

Procesamiento de señales, fundamentos

Maestría en sistemas embebidos

Universidad de Buenos Aires

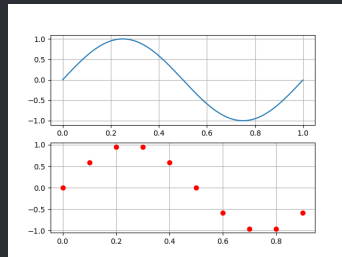
MSE 5Co2020

Clase 1 - Introducción

Ing. Pablo Slavkin

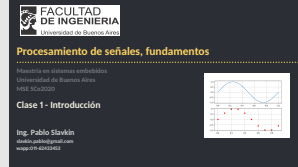
slavkin.pablo@gmail.com

wapp:011-62433453



2020-05-29

Procesamiento de señales, fundamentos



**FACULTAD
DE INGENIERIA**
Universidad de Buenos Aires

Procesamiento de señales, fundamentos

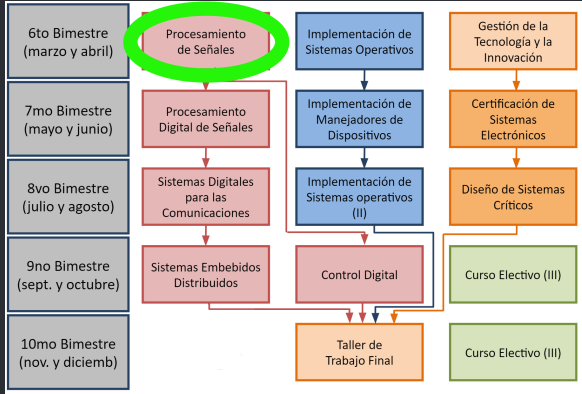
Maestría en sistemas embebidos
Universidad de Buenos Aires
MSE 5Co2020

Clase 1 - Introducción

Ing. Pablo Slavkin
slavkin.pablo@gmail.com
wapp:011-62433453

Plan de vuelo

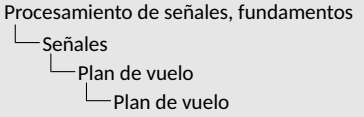
Ud. Está aquí



Colaboradores

- Gonzalo Lavigna
<gonzalolavigna@gmail.com>
- Guillermo Guichal
<guillermo.guichal@gmail.com>
- Federico Giordano Zacchigna
<federico.zacchigna@gmail.com>

2020-05-29



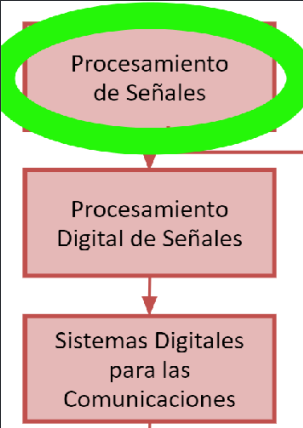
Plan de vuelo
Ud. Está aquí

Colaboradores

- Gonzalo Lavigna
<gonzalolavigna@gmail.com>
- Guillermo Guichal
<guillermo.guichal@gmail.com>
- Federico Giordano Zacchigna
<federico.zacchigna@gmail.com>

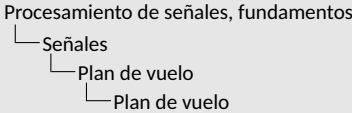
Plan de vuelo

Ud. Esta aquí



- 1.
 - Python, numpy
 - CIAA
 - Sampleo
 - Fourier, DFT
- 2.
 - Python, numpy
 - CIAA
 - VHDL, FPGA
 - Filtrado y ventaneo
- 3.
 - Python
 - VHDL, FPGA
 - Comunicaciones
 - Implementación
 - Hi-Speed

2020-05-29



Plan de vuelo

Ud. Esta aquí

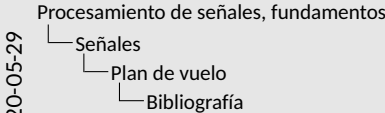
```
graph TD; A([Procesamiento de Señales]) --> B[Procesamiento Digital de Señales]; B --> C[Sistemas Digitales para las Comunicaciones];
```

- 1.
 - Python, numpy
 - CIAA
 - Sampleo
 - Fourier, DFT
- 2.
 - Python, numpy
 - CIAA
 - VHDL, FPGA
 - Filtrado y ventaneo
- 3.
 - Python
 - VHDL, FPGA
 - Comunicaciones
 - Implementación
 - Hi-Speed

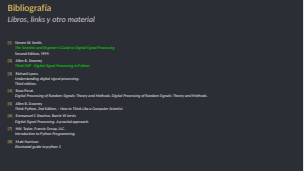
Bibliografía

Libros, links y otro material

- [1] Steven W. Smith.
The Scientist and Engineer's Guide to Digital Signal Processing
Second Edition, 1999.
- [2] Allen B. Downey
Think DSP - Digital Signal Processing in Python
- [3] Richard Lyons.
Understanding digital signal processing.
Third edition.
- [4] Boaz Porat.
Digital Processing of Random Signals: Theory and Methods. Digital Processing of Random Signals: Theory and Methods.
- [5] Allen B. Downey
Think Python, 2nd Edition, - How to Think Like a Computer Scientist
- [6] Emmanuel C Ifeachor, Barrie W Jervis
Digital Signal Processing. A practical approach.
- [7] NW. Taylor, Francis Group, LLC.
Introduction to Python Programming.
- [8] Matt Harrison
Illustrated guide to python 3



2020-05-29



Enuestas

Encuesta anónima clase a clase

Propiciamos este espacio para compartir sus sugerencias, criticas constructivas, oportunidades de mejora y cualquier tipo de comentario relacionado a la clase.

Encuesta anónima



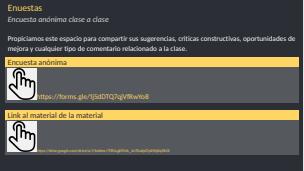
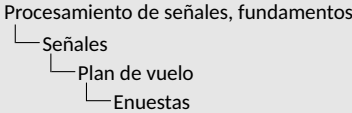
<https://forms.gle/1j5dDTQ7qjVfRwYo8>

Link al material de la material



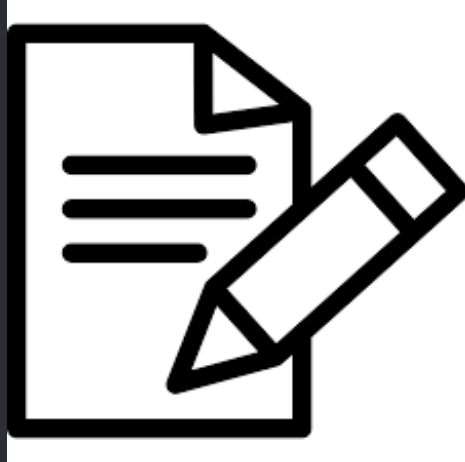
https://drive.google.com/drive/u/1/folders/1TIR2cgDPchL_4v7DxdpS7pZHtjKq38CK

