%PTC 5005 - 2019

%Prof: Maria D. Miranda

%Aluno: Stéfano Albino Vilela Rezende (Ouvinte)

%Exercício Computacional - Filtros Compensadores

%1)b)

N=100;

n=[0:N-1]

x=(.8).^(n);

%Filtro de distorção

A=conv([1 -.95],[1 -.98]);

B=[1 -1/2];

%Filtro compensador

Ac=[1 -1/2];

Bc=conv([1 -.95],[1 -.98]);

y=filter(B,A,x); %Sinal corrompido

x\_est=filter(Bc,Ac,y); %Sinal recuperado

figure(1)

subplot(211); stem(x(1:N)); title('x(n)'); grid

subplot(212); stem(x\_est(1:N)); title('x\_{est}(n)'); grid

figure(2)

subplot(211); plot(abs(fft(x)),'linewidth',2); title('|X(k)|^2'); grid

subplot(212); plot(abs(fft(x\_est)),'linewidth',2); title('|X\_{est}(k)|^2'); grid

figure(3)

subplot(211); plot(angle(fft(x)),'linewidth',2); title('Fase'); grid

subplot(212); plot(angle(fft(x\_est)),'linewidth',2); title('Fase'); grid

%PTC 5005 - 2019

%Prof: Maria D. Miranda

%Aluno: Stéfano Albino Vilela Rezende (Ouvinte)

%Exercício Computacional - Filtros Compensadores

%1)b)

N=100;

n=[0:N-1]

x=(.8).^(n);

%Filtro de distorção

A=conv([1 -.95],[1 -.98]);

B=[1 -1/2];

%Filtro compensador

Ac=[1 -1/2];

Bc=conv([1 -.95],[1 -.98]);

y=filter(B,A,x); %Sinal corrompido

x\_est=filter(Bc,Ac,y); %Sinal recuperado

figure(1)

subplot(211); stem(x(1:N)); title('x(n)'); grid

subplot(212); stem(x\_est(1:N)); title('x\_{est}(n)'); grid

figure(2)

subplot(211); plot(abs(fft(x)),'linewidth',2); title('|X(k)|^2'); grid

subplot(212); plot(abs(fft(x\_est)),'linewidth',2); title('|X\_{est}(k)|^2'); grid

figure(3)

subplot(211); plot(angle(fft(x)),'linewidth',2); title('Fase'); grid

subplot(212); plot(angle(fft(x\_est)),'linewidth',2); title('Fase'); grid

Universidade de São Paulo

Escola Politécnica

Programa de Pós-Graduação em Engenharia Elétrica

PTC5005 – Processamento Digital de Sinais

Exercício Computacional – Filtros Compensadores

Aluno: Stéfano Albino Vilela Rezende

NUSP: “Aluno Ouvinte”

Professor: Maria D. Miranda

São Paulo

2019

**1)****a)** Sim, é possível projetar um filtro *Hc(z)* causal e estável que compense perfeitamente a distorção inserida por *H(z)*. Sendo:

É necessário que *Hc(z)* tenha os módulos de seus polos menores que 1 para ser estável, portanto os zeros de *H(z)* devem ter módulos menores que 1, nesse caso a condição necessária é atendida. Como pode ser visto nas *Figuras 1 e 2.*

*Quadro 1 – Script de MATLAB para o item a).*

%PTC 5005 - 2019

%Prof: Maria D. Miranda

%Aluno: Stéfano Albino Vilela Rezende (Ouvinte)

%Exercício Computacional - Filtros Compensadores

%1)a)

%Filtro de distorção

A=conv([1 -.95],[1 -.98]);

B=[1 -1/2];

%Filtro compensador

Ac=[1 -1/2]; % completar

Bc=conv([1 -.95],[1 -.98]); % completar

figure(1)

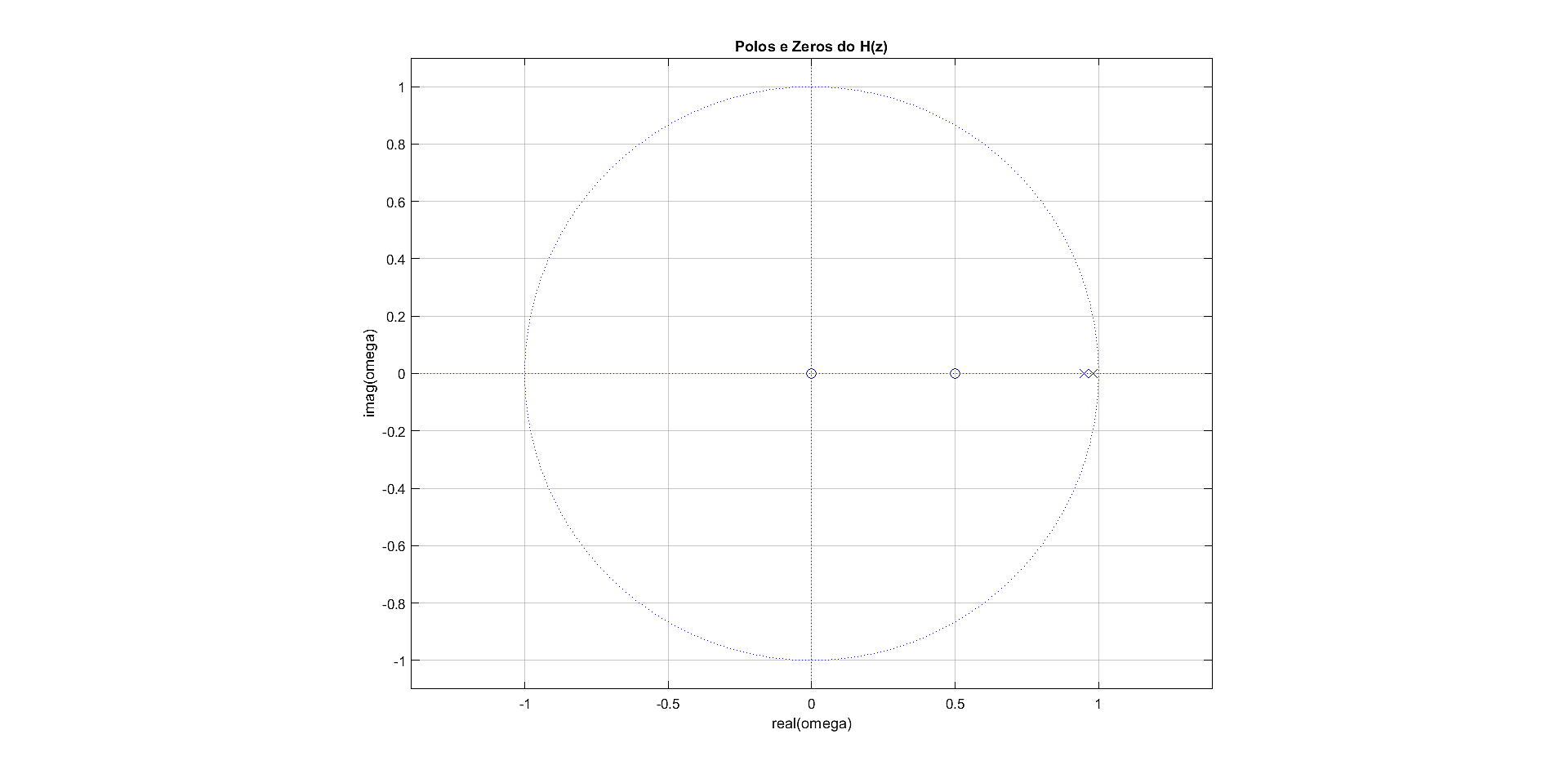
zplane(B,A);grid;title('Polos e Zeros do H(z)')

xlabel('real(omega)'); ylabel('imag(omega)')

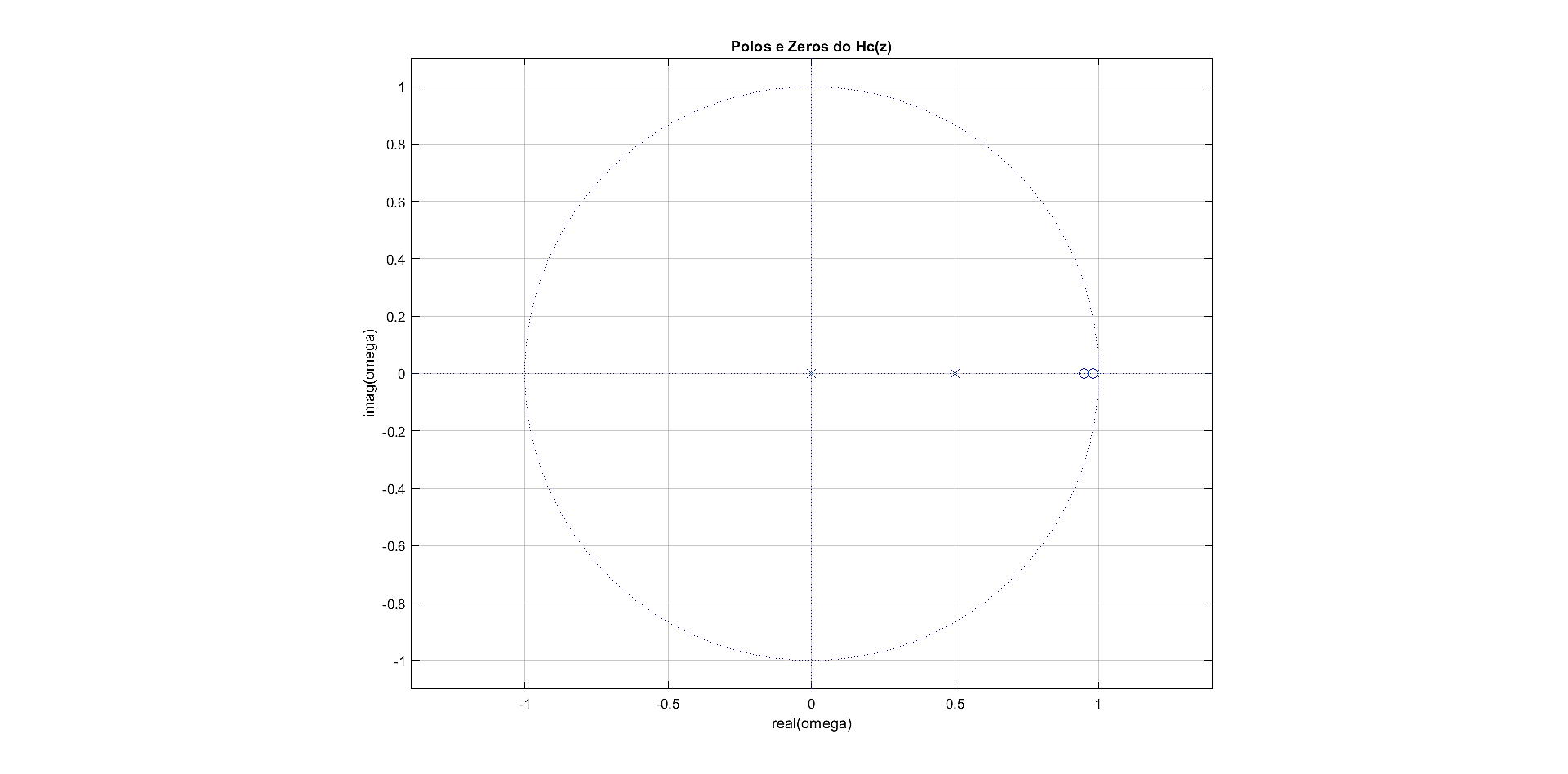
figure(2)

zplane(Bc,Ac);grid;title('Polos e Zeros do Hc(z)')

xlabel('real(omega)'); ylabel('imag(omega)')



