PSI3431 - Processamento Estatístico de Sinais (2022)

Exercício Computacional 1 - Filtragem de um sinal ruidoso

Aluno: Ivan Luiz de Moura Matos

N°USP: 11234162

Data de entrega: 02/05/2022

In [1]:
```
using PyPlot, DSP, WAV
```

Resolução do item a)

In [2]:
```
# Sequencia de instantes em que o sinal x0 foi amostrado:
seq = 0:1/40000:2;

# Constantes utilizadas na função que define o sinal x0:
omega = 2*pi*500; tau = 0.5;
```

Obtenção do sinal discreto $x_0[n]$:

In [3]:
```
x0 = (sin.(omega*seq)).^3 .* exp.(-seq/tau);
```

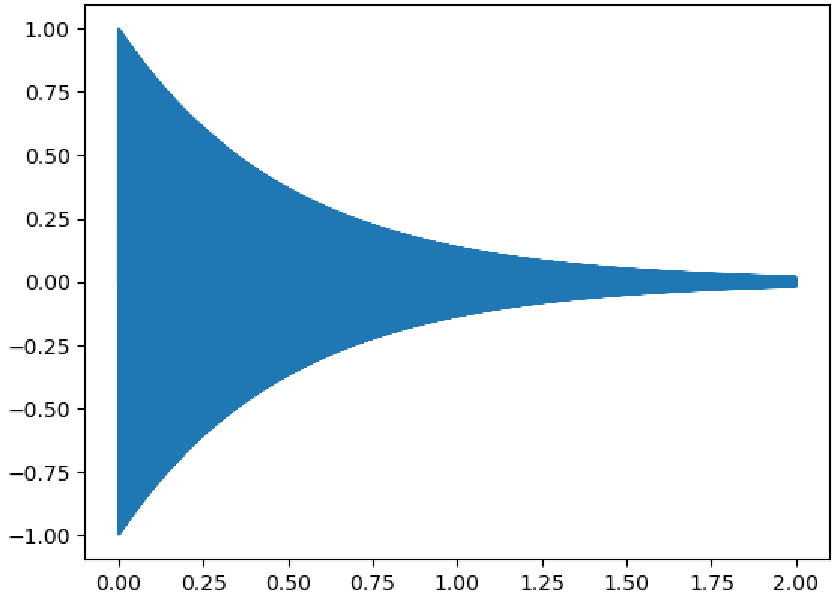
Cálculo do comprimento do sinal discreto $x_0[n]$:

In [4]:
```
print("O comprimento de x0[n] é: ", length(x0))
```

```
O comprimento de x0[n] é: 80001
```

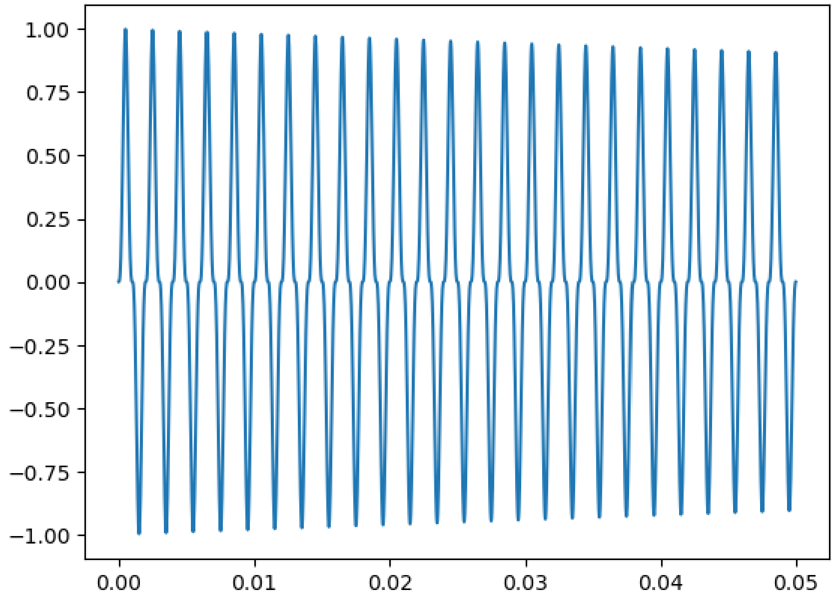
No gráfico abaixo, é plotado o sinal $x_0[n]$:

In [6]:
```
plot(seq, x0);
```



Considerando um trecho menor do sinal, para melhor visualização:

In [8]:
```
plot(seq[1:2000], x0[1:2000]);
```



Para escutar o sinal:

In [9]:
```
wavplay(x0, 40_000)
```

Resolução do item b)

Cálculo da potência P_{sinal} do sinal amostrado $x_0[n]$:

In [10]:
```
Potencia_sinal = sum(x0.^2)/length(x0)
```

Out[10]:
```
0.039048886338914370
```

Podemos representar o ruído gaussiano branco por uma sequência de valores gerados a partir de variáveis aleatórias gaussianas, com média nula e variância σ^2. Nesse caso, a potência do ruído gaussiano branco amostrado é dada por $P_{\text{ruído}} = \sigma^2$. Determinemos a variância σ^2, impondo que a relação sinal-ruído (SNR) deve ser igual a 10 dB:

$$SNR_{(dB)} = 10 \cdot \log(P_{\text{sinal}}/P_{\text{ruído}})$$

$$\Rightarrow \sigma^2 = P_{\text{ruído}} = P_{\text{sinal}}/10^{SNR_{(dB)}/10}$$

