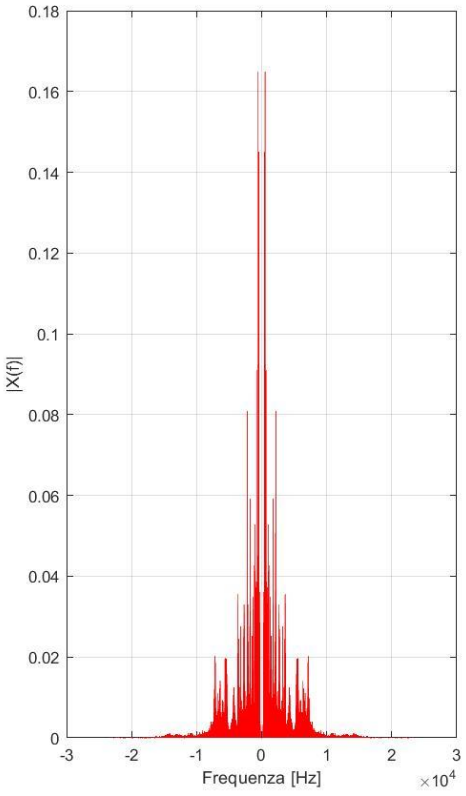
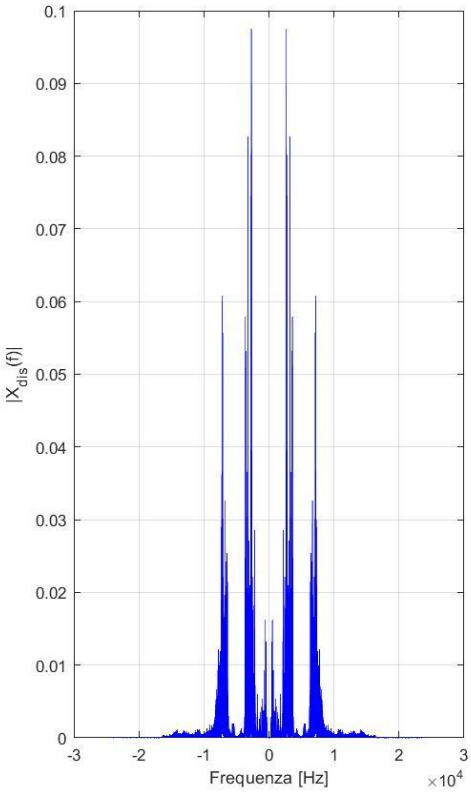
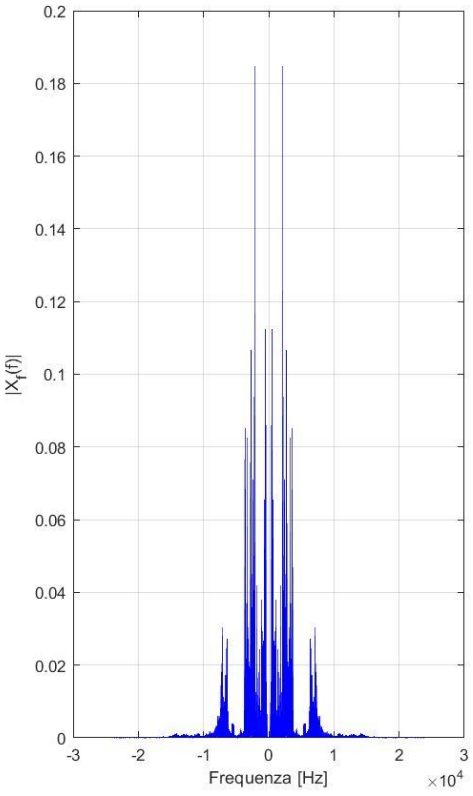
TESINA MATLAB