*Figura 1 – Diagramas de Zeros e Polos de H(z).*



*Figura 2 – Diagramas de Zeros e Polos de Hc(z).*

**b)**

*Quadro 2 – Script de MATLAB para o item b)*

%PTC 5005 - 2019

%Prof: Maria D. Miranda

%Aluno: Stéfano Albino Vilela Rezende (Ouvinte)

%Exercício Computacional - Filtros Compensadores

%1)b)

N=100;

n=[0:N-1]

x=(.8).^(n);

%Filtro de distorção

A=conv([1 -.95],[1 -.98]);

B=[1 -1/2];

%Filtro compensador

Ac=[1 -1/2];

Bc=conv([1 -.95],[1 -.98]);

y=filter(B,A,x); %Sinal corrompido

x\_est=filter(Bc,Ac,y); %Sinal recuperado

figure(1)

subplot(211); stem(x(1:N)); title('x(n)'); grid

subplot(212); stem(x\_est(1:N)); title('x\_{est}(n)'); grid

figure(2)

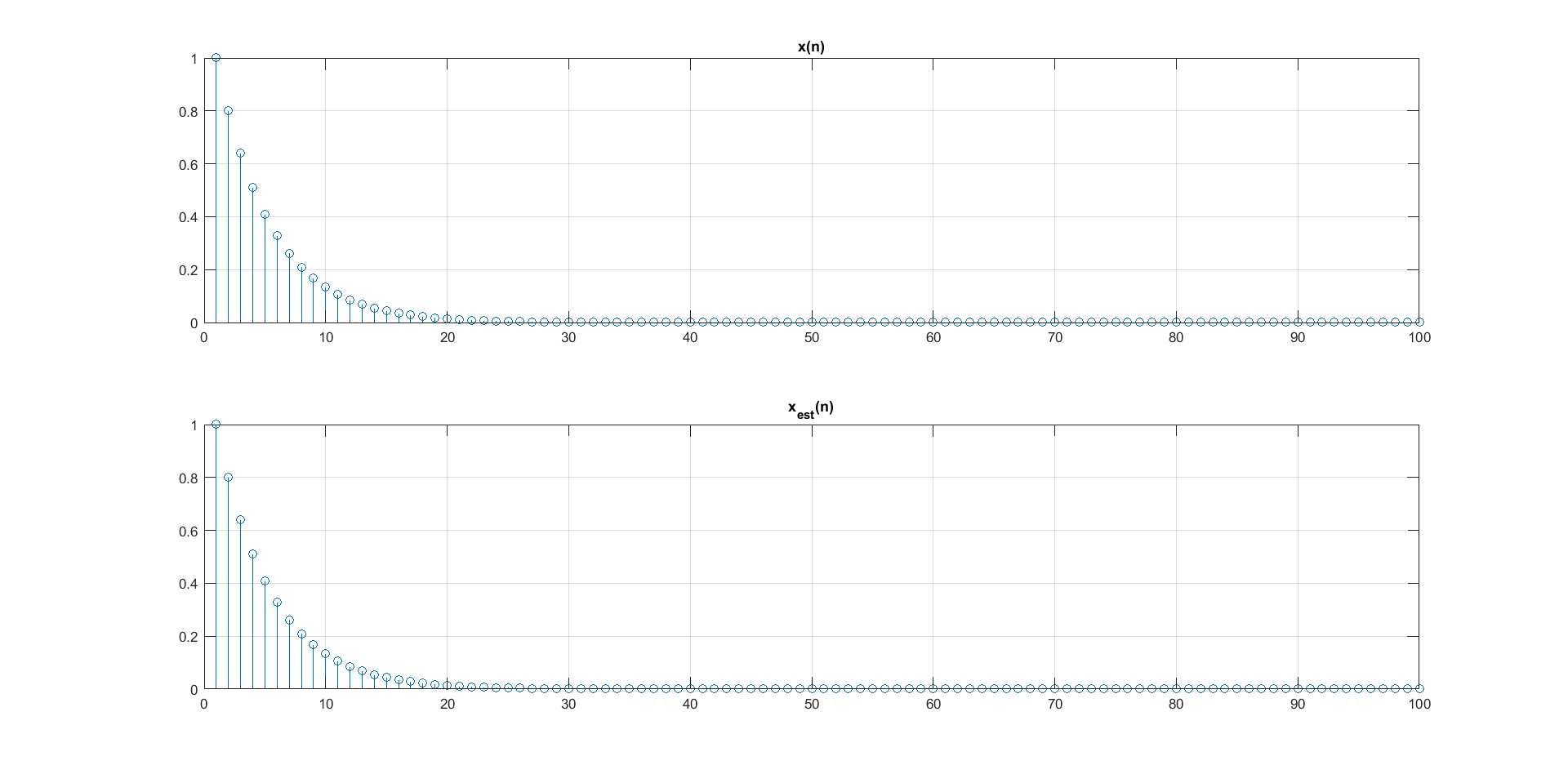
subplot(211); plot(abs(fft(x)),'linewidth',2); title('|X(k)|^2'); grid

subplot(212); plot(abs(fft(x\_est)),'linewidth',2); title('|X\_{est}(k)|^2'); grid

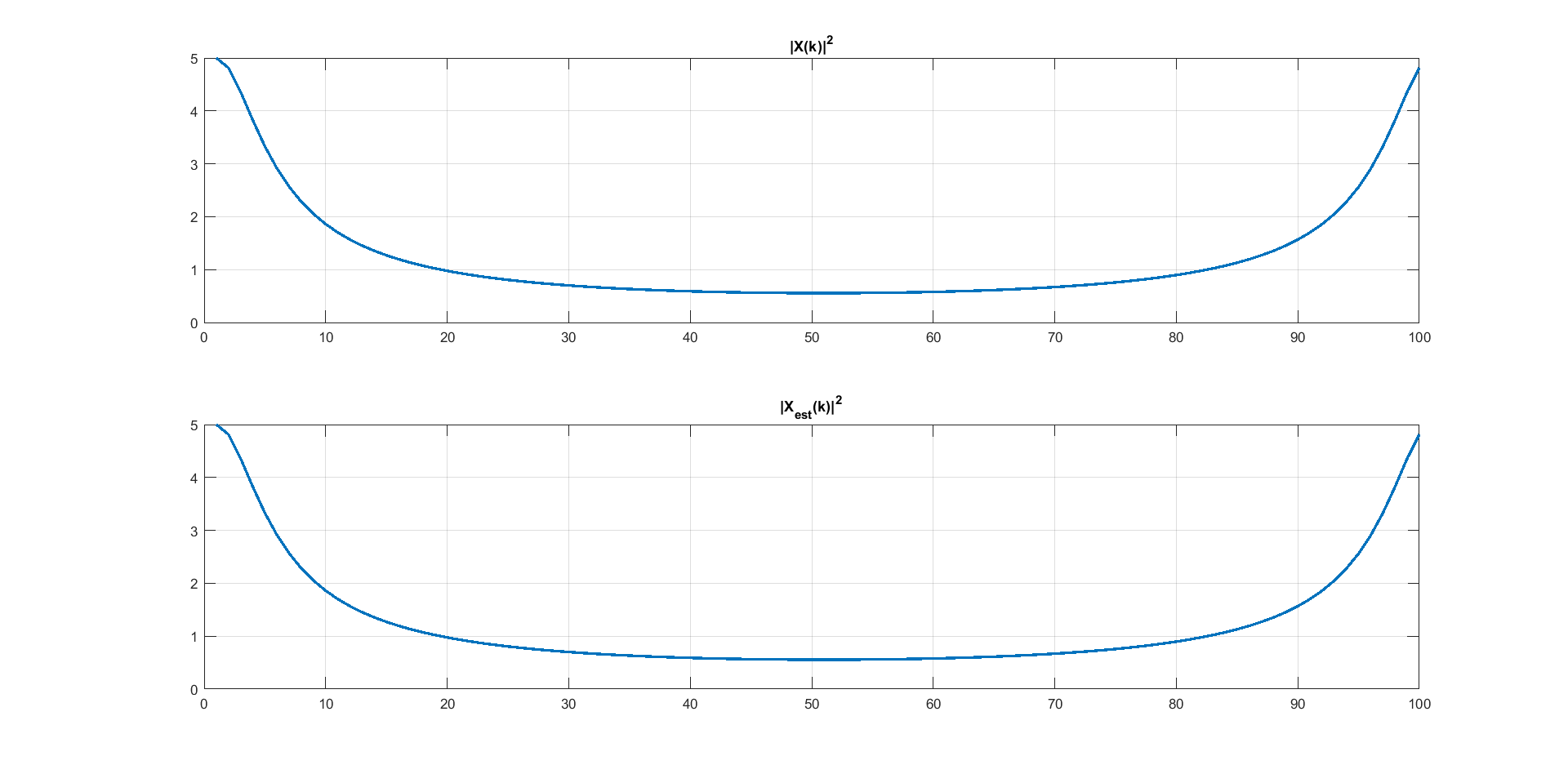
figure(3)

