Odmeravanje zvučnog signala

Predmet: Multimedijalni signali i sistemi

Predavač: dr Nataša Savić

Asistent: Nikola Milutinović

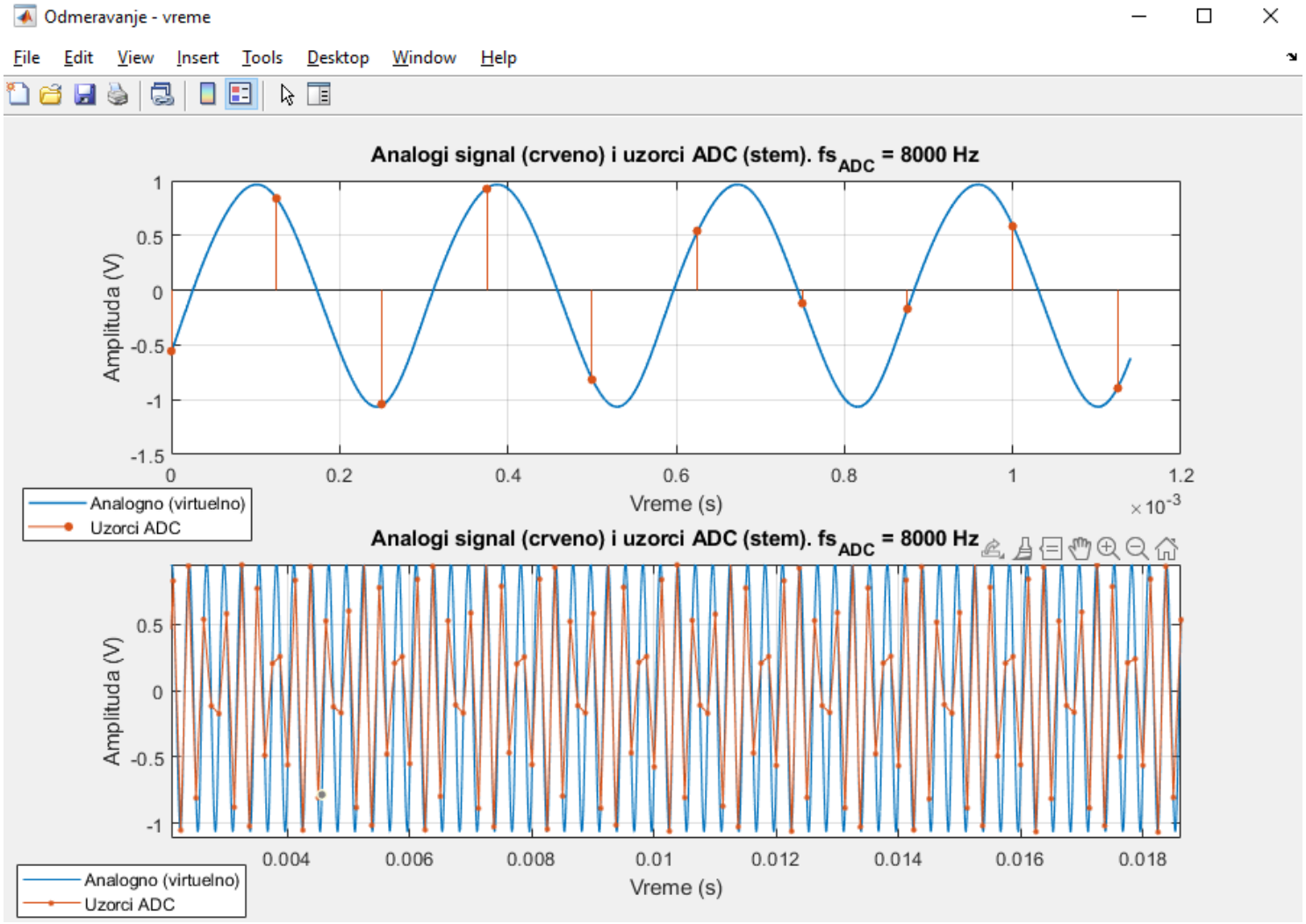
Odmeravanje sinusnog signala

```
% "Analogni" (visoka rezolucija) signal - koristi se samo za prikaz
analog_fs = 200e3;        % "virtuelno analogno" uzorkovanje za crtanje (Hz)
T = 0.02;                 % trajanje signala u sekundama (20 ms)
t_analog = (0:1/analog_fs:T-1/analog_fs)';

% Generisani (istinski) sinusni signal parametri
f_signal = 3500;          % frekvencija sinusa (Hz) - promeni za testove
A_signal = 1.0;           % amplituda (V)
phi_signal = -pi/5;       % faza (radiani)

% Parametri ADC / uzorkovanja (što demonstrira odmeravanje)
fs_adc = 8000;  % frekv. uzorkovanja (Hz) - promeni za effect Nyquist/aliasing
t_adc = (0:1/fs_adc:T-1/fs_adc)';
```

```
% Opcije: dodavanje buke ili harmonika
add_noise = true;
SNR_dB_target = 40;     % samo ako add_noise = true
add_harmonics = true;
harmonics = [2,3];    % harmonici koji će se dodati (relativne amplitude ispod)
harm_rel_amp = [0.05, 0.02];

% "Analogni" signal (vis. rez)
x_analog = A_signal * sin(2*pi*f_signal*t_analog + phi_signal);
if add_harmonics
  for k = 1:length(harmonics)
    x_analog = x_analog + A_signal*harm_rel_amp(k) * ...
          sin(2*pi*harmonics(k)*f_signal*t_analog + 0.3*k);
  end
end
```

```
% Odmeravanje: simulacija ADC uzorkovanja (isključimo višak uzoraka)