A seguir, calculamos a variância σ^2 do ruído:

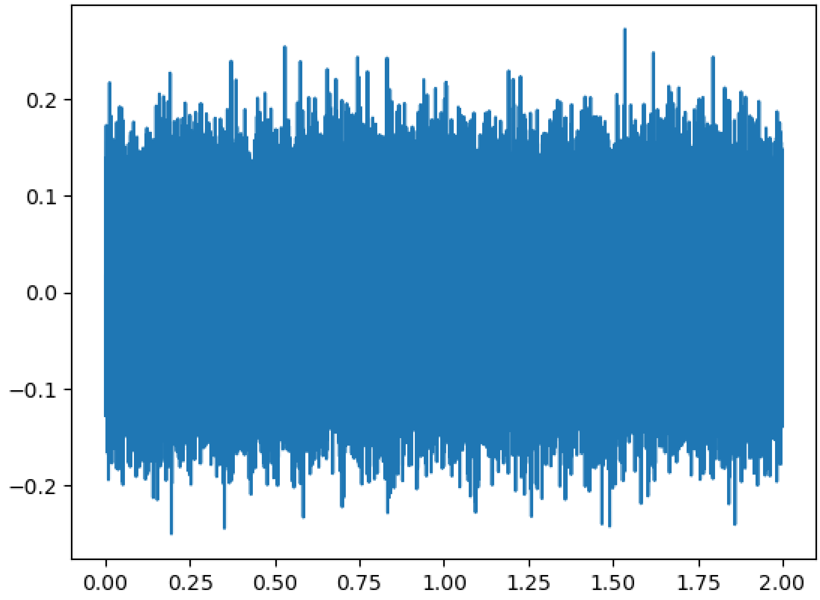
In [11]:
```
# Cálculo da variância do ruído amostrado:
var_ruido = Potencia_sinal / 10^(10/10) # Do enunciado, SNR(dB) = 10 dB
```

Out[11]:
```
0.0039048886338914368
```

Conhecida a variância σ^2 do ruído **(denotada pela variável `var_ruido`)**, podemos construir o vetor `ruido` que representará o ruído branco. Para tanto, utilizaremos a função `randn`, que permite gerar uma sequência de valores extraídos de uma distribuição normal. Tal função considera média nula e variância 1; para obter variânciar igual a σ^2, basta multiplicar por $\sqrt{\sigma^2}$.

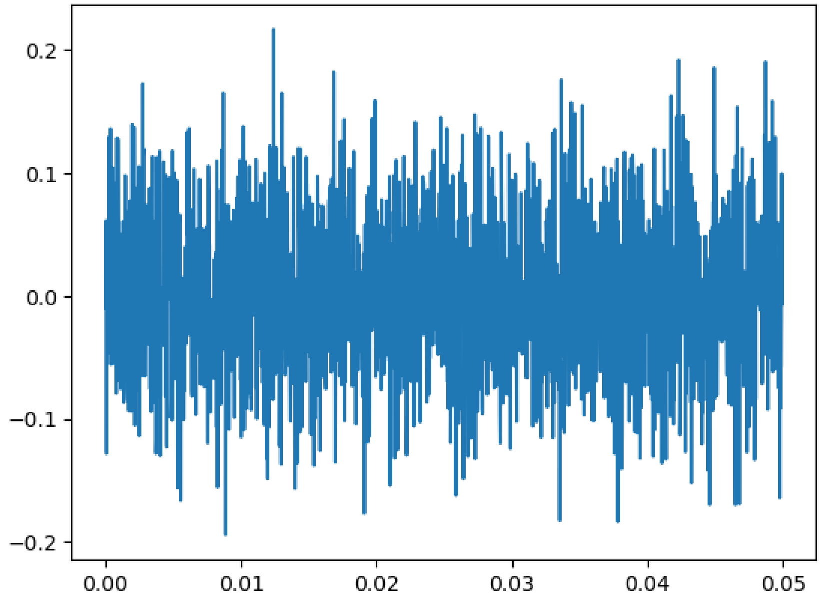
In [12]:
```
ruido = sqrt(var_ruido)*randn(length(seq));
```

In [60]:
```
plot(seq,ruido);
```



Abaixo, é plotado um trecho do gráfico do ruído.

