Kramers-Kronig receivers: supplementary material

ANTONIO MECOZZI,[1,*] CRISTIAN ANTONELLI,[1] AND MARK SHTAIF[2]

[1] *Department of Physical and Chemical Sciences, University of L'Aquila, L'Aquila 67100, Italy*
[2] *School of Electrical Engineering, Tel Aviv University, Tel Aviv 69978, Israel*
[*] *antonio.mecozzi@univaq.it*

Abstract: This document provides supplementary information to [1]. It gives the MatLab codes used to generate the figures of [1].

1. The MatLab code implementing the field reconstruction algorithm with over-sampling

The attached MatLab code was used to generate Figs. 2 – 6, 8 and 13 of [1].

1.1. Input parameters

The input parameters are (please refer to the paper [1]):

1. `M` is the variable $M = BT$, where B is the bandwidth of the signal $a_s(t)$ and T its the period;
2. `xi_dB` is carrier-to-signal power ratio (CSPR) ξ in dB, namely $10\log_{10}(\xi)$;
3. `Ngap` is the number of spectral components set to zero around the carrier. If `Ngap = M/2` there is no overlap between the spectra of the data-carrying signal $2A\mathrm{Re}[a_s(t)]$ and of the SSBN $|a_s(t)|^2$.
4. `n_o` is the up-sampling ratio n_o;
5. `Eps` is the error ϵ in the estimate of the DC component of the intensity. For AC coupling set to zero.

1.2. Figures

1. Figure 1 gives the plot in the complex plane of the upsampled signal $a_s(t)$ (red dashed line) and of the minimum phase waveform (blue solid line);
2. Figure 2 gives with red circles are the zeros of $a_s(t)$, blue crosses are the zeros of the minimum phase waveform. The dashed circle of radius ρ, defined by Eq. (11) of [1], gives the geometric mean of the amplitudes of all zeros;
3. Figure 3 gives the spectrum of the original sequence (red dashed line) and of the minimum phase sequence (blue solid line) in a semilogarithmic scale;
4. Figures 4, 5 and 6 give the sampled original sequence (red circles connected by red dashed lines) and the minimum phase sequence (blue circles connected by blue solid lines); Figure 4 is a plot in the complex plane, whereas Figure 5 and 6 of the real and imaginary parts respectively;
5. Figure 7 is the plot of the spectra (amplitude of the Fourier transform) of the data-carrying signal $2A\mathrm{Re}[a_s(t)]$ (blue) and of the SSBN $|a_s(t)|^2$ (red);

6. Figure 8 is the same as Figure 4, but obtained by the procedure described at the end of Section 9 of [1], namely by performing the inverse Fourier transform with M points, after summing aliases copies of the upsampled spectrum of the logarithm of the minimum phase signal.