% Uzorkujemo analogni signal u vremenima t_adc (uzorci ADC)
x_adc = interp1(t_analog, x_analog, t_adc, 'linear'); % idealni sample iz "analognog"
% Dodaj šum da dobijemo ciljani SNR (opciono)
if add_noise
   sig_pow = mean(x_adc.^2);
   snr_lin = 10^(SNR_dB_target/10);
   noise_pow = sig_pow / snr_lin;
   noise = sqrt(noise_pow) * randn(size(x_adc));
   x_adc_noisy = x_adc + noise;
else
   x_adc_noisy = x_adc;
   noise = zeros(size(x_adc));
end

%% PRIKAZ: vreme i uzorci
figure('Name','Odmeravanje - vreme','NumberTitle','off','Position',[100 100 900 560]);
% 1) prikaži nekoliko prvih perioda (analogni + uzorci)
subplot(2,1,1);
% izaberemo prozor do 4 perioda signala (ako je frekv. poznata)
if f_signal>0
   win_T = min(4/f_signal, T);
else
   win_T = 0.005;
end
idx_a = find(t_analog<=win_T);
plot(t_analog(idx_a), x_analog(idx_a), 'LineWidth', 1.2); hold on;
% plot sampled points over original
idx_s = find(t_adc<=win_T);
stem(t_adc(idx_s), x_adc_noisy(idx_s), 'filled','MarkerSize',4);
xlabel('Vreme (s)');
ylabel('Amplituda (V)');
title(sprintf('Analogi signal (crveno) i uzorci ADC (stem). fs_{ADC} = %.0f Hz', fs_adc));
legend('Analogno (virtuelno)','Uzorci ADC','Location','best');
grid on;

% 2) prikaži celu sekciju signala (svi uzorci) radi opšte slike
subplot(2,1,2);
plot(t_analog, x_analog, 'LineWidth', 0.8); hold on;
plot(t_adc, x_adc_noisy, '.-','MarkerSize',8);
xlabel('Vreme (s)');
ylabel('Amplituda (V)');