2020-05-29



Metodo de evaluacion

- 3 pts - Examen
- 3 pts - TP Python
- 4 pts - Proyecto final



2020-05-29

Procesamiento de señales, fundamentos

- Señales
 - Evaluacion
 - Metodo de evaluacion

Metodo de evaluacion

- 3 pts - Examen
- 3 pts - TP Python
- 4 pts - Proyecto final



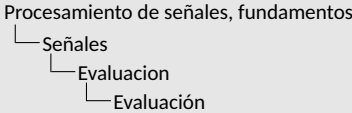
Evaluación

Proyecto final



- Deberá incluir algún tipo de procesamiento Ejemplos:
en hardware. ej. DFT, FIR, IIF, etc.
- Puede utilizar el ADC para samplear, DAC para reconstruir y/o canales de comunicación para adquirir datos previamente digitalizados
- Presentación de 10 minutos.
- Deberá funcionar!
- Filtrado y/o procesamiento de audio, señales biomédicas, etc.
- Técnicas de compresión en dominio de la frecuencia
- Aplicaciones con acelerómetro, magnetómetro, T+H

2020-05-29



Evaluación

Proyecto final

- Deberá incluir algún tipo de procesamiento Ejemplos:
en hardware. ej. DFT, FIR, IIF, etc.
- Puede utilizar el ADC para samplear, DAC para reconstruir y/o canales de comunicación para adquirir datos previamente digitalizados
- Presentación de 10 minutos.
- Deberá funcionar!
- Filtrado y/o procesamiento de audio, señales biomédicas, etc.
- Técnicas de compresión en dominio de la frecuencia
- Aplicaciones con acelerómetro, magnetómetro, T+H

porque digital?

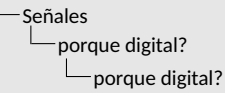
digital vs analógico

- digital
 - Reproducibilidad
 - Tolerancia de componentes
 - Partidas todas iguales
 - Componentes no envejecen
 - Fácil de actualizar
 - Soluciones de un solo chip
- analógico
 - Gran rango dinamico de entrada
 - Alto ancho de banda
 - Alta potencia
 - Baja latencia



2020-05-29

Procesamiento de señales, fundamentos



porque digital? digital vs analógico

- digital
 - Reproducibilidad
 - Tolerancia de componentes
 - Partidas todas iguales
 - Componentes no envejecen
 - Fácil de actualizar
 - Soluciones de un solo chip
- analógico
 - Gran rango dinámico de entrada
 - Alto ancho de banda
 - Alta potencia
 - Baja latencia



Señales y sistemas

Que son?

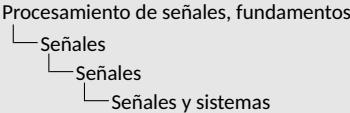
Señal

Una señal, en función de una o más variables, puede definirse como un cambio observable en una entidad cuantificable

Sistema

Un sistema es cualquier conjunto físico de componentes que actúan en una señal, tomando una o más señales de entrada, y produciendo una o más señales de salida.

2020-05-29



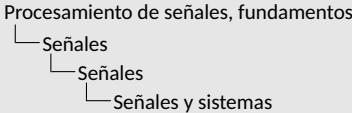
Señales y sistemas
Que son?
Señal
Una señal, en función de una o más variables, puede definirse como un cambio observable en una entidad cuantificable
Sistema
Un sistema es cualquier conjunto físico de componentes que actúan en una señal, tomando una o más señales de entrada, y produciendo una o más señales de salida.

Señales y sistemas

Tipos de señales

- De tiempo continuo
- Pares
- Periódicas
- De energía
- Reales
- De tiempo discreto
- Impares
- Aperiódicas
- De potencia
- Imaginarias

2020-05-29



Señales y sistemas

Tipos de señales

- De tiempo continuo
 - Pares
 - Periódicas
 - De energía
 - Reales
- De tiempo discreto
 - Impares
 - Aperiódicas
 - De potencia
 - Imaginarias

Señales y sistemas

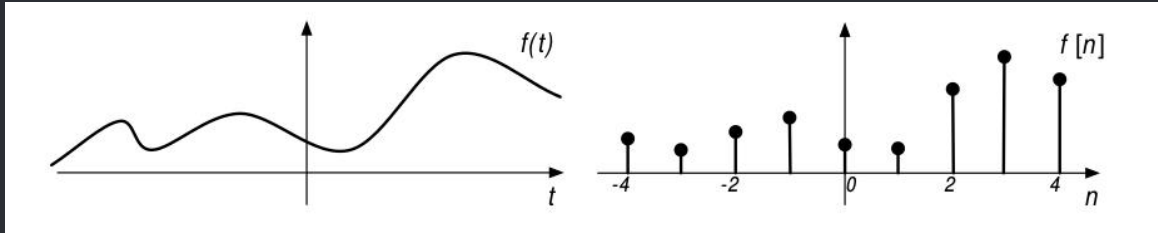
Tipos de señales

- De tiempo continuo

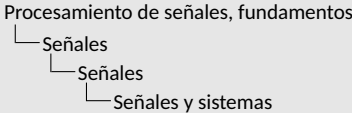
Tiene valores para todos los puntos en el tiempo en algún intervalo (posiblemente infinito)

- De tiempo discreto

Tiene valores solo para puntos discretos en el tiempo



2020-05-29



Señales y sistemas

Tipos de señales

- De tiempo continuo
Tiene valores para todos los puntos en el tiempo en algún intervalo (posiblemente infinito)
- De tiempo discreto
Tiene valores solo para puntos discretos en el tiempo

Generación de señales en Python

Continuo? vs discreto

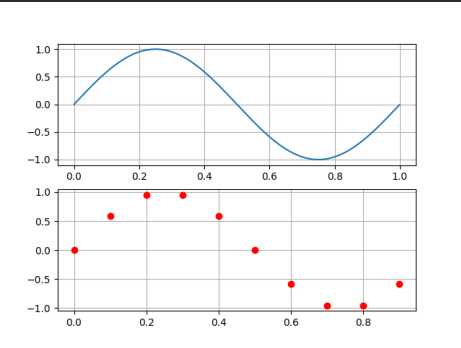
```
import matplotlib.pyplot as plt
import numpy as np

fig = plt.figure(1)
Nc=1000
tc = np.linspace(0, 1, Nc)
ax1 = fig.add_subplot(2,1,1)
ax1.plot(tc, np.sin(2*np.pi*tc),"b-")
ax1.grid(True)

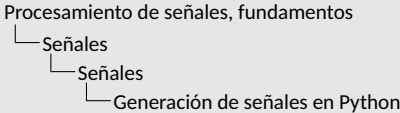
Nd=10
td = np.linspace(0, 1, Nd)
ax2 = fig.add_subplot(2,1,2)
ax2.plot(td, np.sin(2*np.pi*td),"ro")
ax2.grid(True)

plt.show()
```

Podrían pensarse como muestras de una señal de tiempo continuo $x[n] = x(nT)$ donde n es un número entero y T es el período de muestreo.