subplot(211); plot(angle(fft(x)),'linewidth',2); title('Fase'); grid

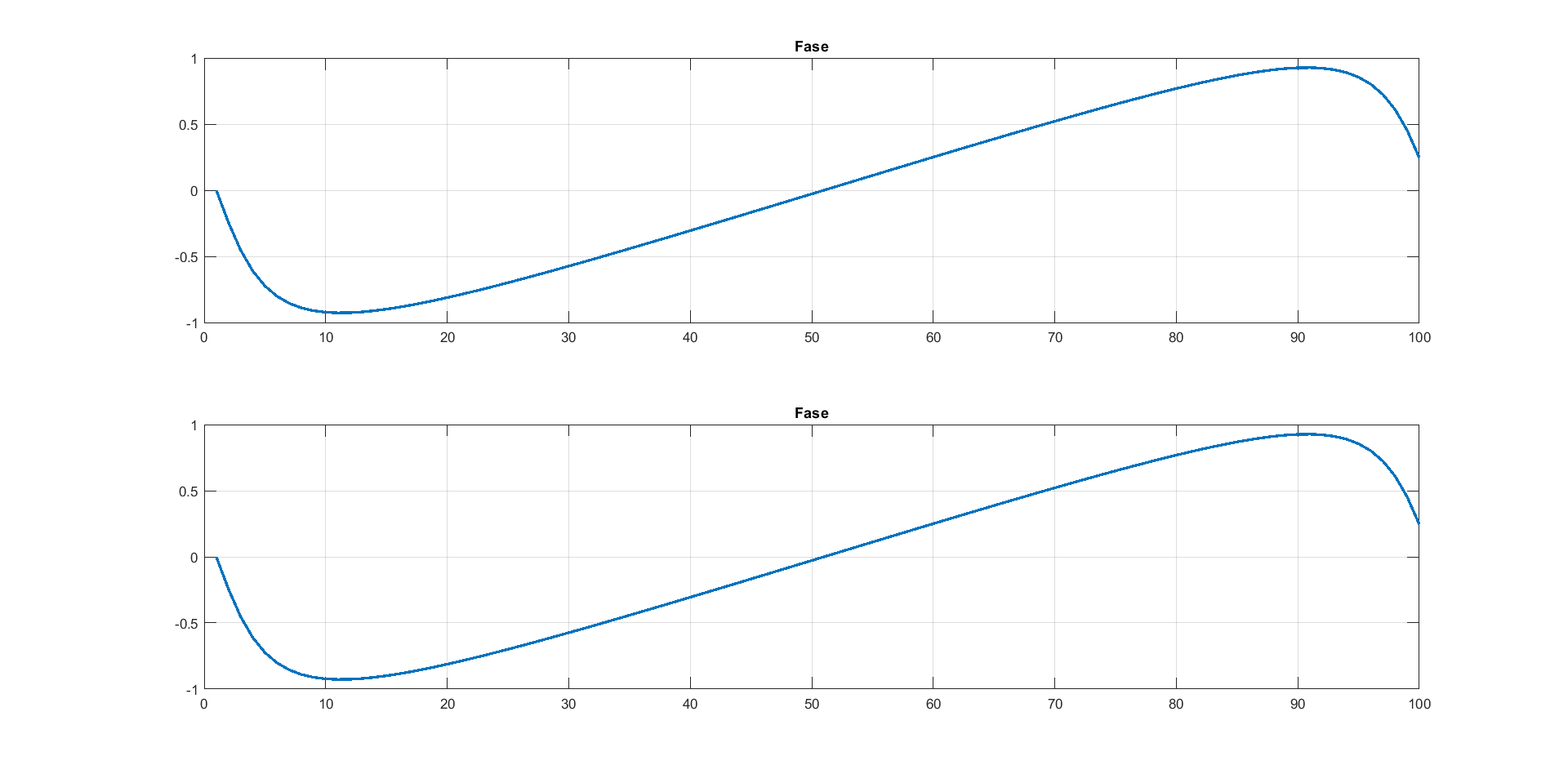
subplot(212); plot(angle(fft(x\_est)),'linewidth',2); title('Fase'); grid



*Figura 3 – Gráficos do sinal original x(n) e do sinal filtrado xest(n) .*

**

*Figura 4 – Gráficos em frequência dos módulos do sinal original x(n) e do sinal filtrado xest(n) .*

**

*Figura 5 – Gráficos em frequência das fases do sinal original x(n) e do sinal filtrado xest(n) .*

**c)**

*Quadro 3 – Script de MATLAB para o item c).*

%PTC 5005 - 2019

%Prof: Maria D. Miranda

%Aluno: Stéfano Albino Vilela Rezende (Ouvinte)

%Exercício Computacional - Filtros Compensadores

%1)c)

%Filtro de distorção

A=conv([1 -.95],[1 -.98]);

B=[1 -1/2]; % ou B=[1 -2]

%Filtro compensador

Ac=[1 -1/2];

Bc=conv([1 -.95],[1 -.98]);

x0 = audioread('Que\_maravilha\_cut.wav');

Fs=44100; % frequência de amostragem

x=x0(2000:5\*Fs,1)';

y=filter(B,A,x); %Sinal corrompido

x\_est=filter(Bc,Ac,y); %Sinal recuperado

sound(x,Fs);

sound(y,Fs);

sound(x\_est,Fs)

N=length(x);

figure(1)

subplot(211); plot(x(1:N),'.'); title('x(n)'); grid

subplot(212); plot(x\_est(1:N),'.'); title('x\_{est}(n)'); grid

figure(2)

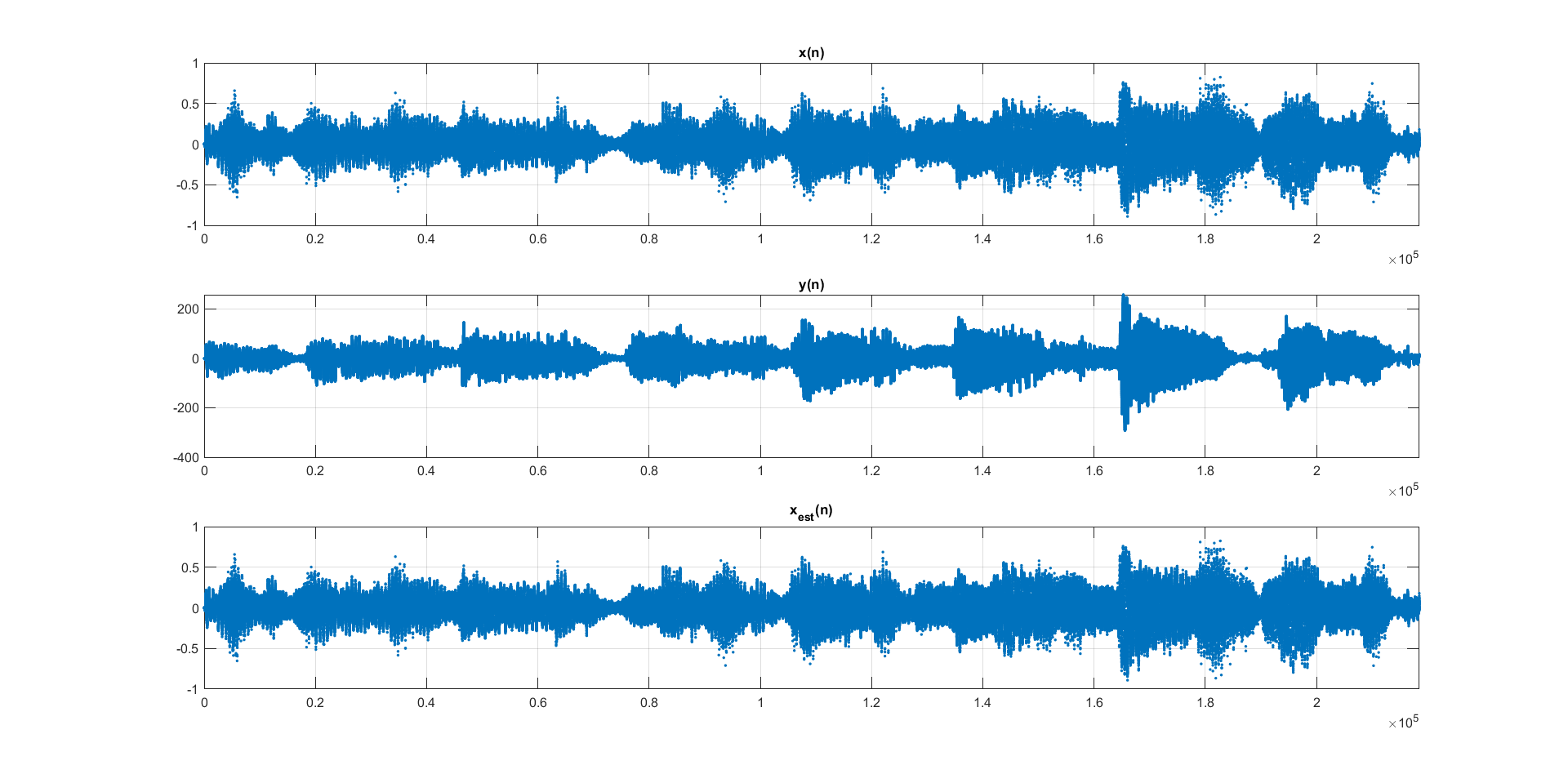
subplot(211); plot(abs(fft(x)),'linewidth',2); title('|X(k)|^2'); grid

subplot(212); plot(abs(fft(x\_est)),'linewidth',2); title('|X\_{est}(k)|^2'); grid

