

unité de performance :

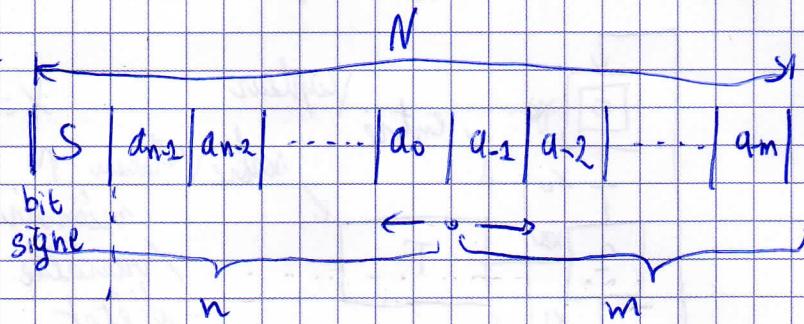
- M FLOPS : Million floating points operation per seconds
- M OPS : " opérations per second ( $f_{op}$ )
- M IPS : " instruction " "
- M BPS <sup>MBps</sup> : Mega Bytes " "
- M MACS : Million MAC " "

~~XX~~ DSP

24-02-2025

Cours

format point fixe



$$N = n + m$$

$$V = (-1)^s \left[ \sum_{i=1}^n a_{n-i} 2^{m-i} + \sum_{j=0}^{m-1} a_{-j} \frac{1}{2^j} \right]$$

$$V = (10, 4) \Rightarrow (10)_B$$

$\begin{array}{r} 10 \\   \\ 0,5 \\   \\ 1 \\   \\ 0 \\   \\ 1 \\   \\ 0 \end{array}$	$\begin{array}{c} \uparrow \\ 0,4 \times 2 = 0,8 \\ \uparrow \\ 0,8 \times 2 = 1,6 \\ \uparrow \\ 1,6 \times 2 = 1,2 \\ \uparrow \\ 0,2 \times 2 = 0,4 \\ \uparrow \\ 0,4 \times 2 = 0,8 \end{array}$	$\begin{array}{c} 0 \\   \\ 1 \\   \\ 1 \\   \\ 0 \end{array}$
		$\begin{array}{c} 0 \\   \\ 1 \\   \\ 1 \\   \\ 0 \end{array}$

$$n = \text{Round} \left[ \frac{\log(|V|)}{2} + 0,5 \right] + 1$$

Pour un vecteur  $v$ , pour avoir une précision optimale, on calcule pour  $V_{max}$  la valeur maximale dans le vecteur

IAE

Cours

Exo → Quelles sont les équations de similitude



DSP

3 - 3 - 2025

Cours

→ Format des données

Format flottante (IEEE 754)

Simple précision (32 bits = N)

$$V = (-1)^S \times (b)^E \times M$$

$N = 32 \text{ bit}$



bit sign   Exposant   Mantisse 23 bits

a) Si  $\begin{cases} e = FF \\ f \neq 0 \end{cases} \Rightarrow V = NaN$  b) Si  $\begin{cases} e = FF \\ f = 0 \end{cases} \Rightarrow \begin{cases} S = 1 \Rightarrow V = -\infty \\ S = 0 \Rightarrow V = +\infty \end{cases}$

c) Si  $\begin{cases} e = 00 \\ f \neq 0 \end{cases} \Rightarrow V = (-1)^S \cdot 2^{(e-126)} \cdot M$  (1-2)  $M = 0.F_{22}2^{-1} + F_{21}2^{-2} + \dots + F_0 2^{-23}$

d) Si  $0 < e < 255 \Rightarrow V = (-1)^S \cdot 2^{(e-127)} \times (1.F) \cdot M$  (2-2)  $M = 1 + F_{22}2^{-1} + F_{21}2^{-2} + \dots + F_0 2^{-23}$

$$\begin{cases} 2^E = 2^{e-126} \\ 2^E = 2^{e-127} \end{cases}$$

## Example 5:

$$V = +314,4$$

$$(314)_{10} \equiv (100111010)_2$$

$$(01100110011001)_2 = (01100110011001)_2$$

Vd = 100 111 010, 011 0 011 100 110 011

$$= \underline{1.00111010_011001100110011} .2^8$$

$$S = 0 \Rightarrow +$$

$$E = f \cdot B$$

$$1.F = 1,00111010011001100110011$$

ide (1-2) on a

$$2^{(\ell-127)} = 2^8 \Rightarrow \ell = 135 \equiv (10000111)_2$$

$$\Rightarrow V = 01000011100111010011001100110011$$

S e M

### Example 2:

$$S=0 \Rightarrow V>0$$

$$E = 2 \quad \text{if } e = 129 \Rightarrow E = 2$$

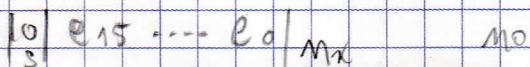
$$M = 1, F = \frac{F_0}{2^0} 2^{-1} + \frac{F_{21}}{2^1} 2^{-2} + \frac{F_{20}}{2^2} 2^{-3} + \frac{F_{19}}{2^3} 2^{-4} = 2^{-1} + 2^{-2} + 2^{-3} + 2^{-4} = 1,9375$$

$$\Rightarrow V = (-1)^0 \cdot 2^2 \cdot (2,9375) = 7,75$$

Exemple à faire au maison :