2020-05-29



Generación de señales en Python

Continuo? vs discreto

```
import matplotlib.pyplot as plt
import numpy as np

fig = plt.figure(1)
Nc=1000
tc = np.linspace(0, 1, Nc)
ax1 = fig.add_subplot(2,1,1)
ax1.plot(tc, np.sin(2*np.pi*tc),"b-")
ax1.grid(True)

Nd=10
td = np.linspace(0, 1, Nd)
ax2 = fig.add_subplot(2,1,2)
ax2.plot(td, np.sin(2*np.pi*td),"ro")
ax2.grid(True)

plt.show()
```

Podrían pensarse como muestras de una señal de tiempo continuo $x[n] = x(nT)$ donde n es un número entero y T es el período de muestreo.

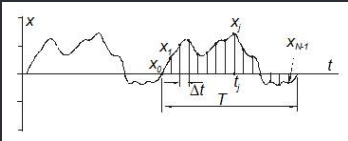
Señales periódicas

Continua periódica

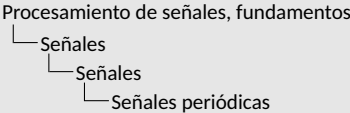
si existe un $T_0 > 0$, tal que $x(t + T_0) = x(t)$, para todo t
 T_0 es el período de $x(t)$ medido en tiempo, y $f_0 = 1/T_0$ es la frecuencia fundamental de $x(t)$

Discreta periódica

si existe un entero $N_0 > 0$ tal que $x[n + N_0] = x[n]$ para todo n
 N_0 es el período fundamental de $x[n]$ medido en espacio entre muestras y $F_0 = \Delta t/N_0$ es la frecuencia fundamental de $x[n]$



2020-05-29



Señales periódicas

Continua periódica

si existe un $T_0 > 0$, tal que $x(t + T_0) = x(t)$, para todo t
 T_0 es el período de $x(t)$ medido en tiempo, y $f_0 = 1/T_0$ es la frecuencia fundamental de $x(t)$

Discreta periódica

si existe un entero $N_0 > 0$ tal que $x[n + N_0] = x[n]$ para todo n
 N_0 es el período fundamental de $x[n]$ medido en espacio entre muestras y $F_0 = \Delta t/N_0$ es la frecuencia fundamental de $x[n]$

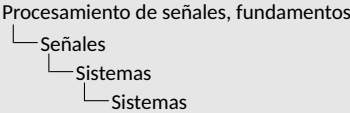
Sistema

Un sistema es cualquier conjunto físico de componentes que actúan en una señal, tomando una o más señales de entrada, y produciendo una o más señales de salida.

En ingeniería, a menudo la entrada y la salida son señales eléctricas.



2020-05-29

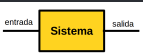


Sistemas

Sistema

Un sistema es cualquier conjunto físico de componentes que actúan en una señal, tomando una o más señales de entrada, y produciendo una o más señales de salida.

En ingeniería, a menudo la entrada y la salida son señales eléctricas.

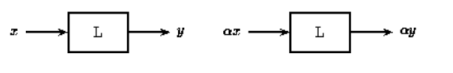


Sistemas

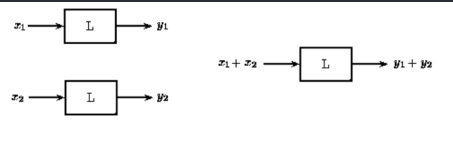
Linealidad

Un sistema es lineal cuando su salida depende linealmente de la entrada. Satisface el principio de superposición.

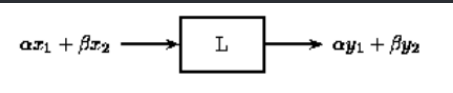
escalado



adición

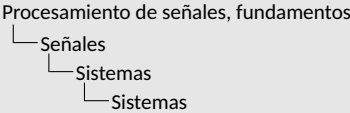


superposición



$y(t) = e^{x(t)}$
 $y(t) = \frac{1}{2}x(t)$

2020-05-29



Sistemas

Linealidad

Un sistema es lineal cuando su salida depende linealmente de la entrada. Satisface el principio de superposición.

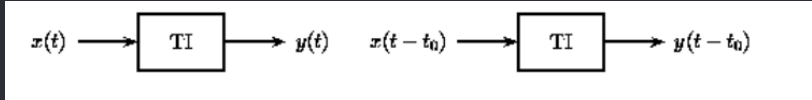
escalado		
adición		$y(t) = e^{x(t)}$ $y(t) = \frac{1}{2}x(t)$
superposición		

Sistemas

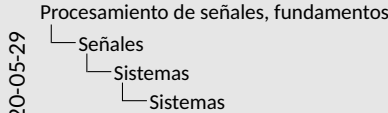
Invariantes en el tiempo

Invariantes en el tiempo

Un sistema es invariante en el tiempo cuando la salida para una determinada entrada es la misma sin importar el tiempo en el cual se aplica la entrada



$$y(t) = x(t) * \cos(t)$$
$$y(t) = \cos(x(t))$$



Sistemas

Invariantes en el tiempo

Invariantes en el tiempo

Un sistema es invariante en el tiempo cuando la salida para una determinada entrada es la misma sin importar el tiempo en el cual se aplica la entrada

$x(t) \rightarrow \boxed{\text{TI}} \rightarrow y(t)$ $x(t - t_0) \rightarrow \boxed{\text{TI}} \rightarrow y(t - t_0)$

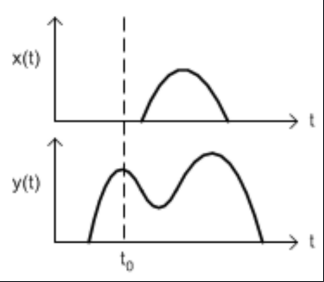
$y(t) = x(t) * \cos(t)$
 $y(t) = \cos(x(t))$

Sistemas

Causalidad

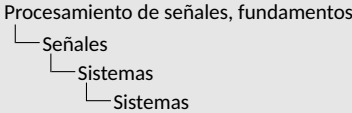
Sistema causal

Un sistema es causal cuando la salida depende solo de los valores presentes y pasados de la entrada



$$y(t) = x(t + 1)$$
$$y(t) = x(t - 2)$$

2020-05-29



Sistemas

Causalidad

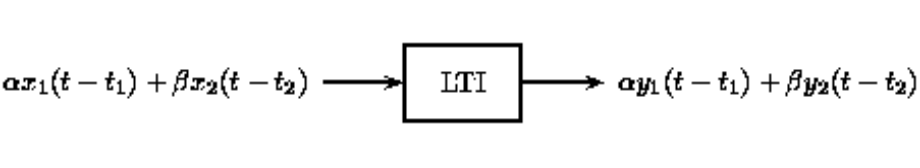
Sistema causal

Un sistema es causal cuando la salida depende solo de los valores presentes y pasados de la entrada.

$y(t) = x(t + 1)$
 $y(t) = x(t - 2)$

Lineales invariantes en el tiempo

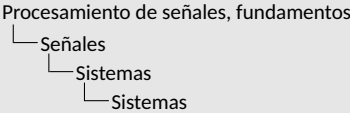
Un sistema es LTI cuando satisface las 2 condiciones anteriores, de linealidad y de invariancia en el tiempo.