title('Ceo signal: analogno vs uzorkovano (sa šumom ako je uključen)');
legend('Analogno','Uzorci ADC','Location','best');
xlim([0 T]);
grid on;
fprintf('\n--- Rezultati merenja ---\n');
fprintf('Zadata frekvencija: %.2f Hz\n', f_signal);
```

Rezultati odmeravanja sinusnog signala



Odmeravanje zvučnog signala

```
% "Analogni" (visoka rezolucija) signal za prikaz i reprodukciju
analog_fs = 192000;     % virtuelno "analogno" uzorkovanje za prikaz i playback (Hz)
T = 2.0;                % trajanje signala u sekundama
t_analog = (0:1/analog_fs:T-1/analog_fs)';

% Sintetički audio signal - kombinacija tonova (može se menjati za eksperimente)
% Primer: mix više tonova (niskofrekv. ton + melodija + harmonici + glissando)
f1 = 440;               % A4 - primer osn. tona (Hz)
f2 = 880;               % oktava
f3 = 1760;              % viša oktava
% Dodaj mali gliss (blagi porast frekvencije) zbog dinamičnijeg prikaza
f_gliss_start = 300;
f_gliss_end   = 900;
t = t_analog;
f_gliss = f_gliss_start + (f_gliss_end - f_gliss_start)*(t/T);

% Kompozicija signala (sum of sines + harmonics + short burst noise)
x_analog = 0.6*sin(2*pi*f1*t + 0.2) ...
      + 0.35*sin(2*pi*f_gliss.*t + 0.9) ...
      + 0.2*sin(2*pi*f2*t - 0.5) ...
      + 0.08*sin(2*pi*3*f1*t + 1.2);   % 3. harmonik

% Dodaj kratki perkusivni ton (burst) u sredini trajanja
burst_len = round(0.05*analog_fs);
burst_start = round(0.5*analog_fs);
x_analog(burst_start:burst_start+burst_len-1)
