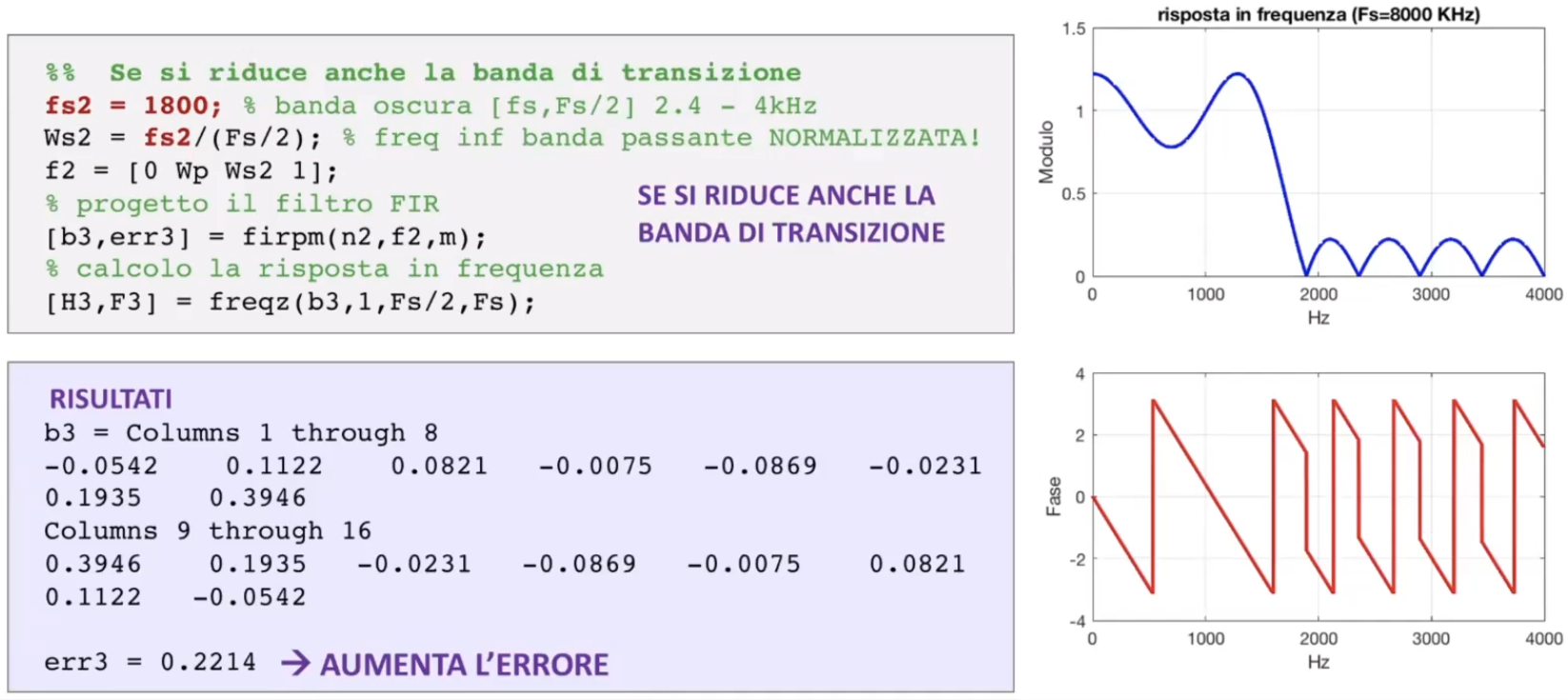


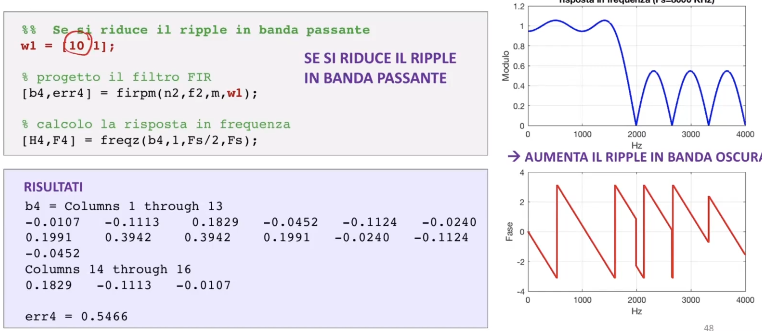


Se provo ad aumentare l’ordine del filtro noto che diminuisce l’errore.



Se provo a diminuire la banda di trasmissione, a parità di ordine del filtro noto un aumento dell’errore (per farla venire più ripida devo generare più ripple tipo:



Provo a dare più peso ai ripple in banda passante (= lo riduco di più) che a quelli in banda oscura:

Aumenta il ripple in banda oscura! Vale il viceversa.

### Filtraggio F-B (=senza cambio di fase)

|  |  |
| --- | --- |
|  | È sufficiente utilizzare la funzione filtfilt. |

#### Esempio – Esercizio 3

Applicare il filtro FIR  alla traccia di un ECG fornito [DatiEs3\_ecg\_hf\_calgaryuni.mat]

|  |  |  |  |
| --- | --- | --- | --- |
| 1. **Caricare il segnale ECG e visualizzarlo** | Fs = 1000 Hz   |  |  | | --- | --- | |  |  | |
| 1. **Scrivere la funzione di trasferimento** e **verificare la stabilità del sistema** | I poli sono tutti in 0 e hanno quindi modulo < 1 → SISTEMA STABILE. |
| 1. **Valutare modulo e fase della risposta in frequenza** | !!! ATTENZIONE: Freqz prende il vettore dei coefficienti valutati come rapporto di polinomi in z^-1. Roots, invece, vuole polinomi in z.    → Notiamo che il filtro amplifica le frequenze basse e attenua le frequenze alte. Si annulla esattamente a 200 e 400 Hz. |