In [13]:
```
plot(seq[1:2000], ruido[1:2000]);
```



O sinal ruidoso $x[n]$ é a soma do sinal original $x_0[n]$ com o ruído:

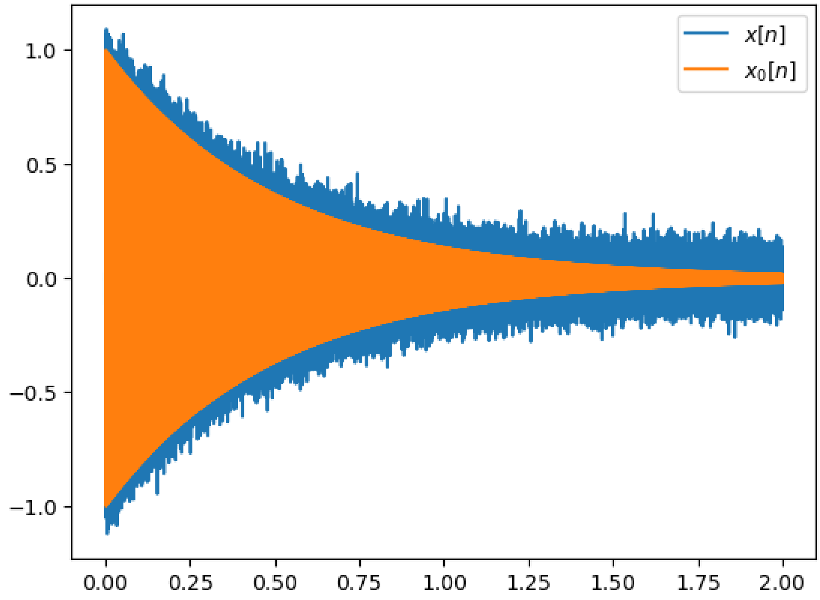
In [14]:
```
x = x0 .+ ruido;
```

Para escutar o sinal ruidoso:

In [15]:
```
wavplay(x, 40_000)
```

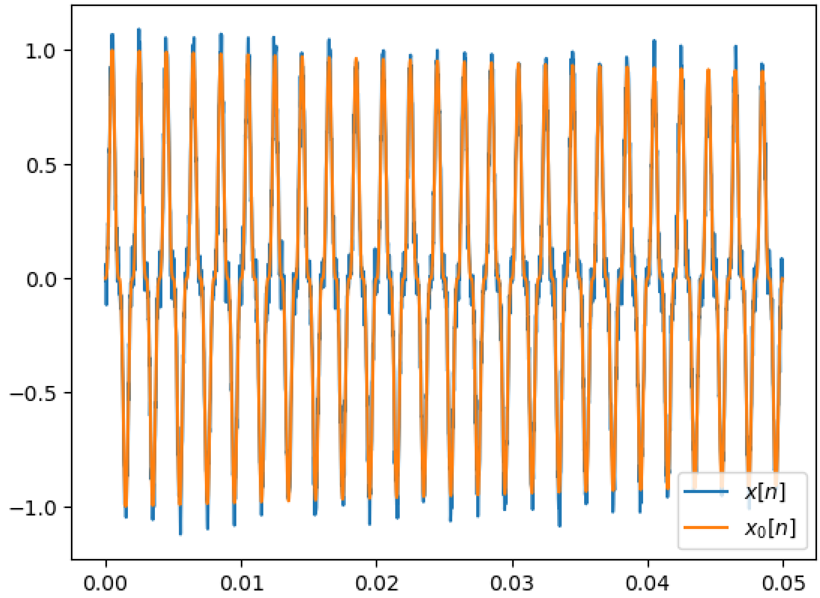
No gráfico abaixo, são plotados $x_0[n]$ e $x[n]$:

In [16]:
```
plot(seq,x, label=L"x[n]")
plot(seq,x0,label=L"x_0[n]")
legend();
```



Considerando um trecho menor do sinal, para melhor visualização:

In [17]:
```
plot(seq[1:2000],x[1:2000], label=L"x[n]");
plot(seq[1:2000],x0[1:2000],label=L"x_0[n]")
legend();
```



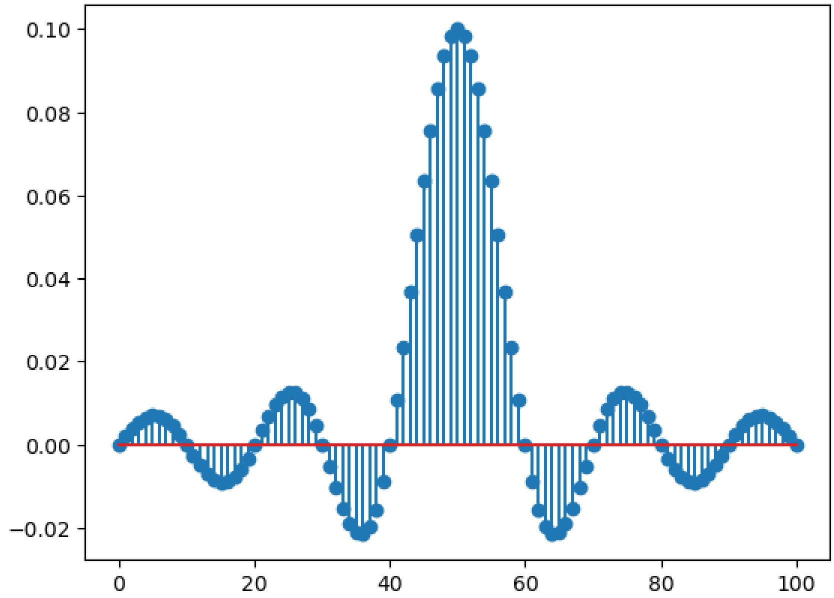
Resolução do item c)

Conforme o enunciado, vamos construir um filtro com resposta ao impulso dada por $h[n] = 0,1 \cdot \text{sinc}\,(0,1 \cdot (n-50))$, para $0 \leq n \leq 100$, e zero caso contrário.