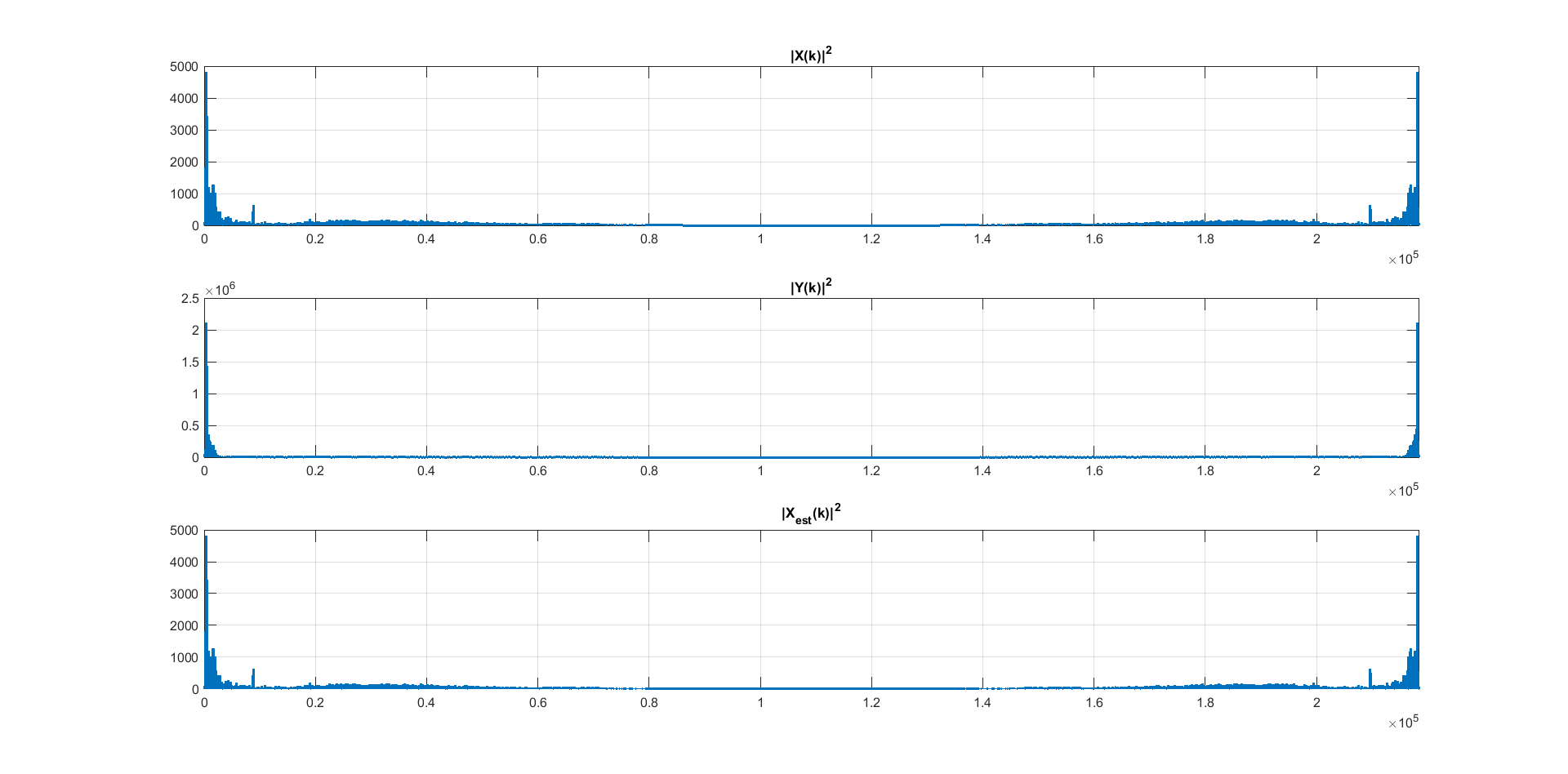
figure (3)

subplot(211); plot(angle(fft(x)),'linewidth',0.3); title('Fase'); grid

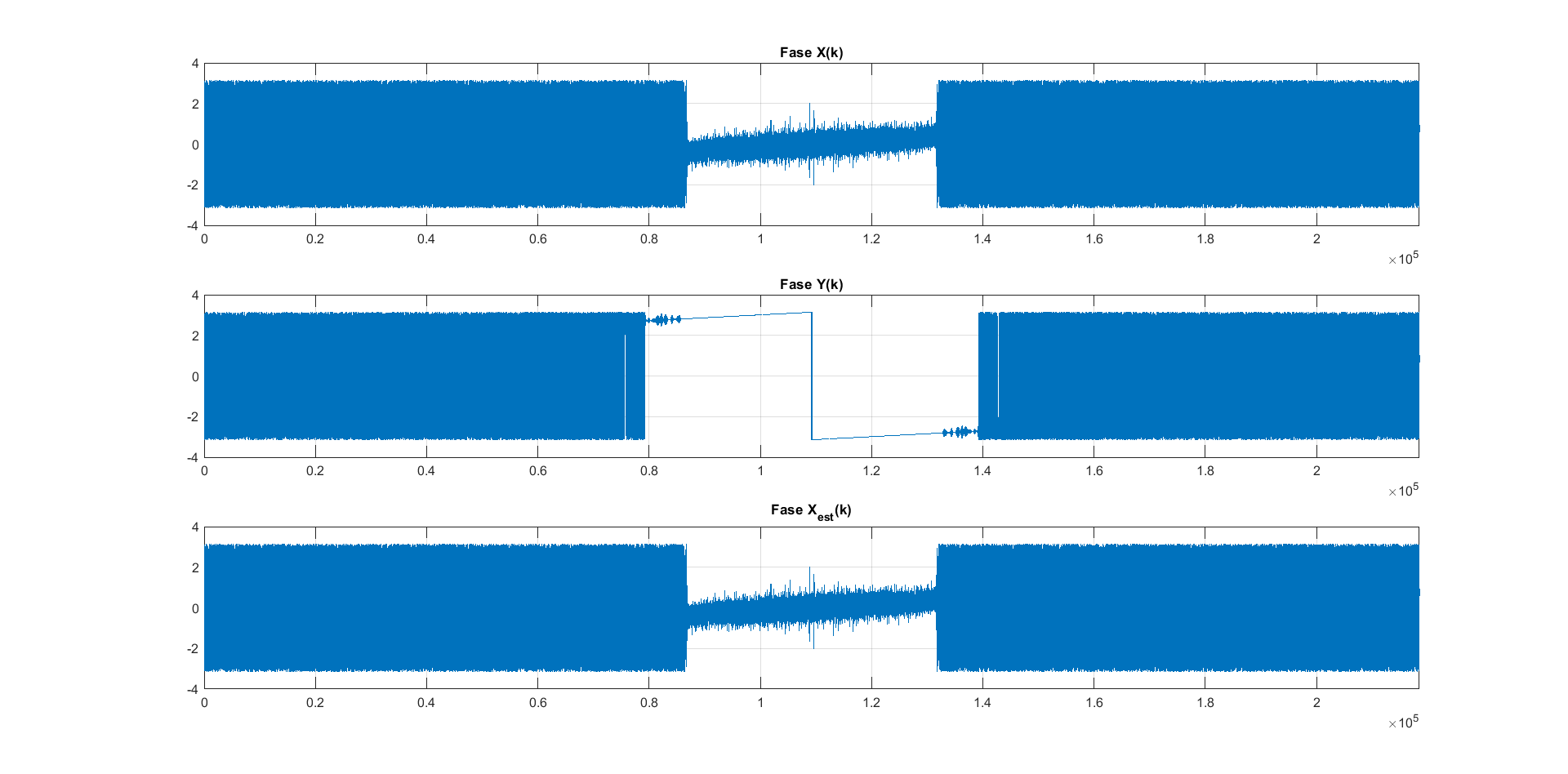
subplot(212); plot(angle(fft(x\_est)),'linewidth',0.3); title('Fase'); grid

**

*Figura 6– Gráficos do áudio original x(n), áudio distorcido y(n) e do áudio filtrado xest(n).*

**

*Figura 7 – Gráficos em frequência dos módulos do áudio original x(n), áudio distorcido y(n) e do áudio filtrado xest(n).*

**

*Figura 8 – Gráficos em frequência das fases do áudio original x(n), áudio distorcido y(n) e do áudio filtrado xest(n).*

**d)**

*Quadro 4 – Script de MATLAB para o item d).*

%PTC 5005 - 2019

%Prof: Maria D. Miranda

%Aluno: Stéfano Albino Vilela Rezende (Ouvinte)

%Exercício Computacional - Filtros Compensadores

%1)d)

%Filtro de distorção

A=conv([1 -.95],[1 -.98]);

B=[1 -1/2]; % ou B=[1 -2]

%Filtro compensador

Ac=[1 -1/2];

Bc=conv([1 -.95],[1 -.98]);

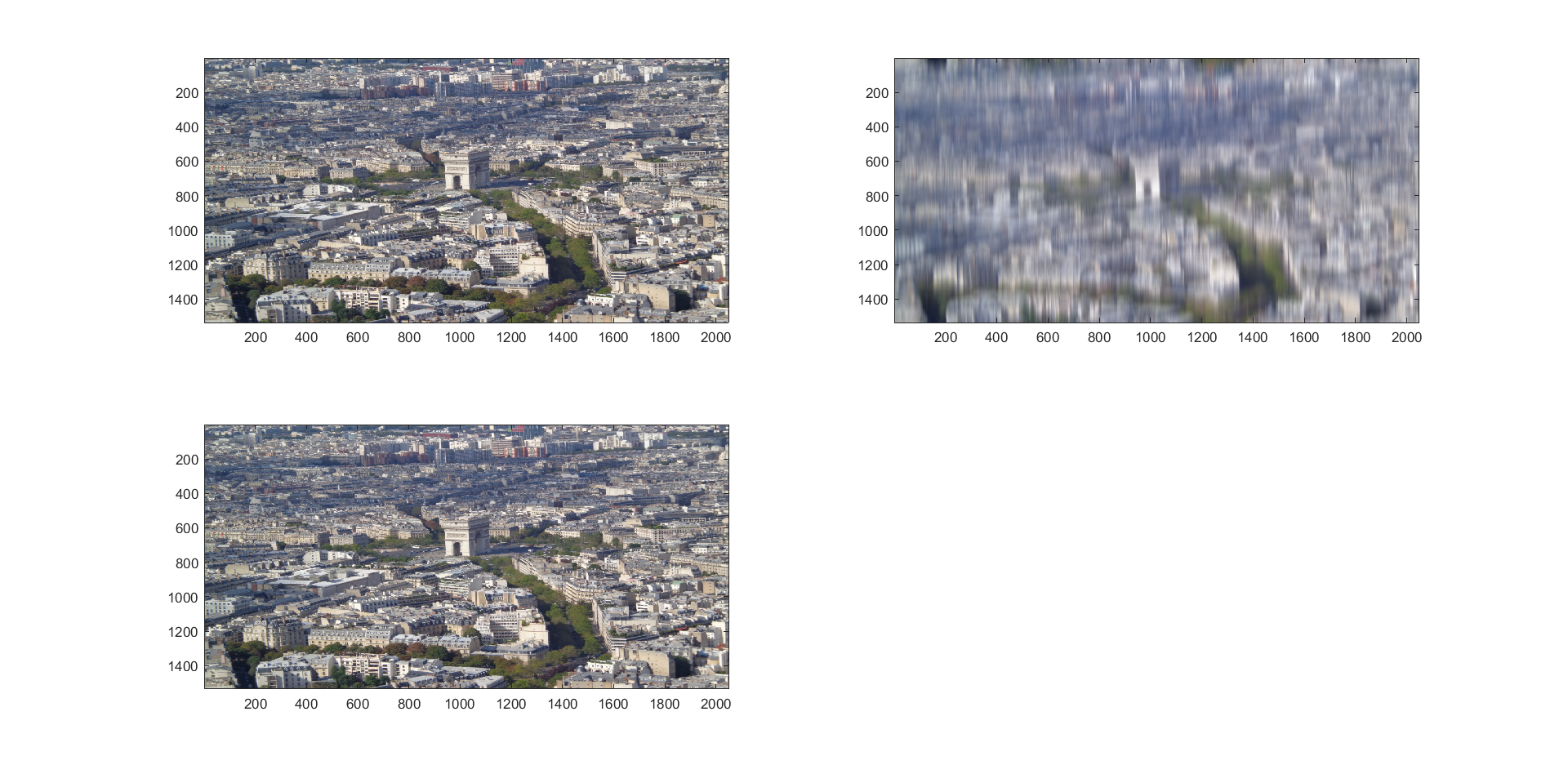
[x,y,xest]=RotImag(A,B,Ac, Bc);

figure(1)

subplot(221); image(x)

subplot(222); image(y)