| 1. Capire dalla risposta in frequenza il **motivo della risposta nulla** (a regime) al segnale sinusoidale per certe frequenze critiche, tipo 200 Hz. | |  |  | | --- | --- | |  | Consideriamo la frequenza critica Fn = 200Hz.  Se la passiamo attraverso il filtro succede l’impensabile:    È colpa dei ripple del filtro che ci mandano a 0 in quella frequenza :( | |
| 1. **Applicare il filtro all’ECG** e confrontare i segnali in ingresso e uscita | È d’uopo, per abitudine, togliere la baseline(media) dal segnale d’ingresso.   |  |  | | --- | --- | |  |  | |
| 1. **Ottenere segnale filtrato con guadagno unitario** | Notiamo a destra che le scale di ampiezza del segnale sono differenti: poiché il **guadagno al centro della banda passante è pari a 5**, l’uscita varia su una scala maggiore [+10 -10] dell’ingresso [-2 +2]. Per avere guadagno unitariobasta moltiplicare tutto per 0.2.   |  |  | | --- | --- | |  |  | |
| 1. Implementiamo il **filtro in versione F-B** e confrontiamo col filtro normale | Tutto uguale ma uso filtfilt anziché filter.   |  |  | | --- | --- | |  |  |   **Valutazione nel tempo** Capire esattamente la differenza con quello di prima può essere complesso; analizziamo la performance del filtro filter vs filtfilt prelevando una piccola finestra e zoommandola.     |  |  | | --- | --- | |  | Notiamo che il filtro filtfilt evita che il segnale filtrato risulti ritardato rispetto all’originale. |   **Valutazione nella risposta in frequenza**  Calcoliamo la risposta in frequenza come prodotto fra H e il suo coniugato. Notiamo la fase lineare del filtfilt. |