PREMESSA:
Nelle ultime pagine è presente il codice matlab: se si vuole ascoltare un segnale, basta rimuovere il commento in una delle righe che presenta il comando play.
ESERCIZIO 1 e 2:
Dopo aver importato e ascoltato il segnale x con il comando play, ho proceduto a calcolare e plottare separatamente il modulo delle trasformate di Fourier dei segnali x e x_dis, prestando attenzione a definire la frequenza di campionamento in frequenza fc = Fc/N (con N=length(x)). Successivamente era necessario, attraverso il banco di filtri, modificare opportunamente i coefficienti α_i. Si è cercato di migliorare sia la rappresentazione dello spettro in frequenza sia la qualità audio. In particolare, l'aumento dei contributi delle basse frequenze ha giovato a entrambi gli obiettivi anche se la distorsione iniziale rimane comunque visibile e abbastanza udibile nel segnale filtrato.
Pertanto, confrontando i tre grafici, si nota una maggiore somiglianza fra la prima rappresentazione (segnale X in rosso) e la terza (segnale X_f filtrato in blu). Ciò è reso possibile scegliendo gli α_i in modo tale che il modulo della TDF di x_f sia quanto più simile a quello della TDF di x, per ogni frequenza considerata. Avendo a disposizione un numero limitato di filtri, tali da costituire un unico banco, è risultato chiaramente meno possibile colmare le differenze fra il segnale originario e quello filtrato in corrispondenza delle bande di transizione.