1.3. Printouts

1. `Eps` is the error ϵ in the estimate of the DC component of the intensity.
2. `CSPR` is the CSPR;
3. `Estimated CSPR_eps` is the CSPR of the signal reconstructed by the KK procedure, different from the CSPR when `Eps` $\neq 0$;
4. `Theoretical CSPR_eps/CSPR` is the ratio of the CSPR of the signal reconstructed by the KK procedure and the actual CSPR as given by the ratio ξ'/ξ obtained by Eq. (70) of [1];
5. `P_s_reconstructed/P_s` is the ratio between the signal power of the reconstructed waveform and the signal power;
6. `Theoretical P_s_reconstructed/P_s` is the ratio between the signal power of the reconstructed waveform and the signal power obtained by Eq. (71) of [1];
7. `min(abs(Z_k))` is the minimum of all zeros of the signal $a_s(t)$, which should be larger than one if the signal is of minimum phase.

```
%
% Kramers Kronig reconstruction of minimum phase Gaussian waveforms
% Code color used in the figures: Blue - reconstructed; Red - original
%
clear;
%
close all;
%
clc;
%
% M = BT, number of samples; T, period
%
M = 2^6;
%
% xi_dB = carrier power/signal power, in dB
%
xi_dB = 8.5;
%
xi = 10^(xi_dB/10);
%
M2 = 2*M;
%
% n_o = upsampling ratio. n_o = 1 no upsampling.
% n_o = 3 is the minimum for good performance
% n_o = 2^6 continuous case
%
n_o = 2^6;
%
Nh = 2*n_o; % Number of samples per symbol
%
%
% Field generation: Gaussian ditribution
```

```
%
af = [randn(1,M)+1i*randn(1,M) zeros(1,M)];
%
% Ngap = 0, no gap; Ngap = Ntot/2 half bandwidth (Lowery)
%
Ngap = 0*floor(M/2);
%
af(1:Ngap) = zeros(1,Ngap); %
%
as = ifft(af);
%
as = as - mean(as);
%
as = as/mean(abs(as).^2)^.5;
%
carrier = sqrt(xi);
%
at = as + carrier;
%
at = at*exp(-1i*mean(angle(at)));
%
af = fft(at);
%
% CSPR: Carrier to signal power ratio
%
CSPR = abs(af(1))^2/sum(abs(af(2:end)).^2);
%
sig = as*carrier + conj(as)*carrier;
%
ssbn = abs(as).^2;
%
Iaat = abs(at).^2;
%
Iaaf = fftshift(fft(Iaat));
%
% Setting of the the AC coupling
%
% Eps: error in the DC intensity if AC coupling is used. Set = 0 with DC
% coupling
%
Eps = 0;
%
Iaaf(M+1) = (1 + Eps)*Iaaf(M+1);
%
C_eps = (xi+2+1/xi)*(1+Eps)^2/2-1;
%
xi_eps = C_eps + sqrt(C_eps^2-1);
%
% End of setting the AC coupling
%
% Oversampling
%
if Nh > 2
    Iaato = ifft([Iaaf(end/2+1:end) Iaaf(1)/2 ...
        zeros(1,(Nh-2)*M-1) Iaaf(1)/2 Iaaf(2:end/2)]);
else
    Iaato = ifft([Iaaf(end/2+1:end) Iaaf(1:end/2)]);
end
%
% End of oversampling
%
% FFT of the log of the oversampled intensity
%
```

```
Lkr = 0.5*fft(log(abs(Iaato)));
%
% Upsampled complex curves
%
Lk = [1 2*ones(1,M*Nh/2-1) 1 zeros(1,M*Nh/2-1)].*Lkr;
%
atmp = exp(ifft(Lk))*sqrt(Nh/2);
%
afmp = fft(atmp);
%
afmp = afmp(1:M);
%
% Estimated signal scaling with imperfect carrier estimation
%
% CSPR_eps = CSPR estimated from the reconstructed waveform
%
CSPR_eps = abs(afmp(1))^2/sum(abs(afmp(2:end)).^2);
%
disp(['Eps = ', num2str(Eps)])
%
disp(['CSPR = ', num2str(CSPR)])
%
disp(['Estimated CSPR_eps = ', num2str(CSPR_eps)])
%
disp(['CSPR_eps/CSPR = ',num2str(CSPR_eps/CSPR)])
%
disp(['Theoretical CSPR_eps/CSPR = ', num2str(xi_eps/xi) ])
%
signal_scaling  = (xi_eps/xi)^.5;
signal_scaling_sim = sum(abs(af(2:end)*Nh/2).^2)/sum(abs(afmp(2:end)).^2);
%
disp(['P_s_reconstructed/P_s = ', num2str(1/signal_scaling_sim)])
%