x_analog(burst_start:burst_start+burst_len-1) ...
   + 0.6 .* hann(burst_len) .* sin(2*pi*2000*(0:burst_len-1)'/analog_fs);
```

=

```
% Dodaj blagi šum (opciono true ili false)
add_noise = true;
if add_noise
   SNR_dB = 40; % ciljni SNR
   sig_pow = mean(x_analog.^2);
   noise_pow = sig_pow / (10^(SNR_dB/10));
   x_analog = x_analog + sqrt(noise_pow)*randn(size(x_analog));
end

% Normalizacija radi sigurnije reprodukcije
x_analog = x_analog / max(abs(x_analog) + eps);

% Parametri ADC / uzorkovanja koje demonstriramo
fs_adc_list = [44100, 22050, 8000]; % liste stope uzorkovanja za poređenje (Hz)
% Izaberemo jedan od njih kao "radni" fs_adc; u vežbi menjati i upoređivati
fs_adc = fs_adc_list(2); % npr. 22050 (menični izbor, studenti menjaju)
t_adc = (0:1/fs_adc:T-1/fs_adc)';

% Downsampling flag (simulacija ADC: reuzorkovati sa analog_fs -> fs_adc)
do_downsample = true;

% Anti-alias (pre-decimation) opcije
apply_antialias_filter = true;
aa_order = 6;               % red Butterworth
aa_cutoff = 0.45 * fs_adc;      % cutoff frekvencija lowpass pre decimacije

% Rekonstrukcija (interpolacija) za prikaz
reconstruct_method = 'pchip';   % 'pchip', 'spline' ili 'sinc' (sinc sporo)
```

```
% Playback opcije
play_original = true;
play_sampled  = true;
pause_between = 1.0; % sec

% Simuliramo ADC: reuzorkujemo "analogni" signal na fs_adc
if do_downsample % Direktna decimacija indeksima (loš primer koji pokazujemo studentima)