In [18]:
```
h = 0.1 * sinc.(0.1*((0:100) .- 50));
```

Abaixo, é apresentada a **resposta ao impulso** do filtro:

In [19]:
```
stem(0:100,h);
```



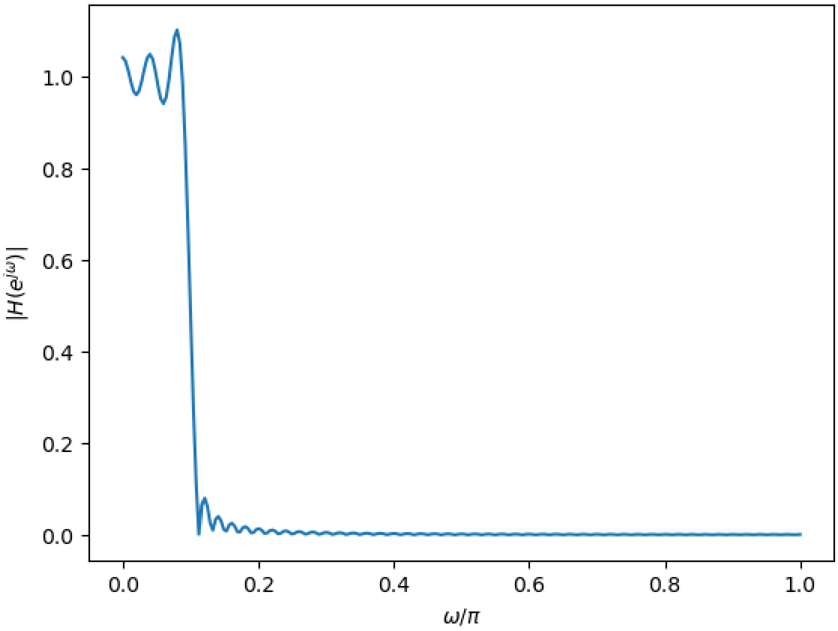
In [20]:
```
h_filtro = PolynomialRatio(h,[1.0]);

omega_filtro= range(0, stop=pi, length = 250);
H_filtro = freqz(h_filtro, omega_filtro);

#H_filtro, omega_filtro = freqresp(h_filtro);
```

Abaixo, é apresentada a **resposta em frequência** do filtro:

In [21]:
```
plot(omega_filtro/π, abs.(H_filtro));
xlabel(L"\omega/\pi");
ylabel(L"|H(e^{j\omega})|");
```



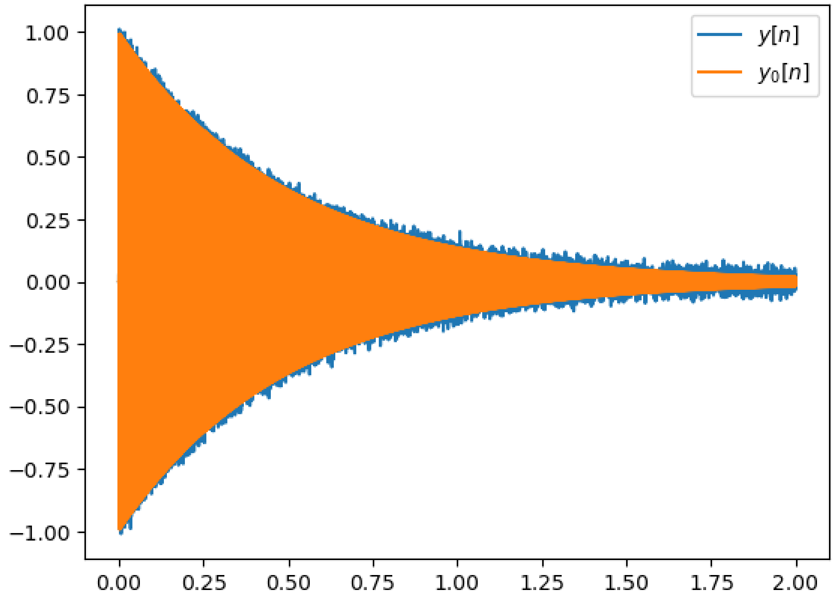
Resolução do item d)

Vamos obter as saídas $y_0[n]$ e $y[n]$ do filtro, considerando que o sinal de entrada é $x_0[n]$ e $y[n]$, respectivamente :