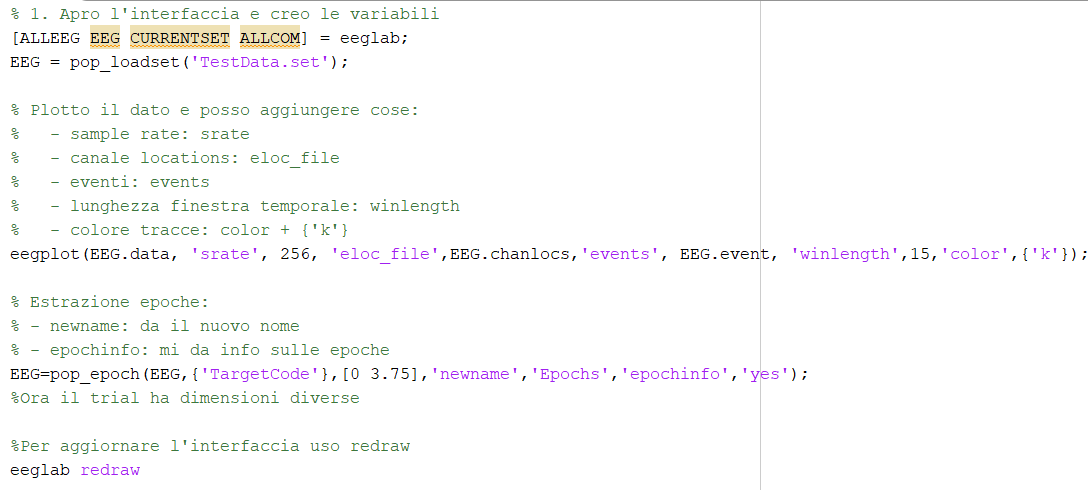
https://univr.cloud.panopto.eu/Panopto/Pages/Viewer.aspx?id=99a7c76e-aa0f-4125-8ac4-ac8700e75279

# Lab #3: EEGlab – Analisi completa via GUI

#### Es.1 : Estrarre epoche da segnale simulato.

Segnale simulato vuol dire che si parte da un segnale di fondo a cui si aggiunge una componetne a 10 Hz su alcuni canali (in questo caso solo il C3, quinta traccia, su alcune porzioni). Il file ha una matrice di 16x46080 canali, dove 16 è il numero di canali e il resto il numero di campioni. La frequenza di campionamento è 256 Hz

|  |  |
| --- | --- |
| **Estraiamo le epoche**  Tools → Extract epochs | * + Time-locking event type: è un’impostazione che ci viene data   + Epoch limits: determina gli intervalli di tempo da amntenere prima e dopo il marker selezionato (noi mettiamo 0,3.75 quindi partiamo dal marker e guardiamoo i successivi 3.75 secondi)     - Name: sarà un nuovo dataset in cui sono presenti i nuovi trial.   + Epoch limits: possiamo specificare dove c’è la parte di pausa pre-task; serve a rimuovere la baseline, ovvero la media del segnale da ogni singola epoca Baseline: è l’attività di base di riferimento. Qui sono millisecondi!! Con lo script possiamo anche toglierla da tuttoil segnale anziché solo da una eèpca-     - Fa la media dalla zona a riposo e la rimuove dall’epoca * Questo genera un nuovo dataset, separato dal primo. Questo ci permette di avere un versionamento del dataset e di tornare al tracciato grezzo. |
| **Visualizzazione plot**  Plot→channel data scroll | * Ora che abbiamo estratto le epoche, la visualizzazione via mostrerà il tempo diviso in epoche (e non solo in ms).   Osservando le epoche noto che solo alcune epoche hanno la modulazione in banda alfa. |
| **Distinzione epoche**  Edit→select epoch or event | Abbiamo delle epoche in posizione 1 o 2;  1= no attività  2= sì attività  Le salvo in due dataset separati (quindi quando estraggo i 2 devo tornare a lavorare sul dataset intero)   |  |  | | --- | --- | | 1: solo epoche no banda alfa | 2: solo epoche con banda alfa | |
| **Analisi in frequenza**  Plot→channel spectra and maps | * Siamo interessati solo alla parte eseguita, quindi vogliamo eliminare il pretrial. * Le mappe non sono normalizzate, ma mettendo il limite con ‘maplimits’ posso dare lo stesso limite ad entrambe e renderle confrontabili.      |  |  | | --- | --- | | Epoche 1 (banda alfa): | Epoche 2 (no banda alfa): | |