Convert an decimal

→ double precision



10-3-2023

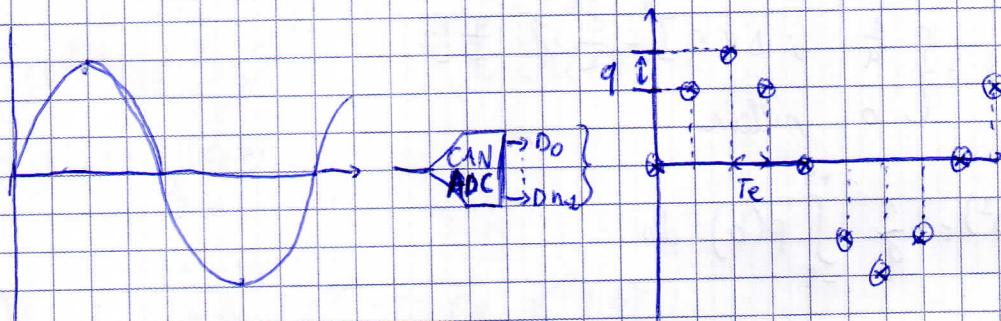


CAN : est l'ensemble de 3 processus :

→ échantillonage : discrétisation dans le temps

→ quantification : - - - l'amplitude

→ Codage : codage vers un format des données



### 1) échantillonage:

$T_e$  = temps d'échantillonage

$$T_e = \frac{1}{F_e} \quad F_e \geq 2 f_s$$

$F_e$  = fréquence ..

$$X(nT_e) = x(t) T_e \sum_{n=-\infty}^{+\infty} \delta(t - nT_e)$$

### 2) Quantification:

$q$  = pas de quantification

→ algorithme d'arrondissement :

>> math.floor(7.5) # arrondis supérieur

>> math.round(2.6653) # arrondi normal

3

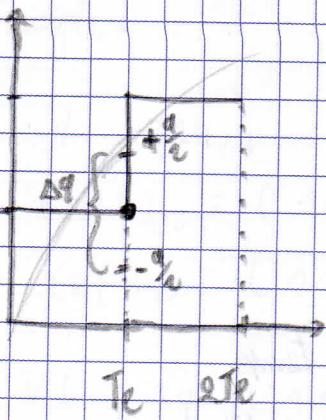
>> math.round(2.6653, 2) # " 2 chiffres après la virgule

2.67

>> math.floor(2.665) # " inférieur

TP3  
évaluation : 5/5 (2)

29



$$\Delta q \in ]-\frac{q}{2}; +\frac{q}{2}[$$

l'arrondissement crée un ~~peu de~~ bruit "bruit de quantification", on peut calculer la probabilité comme suivant

$$p(b) = \begin{cases} \frac{1}{q} & ; \Delta q \in ]-\frac{q}{2}; +\frac{q}{2}[ \\ 0 & \text{ailleurs} \end{cases}$$

$$E(b^2) = \frac{1}{q} \int_{-\frac{q}{2}}^{+\frac{q}{2}} p(b) db$$

puissance de bruit est donné comme :

$$P_d = E(b^2) = \frac{q^2}{12}$$

Rapport Signal Bruit

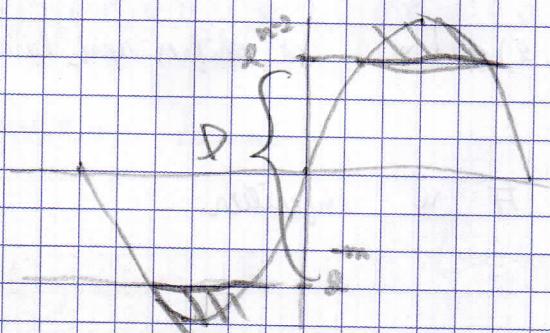
$$RSB = 10 \log \left( \frac{P_s}{P_b} \right)$$