In [22]:
```
y0 = filt(h_filtro, x0);
y = filt(h_filtro, x);
```

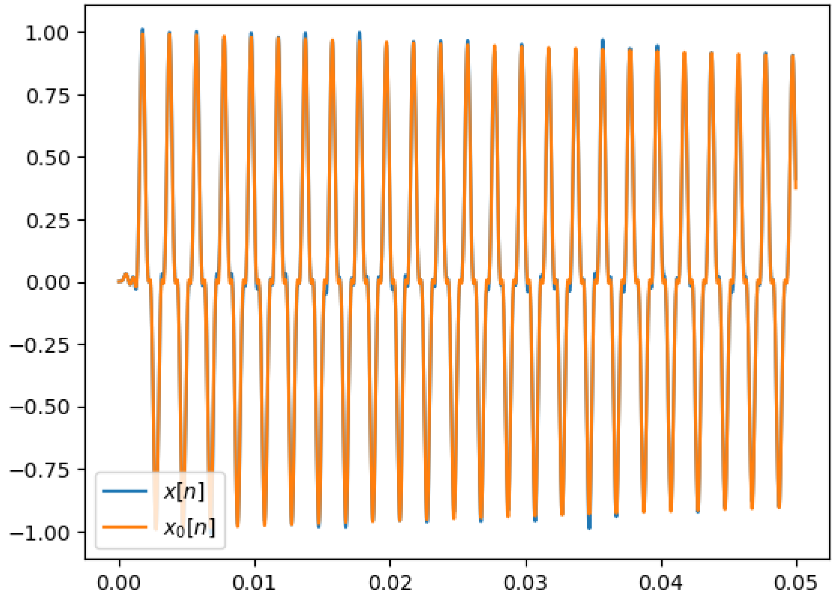
No gráfico abaixo, são plotados $y_0[n]$ e $y[n]$:

In [23]:
```
plot(seq,y, label=L"y[n]");
plot(seq,y0, label=L"y_0[n]");
legend();
```



Considerando um trecho menor do sinal, para melhor visualização:

In [24]:
```
plot(seq[1:2000],y[1:2000], label=L"x[n]");
plot(seq[1:2000],y0[1:2000],label=L"x_0[n]")
legend();
```



Para escutar os sinais $y_0[n]$ e $y[n]$:

In [25]:
```
wavplay(y0, 40_000)
```

In [26]:
```
wavplay(y, 40_000)
```

Resolução do item e)

Vamos explicar por qual motivo $x[n]$ e $y[n]$ não são estacionários em nenhum sentido:

O sinal de entrada $x_0[n]$ é determinístico, enquanto $x[n]$, que é o sinal afetado por ruído, pode ser visto como um processo estacionário $X[n]$. A média do ruído $v_x[n]$ é nula (conforme mostrado abaixo). O valor do sinal $x_0[n]$ varia conforme o instante n, de modo que o valor esperado de $X[n]$ é dado por $\mathrm{E}\{X[n]\} = x_0[n] + \mathrm{E}\{V_x[n]\} = x_0[n] + 0 = x_0[n]$, para cada n.

Como a média de $X[n]$ não é constante, resulta que esse processo **não** é estacionário no sentido amplo, ou seja: **não é estacionário em nenhum sentido**.

Por outro lado, a partir de um argumento similar (também baseado no fato que $\mathrm{E}\{V_x[n]\} = 0$), concluímos que a média do processo $Y[n]$ não é constante, e, portanto, $y[n]$ também **não** é estacionário em nenhum sentido.

Vamos mostrar que o ruído $v_x[n]$ é estacionário no sentido amplo:

O ruído $v_x[n] = x[n] - x_0[n]$ consiste na realização de um processo estocástico $V_x[n]$, onde $V_x \sim N(0;\sigma^2)$, isto é, a função distribuição de probabilidade de $V_x[n]$ é dada por:

$$f_{V_x[n]}(x;n) = \frac{1}{\sqrt{2\pi\sigma^2}} \cdot \exp(-\frac{x^2}{2\sigma^2})$$

O **valor esperado** de $V_x[n]$ é : $\mathrm{E}\{V_x[n]\} = 0 = \mathrm{constante}$, para todo instante n.

Além disso, levando em conta que $V_x[n]$ é $i.i.d.$ calculamos a sua **autocorrelação** como segue:

$r_{V_x}[n_1,n_2] = \mathrm{E}\{V_x[n_1] \cdot V_x[n_2]\} = \mathrm{E}\{V_x[n_1]\} \cdot \mathrm{E}\{V_x[n_1]\} = 0 \cdot 0 = 0$, se $n_1 \neq n_2$

e

$r_{V_x}[n_1,n_2] = \mathrm{E}\{V_x[n_1] \cdot V_x[n_1]\} = \mathrm{E}\{V_x^2[n_1]\} = \sigma^2 + \mathrm{E}^2\{V_x[n_1]\} = \sigma^2 + 0 = \sigma^2$, se $n_1 = n_2$. Podemos escrever, então, $r_{V_x}[n_1,n_2] = \sigma^2 \cdot \delta[n_1 - n_2] = \sigma^2 \cdot \delta[\tau]$, para quaisquer dois instantes n_1 e n_2. Observamos que a autocorrelação depende apenas da diferença $\tau = n_1 - n_2$ entre os instantes considerados.

Nota-se, assim, que ficam satisfeitas as duas condições que caracterizam o processo como estacionário em sentido amplo.

Agora, vamos explicar por qual motivo o ruído $v_y[n]$ é estacionário no sentido amplo (desconsiderando transitórios):

O filtro $H(z)$ implementado consiste em um **sistema linear e invariante no tempo (SLIT)**. O ruído $v_y[n]$ de saída consiste, portanto, no sinal obtido ao se aplicar o ruído $v_x[n]$ na entrada do sistema. Como sabemos, se a entrada de um SLIT é um processo estocástico no sentido amplo, então sua saída também o é, como mostrado nas páginas 59 e 60 da apostila do curso. Concluímos, assim, que o ruído $v_y[n]$ é estacionário no sentido amplo.