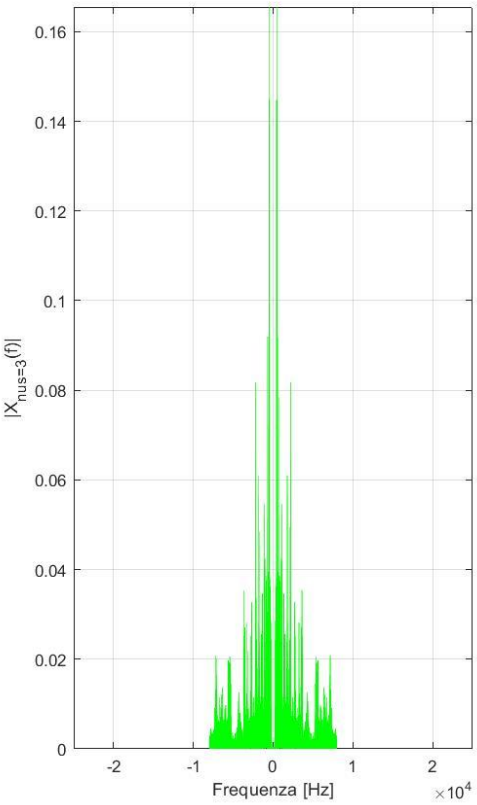


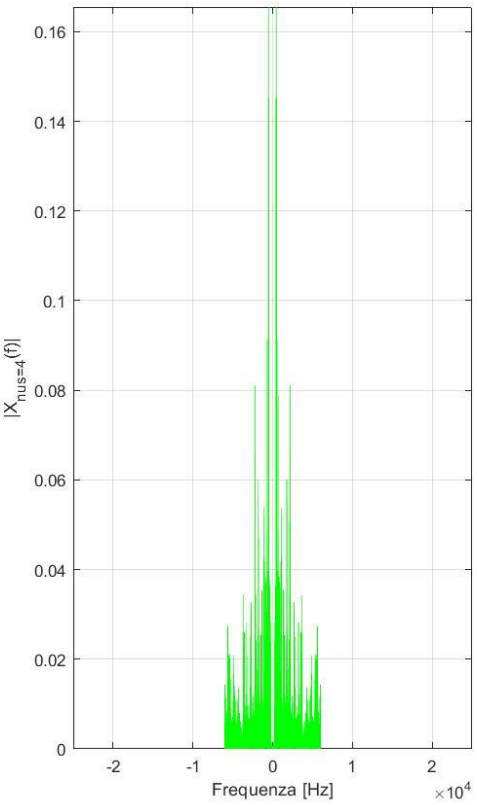


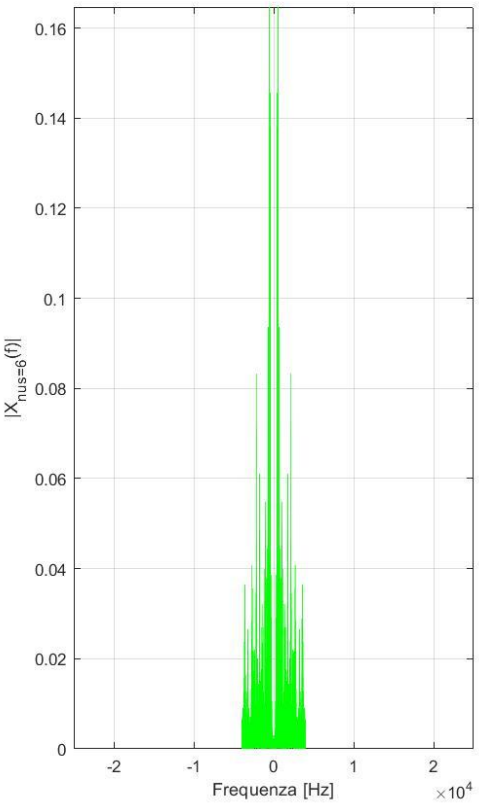


ESERCIZIO 3:

É stata generata una figura in cui sono presenti i grafici della TDF che si ottengono sotto campionando il segnale originale x alle frequenze 16 kHz ($x_{nus=3}$), 12 kHz ($x_{nus=4}$) e 8 kHz($x_{nus=6}$). Si è fatto in modo di scegliere lo stesso intervallo di frequenze da riportare in ascissa, in modo da rendere più agevole il confronto fra grafici. I tre segnali sono stati generati trattenendo solo 1 campione su 3, 4 o 6 del segnale x originario. Il risultato è la conseguente perdita di contributi per alcune frequenze. Tale perdita si accentua maggiormente quanto più si riduce la frequenza di campionamento. Ascoltando i tre segnali, si nota un significativo peggioramento della qualità del suono soprattutto per $x_{nus=6}$ (per $x_{nus=3}$ si percepisce ancora poco il peggioramento). In tal caso l'ampiezza di banda in frequenza è ridotta a tal punto da generare un fenomeno di aliasing facilmente udibile per alte frequenze: esso si traduce principalmente nella presenza di "fischi".







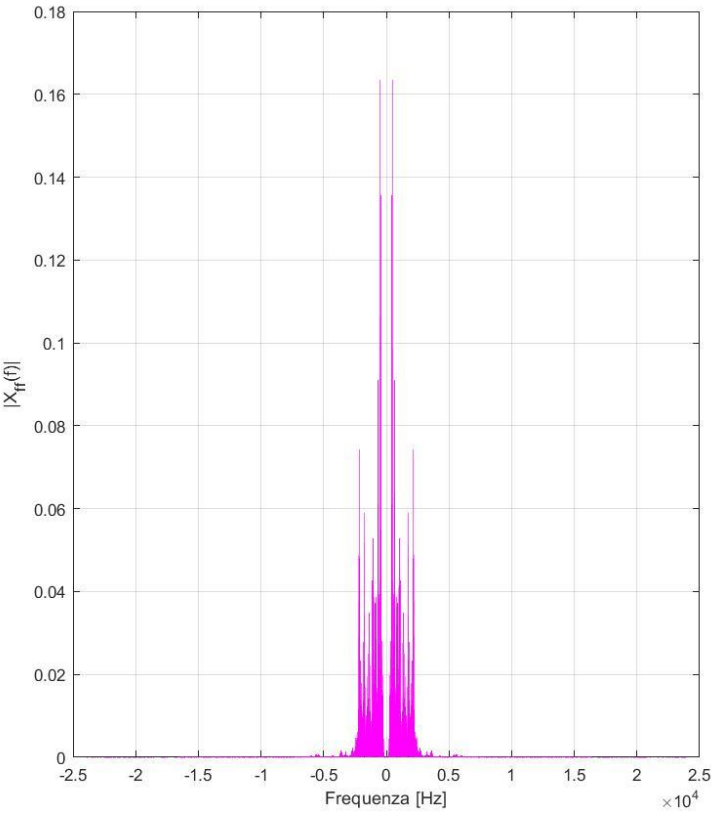
ESERCIZIO 4:

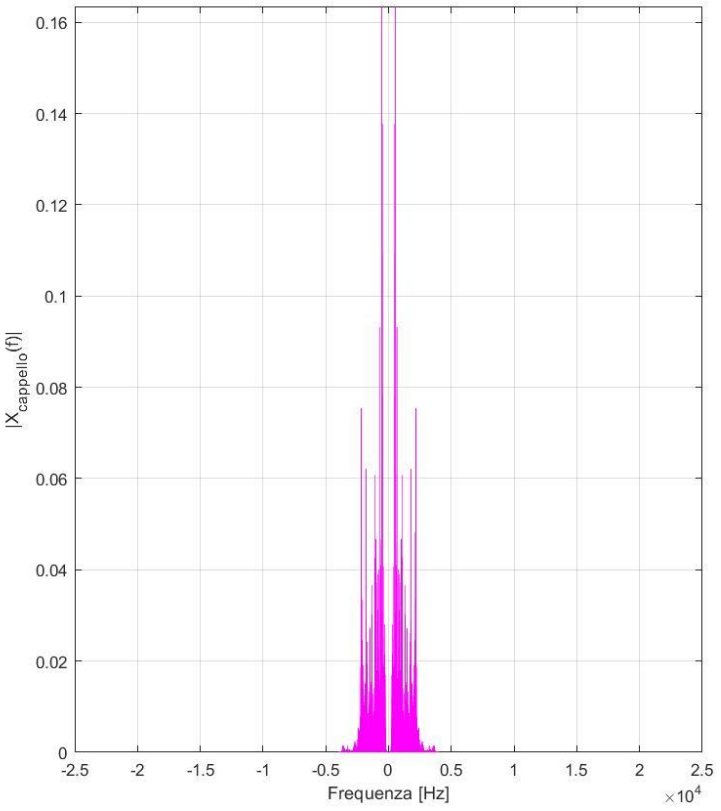
Per risolvere o quantomeno ridurre il problema di aliasing visibile nel segnale $x_{nus=6}$, è stato necessario ricorrere al filtro antialiasing. Si è quindi proceduto a creare il banco di filtri, con l'apposita funzione, applicandolo al segnale originale x. I diversi coefficienti α_i sono stati scelti, anche in questo caso, per rimuovere l'aliasing e migliorare il suono (rendendolo quanto più simile a quello di x). Per fare ciò ci si è focalizzati sulla causa dell'aliasing. Trascurando frequenze che contribuiscono con componenti trascurabili, si può affermare, notando lo spettro di X, che la frequenza minima di campionamento è circa 30 kHz. Pertanto vale che Fc > 30 kHz ma Fnus = Fc/6 = 8 kHz < 30 kHz (ciò si percepisce a livello sonoro nella presenza di "fischi" per determinate frequenze).

Ponendo molto bassi (o anche nulli) i coefficienti che non si riferiscono alla banda centrale (almeno quella da preservare: quindi si possono annullare tutti i coefficienti a parte i primi due), si eliminano i contributi di alcune frequenze dallo spettro originale ma si agisce anche sulla frequenza minima di campionamento: guardando il grafico del segnale filtrato (sotto a sinistra) appare di circa 8 kHz. Pertanto ciò permette di campionare il segnale x_ff (ottenuto filtrando x) a Fnus = Fc/6 = 8 kHz evitando aliasing.