*** LTI ***

En este curso, solo estudiaremos sistemas lineales invariantes en el tiempo.

2020-05-29



Sistemas

Lineales invariantes en el tiempo

Un sistema es LTI cuando satisface las 2 condiciones anteriores, de linealidad y de invariancia en el tiempo.

*** LTI ***

En este curso, solo estudiaremos sistemas lineales invariantes en el tiempo.

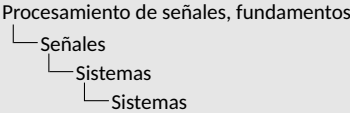
Fidelidad senoidal

En todo sistema LTI para una entrada senoidal la salida es **siempre** senoidal.

Linealidad estática

En todo sistema LTI para una entrada constante (DC) la salida es **siempre** la entrada multiplicada por una constante.

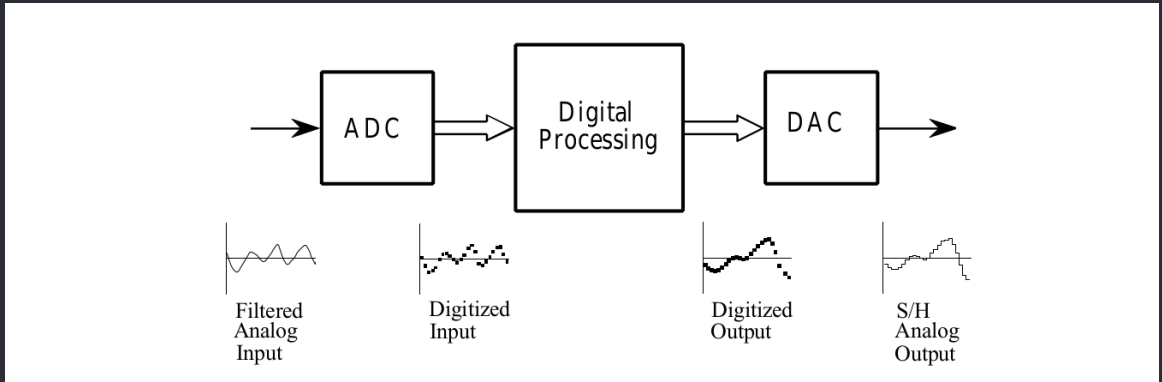
2020-05-29



Sistemas
Fidelidad senoidal
En todo sistema LTI para una entrada senoidal la salida es siempre senoidal.
Linealidad estática
En todo sistema LTI para una entrada constante (DC) la salida es siempre la entrada multiplicada por una constante.

ADC

Bloque incompleto de procesamiento



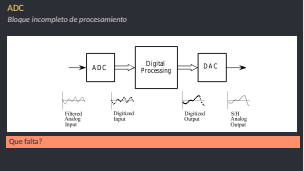
Que falta?

2020-05-29

Procesamiento de señales, fundamentos

└ ADC

└ ADC

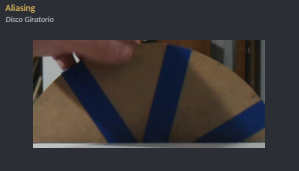
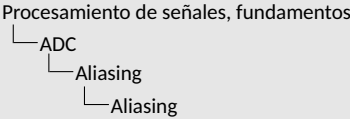


Aliasing

Disco Giratorio



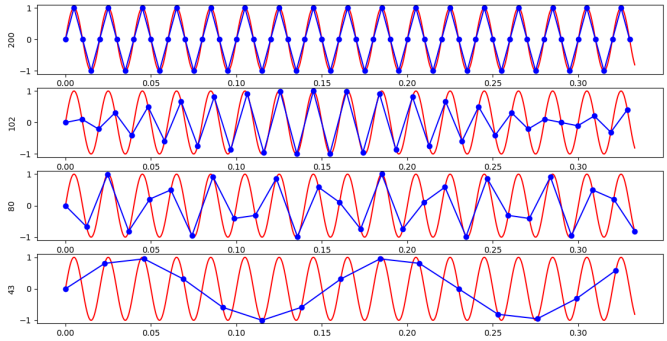
2020-05-29



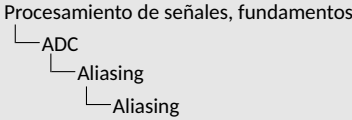
Aliasing

Simulando en Python

Diferentes frecuencias de sampleo para capturar una señal de 50hz



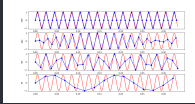
2020-05-29



Aliasing

Simulando en Python

Diferentes frecuencias de sampleo para capturar una señal de 50hz



Aliasing

Simulando en Python



Diferentes frecuencias de sampleo para capturar una señal de 50hz

```
import numpy as np
import matplotlib.pyplot as plt

signalFreq = 50
NC          = 1000
fsC         = 3000
tC          = np.arange(0,NC/fsC,1/fsC)
signalC     = np.sin(2*np.pi*signalFreq*tC)
fsD         = [200,102,80,43]

fig         = plt.figure()
signalC     = np.sin(2*np.pi*signalFreq*tC)+0.5*np.sin(2*np.pi*210*tC)
for i in range(len(fsD)):
    contiAxe = fig.add_subplot(4,1,i+1)
    plt.plot(tC,signalC,'r-',tC[::fsC//fsD[i]],signalC[::fsC//fsD[i]],'b-o')
    contiAxe.set_ylabel(fsD[i])

plt.show()
```

2020-05-29

Procesamiento de señales, fundamentos

ADC

Aliasing

Aliasing

Aliasing

Simulando en Python



```
Diferentes frecuencias de sampleo para capturar una señal de 50hz
import numpy as np
import matplotlib.pyplot as plt

signalFreq = 50
NC          = 1000
fsC         = 3000
tC          = np.arange(0,NC/fsC,1/fsC)
signalC     = np.sin(2*np.pi*signalFreq*tC)
fsD         = [200,102,80,43]

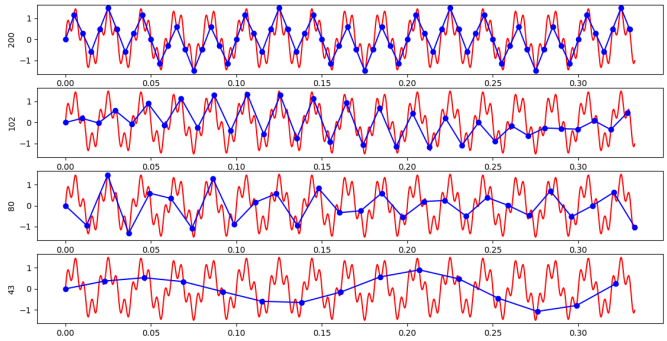
fig         = plt.figure()
signalC     = np.sin(2*np.pi*signalFreq*tC)+0.5*np.sin(2*np.pi*210*tC)
for i in range(len(fsD)):
    contiAxe = fig.add_subplot(4,1,i+1)
    plt.plot(tC,signalC,'r-',tC[::fsC//fsD[i]],signalC[::fsC//fsD[i]],'b-o')
    contiAxe.set_ylabel(fsD[i])

plt.show()
```

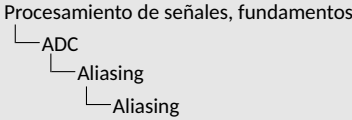
Aliasing

Simulando en Python

Que pasa si se suma ruido de alta frecuencia?