   dec_factor = round(analog_fs / fs_adc);
   if dec_factor < 1
      error('Izaberite fs_adc manji od analog_fs (%d).', analog_fs);
   end
   x_naive = x_analog(1:dec_factor:end);
   t_naive = (0:length(x_naive)-1)'/fs_adc;

   % Ispravan postupak: anti-aliasing filter pa decimacija
   if apply_antialias_filter
      Wn = aa_cutoff / (analog_fs/2);
      [b,a] = butter(aa_order, Wn, 'low');  % filtfilt za zero-phase radi lepšeg prikaza (nije uvek dostupno u realnoj ADC)
      x_filt = filtfilt(b,a,x_analog);
      x_down = x_filt(1:dec_factor:end);
      t_down = (0:length(x_down)-1)'/fs_adc;
   else
      x_down = x_naive;
      t_down = t_naive;
   end
else
   x_down = interp1((0:length(x_analog)-1)'/analog_fs, x_analog, t_adc, 'pchip');
   t_down = t_adc;
   dec_factor = analog_fs / fs_adc;
end

% Normalizacija downsampled signala
x_down = x_down / max(abs(x_down)+eps);

% Rekonstruisemo signal iz uzoraka natrag na "analognu" mrežu t_analog radi poređenja
if strcmpi(reconstruct_method,'sinc') % Idealna (aproksimativna) sinc rekonstrukcija - SPORO za duge signale
   Ts_new = 1/fs_adc;
   x_rec = zeros(size(t_analog));
   t_samples = t_down;
   for k=1:length(t_samples)
      x_rec = x_rec + x_down(k) * sinc((t_analog - t_samples(k))/Ts_new);
   end
else
   x_rec = interp1(t_down, x_down, t_analog, reconstruct_method, 0);
end

% FFT originalnog i downsampled signala (za prikaz)
Nfft1 = 2^nextpow2(length(x_analog));