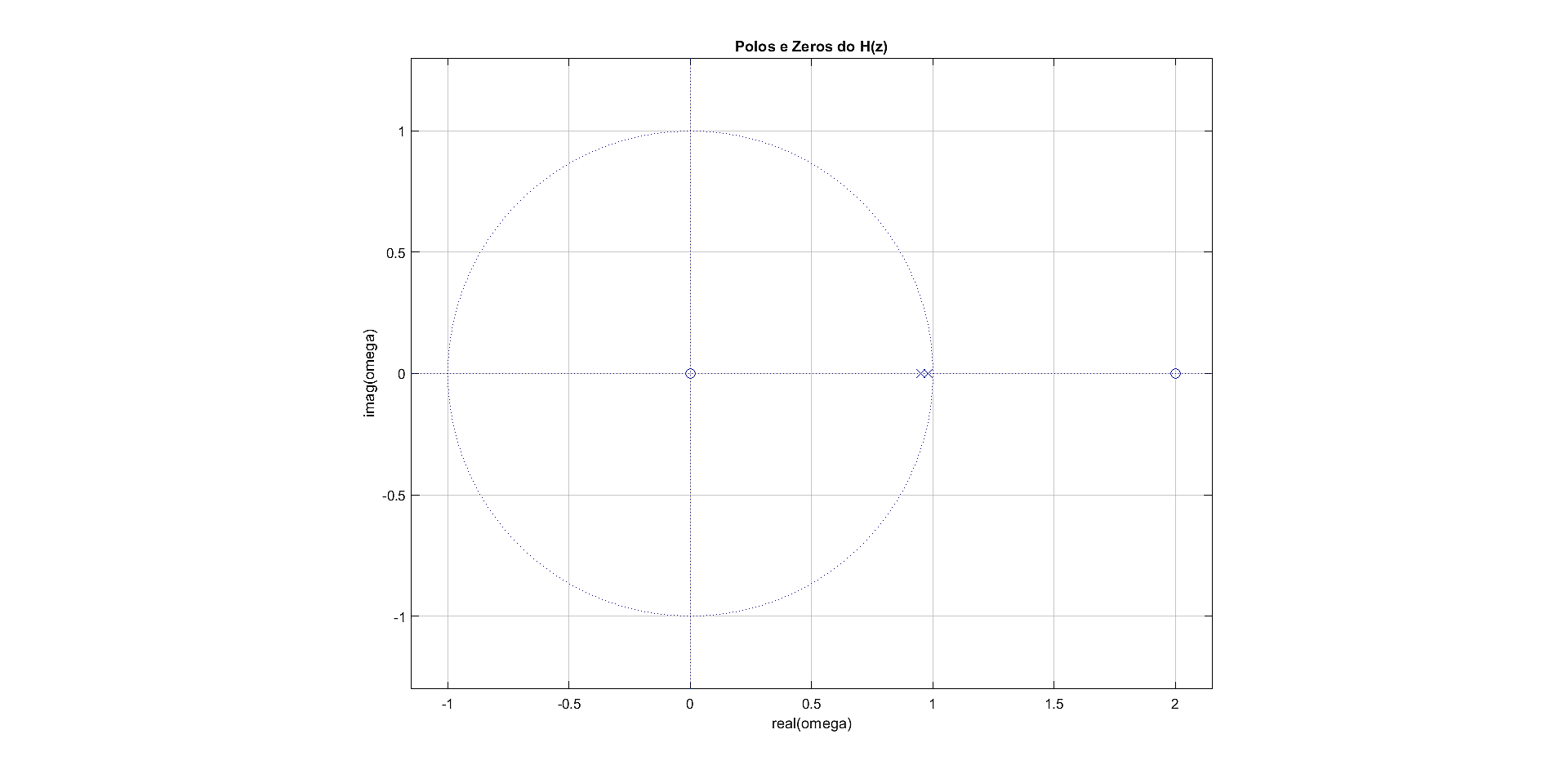
subplot(223); image(xest)

**

*Figura 9 – Reprodução da imagem arco\_triunfo.jpg original, distorcida e recuperada*

Analise a imagem original, distorcida e recuperada.

**2)****a)** Não, esse filtro não admite projetar um filtro compensador ideal causal. Pois um de seus Zeros possui módulo maior que 1, como mostrado na *Figura 1*. Gerando assim, filtro *Hc(z)* com polo fora da circunferência unitária.



*Figura 10 – Diagramas de Zeros e Polos de H(z).*

Para projetar um filtro que possa recuperar pelo menos a magnitude do sinal de entrada antes da distorção de *H(z)* pode-se adotar a seguinte estratégia:

*H(z)* tem sua equação reescrito em dois novos filtros, um passa-tudo(PT) e outro de fase mínima (FM). Sendo:

É calculado o polinômio recíproco do numerador da equação e, então o complexo conjugado do seu recíproco para construir a equação do filtro passa-tudo. Como demonstrado abaixo:

Sendo possível reescrever *H(z)* como:

Sendo:

Assim, podemos construir o filtro compensador que recuperará a magnitude do sinal de entrada e terá uma mínima alteração na fase. Como demonstrado abaixo:

**b)**

*Quadro 5 – Script de MATLAB para o item b)*

%PTC 5005 - 2019

%Prof: Maria D. Miranda

%Aluno: Stéfano Albino Vilela Rezende (Ouvinte)

%Exercício Computacional - Filtros Compensadores

%2)b)

N=100;

n=[0:N-1]

x=(.8).^(n);

%Filtro de distorção

A=conv([1 -.95],[1 -.98]);

B=[1 -2];

%Filtro compensador

Ac=[-2 1];

Bc=conv([1 -.95],[1 -.98]);

y=filter(B,A,x); %Sinal corrompido

x\_est=filter(Bc,Ac,y); %Sinal recuperado

figure(1)

subplot(211); stem(x(1:N)); title('x(n)'); grid

subplot(212); stem(x\_est(1:N)); title('x\_{est}(n)'); grid

figure(2)

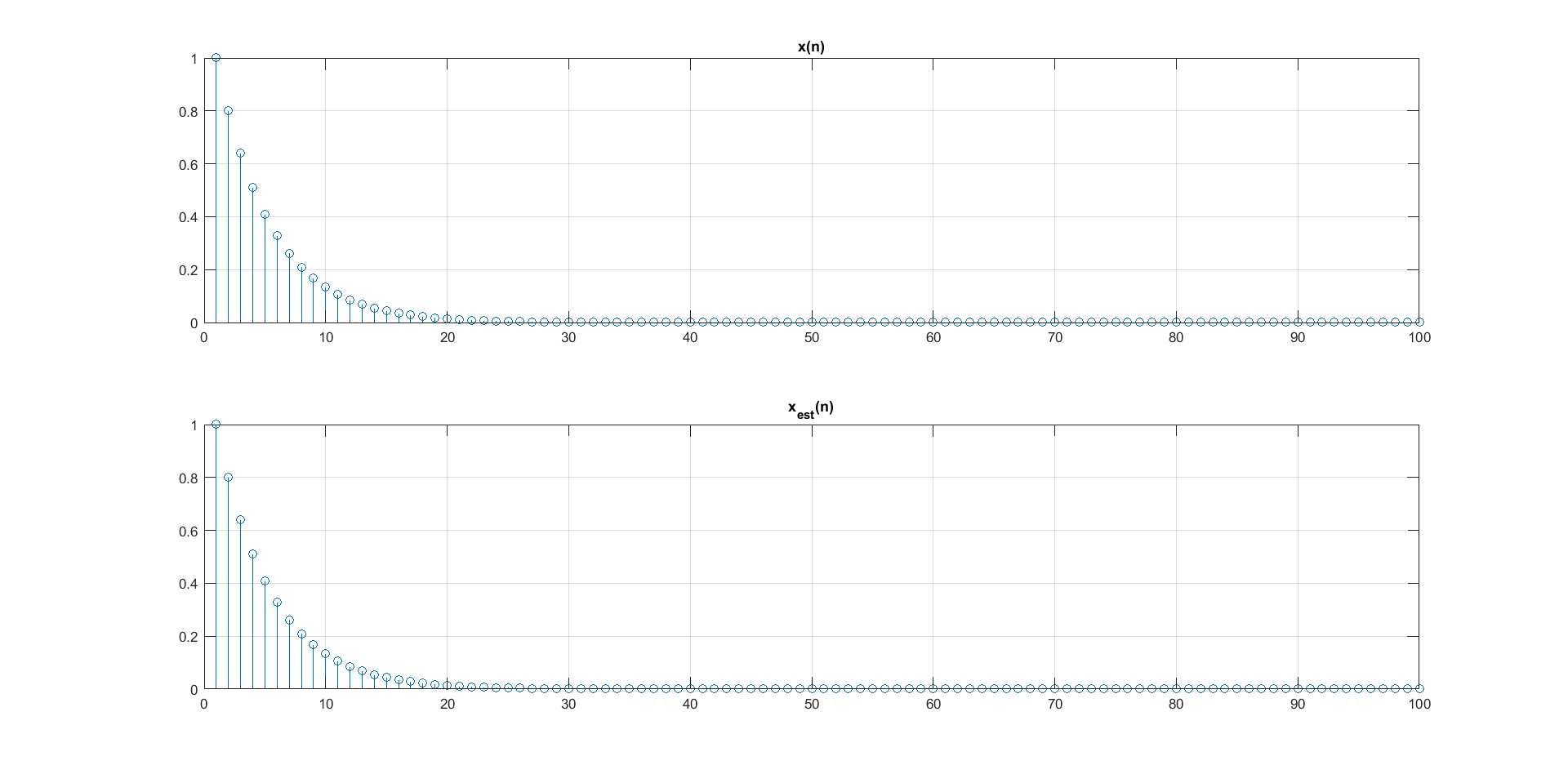
subplot(211); plot(abs(fft(x)),'linewidth',2); title('|X(k)|^2'); grid