2020-05-29



Aliasing

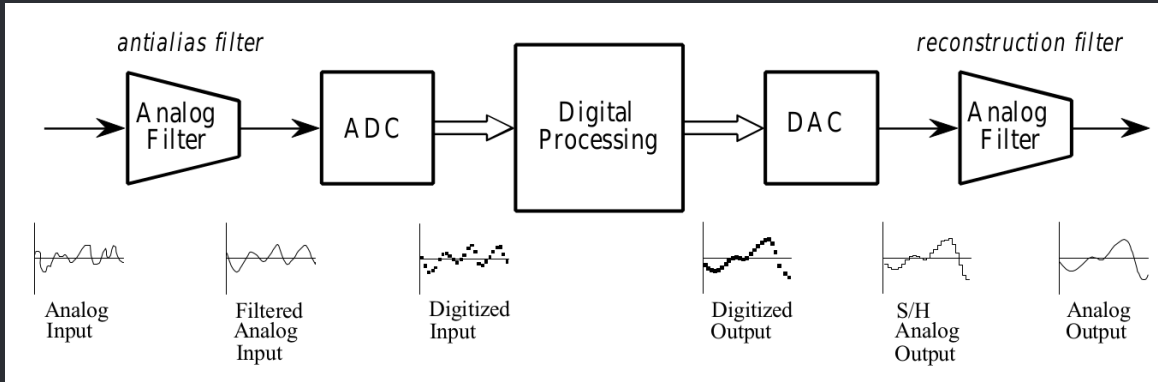
Simulando en Python

Que pasa si se suma ruido de alta frecuencia?

This block contains a small version of the figure shown in the main content area, illustrating the effect of adding high-frequency noise to a signal and how it affects the sampled data (blue dots) relative to the original signal (red line).

ADC

Bloque genérico de procesamiento



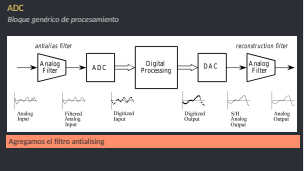
Agregamos el filtro antialiasing

2020-05-29

Procesamiento de señales, fundamentos

└ ADC

└ ADC



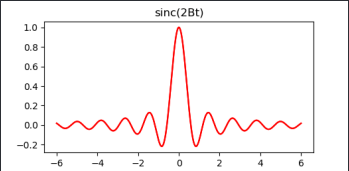
Teorema de sampleo

Teorema de Shannon

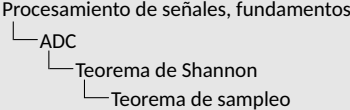
Teorema

La reconstrucción exacta de una señal periódica continua en banda base a partir de sus muestras, es matemáticamente posible si la señal está limitada en banda y la tasa de muestreo es superior al doble de su ancho de banda

$$x(t) = \sum_{n=-\infty}^{\infty} x_n \frac{\sin \pi(2Bt - n)}{\pi(2Bt - n)}.$$



2020-05-29



Teorema de sampleo

Teorema de Shannon

Teorema

La reconstrucción exacta de una señal periódica continua en banda base a partir de sus muestras, es matemáticamente posible si la señal está limitada en banda y la tasa de muestreo es superior al doble de su ancho de banda

Teorema de sampleo

Teorema de Shannon



Sampleo e interpolado

```
#!/usr/bin/ip3
import numpy as np
import matplotlib.pyplot as plt

signalFrec = 50
NC = 300
fsC = 1000
tC = np.arange(0,NC/fsC,1/fsC)
signalC = np.sin(2*np.pi*signalFrec*tC)
#signalC = np.sin(2*np.pi*signalFrec*tC)+0.5*np
        .sin(2*np.pi*210*tC)
fsD = np.array([200,102,80,45])
fig = plt.figure()

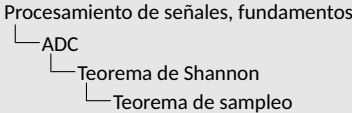
def interpolate(x, s, u):
    y=[]
```

```
        B = 1/(2*(s[1] - s[0]))
        for t in u:
            prom=0
            for n in range(len(x)):
                prom+=x[n]*np.sinc(2*B*t-n)
            y.append(prom)
        return y

for i in range(len(fsD)):
    contiAxe = fig.add_subplot(4,1,i+1)
    Xt=interpolate(signalC[:,fsC//fsD[i]],tC[:,
        fsC//fsD[i]],tC)
    plt.plot(tC,signalC,'r-',tC,Xt,'b-')
    contiAxe.set_ylabel(fsD[i])

plt.show()
```

2020-05-29



Teorema de sampleo

Teorema de Shannon

Sampleo e Interpolado

```
#!/usr/bin/ip3
import numpy as np
import matplotlib.pyplot as plt

signalFrec = 50
NC = 300
fsC = 1000
tC = np.arange(0,NC/fsC,1/fsC)
signalC = np.sin(2*np.pi*signalFrec*tC)
#signalC = np.sin(2*np.pi*signalFrec*tC)+0.5*np
        .sin(2*np.pi*210*tC)
fsD = np.array([200,102,80,45])
fig = plt.figure()

def interpolate(x, s, u):
    y=[]

    B = 1/(2*(s[1] - s[0]))
    for t in u:
        prom=0
        for n in range(len(x)):
            prom+=x[n]*np.sinc(2*B*t-n)
        y.append(prom)
    return y

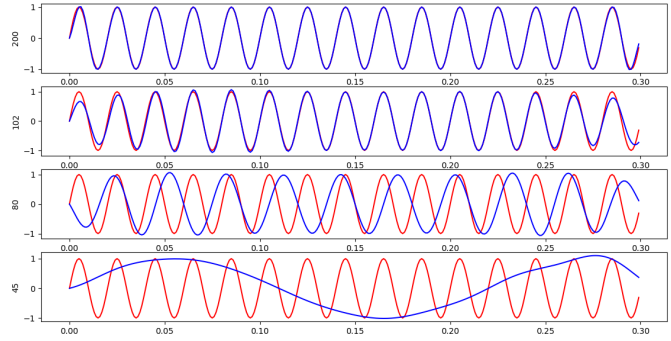
for i in range(len(fsD)):
    contiAxe = fig.add_subplot(4,1,i+1)
    Xt=interpolate(signalC[:,fsC//fsD[i]],tC[:,
        fsC//fsD[i]],tC)
    plt.plot(tC,signalC,'r-',tC,Xt,'b-')
    contiAxe.set_ylabel(fsD[i])

plt.show()
```