X1 = fft(x_analog, Nfft1); f1 = (0:Nfft1-1)*(analog_fs/Nfft1);
half1 = 1:floor(Nfft1/2);

Nfft2 = 2^nextpow2(length(x_down));
X2 = fft(x_down, Nfft2); f2 = (0:Nfft2-1)*(fs_adc/Nfft2);
half2 = 1:floor(Nfft2/2);
```

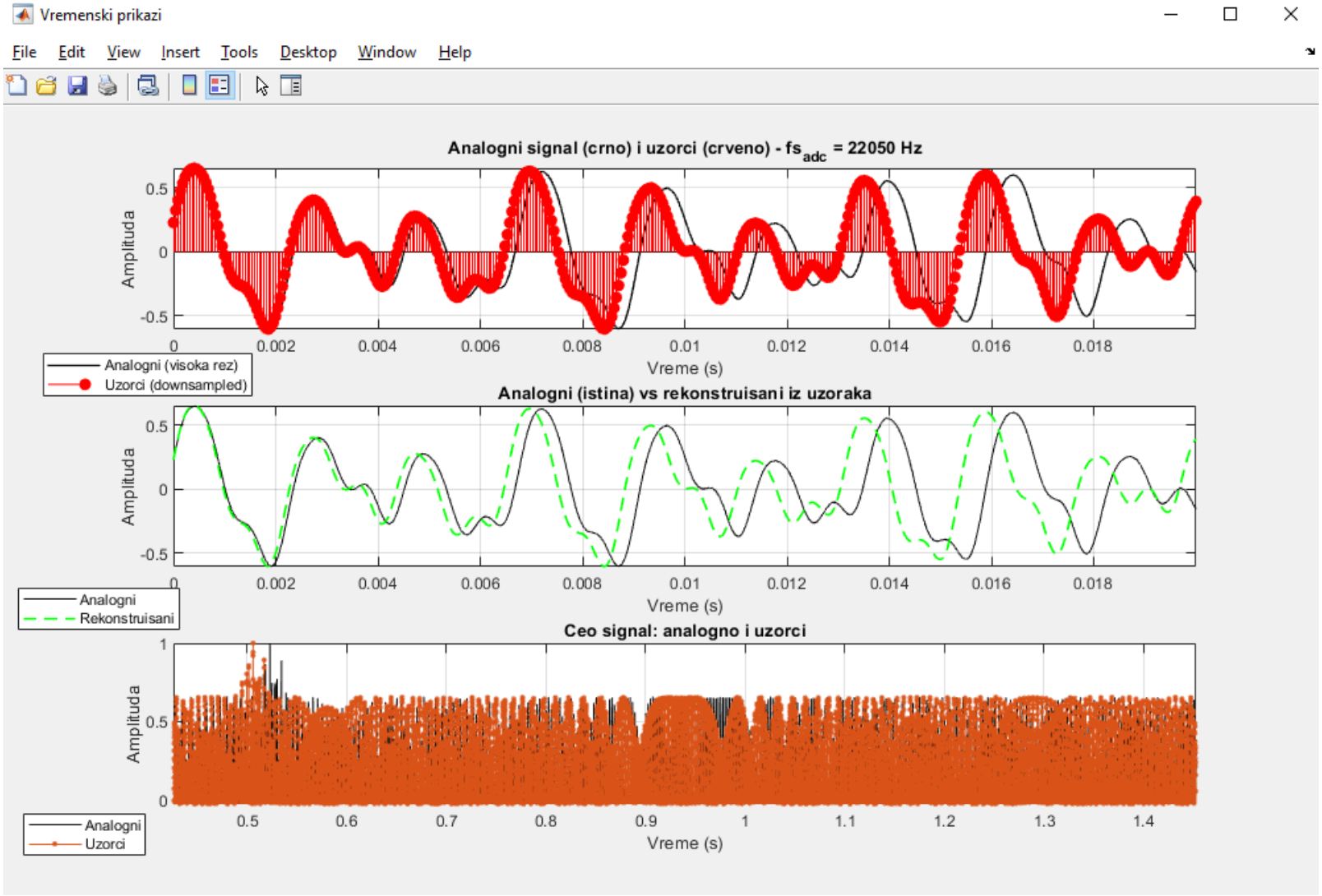
```
% 1) Vremenski prikazi: kratki prozor (par desetina ms) za bolju vidljivost
zoomT = 0.02; % 20 ms window
idx_zoom_a = find(t_analog <= zoomT);
idx_zoom_d = find(t_down <= zoomT);

figure('Name','Vremenski prikazi','NumberTitle','off','Position',[100 100 1000 600]);
subplot(3,1,1);
plot(t_analog(idx_zoom_a), x_analog(idx_zoom_a),'k','LineWidth',1.0); hold on;
stem(t_down(idx_zoom_d), x_down(idx_zoom_d),'r','filled');
xlabel('Vreme (s)');
ylabel('Amplituda');
title(sprintf('Analogni signal (crno) i uzorci (crveno) - fs_{adc} = %d Hz', fs_adc));
legend('Analogni (visoka rez)','Uzorci (downsampled)','Location','best');
grid on;

subplot(3,1,2);
plot(t_analog(idx_zoom_a), x_analog(idx_zoom_a),'k'); hold on;
plot(t_analog(idx_zoom_a), x_rec(idx_zoom_a),'g--','LineWidth',1.2);
xlabel('Vreme (s)');
ylabel('Amplituda');
title('Analogni (istina) vs rekonstruisani iz uzoraka');
legend('Analogni','Rekonstruisani','Location','best');
grid on;

subplot(3,1,3);
% cela sekvenca: original vs downsampled (pojednostavljeno)
plot((0:length(x_analog)-1)/analog_fs, x_analog,'k'); hold on;
plot(t_down, x_down,'.-','MarkerSize',8);
xlabel('Vreme (s)'); ylabel('Amplituda');
title('Ceo signal: analogno i uzorci');