disp(['Theoretical  P_s_reconstructed/P_s = ', num2str(1/signal_scaling)])
%
% End of estimated signal scaling
%
af = fft(at);
%
afs = fftshift(af);
%
ato = ifft([afs(end/2+1:end) ...
    zeros(1,(Nh-2)*M-1) afs(1:end/2)])*Nh/2;
%
%
atmp1 = (atmp-mean(atmp))*sqrt(signal_scaling_sim) + mean(ato);
%
figure(1); plot(real(atmp1), imag(atmp1),'- b',...
    real(ato), imag(ato), '-- r',...
    'MarkerSize', 6, 'LineWidth', 2);
set(gca,'fontsize',20, 'LineWidth',2);
grid on;
%
% End upsanpled complex curves
%
% Roots in the complex plane
%
Ra = sort(roots(af(1:M)));
%
Ra = 1./conj(Ra);
%
disp(['min(abs(Z_k)) = ', num2str(min(abs(Ra))),...
    '; Minimun phase if min(abs(Z_k)) > 1']);
```

```
%
Ram = sort(roots(afmp));
%
Rap = 1./conj(Ram);
%
rds = abs(af(1)/af(M))^(1/(M-1));
%
figure(2); plot(real(Ra), imag(Ra), 'o r', ...
    real(Rap), imag(Rap), 'x b', ...
    cos(0:0.01:2*pi),sin(0:0.01:2*pi), '-b',...
    rds*cos(0:0.01:2*pi),rds*sin(0:0.01:2*pi), '--r',...
    'MarkerSize', 6, 'LineWidth', 1);
axis('equal');set(gca,'fontsize',20, 'LineWidth',2);
grid on;
%
% End of roots in the complex plane
%
% Sampled curves
%
Lk = [1 2*ones(1,M*Nh/2-1) 1 zeros(1,M*Nh/2-1)].*Lkr;
%
atmp = exp(ifft(Lk));
%
afmp = fft(atmp);
%
atmp = ifft(afmp(1:M2));
%
% Spectra
%
figure(3);semilogy(1:M2, ...
    abs(fftshift(afmp(1:M2)))/(mean(abs(afmp(1:M2)).^2))^.5,'r--', ...
    1:M2, abs(fftshift(af(1:M2)))/(mean(abs(af(1:M2)).^2))^.5, 'b-', ...
    'MarkerSize', 6, 'LineWidth', 2);  xlim([1 M2])
set(gca,'fontsize',20, 'LineWidth',2);
grid on;
%
% End spectra
%
afmp = afmp(1:M);
%
atmp = atmp/sqrt(Nh/2);
%
atmp1 = (atmp-mean(atmp))*sqrt(signal_scaling_sim) + mean(at);
%
figure(4); plot(real(atmp1), imag(atmp1),'-o b',...
    real(at), imag(at), '--o r',...
    'MarkerSize', 6, 'LineWidth', 2);
set(gca,'fontsize',20, 'LineWidth',2);
grid on;
%
% End sampled curves
%
atmp1 = (atmp-mean(atmp))*sqrt(signal_scaling_sim);
%
at1 = at - mean(at);
%
figure(5); plot(1:M2,real(at1),'- o b',1:M2,real(atmp1),'--o r', ...
    'MarkerSize', 6, 'LineWidth', 1); xlim([0 M2])
set(gca,'fontsize',20, 'LineWidth',2);
%
figure(6); plot(1:M2,imag(at1),'-o b',1:M2,imag(atmp1),'--o r', ...
    'MarkerSize', 6, 'LineWidth', 1); xlim([0 M2])
set(gca,'fontsize',20, 'LineWidth',2);
```

```
%
figure(7); plot(-M:M-1, abs(fftshift(fft(sig))),'b',...
    -M:M-1, abs(fftshift(fft(ssbn))),'r', ...
    'MarkerSize', 6, 'LineWidth', 2); xlim([-M M-1])
set(gca,'fontsize',20, 'LineWidth',2);
%
% End spectra
%
if Nh > 2
    %
    % Downsampling to 1 sample/symbol before IFFT and exponentiating
    %
    Lk = [1 2*ones(1,M*Nh/2-1) 1 zeros(1,M*Nh/2-1)].*Lkr;
    %
    Lka = zeros(1,M*Nh);
    %
    Nhmin = Nh/2-2; % Rigorous, but not necessary
    %
    for nn = Nh-Nhmin:Nh
        %
        Lka = Lka + Lk([M*(Nh-nn)+1:M*Nh 1:M*(Nh-nn)]);
        %
    end
    %
    Lk = Lka(1:M)/Nh;
    %
    atmp = exp(ifft(Lk));
    %
    atmp = atmp*sqrt(Nh/2);
    %
    at = at(1:2:M2);
    %
    atmp1 = (atmp-mean(atmp))*sqrt(signal_scaling_sim) + mean(at);
    %
    figure(8); plot(real(atmp1), imag(atmp1),'-o b',...
        real(at), imag(at), '--o r',...
        'MarkerSize', 6, 'LineWidth', 2);
    set(gca,'fontsize',20, 'LineWidth',2);
    grid on;
    %
    % End downsampling to M before exponentiating
    %
end
```