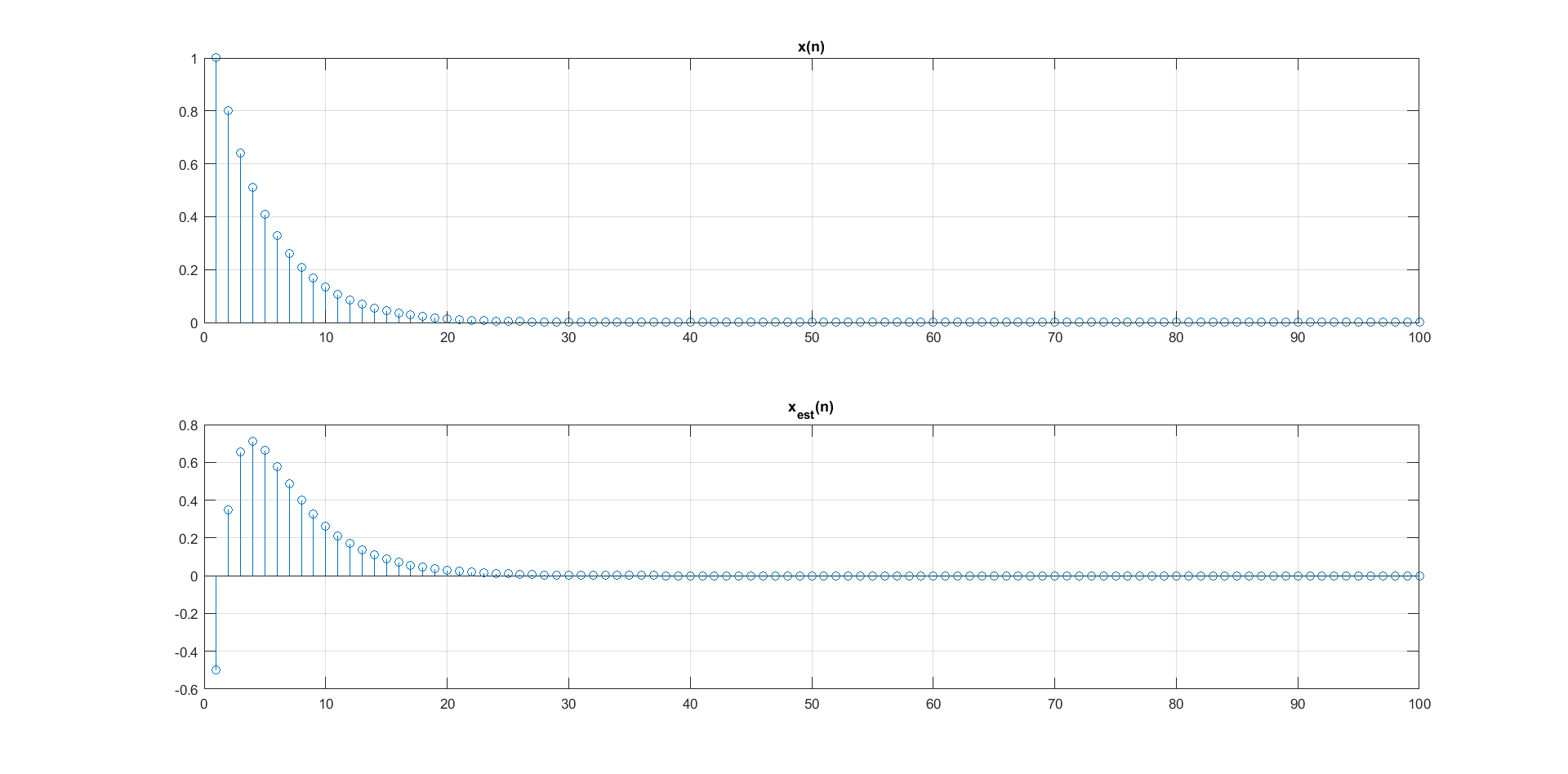
subplot(212); plot(abs(fft(x\_est)),'linewidth',2); title('|X\_{est}(k)|^2'); grid

figure(3)

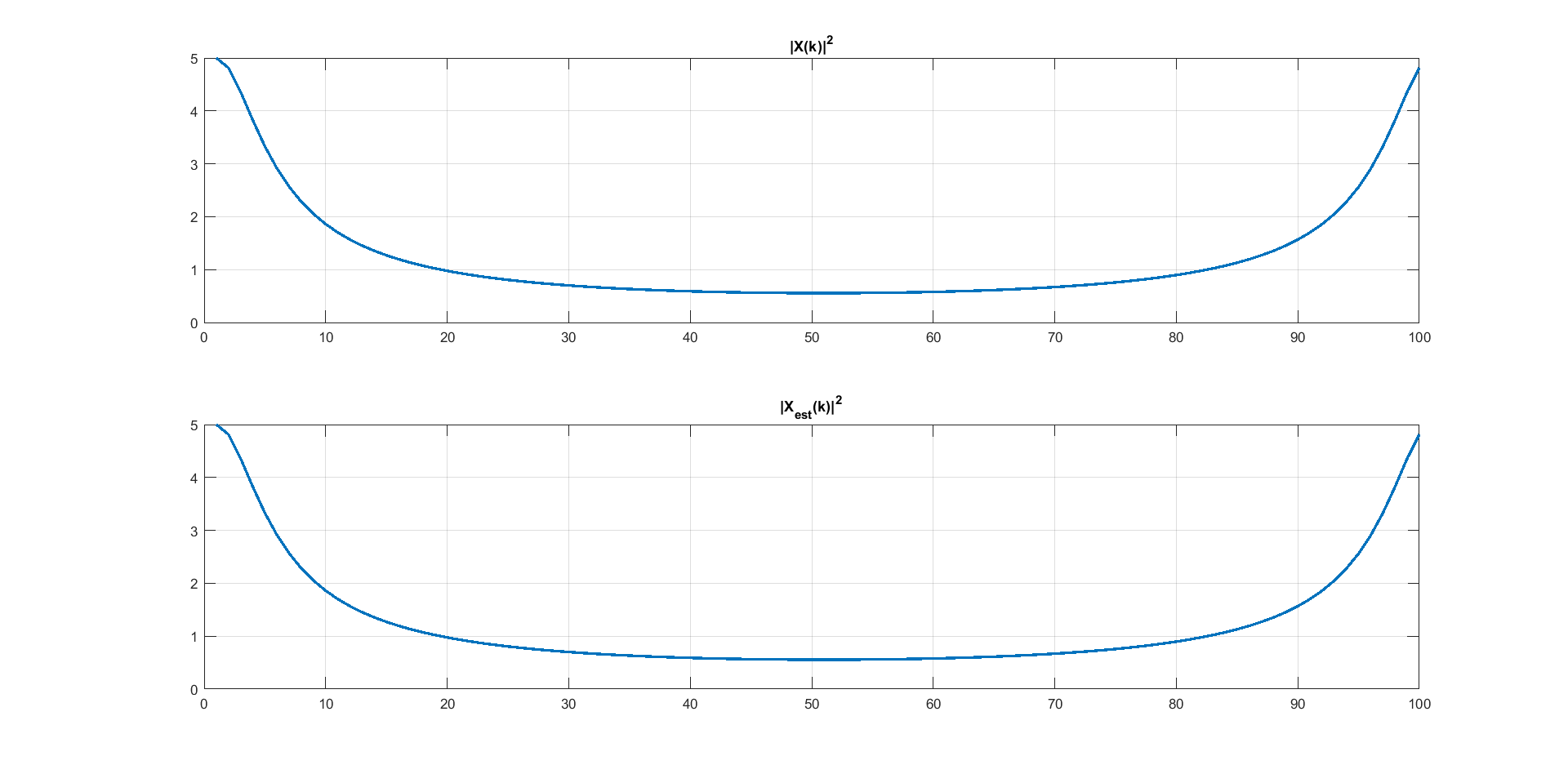
subplot(211); plot(angle(fft(x)),'linewidth',2); title('Fase'); grid