Teorema de sampleo

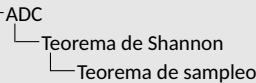
Teorema de Shannon

Sampleo e interpolado



2020-05-29

Procesamiento de señales, fundamentos



Teorema de sampleo

Teorema de Shannon

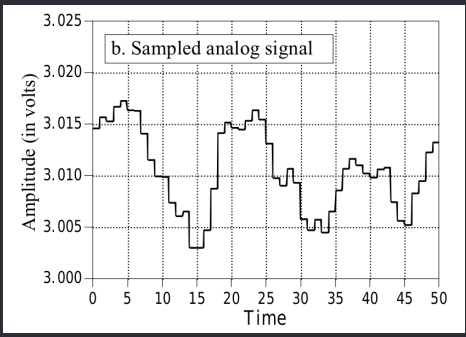
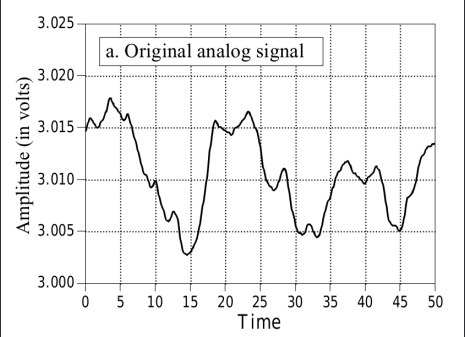
Sampleo e interpolado

Sampleo

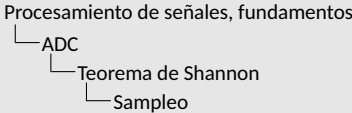
Filtro Antialias

FAA

Filtro **analógico** Pasabajos que elimina o al menos mitiga el efecto de aliasing



2020-05-29



Sampleo

Filtro Antialias

FAA

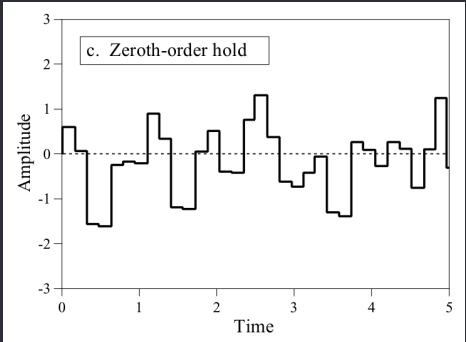
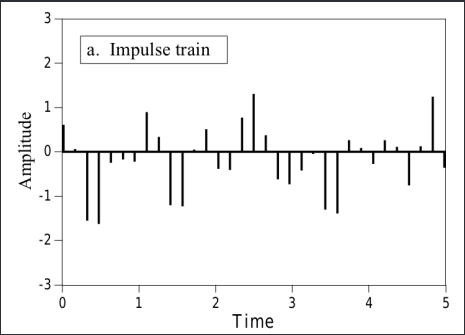
Filtro **analógico** Pasabajos que elimina o al menos mitiga el efecto de aliasing

Sampleo

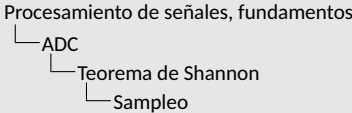
Filtro reconstructor

Filtro reconstructor

Filtro **analógico** Pasabajos que suaviza la salida del DAC eliminando frecuencias mas alla de la $F_s/2$



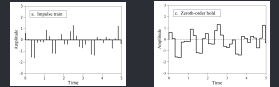
2020-05-29



Sampleo

Filtro reconstructor

Filtro **analógico** Pasabajos que suaviza la salida del DAC eliminando frecuencias mas alla de la $F_s/2$

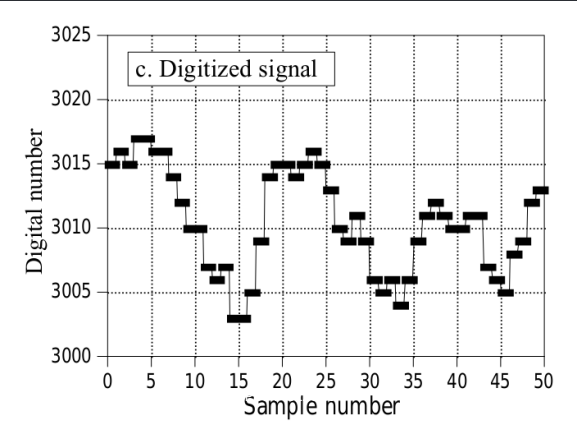


Sampleo

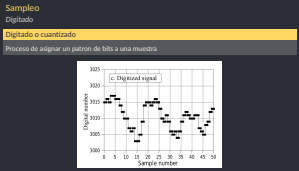
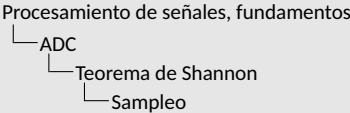
Digitado

Digitado o cuantizado

Proceso de asignar un patron de bits a una muestra



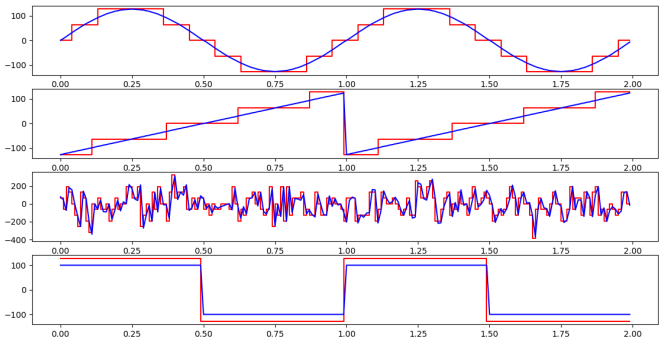
2020-05-29



Ruido de cuantización

Ejemplo de cuantización

Diferentes formas de onda cuantizadas



2020-05-29

- Procesamiento de señales, fundamentos
 - Cuantización
 - Ejemplos
 - Ruido de cuantización

Ruido de cuantización

Ejemplo de cuantización

Diferentes formas de onda cuantizadas



Ruido de cuantización

Cuantización en python

```
import numpy as np
import scipy.signal as sc
import matplotlib.pyplot as plt

signalFrec = 1
NC = 200
fsC = 100
Bits = 2
tC = np.arange(0,NC/fsC,1/fsC)
signalC = np.array([(2**7-1)*np.sin(2*np.pi*signalFrec*tC),
(2**7-1)*sc.sawtooth(2*np.pi*tC,1),
(2**7-1)*np.random.normal(0,1,len(tC)),
100*sc.square(2*np.pi*tC,0.5)],dtype='int16')

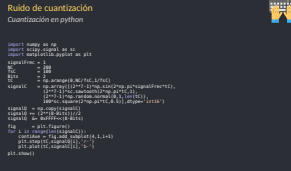
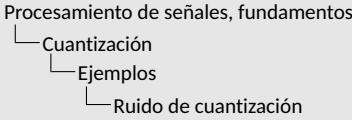
signalQ = np.copy(signalC)
signalQ += (2**((8-Bits)))/2
signalQ &= 0xFFFF<<(8-Bits)

fig = plt.figure()
for i in range(len(signalC)):
    contiAxe = fig.add_subplot(4,1,i+1)
    plt.step(tC,signalQ[i], 'r-')
    plt.plot(tC,signalC[i], 'b-')

plt.show()
```