Resolução do item f)

Sejam $S_{V_x}(\omega)$ e $S_{V_y}(\omega)$ as densidades espectrais de potência do ruído na entrada e saída do filtro, respectivamente. Determinemos as expressões teóricas de tais funções.

Função densidade de probabilidade do ruído *na entrada* do filtro:

Temos

$$S_{V_x}(\omega) = \sum_{\ell=-\infty}^{\infty} [r_{V_x}[\ell] \cdot \exp(-j\omega\ell)]$$

A partir do exposto no item **e)**, podemos escrever: $r_{V_x}[\ell] = \sigma^2 \cdot \delta[\ell]$. Daí, segue que

$$S_{V_x}(\omega) = \sum_{\ell=-\infty}^{\infty} [(\sigma^2 \cdot \delta[\ell]) \cdot \exp(-j\omega\ell)] = \sigma^2 \cdot \delta[0] \cdot \exp(-j\omega \cdot 0) = \sigma^2 \cdot 1 \cdot \exp(0)$$

isto é:

$$S_{V_x}(\omega) = \sigma^2$$

para todo ω.

Função densidade de probabilidade do ruído *na saída* do filtro:

Como se sabe (página 62 da apostila do curso), a densidade de probabilidade $S_{V_y}(\omega)$ do ruído depois do filtro será igual ao produto entre o módulo da resposta em frequência do filtro e $S_{V_x}(\omega)$:

$$S_{V_y}(\omega) = |H(e^{j\omega})|\, S_{V_y}(\omega)$$

Como a resposta ao impulso do filtro é $h[n] = 0,1 \cdot \mathrm{sinc}\,(0,1 \cdot (n-50))$, temos

$$H(e^{j\omega}) = \frac{0,1}{0,1} \cdot \mathrm{rect}(\frac{\omega}{2\pi \cdot 0,1}) \cdot e^{-j\omega \cdot 50} = \mathrm{rect}(\frac{\omega}{0,2\pi}) \cdot e^{-j\omega \cdot 50}$$

onde $\mathrm{rect}(x)$ vale 1 para $|x| \leq 1/2$, e vale 0 para $|x| > 1/2$.

Daí,

$$|H(e^{j\omega})| = |\mathrm{rect}(\frac{\omega}{0,2\pi}) \cdot e^{-j\omega \cdot 50}| = \mathrm{rect}(\frac{\omega}{0,2\pi})$$

Logo,

$$S_{V_y}(\omega) = |H(e^{j\omega})|\, S_{V_y}(\omega) = \sigma^2 \mathrm{rect}(\frac{\omega}{0,2\pi})$$

ou, ainda,

$$S_{V_y}(\omega) = \begin{cases} \sigma^2 & \text{se} \quad |\omega| \leq 0,1\pi \\ 0 & \text{se} \quad |\omega| > 0,1\pi \end{cases}$$

Potência média do ruído *na entrada* do filtro:

A potência (média) do ruído na entrada do filtro é dada por

$$\frac{1}{2\pi}\int_{-\pi}^{\pi} S_{V_x}(\omega)\,d\omega = \frac{1}{2\pi}\int_{-\pi}^{\pi} \sigma^2\,d\omega = \frac{1}{2\pi}\cdot 2\pi\cdot\sigma^2 = \sigma^2$$

(tal resultado já havia sido utilizado no item **b)**.

Potência média do ruído *na saída* do filtro:

A potência (média) do ruído na saída do filtro é dada por

$$\frac{1}{2\pi}\int_{-\pi}^{\pi} S_{V_y}(\omega)\,d\omega = \frac{1}{2\pi}\int_{-0,1\pi}^{0,1\pi} \sigma^2\,d\omega = \frac{1}{2\pi}\cdot 0,2\pi\cdot\sigma^2 = 0,1\cdot\sigma^2$$

Nota-se, portanto, que a potência do ruído na saída do filtro é inferior à potência do ruído na entrada.

Abaixo, calcula-se a potência média do ruído na entrada do filtro:

In [27]:
```
Pot_Media_Vx_calculada = var_ruido
```

Out[27]:
```
0.0039048886338914368
```

Abaixo, calcula-se a potência média do ruído na saída do filtro:

In [28]:
```
Pot_Media_Vy_calculada = 0.1*var_ruido
```

Out[28]:
```
0.00039048886633891437
```

Resolução do item g)

(a) Vamos começar pelo primeiro procedimento proposto, considerando:

$$r_{V_x}[0] \approx \frac{1}{L}\sum_{\ell=1}^{L}(v_x^{(\ell)}[n])^2$$

e

$$r_{V_y}[0] \approx \frac{1}{L}\sum_{\ell=1}^{L}(v_y^{(\ell)}[n])^2$$

Cálculo de $r_{V_x}[0]$ com $L = 1000$ e para todos os valores de n:

In [31]:
```
L = 1000;
soma = zeros(length(seq));

for i in 1:L
    #Geracao do vetor de ruido na i-esima iteracao
    v_x = sqrt(var_ruido)*randn(length(seq));

    soma = soma .+ v_x.^2
end

Pot_Vx_estimada_A = (1/L) * soma
```