I grafici in frequenza di x_ff e x_cappello appaiono particolarmente simili (infatti campiono alla frequenza minima di campionamento) proprio perchè campiono x_ff alla Fc_{minima} e quindi il segnale viene riprodotto senza aliasing.

Pertanto il miglioramento in termini di suono di x_ff rispetto a x_cappello è visibile nonostante la perdita di certe frequenze del segnale originario.





Codice utilizzato commentato:

```
%% *** TESINA SEGNALI E SISTEMI ***
clear
close all
clc

%% PRIMA PARTE: ascoltare i segnali audio

load audio;
player = audioplayer(x,Fc);
%play(player);
player_dis = audioplayer(x_dis,Fc);
%play(player_dis);


%% SECONDA PARTE:
% calcolo trasformata di Fourier dei segnali x e x_dis con plot associato.
N = length(x);
fc = Fc/N;
X = (1/Fc)*fft(x);
X = fftshift(X);
f = fc*(-N/2:N/2-1);
figure;
subplot(1,3,1);
plot(f,abs(X),'r');
grid on
xlabel('Frequenza [Hz]');
ylabel('|X(f)|');

% plot del modulo della trasformata di Fourier del segnale x_dis
N_dis = length(x_dis);
X_dis = (1/Fc)*fft(x_dis);
X_dis = fftshift(X_dis);
f = fc*(-N_dis/2:N_dis/2-1);
subplot(1,3,2);
plot(f,abs(X_dis),'b');
grid on
xlabel('Frequenza [Hz]');
ylabel('|X_d_i_s(f)|');

% definisco i vettori delle frequenze e dei coefficienti.
% creo il banco di filtri con l'apposita funzione, infine applico il filtraggio
f_vals = [1,2e3,3.25e3,6e3,8e3,10e3,23.999e3];
eq_vals = [7,1,2,0.5,1,1];
H = filter_bank(eq_vals, f_vals, Fc, false);
x_f = filter(H.Numerator{:},H.Denominator{:},x_dis);

% plot del modulo della trasformata di Fourier del segnale x_f
```

```
N_f = length(x_f);
fc = Fc/N_f;
X_f = (1/Fc)*fft(x_f);
X_f = fftshift(X_f);
f = fc*(-N_f/2:N_f/2-1);
subplot(1,3,3);
plot(f,abs(X_f),'b');
grid on
xlabel('Frequenza [Hz]');
ylabel('|X_f(f)|');

% ascolto il nuovo segnale ottenuto filtrando il segnale x_dis
player_f = audioplayer(x_f,Fc);
%play(player_f);


%% TERZA PARTE:
%ripeto tre analisi separate per nus = 3,4,6, generando i segnali x_nus3,
%x_nus4, x_nus6 rispettivamente (sono generati scegliendo 1 campione su
%3,4 o 6 del segnale x originario). Tali segnali possono essere ascoltati e
%vengono plottati nel dominio delle frequenze (stabilendo uguali i limiti
%dell'asse x in modo che si possano vedere immediatamente gli effetti dei
%diversi campionamenti confrontando i grafici).

%caso nus = 3
nus = 3;
Fus = Fc/nus;
N = ceil(length(x)/nus);
fc = Fus/N;
x_nus3 = zeros(N,1);
i=1; k=1;
while i< length(x)
    x_nus3(k) = x(i);
    k=k+1;
    i=i+nus;
end
X_nus3 = (1/Fus)*fft(x_nus3);
X_nus3 = fftshift(X_nus3);