subplot(212); plot(angle(fft(x\_est)),'linewidth',2); title('Fase'); grid

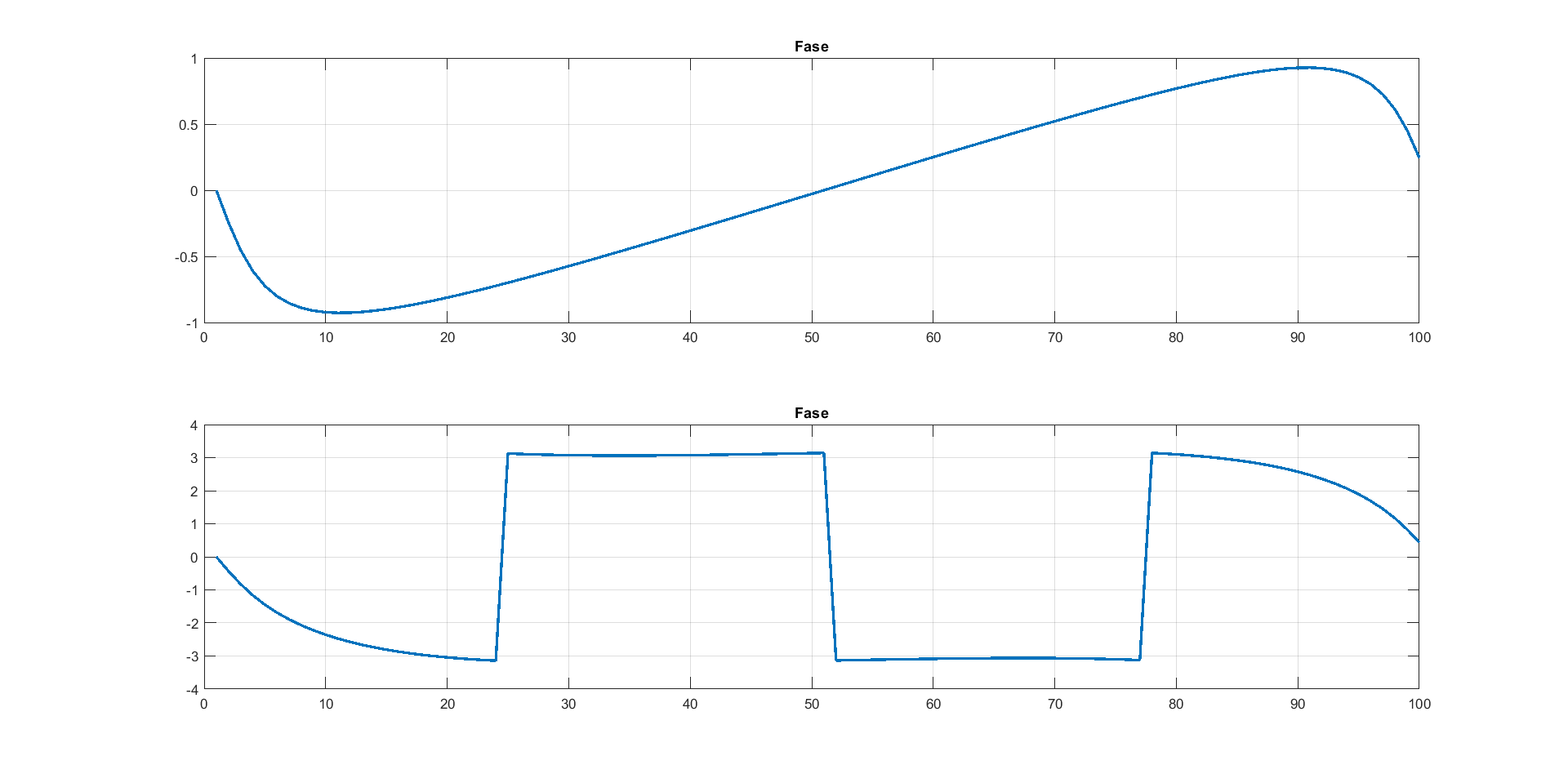


**

*Figura 11 – Gráficos do sinal original x(n) e do sinal filtrado xest(n)*

*.*

*Figura 12 – Gráficos em frequência dos módulos do sinal original x(n) e do sinal filtrado xest(n) .*

**

*Figura 13 – Gráficos em frequência das fases do sinal original x(n) e do sinal filtrado xest(n) .*

**c)**

*Quadro 6 – Script de MATLAB para o item c).*

%PTC 5005 - 2019

%Prof: Maria D. Miranda

%Aluno: Stéfano Albino Vilela Rezende (Ouvinte)

%Exercício Computacional - Filtros Compensadores

%2)c)

%Filtro de distorção

A=conv([1 -.95],[1 -.98]);

B=[1 -2];

%Filtro compensador

Ac=[-2 1];

Bc=conv([1 -.95],[1 -.98]);

x0 = audioread('Que\_maravilha\_cut.wav');

Fs=44100; % frequência de amostragem

x=x0(2000:5\*Fs,1)';

y=filter(B,A,x); %Sinal corrompido

x\_est=filter(Bc,Ac,y); %Sinal recuperado

%sound(x,Fs);

%sound(y,Fs);

sound(x\_est,Fs)

N=length(x);

figure(1)

subplot(311); plot(x(1:N),'.'); title('x(n)'); grid; xlim([0 2.185\*10^5])

subplot(312); plot(y(1:N),'.'); title('y(n)'); grid; xlim([0 2.185\*10^5])

subplot(313); plot(x\_est(1:N),'.'); title('x\_{est}(n)'); grid; xlim([0 2.185\*10^5])

figure(2)

subplot(311); plot(abs(fft(x)),'linewidth',2); title('|X(k)|^2'); grid; xlim([0 2.185\*10^5])

subplot(312); plot(abs(fft(y)),'linewidth',2); title('|Y(k)|^2'); grid; xlim([0 2.185\*10^5])

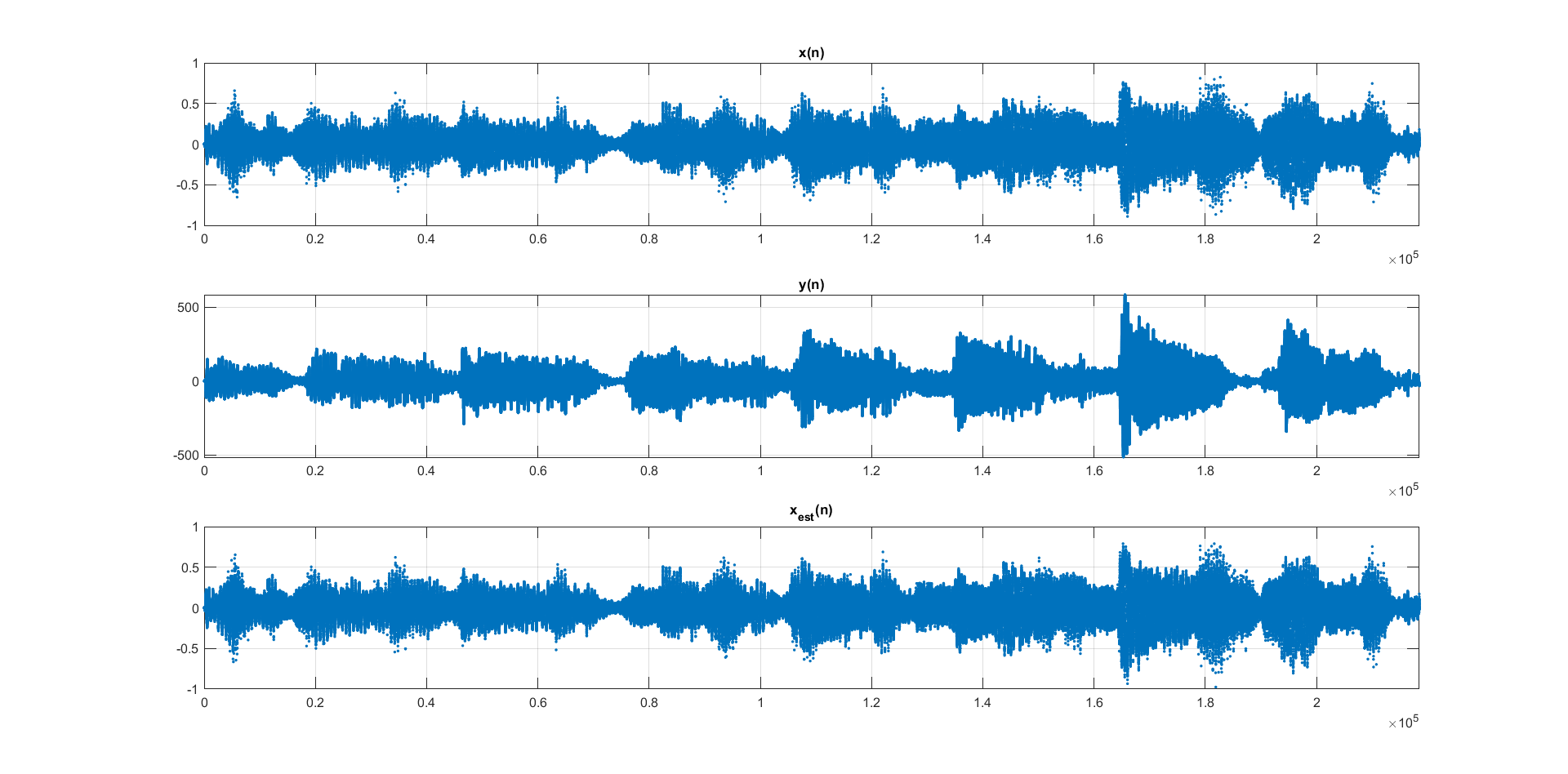
subplot(313); plot(abs(fft(x\_est)),'linewidth',2); title('|X\_{est}(k)|^2'); grid; xlim([0 2.185\*10^5])

figure (3)

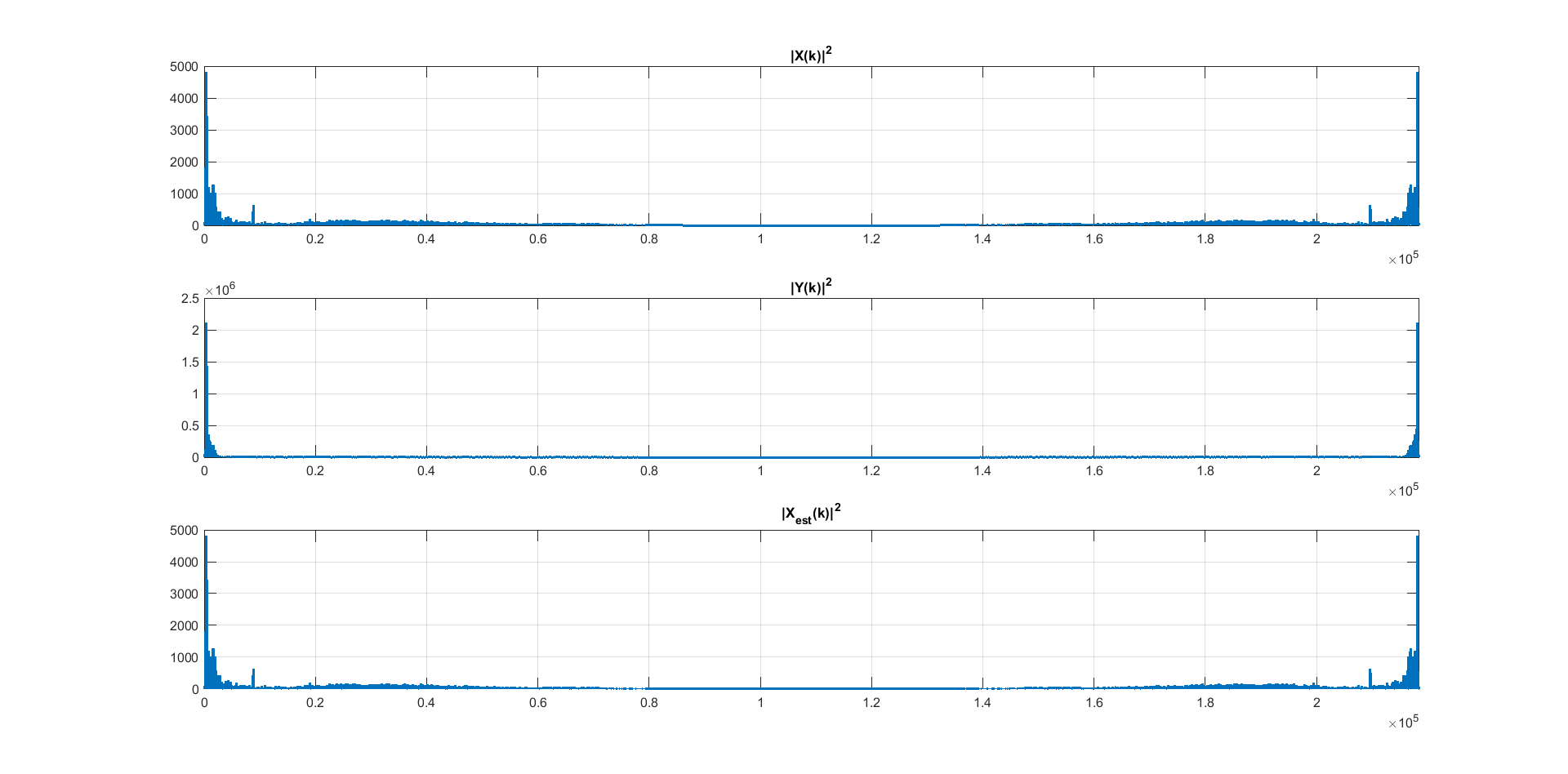
subplot(311); plot(angle(fft(x)),'linewidth',0.3); title('Fase X(k)'); grid; xlim([0 2.185\*10^5])

subplot(312); plot(angle(fft(y)),'linewidth',0.3); title('Fase Y(k)'); grid; xlim([0 2.185\*10^5])