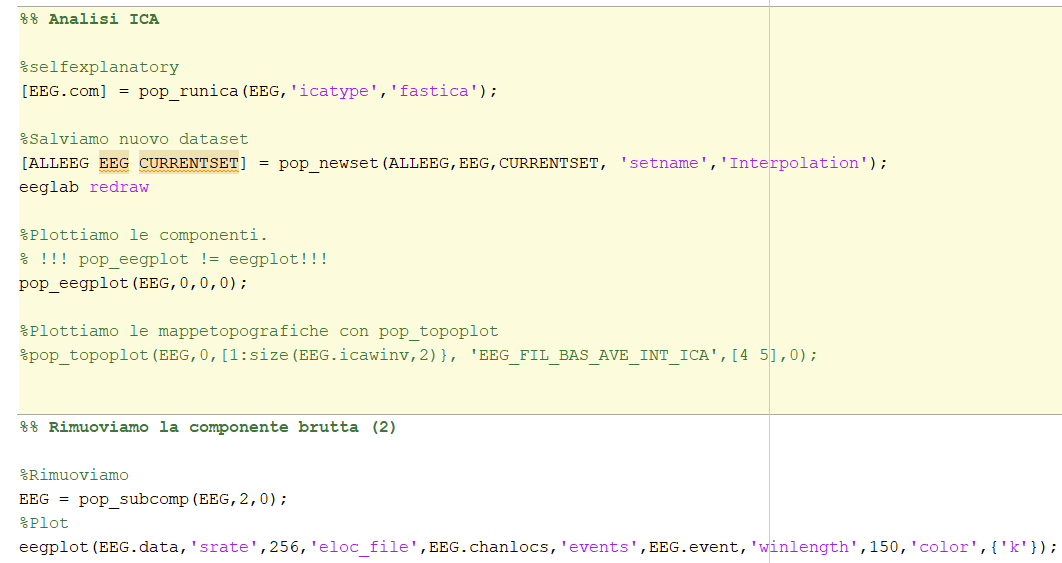
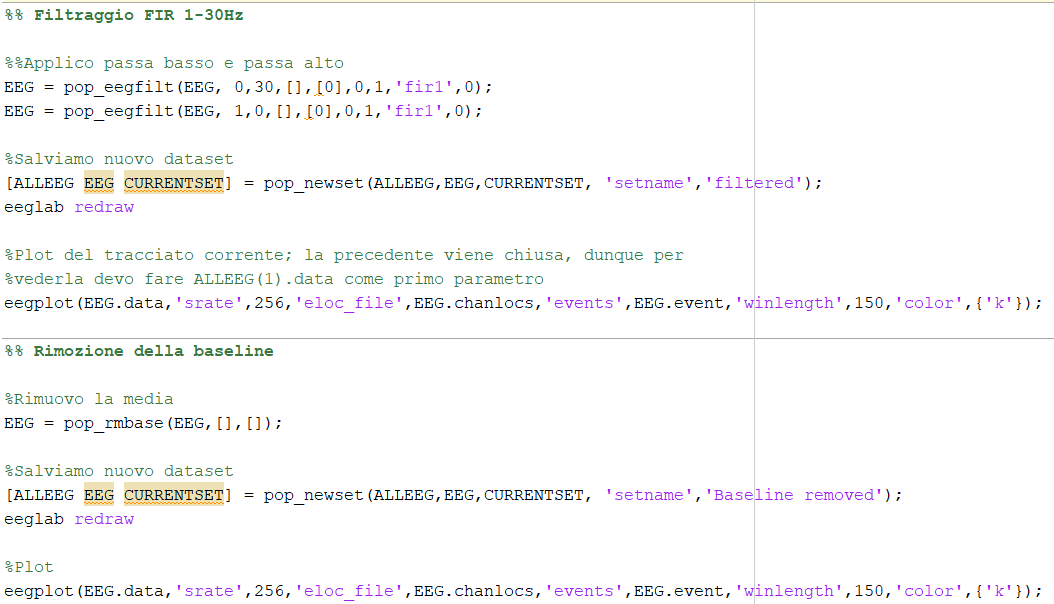