Out[31]:
```
80001-element Vector{Float64}:
 0.004014579168785768
 0.003904151681004828
 0.004427573404025718
 0.004194378516181004
 0.003796779144203173
 0.0038560223934413513
 0.004228344339059466
 0.0036892076153429306
 0.0038417401832813495
 0.004022247628705837
 0.0033441419731832086
 0.003718686696366693
 0.004052139725856822
 ⋮
 0.004042549516016283
 0.0039957355113176855
 0.004130349435189255
 0.004015495250549348
 0.004006074062236548
 0.0038101139625680703
 0.004087402461113508
 0.0037832084358373442
 0.00401048296330216
 0.003892652737393504
 0.003716315005769439
 0.003892209012267026
```

Vamos agora considerar $L = 10000$:

In [34]:
```
L = 10000;
soma = zeros(length(seq));

for i in 1:L
    #Geracao do vetor de ruido na i-esima iteracao
    v_x = sqrt(var_ruido)*randn(length(seq));

    soma = soma .+ v_x.^2
end

Pot_Vx_estimada_A = (1/L) * soma
```

Out[34]:
```
80001-element Vector{Float64}:
 0.0039266426569090525
 0.0038618955455948377
 0.0038615703866958953
 0.0038799742736986037
 0.003922663401486416
 0.00391806250776194
 0.0038571469495275405
 0.003877806296651594
 0.0039400151991511295
 0.003798603570091412
 0.0038067363174205384
 0.0038798262780388604
 0.0037666749649288324
 ⋮
 0.00391563915590679
 0.003951020055571184
 0.003954799580050118
 0.003856075351170323
 0.0038263954866887013
 0.0039662930013545815
 0.0039779902952998564
 0.0037850905912219235
 0.003926517802529616
 0.003923145468319118
 0.003966531123533809
 0.0038793533135098664
```

Vamos agora considerar $L = 100000$:

In [35]:
```
L = 100000;
soma = zeros(length(seq));

for i in 1:L
    #Geracao do vetor de ruido na i-esima iteracao
    v_x = sqrt(var_ruido)*randn(length(seq));

    soma = soma .+ v_x.^2
end

Pot_Vx_estimada_A = (1/L) * soma
```

Out[35]:
```
80001-element Vector{Float64}:
 0.003892048324572692
 0.003904290072817606
 0.0039041874965764665
 0.003914219084761254
 0.003907480414520672
 0.0038961442336698858
 0.0039054094507821654
 0.003924176561147666
 0.003919395609262783
 0.003872170038743886
 0.003909274563408068
 0.003917943426107885
 0.0039029898649873204
 ⋮
 0.0038796085091919647
 0.00387374776278717
 0.003894555076286878
 0.003883134518212224
 0.0038856784491709036
 0.003913294343759801
 0.003926478707966195
 0.0038786837292263524
 0.003938230292451524
 0.0039016848327082685
 0.003890003982454634
 0.003912750142681772
```

Nota-se que os valores dos vetores calculados acima são aproximadamente constantes, e valem cerca de 0.0039. Tal valor é compatível com o calculado teoricamente: $P_{\text{ruído, entrada}} = \sigma^2 \approx 0,0039$, conforme o item **f)**.

Nota-se uma melhora dos valores das estimativas, ao se aumentar L para 10000 e 100000. Ainda assim, tal melhora não é altamente significativa, uma vez que em todos os casos, os valores obtidos são próximos ao esperado.

Cálculo de $r_{V_y}[0]$ com $L = 1000$ e para todos os valores de n:

In [37]:
```
L = 1000;
soma = zeros(length(seq));

for i in 1:L
    #Geracao do vetor de ruido na i-esima iteracao
    v_x = sqrt(var_ruido)*randn(length(seq));
    v_y =  filt(h_filtro, v_x);

    soma = soma .+ v_y.^2
end

Pot_Vy_estimada_A = (1/L) * soma
```

Out[37]:
```
80001-element Vector{Float64}:
 0.0
 1.623183152413176e-8
 7.759969080574834e-8
 1.964070628147368e-7
 3.6425585347179814e-7
 5.542804024657695e-7
 7.408462736362166e-7
 8.785986367803201e-7
 9.593702676595434e-7
 9.849512708931517e-7
 9.87527807017577e-7
 1.0097548774888968e-6
 1.117289340960085e-6
 ⋮
 0.0003758351150988676
 0.00037882143210303683
 0.0003829455252656992
 0.00038720724303543887
 0.0003909398914158
 0.00039336335963120945
 0.00039474644705095877
 0.0003947207895138171
 0.0003934682960581493
 0.0003917182839102501
 0.0003896815436133828
 0.0003880356754495063
```

Vamos agora considerar $L = 10000$:

In [38]:
```
L = 10000;
soma = zeros(length(seq));

for i in 1:L
```

```
    #Geracao do vetor de ruido na i-esima iteracao
    v_x = sqrt(var_ruido)*randn(length(seq));
    v_y =  filt(h_filtro, v_x);

    soma = soma .+ v_y.^2
end

Pot_Vy_estimada_A = (1/L) * soma
```