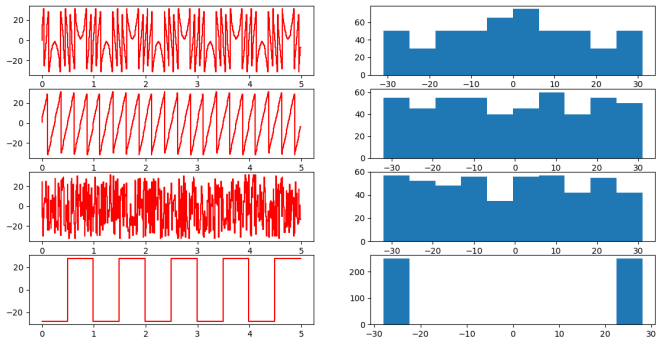
2020-05-29



Ruido de cuantización

Histogramas

Histogramas de ruido para cada señal



2020-05-29

- Procesamiento de señales, fundamentos
 - Cuantización
 - Ejemplos
 - Ruido de cuantización



Ruido de cuantización

Histogramas

Histogramas en Python

```
import numpy as np
import scipy.signal as sc
import matplotlib.pyplot as plt

signalFrec = 1
NC = 500
fsC = 100
Bits = 2
tC = np.arange(0,NC/fsC,1/fsC)
signalC = np.array([(2**7-1)*np.sin(2*np.pi*signalFrec*tC),
                    (2**7-1)*sc.sawtooth(2*np.pi*tC,1),
                    (2**7-1)*np.random.normal(0,1,len(tC)),
                    100*sc.square(2*np.pi*tC,0.5)],dtype='int16')

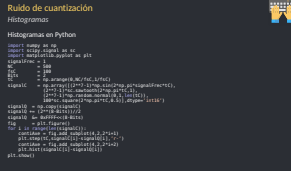
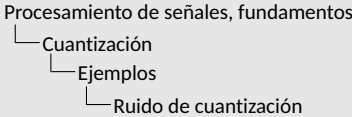
signalQ = np.copy(signalC)
signalQ += (2**(8-Bits))/2
signalQ &= 0xFFFF<<(8-Bits)

fig = plt.figure()
for i in range(len(signalC)):
    contiAxe = fig.add_subplot(4,2,2*i+1)
    plt.step(tC,signalC[i]-signalQ[i],'r-')
    contiAxe = fig.add_subplot(4,2,2*i+2)
    plt.hist(signalC[i]-signalQ[i])

plt.show()
```



2020-05-29



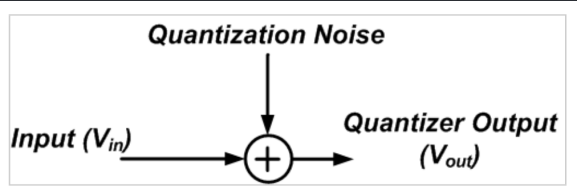
Ruido de cuantización

Modelo estadístico

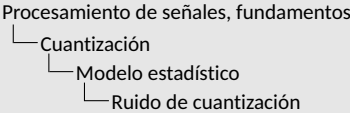
En el caso de que se cumplan las siguientes premisas:

- La entrada se distancia de los diferentes niveles de cuantización con igual probabilidad
- El error de cuantización NO esta correlacionado con la entrada
- El cuantizador cuanta con un numero relativamente largo de niveles
- Los niveles de cuantización son uniformes

Se puede considerar la cuantización como un ruido aditivo a la señal según el siguiente esquema:



2020-05-29



Ruido de cuantización
Modelo estadístico

En el caso de que se cumplan las siguientes premisas:

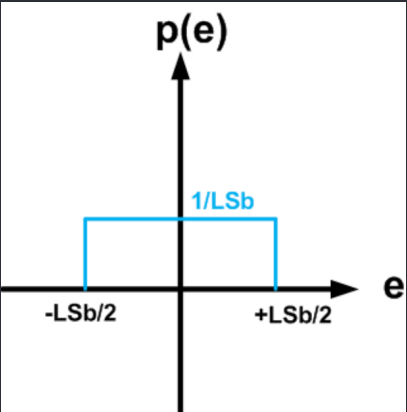
- La entrada se distancia de los diferentes niveles de cuantización con igual probabilidad
- El error de cuantización NO esta correlacionado con la entrada
- El cuantizador cuanta con un numero relativamente largo de niveles
- Los niveles de cuantización son uniformes

Se puede considerar la cuantización como un ruido aditivo a la señal según el siguiente esquema:

A small block diagram showing a quantizer model. An input signal, labeled V_{in} , enters a circular block containing a plus sign (+). A vertical arrow labeled "Quantization Noise" points down into the top of this block. An output arrow labeled V_{out} exits the right side of the block.

Ruido de cuantización

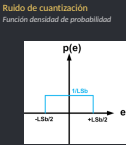
Función densidad de probabilidad



$$\int_{-\frac{LSb}{2}}^{\frac{LSb}{2}} p(e) de = 1$$

2020-05-29

- Procesamiento de señales, fundamentos
 - Cuantización
 - Modelo estadístico
 - Ruido de cuantización



$$\int_{-\frac{LSb}{2}}^{\frac{LSb}{2}} p(e) de = 1$$

Ruido de cuantización

Potencia de ruido de cuantización

$$P_q = \int_{-\frac{lsb}{2}}^{\frac{lsb}{2}} e^2 p(e) de$$

$$P_q = \int_{-\frac{lsb}{2}}^{\frac{lsb}{2}} e^2 \frac{1}{lsb} de$$

$$P_q = \frac{1}{lsb} \left(\frac{e^3}{3} \Big|_{-\frac{lsb}{2}}^{\frac{lsb}{2}} \right)$$

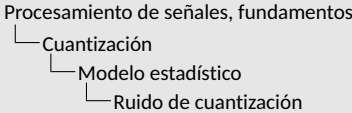
$$P_q = \frac{1}{lsb} \left(\frac{(\frac{lsb}{2})^3}{3} - \frac{(\frac{-lsb}{2})^3}{3} \right)$$

$$P_q = \frac{1}{lsb} \left(\frac{lsb^3}{24} + \frac{lsb^3}{24} \right)$$

Potencia de ruido de cuantización

$$P_q = \frac{lsb^2}{12}$$

2020-05-29



Ruido de cuantización

Potencia de ruido de cuantización

$$P_q = \int_{-\frac{lsb}{2}}^{\frac{lsb}{2}} e^2 p(e) de$$
$$P_q = \frac{1}{lsb} \left(\frac{(\frac{lsb}{2})^3}{3} - \frac{(\frac{-lsb}{2})^3}{3} \right)$$
$$P_q = \int_{-\frac{lsb}{2}}^{\frac{lsb}{2}} e^2 \frac{1}{lsb} de$$
$$P_q = \frac{1}{lsb} \left(\frac{lsb^3}{24} + \frac{lsb^3}{24} \right)$$
$$P_q = \frac{1}{lsb} \left(\frac{e^3}{3} \Big|_{-\frac{lsb}{2}}^{\frac{lsb}{2}} \right)$$