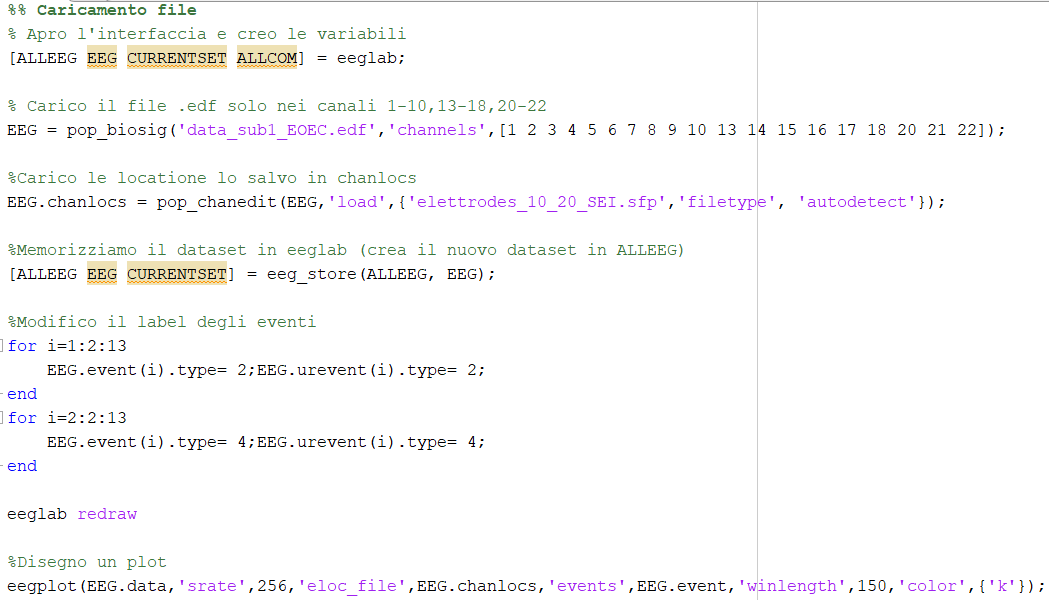
Via script:  