legend('Analogni','Uzorci','Location','best');
xlim([0 T]); grid on;
```

```
% 2) FFT prikaz (normalizovano dB)
figure('Name','Spektri','NumberTitle','off','Position',[150 120 900 500]);
subplot(2,1,1);
plot(f1(half1), 20*log10(abs(X1(half1))/max(abs(X1(half1)))+eps));
xlabel('Frekvencija (Hz)'); ylabel('Magnitude (dB)');
title('Spektar ORIGINALNOG signala (normalizovano)');
xlim([0 20000]); grid on;

subplot(2,1,2);
plot(f2(half2), 20*log10(abs(X2(half2))/max(abs(X2(half2)))+eps));
xlabel('Frekvencija (Hz)'); ylabel('Magnitude (dB)');
title(sprintf('Spektar UZORAKA (fs = %d Hz)', fs_adc));
xlim([0 min(fs_adc/2,20000)]); grid on;

% 3) Spektrogram (vizuelna potvrda aliasinga / gubitka)
figure('Name','Spektrogrami','NumberTitle','off','Position',[200 150 1000 600]);
subplot(2,1,1);
spectrogram(x_analog, round(0.02*analog_fs), round(0.01*analog_fs), 1024,
analog_fs, 'yaxis');
title('Spektrogram ORIGINALNOG (visoka rezolucija)');
subplot(2,1,2);
spectrogram(x_down, max(64,round(0.02*fs_adc)),
round(0.5*round(0.02*fs_adc)), 512, fs_adc, 'yaxis');
title(sprintf('Spektrogram UZORAKA (fs = %d Hz)', fs_adc));

%% ============================== REPRODUKCIJA
==============================
% Reprodukuj originalni, uzorkovani (pravilno decimovan) i rekonstruisani
if play_original
  fprintf('Reprodukujem ORIGINALNI (analog_fs = %d Hz)...\n', analog_fs);
  % presample/konvert na analog_fs ako treba (već jeste)
  soundsc(x_analog, analog_fs);