2. The MatLab code implementing the field reconstruction algorithm without oversampling

The attached MatLab code was used to generate Figs. 9 – 12 of the paper [1].

2.1. *Input parameters*

The input parameters are (please refer to the paper [1]):

1. `M` is the variable $M = BT$, where B is the bandwidth and T is the period;
2. `xi_dB` is carrier-to-signal power ratio ξ in dB, $10\log(\xi)$;
3. `n_o` is the up-sampling ratio n_o, fixed to `n_o = 1` in this program;
4. `cnt` is a control string, that can assume the values `'KK_corr'`, `'KK_nocorr'` and `'noKK'`. When `cnt = 'KK_corr'`, the mitigation algorithm of the phase noise

introduced by the absence of upsampling, proposed in Section 10 of [1], is applied. Figures 9 – 11 of [1] have been obtained with this setting. When `cnt = 'KK_nocorr'` no correction of the phase noise added on the reconstructed signal by the absence of upsampling is applied. Figure 12 of [1] has been obtained with this setting. When `cnt = 'noKK'`, a simple decoding without any mitigation of the signal-signal beat interference is applied. Figure 13 of [1] has been obtained with this setting.

2.2. Figures

1. Figures 1, 2 and 3 give the sampled original sequence (red circles connected by red dashed lines) and the minimum phase sequence (blue circles connected by blue solid lines); Figure 1 is a plot in the complex plane, whereas Figure 2 and 3 are the real and imaginary parts respectively;

2. Figure 4 gives the spectrum of the original sequence (red dashed line) and of the minimum phase sequence (blue solid line) in a semilogarithmic scale;

3. Figure 5 gives with red circles are the zeros of $a_s(t)$, blue crosses are the zeros of the minimum phase waveform. The dashed circle of radius ρ, defined by Eq. (11) of [1], gives the geometric mean of the amplitudes of all zeros.