#### Es.2 : Analisi di segnale REALE.

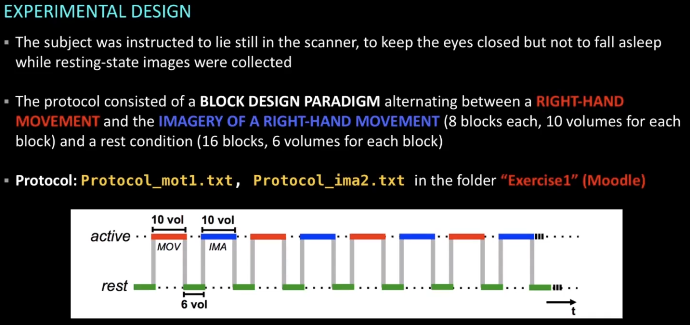
|  |  |
| --- | --- |
| **Carichiamo file EDF+**  Load→ BIOSIG  EEG = pop\_biosig('data\_sub1\_EOEC.edf'); | Dato di tipo edf+, ovvero al suo interno ha dati rigurado agli eventi   * 35x61750 con sr=250 Hz * Ci interessano solo alcuni canali 1:10 13:18 20:22     Mooolto rumoroso (es. componenti lente) |
| **Carichiamo locazioni**  Edit→Channel location  Matlab a dx | Carichiamo le locations, salvate nel file sfp  EEG.chanlocs = pop\_chanedit(EEG,'load',{'elettrodes\_10\_20\_SEI.sfp','filetype', 'autodetect'});  [ALLEEG EEG CURRENTSET] = eeg\_store(ALLEEG, EEG) |
| **Ridenominazione eventi**  (mini script preso dalle slide) | |  |  | | --- | --- | |  | Eventi: A seconda della versione possiamo avere i dati caricati con numerazione diversa. Tipo, anche a me li chiama Drowsy. | |  | Per uniformarli li rinominiamo :) | |
| Primo filtraggio  Tools→Filter→Basic FIR filter.  EEG = pop\_eegfilt(EEG, 0,30,[],[0],0,1,'fir1',0);  EEG = pop\_eegfilt (EEG, 1,0,[],[0],0,1,'fir1',0); | Non è un filtro ideale, quindi non perfetto, ma tendenzialmente riusciamo ad applicare un filtro passabanda che mantiene 1-30 Hz. Ora è molto più pulito!!!   * Bande troppo lente sono weird * Bande troppo alte sono robabilmente contrazioni muscolari.   Sono ancora presenti artefatti:   * Chiusura occhi: si genera attività ampia in banda alfa. * Blinking: si vedeeee   Possiamo capire che è blinking se abbiamo un video accanto all’EEG.  Se la frequenza fosse stata sotto i 30Hz avrei avuto bisogno di un filtro notch(??) |
| **Cambiare la referenza fra elettrodi**  Tools→Rereference→Compute average reference | → Calcolo rispetto alla media  Tools->interpolate electrodes  → Un elettrodo non registra bene (es. perde aderenza): se l’artefatto persiste su tutta la registrazione conviene interpolare le tracce nelle vicinanze. |
| **Analisi delle componenti indipendenti**  Tools→Decompose data by ICA→fastICA. | Togliere il blinking con il passabanda è difficile perché il blinking sta proprio nelle frequenze che mi interessano. Devo fare una scomposizione.  Quindi scompongo il segnale e seleziono le componenti del rumore. |
| **Scomposizione e selezione delle componenti**  Tools→Classify components using ICLabels→View extended components properties | Proprio *visivamente* dovrò riconoscere le componenti sminchiate e dirgli di levarle.   |  |  |  | | --- | --- | --- | | 🤔 - thinking face emoji | What does the thinking face emoji mean? | Eeglab mi chiede dei parametri che alla prof non chiede, e poi mi manda a quel paese. | 🤔 - thinking face emoji | What does the thinking face emoji mean? | |
| **Visualizzazione delle componenti**  Plot→ component activation scroll | L’algoritmo ICA può creare ad ogni iterazione componenti con numeri diversi!! Quindi i numeri a sinistra possono variare da una iterazione all’altra.    Questa componente (2) modula bene la condizione occhi aperti occhi chiusi:    Quest’altra (1) invece è quella legata al blink:    infatti   * Non c’è in corrispondenza di occhi chiusi * Potenziale stranino quando chiude gli occhi (ha senso, il movimento è simile anche se volontario) |
| **Rimuovere le componenti brutte**  Tools→ Remove components from data | Non c’è piuuuuuuuuù!!!! :D |
| **Estraiamo epoche**  Tools → extract epochs | Se ci chiede roba di baseline lasciamo tutto vuoto per rimuovere la media da tutto il tracciato automaticamente :)  Prendiamo epoche da 18 secondi (lunghette, ma whatever bro) |
| **Estraiamo le epoche separatamente per tipo**    Edit→ select epochs or events | Prendiamo prima solo le type 2 e poi solo le type 4.     |  |  | | --- | --- | | Type 2 (closed) | Type 4 (open) | |
| **Analisi in frequenza**  Plot→ Channel Spectra and Maps | * Epoch time: 18 secondi come prima * 100% da analizzare * (a me non faceva scegliere le frequenze tbf)  |  |  | | --- | --- | | Type 2 (closed) | Type 4 (open) | |