Potencia de ruido de cuantización

$$P_q = \frac{lsb^2}{12}$$

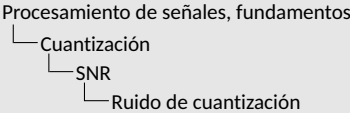
Ruido de cuantización

Relación señal a ruido

$$\begin{aligned}input &= \frac{Amp}{2} \sin(t) \\ P_{input} &= \frac{1}{T} \int_0^T \left(\frac{Amp}{2} \sin(t) \right)^2 dt \\ P_{input} &= \frac{1}{T} \left(\frac{Amp}{2} \right)^2 * \left(\frac{t}{2} - \frac{\sin(2t)}{4} \right) \Big|_0^T \\ P_{input} &= \frac{Amp^2 T}{4T} \frac{1}{2} \\ P_{input} &= \frac{Amp^2}{8}\end{aligned}$$

$$\begin{aligned}lsb &= \frac{Amp}{2^N} \\ P_{ruido} &= \frac{lsb^2}{12} \\ P_{ruido} &= \frac{\left(\frac{Amp}{2^N} \right)^2}{12} \\ P_{ruido} &= \frac{Amp^2}{12 * 2^{2N}}\end{aligned}$$

2020-05-29



Ruido de cuantización

Relación señal a ruido

$$\begin{aligned}input &= \frac{Amp}{2} \sin(t) \\ P_{input} &= \frac{1}{T} \int_0^T \left(\frac{Amp}{2} \sin(t) \right)^2 dt \\ P_{input} &= \frac{1}{T} \left(\frac{Amp}{2} \right)^2 * \left(\frac{t}{2} - \frac{\sin(2t)}{4} \right) \Big|_0^T \\ P_{input} &= \frac{Amp^2 T}{4T} \frac{1}{2} \\ P_{input} &= \frac{Amp^2}{8}\end{aligned}$$

$$\begin{aligned}lsb &= \frac{Amp}{2^N} \\ P_{ruido} &= \frac{lsb^2}{12} \\ P_{ruido} &= \frac{\left(\frac{Amp}{2^N} \right)^2}{12} \\ P_{ruido} &= \frac{Amp^2}{12 * 2^{2N}}\end{aligned}$$

Ruido de cuantización

Relación señal a ruido

$$SNR = 10 \log_{10} \left(\frac{P_{input}}{P_{ruido}} \right)$$

$$SNR = 10 \log_{10} \left(\frac{\frac{Amp^2}{8}}{\frac{Amp^2}{12 * 2^{2N}}} \right)$$

$$SNR = 10 \log_{10} \left(\frac{3 * 2^{2N}}{2} \right)$$

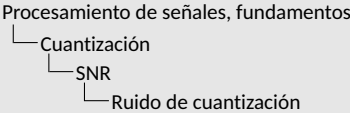
$$SNR = 10 \log_{10} \left(\frac{3}{2} \right) + 10 \log_{10} (2^{2N})$$

$$SNR_{N=10} \approx 62dB$$

$$SNR_{N=11} \approx 68dB$$

$$SNR = 1,76 + 6,02 * N$$

2020-05-29



Ruido de cuantización

Relación señal a ruido

$$SNR = 10 \log_{10} \left(\frac{P_{input}}{P_{ruido}} \right)$$
$$SNR = 10 \log_{10} \left(\frac{\frac{Amp^2}{8}}{\frac{Amp^2}{12 * 2^{2N}}} \right)$$

$$SNR = 10 \log_{10} \left(\frac{3 * 2^{2N}}{2} \right)$$
$$SNR = 10 \log_{10} \left(\frac{3}{2} \right) + 10 \log_{10} (2^{2N})$$

SNR	
$SNR = 1,76 + 6,02 * N$	$SNR_{N=10} \approx 62dB$ $SNR_{N=11} \approx 68dB$

Ruido de cuantización

Densidad espectral de potencia de ruido

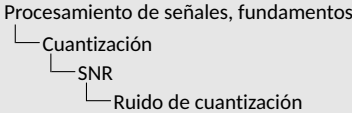
Si consideramos la potencia de ruido uniformemente distribuido en todo el espectro desde $-F_s$ hasta $+F_s$, nos queda que:

Densidad espectral de potencia de ruido

$$S_{espectral}(f) = \frac{P_q}{F_s}$$

Entonces como puedo mejorar la SNR de un sistema?

2020-05-29



Ruido de cuantización

Densidad espectral de potencia de ruido

Si consideramos la potencia de ruido uniformemente distribuido en todo el espectro desde $-F_s$ hasta $+F_s$, nos queda que:

Densidad espectral de potencia de ruido

$$S_{espectral}(f) = \frac{P_q}{F_s}$$

Entonces como puedo mejorar la SNR de un sistema?

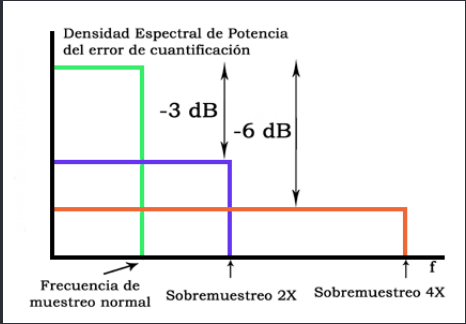
Sobremuestreo

Densidad espectral de potencia de ruido

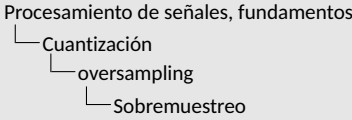
Oversampling x4

$$S_{espectral}(f) = \frac{P_q}{4 * F_s}$$

Que hago si tengo un AD de 10bits y deseo una SNR de 68dB? $SNR_{10} \approx 62dB$ Pero si sobremuestreo a 4x obtengo **6dB** extras



2020-05-29



Sobremuestreo
Densidad espectral de potencia de ruido

Oversampling x4

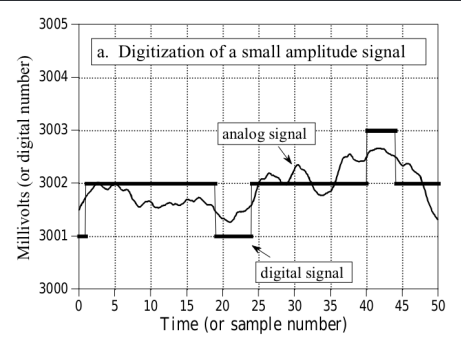
$$S_{espectral}(f) = \frac{P_q}{4 * F_s}$$

Que hago si tengo un AD de 10bits y deseo una SNR de 68dB? $SNR_{10} \approx 62dB$ Pero si sobremuestreo a 4x obtengo **6dB** extras

Dithering

Dithering

Tecnica de agregado de ruido antes del ADC para prevenir que señales con poca variacion sean samoleadas siempre con el mismo valor



2020-05-29

- Procesamiento de señales, fundamentos
 - Cuantización
 - oversampling
 - Dithering

Dithering

Técnica de agregado de ruido antes del ADC para prevenir que señales con poca variación sean samoleadas siempre con el mismo valor