Out[38]:

```
80001-element Vector{Float64}:
 0.0
 1.583581701359644e-8
 7.447832596656689e-8
 1.9052287516974362e-7
 3.574077351942994e-7
 5.476432326533744e-7
 7.24185180121955e-7
 8.559725755950742e-7
 9.346540380116729e-7
 9.639479859889405e-7
 9.74561911986277e-7
 1.0008320174974733e-6
 1.0957863754217447e-6
 ⋮
 0.00038290538265692357
 0.00038267754742428016
 0.0003833666507311688
 0.00038480482235069577
 0.0003867696555429959
 0.0003885613890775101
 0.0003898398873834539
 0.0003904343954922608
 0.00039034963415197
 0.000389612695070423
 0.00038855715792765775
 0.00038752222064660847
```

Vamos agora considerar $L = 100000$:

In [39]:

```
L = 100000;
soma = zeros(length(seq));

for i in 1:L
    #Geracao do vetor de ruido na i-esima iteracao
    v_x = sqrt(var_ruido)*randn(length(seq));
    v_y =  filt(h_filtro, v_x);

    soma = soma .+ v_y.^2
end

Pot_Vy_estimada_A = (1/L) * soma
```

Out[39]:

```
80001-element Vector{Float64}:
 0.0
 1.565565139236163e-8
 7.497605847814117e-8
 1.9184487647892676e-7
 3.614620533440105e-7
 5.575928066394993e-7
 7.433302388597617e-7
 8.848427102129436e-7
 9.64045226718614e-7
 9.874456535747567e-7
 9.878937545908594e-7
 1.0135582278191966e-6
 1.1072778815948241e-6
 ⋮
 0.00038596307123704417
 0.00038586449043436714
 0.0003857363324602098
 0.0003856507260606789
 0.000385576983684432
 0.0003854030460115337
 0.0003851138992213447
 0.00038473237179444884
 0.0003842624962016103
 0.0003837150623701405
 0.0003831640235065704
 0.000382623186946296
```

Nota-se que, após o término dos transitórios, os valores do vetor calculado acima tornam-se aproximadamente constantes (e valem cerca de 0.00039, que é compatível ao valor previsto: $P_{\text{ruído, saída}} = 0,1 \cdot \sigma^2 \approx 0,00039$, conforme o item **f)**.

Não se nota uma melhora altamente significativa das estimativas, ao se aumentar L para 10000 e 100000. Em todos os casos, os valores obtidos são próximos ao esperado.

(b) Vamos agora adotar o segundo procedimento proposto. Usando o fato de os ruídos serem ergódicos, vamos usar as expressões

$$r_{V_x}[0] \approx \frac{1}{N}\sum_{n=0}^{N-1}(v_x^{(\ell)}[n])^2$$

e

$$r_{V_y}[0] \approx \frac{1}{N}\sum_{n=0}^{N-1}(v_y^{(\ell)}[n])^2$$

Cálculo de $r_{V_x}[0]$ com $N = \text{comprimento do sinal}$:

In [43]:
```
# Constrói-se uma realização do ruído na entrada:
v_x = sqrt(var_ruido)*randn(length(seq));

N = length(v_x);

# Em seguida, estima-se sua potência média pela fórmula indicada acima:
Pot_Vx_estimada_B = (1/N) * sum(v_x.^2)
```

Out[43]:
```
0.00390264124551737
```

Vamos agora considerar $N = 10000$:

In [45]:
```
N = 10000;
v_x = sqrt(var_ruido)*randn(N);
Pot_Vx_estimada_B = (1/N) * sum(v_x.^2)
```

Out[45]:
```
0.0039831690298021945
```

Vamos agora considerar $N = 100000$:

In [46]:
```
N = 100000;
v_x = sqrt(var_ruido)*randn(N);
Pot_Vx_estimada_B = (1/N) * sum(v_x.^2)
```

Out[46]:
```
0.003912593002332618
```

Nota-se que o valor "medido" da potência do ruído na entrada é compatível ao valor teórico: $P_{\text{ruído, entrada}} = \sigma^2 \approx 0,0039$. Não se nota uma melhora altamente significativa das estimativas, ao se aumentar N para 10000 e 100000. Em todos os casos, os valores obtidos são próximos ao esperado.

Cálculo de $r_{V_y}[0]$ com $N = \text{comprimento do sinal}$:

In [56]:
```
# Constrói-se uma realização do ruído na saída:
v_y = filt(h_filtro, v_x);

# Em seguida, estima-se sua potência média pela fórmula indicada acima:
Pot_Vy_estimada_B = (1/N) * sum(v_y.^2)
```

Out[56]:
```
0.0003862275248672492
```

Vamos agora considerar $N = 10000$:

In [57]:
```
N = 10000;
v_x = sqrt(var_ruido)*randn(N);
v_y = filt(h_filtro, v_x);

Pot_Vy_estimada_B = (1/N) * sum(v_y.^2)
```

Out[57]:
```
0.0003882977252964052
```

Vamos agora considerar $N = 100000$:

In [58]:
```
N = 100000;
v_x = sqrt(var_ruido)*randn(N);
v_y = filt(h_filtro, v_x);

Pot_Vy_estimada_B = (1/N) * sum(v_y.^2)
```

Out[58]:
```
0.0003872999261858583
```

Nota-se que o valor "medido" da potência do ruído na saídia é compatível ao valor teórico: $P_{\text{ruído, saída}} = 0,1 \cdot \sigma^2 \approx 0,00039$. Não se nota uma melhora altamente significativa das estimativas, ao se aumentar N para 10000 e 100000. Em todos os casos, os valores obtidos são próximos ao esperado.