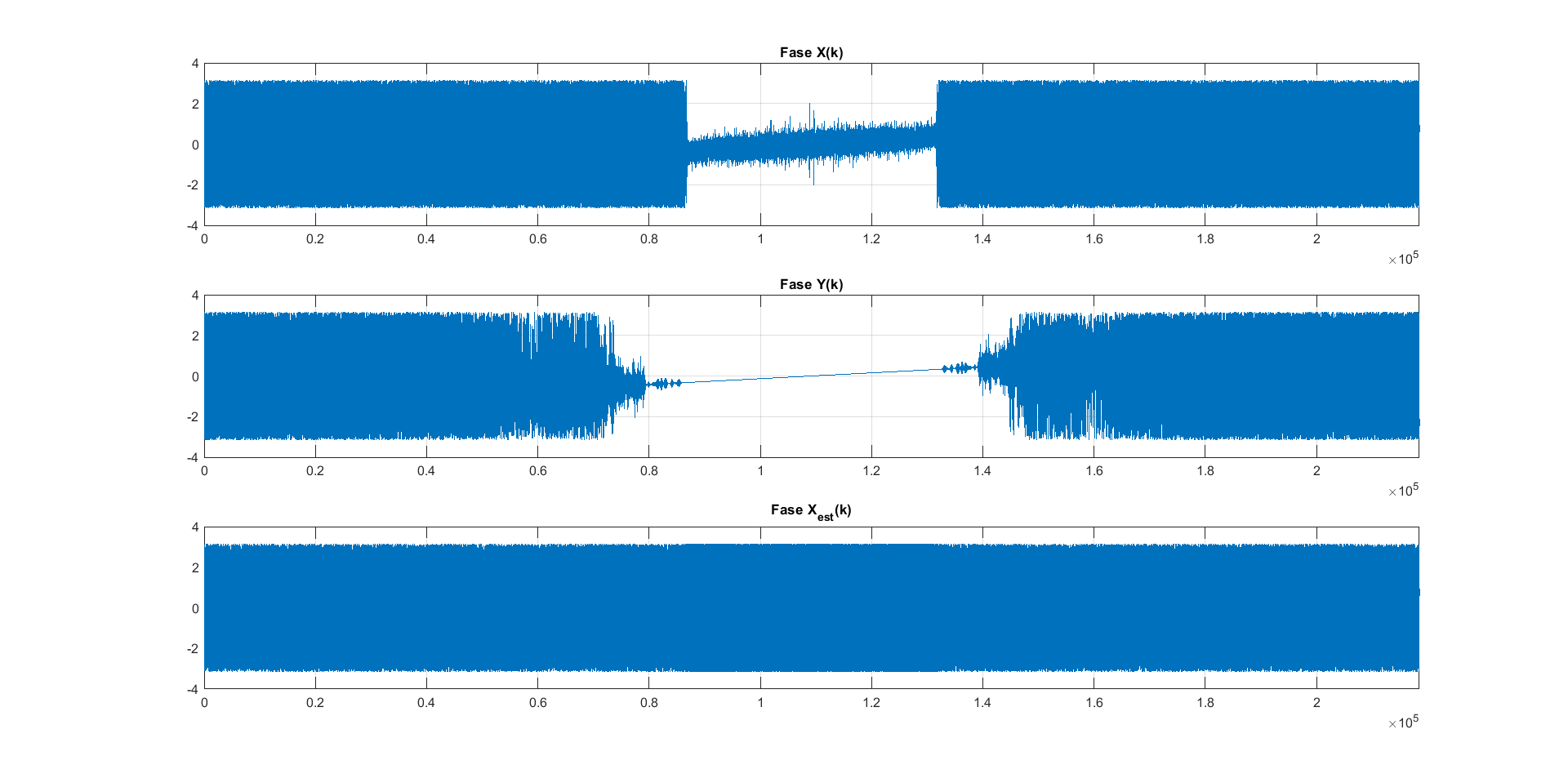
subplot(313); plot(angle(fft(x\_est)),'linewidth',0.3); title('Fase X\_{est}(k)'); grid; xlim([0 2.185\*10^5])

**

*Figura 14– Gráficos do áudio original x(n), áudio distorcido y(n) e do áudio filtrado xest(n).*

**

*Figura 15 – Gráficos em frequência dos módulos do áudio original x(n), áudio distorcido y(n) e do áudio filtrado xest(n).*

**

*Figura 16 – Gráficos em frequência das fases do áudio original x(n), áudio distorcido y(n) e do áudio filtrado xest(n).*

**d)**

*Quadro 7 – Script de MATLAB para o item d).*

%PTC 5005 - 2019

%Prof: Maria D. Miranda

%Aluno: Stéfano Albino Vilela Rezende (Ouvinte)

%Exercício Computacional - Filtros Compensadores

%2)d)

%Filtro de distorção

A=conv([1 -.95],[1 -.98]);

B=[1 -2];

%Filtro compensador

Ac=[-2 1];

Bc=conv([1 -.95],[1 -.98]);

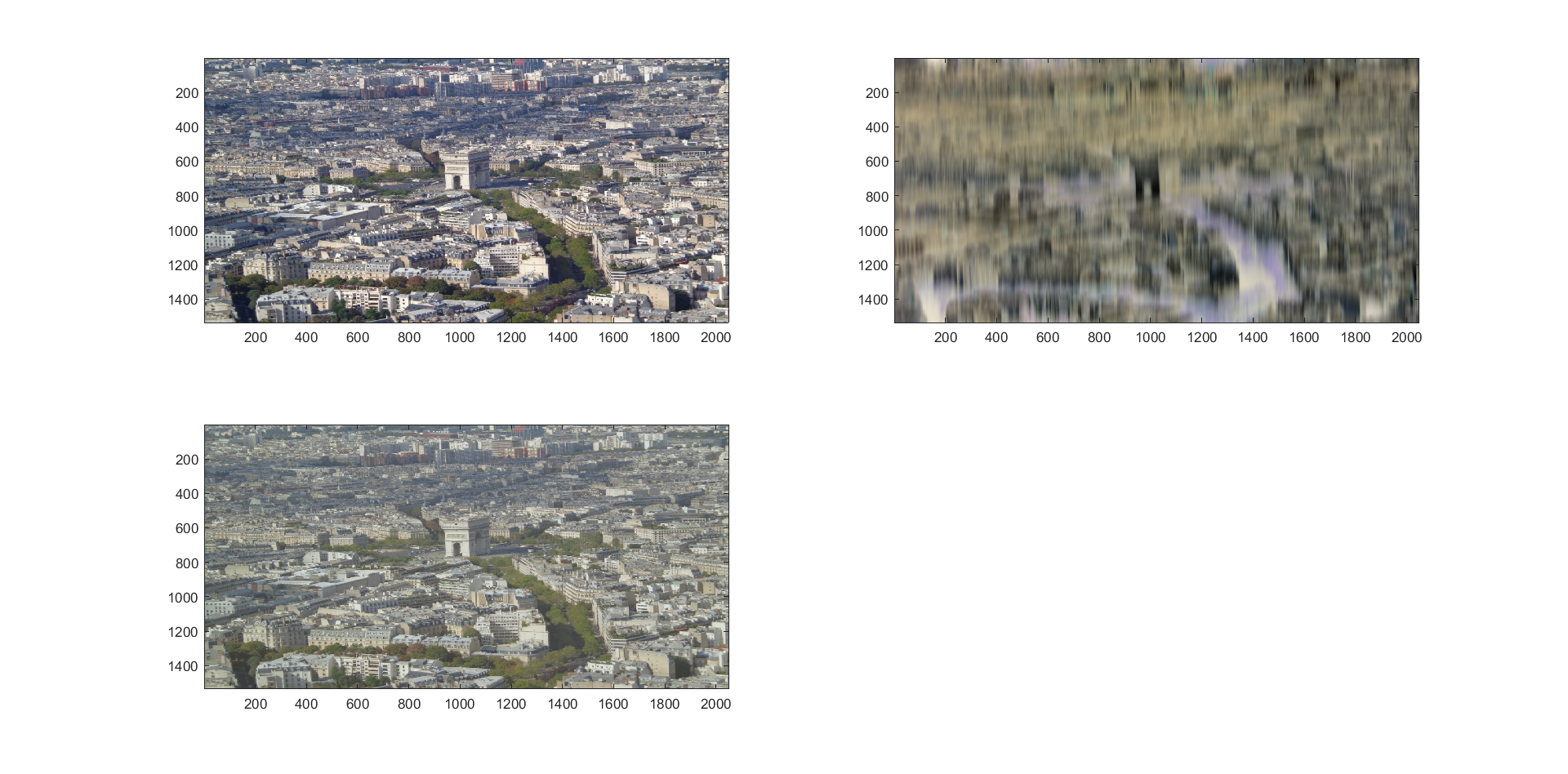
[x,y,xest]=RotImag(A,B,Ac, Bc);

figure(1)

subplot(221); image(x)

subplot(222); image(y)

subplot(223); image(xest)

**

*Figura 17 – Reprodução da imagem arco\_triunfo.jpg original, distorcida e recuperada*

Analise a imagem original, distorcida e recuperada.

**3)**

Compare e comente o resultados nos itens 1 e 2. Principalmente os efeitos de distorção de fase nos sinais de audio e imagem