2.3. Printouts

1. `min(abs(Z_k))` is the minimum of all zeros of the signal $a_s(t)$, which should be larger than one if the signal is of minimum phase.

```
%
% Kramers Kronig reconstruction of minimum phase waveforms
% with no oversampling
%
% clear;
%
close all;
%
clc;
%
% M = BT, number of samples; T, period
%
M = 2^6;
%
% xi_dB = carrier power/signal power, in dB
%
xi_dB = 13;
%
xi = 10^(xi_dB/10);
%
M2 = 2*M;
%
% Field generation: Gaussian ditribution
%
af = [randn(1,M)+1i*randn(1,M) zeros(1,M)];
%
as = ifft(af);
%
as = as - mean(as);
%
as = as/mean(abs(as).^2)^.5;
%
```

```
carrier = sqrt(xi);
%
at = as + carrier;
%
at = at*exp(-1i*mean(angle(at)));
%
af = fft(at);
%
disp(['CSPR = ', num2str(abs(af(1))^2/sum(abs(af(2:end)).^2))])
%
sig = as*carrier + conj(as)*carrier;
%
ssbn = abs(as).^2;
%
% n_o = upsampling ratio. n_o = 1 no upsampling.
%
n_o = 1; % DO NOT CHANGE FROM n_o = 1
%
Nh = 2*n_o; % Number of samples per symbol
%
Iaat = abs(at).^2;
%
Iaaf = fftshift(fft(Iaat));
%
% Options: cnt = KK_corr, KK_nocorr, noKK.
% KK_corr, the KK is applied with the correction decribed in the paper
% KK_nocorr, the KK is applied with no correction
% noKK: there is no cancellation of the SSBI
%
cnt = 'KK_corr';
%
if strcmp(cnt,'KK_corr') % First order correction of the phase error
    %
    Lkr = 0.5*fft(log(abs(Iaat)));
    %
    Lk = [1 2*ones(1,M*Nh/2-1) 1 zeros(1,M*Nh/2-1)].*Lkr;
    %
    atmp = exp(ifft(Lk));
    %
    afmp = fft(atmp);
    %
    % First order correction of the phase error caused by the absence
    % of oversampling
    %
    afmp = [afmp(1) afmp(2:M)+conj(afmp(M2:-1:M+2)) zeros(1,M)];
    %
    atmp = ifft(afmp);
    %
    % Intensity renormalization
    %
    atmp = (Iaat).^0.5.*atmp./abs(atmp);
    %
elseif strcmp(cnt,'KK_nocorr') % No first order correction
    %
    Lkr = 0.5*fft(log(abs(Iaat)));
    %
    Lk = [1 2*ones(1,M*Nh/2-1) 1 zeros(1,M*Nh/2-1)].*Lkr;
    %
    atmp = exp(ifft(Lk));
    %
elseif strcmp(cnt,'noKK') % Decoding without KK
    %
    Lkr = fft(Iaat);
```

```
    %
    Lk = [ones(1,M*Nh/2) zeros(1,M*Nh/2)].*Lkr;
    %
    atmp = ifft(Lk)/sqrt(Lkr(1)/M2);
    %
end
%
figure(1); plot(real(at), imag(at),'- o b',...
    real(atmp), imag(atmp), '-- o r',...
    'MarkerSize', 6, 'LineWidth', 2);
axis('equal'); set(gca,'fontsize',20, 'LineWidth',2);
grid on;
%
figure(2); plot(1:M2,real(at),'- o b',1:M2,real(atmp),'--o r', ...
    'MarkerSize', 6, 'LineWidth', 1); xlim([0 M2])
set(gca,'fontsize',20, 'LineWidth',2);
%
figure(3); plot(1:M2,imag(at),'-o b',1:M2,imag(atmp),'--o r', ...
    'MarkerSize', 6, 'LineWidth', 1); xlim([0 M2])
set(gca,'fontsize',20, 'LineWidth',2);
%
afmp = fft(atmp);
%
figure(4);
semilogy(1:M2, abs(fftshift(afmp))/(mean(abs(afmp).^2))^.5,'--o r', ...
    1:M2, abs(fftshift(af))/(mean(abs(af).^2))^.5,'-o b', ...
    'MarkerSize', 6, 'LineWidth', 1); xlim([0 M2])
set(gca,'fontsize',20, 'LineWidth',2); grid on;
%
% Roots in the complex plane
%
Ra = sort(roots(af(1:M)));

Ra = 1./conj(Ra);
%
disp(['min(abs(Z_k)) = ', num2str(min(abs(Ra))),...
    '; Minimun phase if min(abs(Z_k)) > 1']);
%
Ram = sort(roots(afmp(1:M)));
%
Rap = 1./conj(Ram);
%
rds = abs(af(1)/af(M))^(1/(M-1));
%
figure(5); plot(real(Ra), imag(Ra), 'o r', ...
    real(Rap), imag(Rap), 'x b', ...
    cos(0:0.01:2*pi),sin(0:0.01:2*pi), '-b',...
    rds*cos(0:0.01:2*pi),rds*sin(0:0.01:2*pi), '--r',...
    'MarkerSize', 6, 'LineWidth', 1);
axis('equal');set(gca,'fontsize',20, 'LineWidth',2);
grid on;
%
% End of roots in the complex plane
%
```

References

1. A. Mecozzi, C. Antonelli, and M. Shtaif, “Kramers-Kronig receivers,” Advanced in Optics and Photonics (2019).