  pause(T + pause_between);
end
```

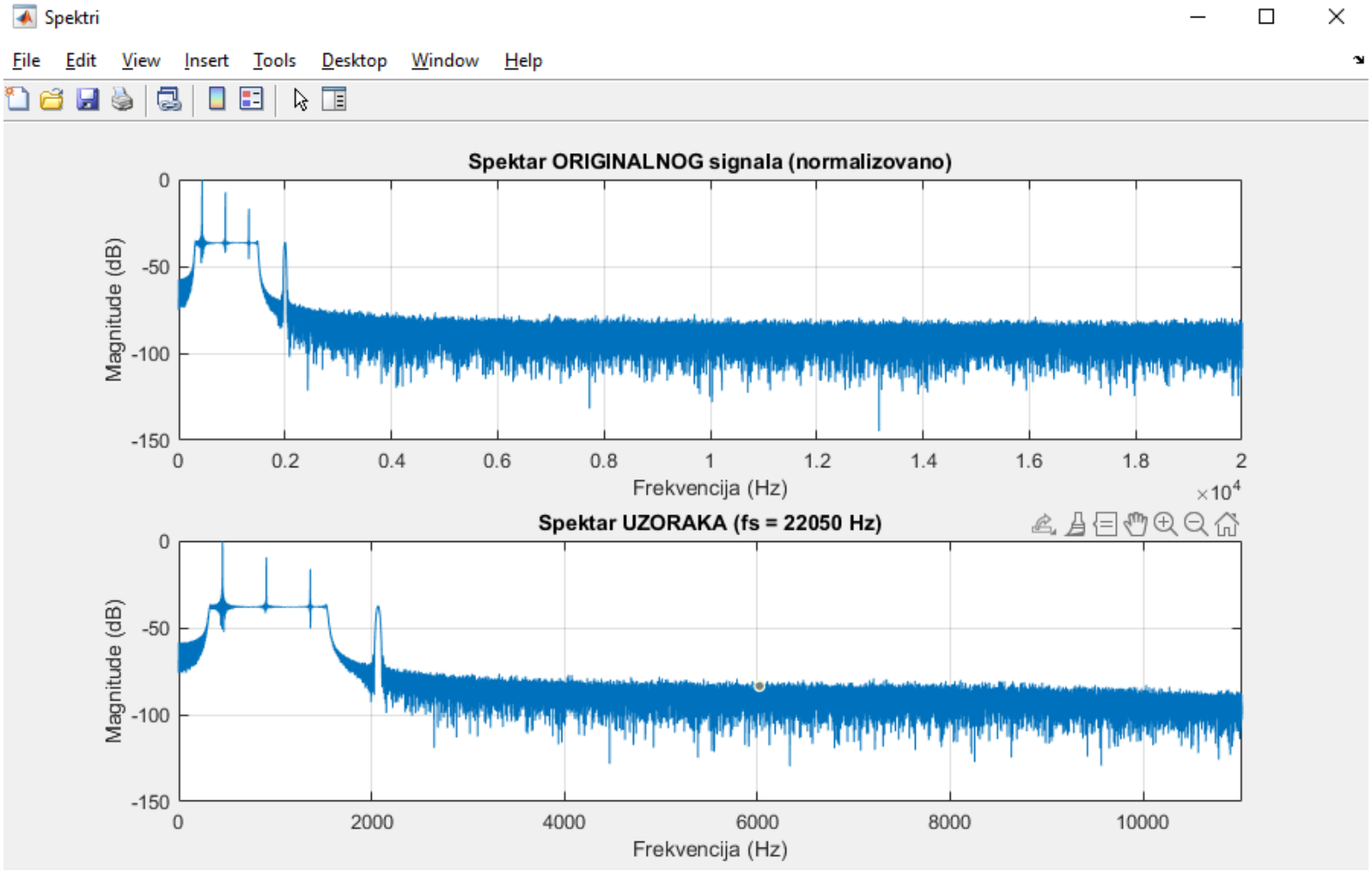
```
% Reprodukuj originalni, uzorkovani (pravilno decimovan) i rekonstruisani
if play_original
   fprintf('Reprodukujem ORIGINALNI (analog_fs = %d Hz)...\n', analog_fs);
% presample/konvert na analog_fs ako treba (već jeste)
   soundsc(x_analog, analog_fs);
   pause(T + pause_between);
end

if play_sampled
   fprintf('Reprodukujem UZORKOVANI signal direktno (fs = %d Hz)...\n', fs_adc);
   soundsc(x_down, fs_adc);
   pause(length(x_down)/fs_adc + pause_between);

   fprintf('Reprodukujem REKONSTRUKCIJU (interp na analog_fs)...\n');
   soundsc(x_rec, analog_fs);
   pause(T + pause_between);
end
```

```
%% ISPIS NA EKRANU
fprintf('\n--- Rezime vežbe ---\n');
fprintf('Trajanje signala: %.2f s\n', T);
fprintf('Analogno uzorkovanje (za prikaz): %d Hz\n', analog_fs);
fprintf('Simulovano uzorkovanje: fs_adc = %d Hz (decimacija ~ %d)\n', fs_adc, dec_factor);
if apply_antialias_filter
   fprintf('Primena anti-alias filtera: Butterworth red %d, cutoff %.1f Hz\n', aa_order, aa_cutoff);
else
   fprintf('ANTI-ALIAS FILTER ISKLJUCEN -> ocekivati aliasing\n');
end
fprintf('Kraj skripte. Studenti: promenite fs_adc, apply_antialias_filter i reconstruct_method.\n');
```

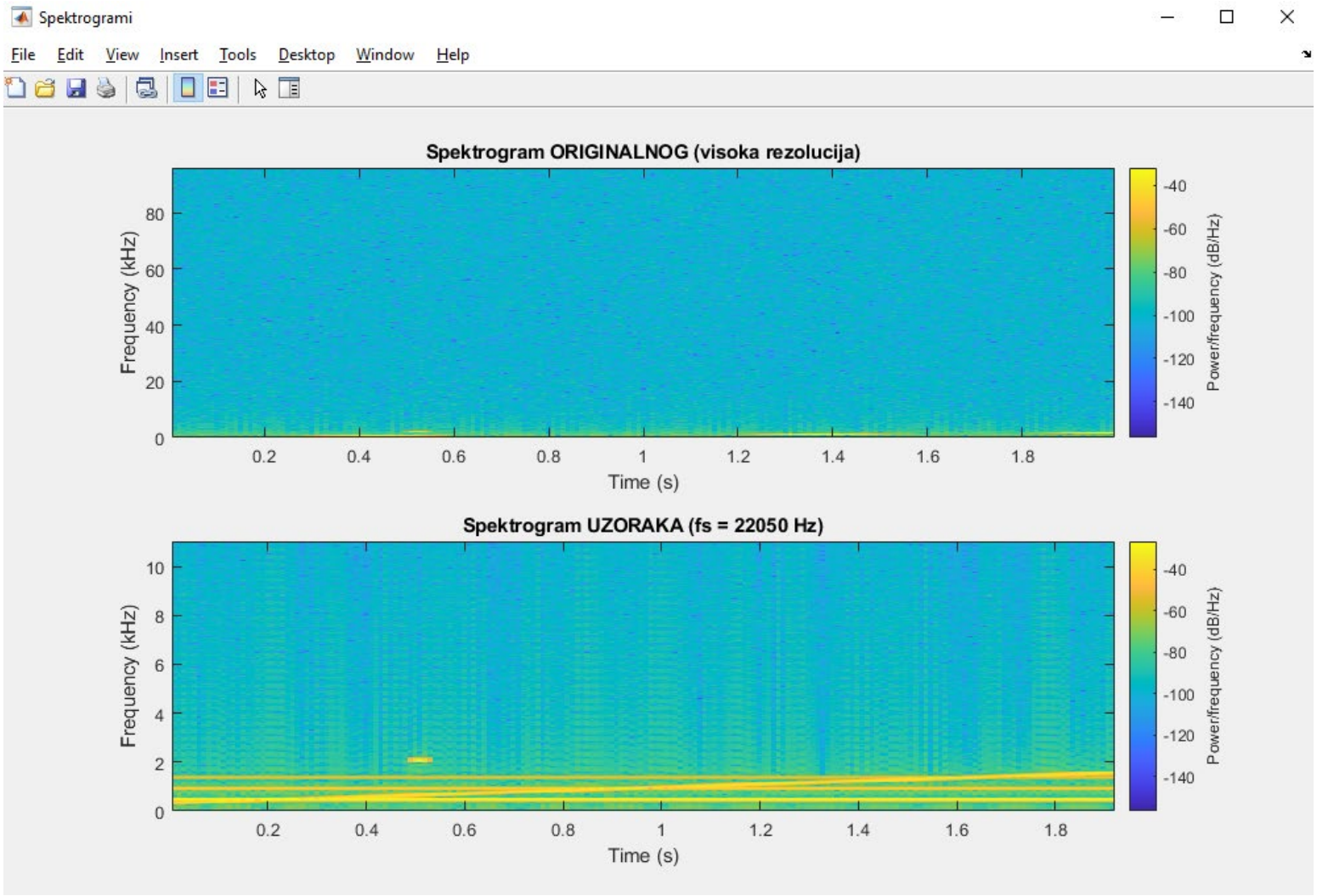
Rezultati odmeravanja zvučnog signala



Rezultati odmeravanja zvučnog signala



Rezultati odmeravanja zvučnog signala



Hvala na pažnji!

PITANJA?