# Lab #4: EEGlab – Analisi completa via scripting

#### Es.1 : Fai l’esercizio precedente via scripting



# Lab #5: FSL e FSLeyes – elaborazione di mappe statistiche via GLM



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Strutture** **dati** | |  | | | |
| **structuralMRI.nii.gz:** risonanza magnetica di base; sequenza T1 (sequenza volumetrica strutturale). Dato in 3 dimensioni (volume).  **fMRI.nii.gz :** volumi cerebrali durante la sequenza magnetica funzionale. Dato in 4 dimensioni (volume \* 256 campioni nel tempo). | | Acquisizione da soggetto sano volontario magnete 3 tesla.  256 volumi Ripetizioni da 2.2 secondi 36 fette Voxel da 3x3x3 mm2 | | | |
|  | | |  |  |
| **Visualizzare MRI e fMRI** | fsleyes <file>  Conviene aprire un fsleyes per ciascuna delle due.    Posso visualizzare le due immagini sovrapposte mettendo entrambi i file dopo fsleyes. | | | |
| **Riorientare l’imamgine MRI per renderla omogenea con la fMRI** | fslswapdim <input> <a> <b> <c> [output]  Prende in input l’immagine e 3 valori a,b,c che indicano la rotazione e il nome del risultato. A,b,c sono i nuovi assi in termine dei vecchi assi; possiamo dare -x/x,-y/y,-z/z o anche RL LR AP PA SI IS (right left, anterior superior, superior inferior). Noi usiamo RL PA IS. | | | |
| **Estrarre dalle immagini MRI il contenuto dei tessuti cerebrali** (Per esempio, rimuovere ossa e collo). | bet structuralMRI\_flip.nii.gz structuralMRI\_flip\_brain\_t05  -f 0.5 -R -m -v  Bisogna fare molto attenzione perché potrebbe succedere di portare via pezzi di tessuto che ci interessano, in particolare dato che quello che ci interessa è proprio il limite esterno del cervello (la corteccia!)    Interfaccia: | | | |
| **Analisi - Preparazione all’analisi** | Scrivere il file da zero è molto laborioso, quindi è meglio eseguire l’interfaccia una prima volta e poi modificare il file.      First level analysis = se selezionata, sto facendo analisi u un solo soggetto e non su un gruppo  Full analysis: settiamo sia il preprocessing che l’analisi statistica  Potrebbe essere necessario eliminare i primi volumi (potrebero non essere ancora stabili). | | | |
| **Analisi - Pre processing (pre-stats)** | * **Slice timing correction**: bisogna dire come sono state acquisite; l’informazione ci viene fornita. * Estrazione **BET** vista prima * **Smoothing spaziale**: 6 mm (in generale deve essere il doppio della dimensione del voxel) * **Filtro highpass** | | | |
| **Analisi – Registrazione** | * Carichiamo l’immagine T1 nel campo main structural image (Già filtrata e flippata) * Possiamo scegliere uno spazio standard, ovvero a quale “media di cervelli” fare riferimento. * I gradi di libertà nella seconda devono essere alti (not sure why 31.27) | | | |
| **Analisi – Costruire il GLM** | →  Bisogna inserire tutti i predittori, dettati dal protocollo (che trovi nel file .txt). Glielo passo direttamente e fa tutto lui! :D Il protocollo txt è sostanzialmente l’onda quadra, solo che dovremo dare ogni stato separatamente.   |  |  | | --- | --- | | **Movimento** | **Movimento** **immaginato** |   Creati questi due regressori posso applicare il contrasto; posso crearne due tipi (uno per stato).    Risultato bellissimissimo: essa è la descrizione dalla design matrix! | | | |
| **Visualizzare la HRF (Hempodinamic response function)** | **Make\_flobs** (sulla bash) | | | |