f = fc*(-N/2:N/2-1);
figure;
subplot(1,3,1);
plot(f,abs(X_nus3),'g');
grid on
xlabel('Frequenza [Hz]');
ylabel('|X_n_u_s_=_3(f)|');
axis([-2.5*10^4 2.5*10^4 0 inf]);
playerus3 = audioplayer(x_nus3,Fus);
%play(playerus3);
%caso nus = 4
nus = 4;
```

```
Fus = Fc/nus;
N = ceil(length(x)/nus);
fc = Fus/N;
x_nus4 = zeros(N,1);
i=1; k=1;
while i< length(x)
    x_nus4(k) = x(i);
    k=k+1;
    i=i+nus;
end
X_nus4 = (1/Fus)*fft(x_nus4);
X_nus4 = fftshift(X_nus4);
f = fc*(-N/2:N/2-1);
subplot(1,3,2);
plot(f,abs(X_nus4),'g');
grid on
xlabel('Frequenza [Hz]');
ylabel('|X_n_u_s_=_4(f)|');
axis([-2.5*10^4 2.5*10^4 0 inf]);
playerus4 = audioplayer(x_nus4,Fus);
%play(playerus4);

%caso nus = 6
nus = 6;
Fus = Fc/nus;
N = ceil(length(x)/nus);
fc = Fus/N;
x_nus6 = zeros(N,1);
i=1; k=1;
while i< length(x)
    x_nus6(k) = x(i);
    k=k+1;
    i=i+nus;
end
X_nus6 = (1/Fus)*fft(x_nus6);
X_nus6 = fftshift(X_nus6);
f = fc*(-N/2:N/2-1);
subplot(1,3,3);
plot(f,abs(X_nus6),'g');
grid on
xlabel('Frequenza [Hz]');
ylabel('|X_n_u_s_=_6(f)|');
axis([-2.5*10^4 2.5*10^4 0 inf]);
playerus6 = audioplayer(x_nus6,Fus);
%play(playerus6);


%% PARTE QUATTRO:
% prima filtro x e ottengo x_ff
```

```
% poi sottocampiono x_ff (ottenendo x_cappello) e analizzo, nel dominio
% delle frequenze, che x_ff non dovrebbe presentare aliasing.
% Ascolto e confronto (in frequenza) x_ff e x_cappello.

%generazione e applicazione del filtro per la creazione di x_ff
nus = 6;
Fus = Fc/nus;
N = ceil(length(x)/nus);
eq_vals = [1,0.05,0.03,0.01,0.01,0.01];
H = filter_bank(eq_vals, f_vals, Fc, false);
% applico il filtro al segnale x sottocampionato (Fus = Fc/6), ottengo x_ff
x_ff = filter(H.Numerator{:},H.Denominator{:},x);

%rappresentazione segnale x_ff in frequenza
N = length(x_ff);
fc = Fc/N;
X_ff = (1/Fc)*fft(x_ff);
X_ff = fftshift(X_ff);
f = fc*(-N/2:N/2-1);
figure;
subplot(1,2,1);
plot(f,abs(X_ff),'m');
grid on
xlabel('Frequenza [Hz]');
ylabel('|X_f_f(f)|');
playerff = audioplayer(x_ff,Fc);
%play(playerff);

%rappresentazione segnale x_cappello in frequenza
fc = Fus/N;
x_cappello = zeros(N,1);
i=1; k=1;
while i< length(x_ff)
    x_cappello(k) = x_ff(i);
    k=k+1;
    i=i+nus;
end
X_cappello = (1/Fus)*fft(x_cappello);
X_cappello = fftshift(X_cappello);
f = fc*(-N/2:N/2-1);
subplot(1,2,2);
plot(f,abs(X_cappello),'m');
grid on
xlabel('Frequenza [Hz]');
ylabel('|X_c_a_p_p_e_l_l_o(f)|');
axis([-2.5*10^4 2.5*10^4 0 inf]);
%verifico il miglioramento del suono a seguito del filtro anti aliasing
player6 = audioplayer(x_cappello,Fus);
%play(player6);
```