| **Analisi - Applicazione della soglia statistica** |  | | | |
| **Avvia l’analisi** | Si crea una fighissima pagina HTML che si aggiorna mano mano facendoti vedere cosa sta facendo e il log e tutto 😍😍😍😍😍😍😍😍    Ci mette una vita. | | | |
| **Report – Mappe statistiche** | Viene proposto il volume cerebrale affettato.  **Movimento**: come prevedibile si attiva molto bene l’attivazione sul lato opposto del movimento  **Imagery**: abbiamo un’attivazione ma mooolto più piccina. | | | |
| **Report – time series** | Crea vari grafici, fra cui   * Per il voxel con valore statistico più alto * Per il segnale mediato, il cluster più intenso     Il dato in rosso è il dato grezzo, e sovrapposto in blu vediamo il full model fit (ovvero il modello creato dalla design matrix) moltiplicato per i valori trovate per quel voxel o per cluster di voxel. | | | |
| **File design.fsf** | feat design.fsf  Con tutte queste belle operazioni abbiamo creato un file che si chiama design.fsf. Questo tipo di file può essere aperto e modificato per riadattare protocollo e immagine al nuovo dato.  Lanciando feat design.fsf avviamo direttamente la funzione feat dal file, senza passare dalla interfaccia grafica. | | | |
| **Regustrazione delle mappe di attivazione sulla MRI strutturale** | flirt [options] -in <inputvol> -ref <refvol> -applyxfm -init <matrix> -out <outputvol>  I parametri necessari si troveranno in .feat, dove in particolare avremo le due mappe (zstat1 e zstat2). Uso la funzione flirt con:   * -in (input) : output delle operazioni precedenti (tresh\_zstat1.gz) * -ref (target): la MRI strutturale, che viene automaticamente copiata in reg/highres; * -applyxfm -init: prende i parametri (generati in automatico dalle operazioni precedenti). * -out: il nome da dare in output.   Questo bordello è necessario per spostare/adattare le mappe statistiche alle mappe strutturali; in particolare, al paziente vengono fatte prima quelle strutturali e poi quelle funzionali. Ciò significa che potenzialmente il soggetto potrebbe essersi spostato.  Possiamo impostare dal visualizzatore delle colormap diverse. | | | |
| **Registrare le mappe statistiche allo spazio standard (MNI?)** | Si vogliono unire le mappe generate a una MRI *standardizzata.* Fom la stessa cosa di prima ma:  Si prende l’immagine BOLD (non quella grezza ma quella già unita a una MRI strutturale   * -in (input) : in questo caso vuole la versione **già registrata alla fMRI strutturale singola** * -ref (target): vorremo la MRI standard, * -applyxfm -init:qui vorremo passare dallo spazio highres del soggetto a quello standard.. quindi. * -out: il nome da dare in output.     L’anatomia è template! Possiamo aggiungere un pannello atlante per vedere subito a quale zona funzionale corrispondono le zone attivate | | | |
| **Creare e applicare una maschera binaria.** | fslmaths  È utile in quanto può aiutarci a eliminare quelle attivazioni che andrebbero al di fuori della mappa standard; questo potrebbe accadere se lo smoothing è stato eccessivo.  →  Innanzitutto creiamo la maschera:    Poi applichiamo la maschera alla mappa di attivazione con l’opzione -mas: | | | |
| **Cluster** | Quantifica le attivazioni cerebrali | | | |

Note da Eleonora:

* Importante non avere spazi o caratteri speciali nel percorso; esplode.