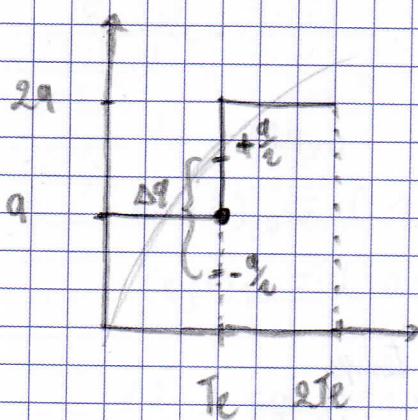
### 3) Codage

→ Codage formate fixe:

$$N = n + m \Rightarrow q = 2^{-m}$$

Domaine de définition:  $D = [2^{-m}; 2^{n-1}]$





$$\Delta q \in ]-\frac{q}{2}; +\frac{q}{2}[$$

l'arrondissement crée un ~~bruit~~ bruit "bruit de quantification", on peut calculer la probabilité comme suivant

$$p(b) = \begin{cases} \frac{1}{q} & ; \Delta q \in ]-\frac{q}{2}; +\frac{q}{2}[ \\ 0 & \text{ailleurs} \end{cases}$$

$$E(b^2) = \frac{1}{q} \int_{-\frac{q}{2}}^{+\frac{q}{2}} p(b) db$$

puissance de bruit est donné comme :

$$P_d = E(b^2) = \frac{q^2}{12}$$

Rapport Signal Bruit

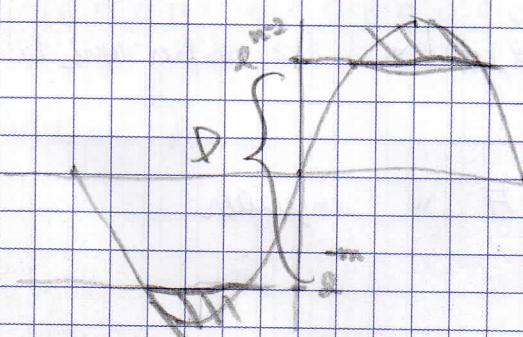
$$RSB = 10 \log \left( \frac{P_s}{P_b} \right)$$

### 3) Codage

→ Codage formate fixe:

$$N = n + m \Rightarrow q = 2^{-m}$$

Domaine de définition:  $D = [2^{-m}; 2^{n-1}]$



## Dynamique

$$D_N = 20 \log \left( \frac{2^{n-1}}{2^{-m}} \right) \approx 6,02 (N-1)$$

→ codage format flottant

$$N = \underbrace{N_E}_{\substack{\text{nbr} \\ \text{bit}}} + \underbrace{N_M}_{\substack{\text{nbr} \\ \text{bit}}} + \underbrace{1}_{\substack{\text{bit signe} \\ \text{exponent montice}}}$$

$$b = 2^{E-1} - 1$$

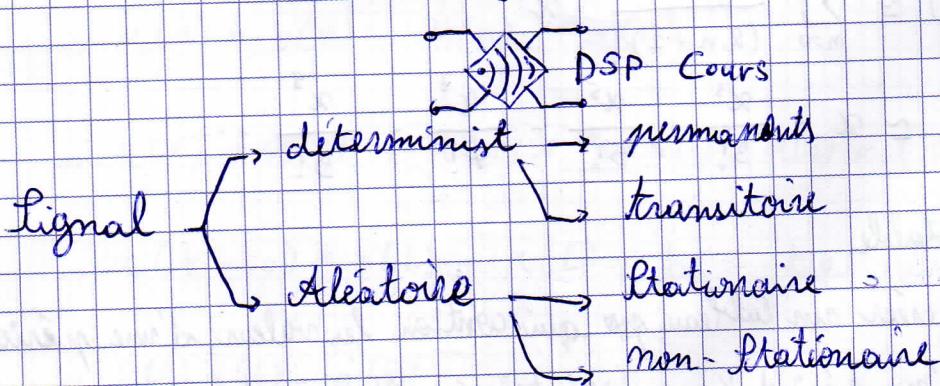
$$q = 2^{-F} \cdot 2^{(E-b)}$$

$$E \begin{cases} -126 + e \\ -127 + e \end{cases}$$

$$P = [-(2^{-E}) 2^b ; (2^{-E}) 2^b]$$

$$D_N \stackrel{?}{=} 20 \log \left( \frac{2^b}{2^{-E}} \right)$$

17-03-2025



Puissance moyen :  $P(t_1, t_2) = \frac{1}{t_2 - t_1} \int_{t_1}^{t_2} |x(t)|^2 dt$

Puissance total :  $\lim_{T \rightarrow \infty} \int_{-\frac{T}{2}}^{\frac{T}{2}} |x(t)|^2 dt$

Puissance d'un signal discret :  $P = \frac{1}{N} \sum_{n=0}^{N-1} |x(n)|^2$

Energie de signal discret  $E = \sum_{n=-\infty}^{+\infty} |x(n)|^2$

from an signal  $x(t)$

Il y a les caractéristiques suivantes

$$x(t) = A \sin(\omega_0 t + \varphi_0)$$

Amplitude

La fréquence

phase initiale

$$\rightarrow \phi(t) = (2\pi f_s t + \varphi_0) \quad \leftarrow \text{phase instantanée}$$

→ La création d'un signal périodique numériquement :

- ## • Série de Taylor (Développement limité)

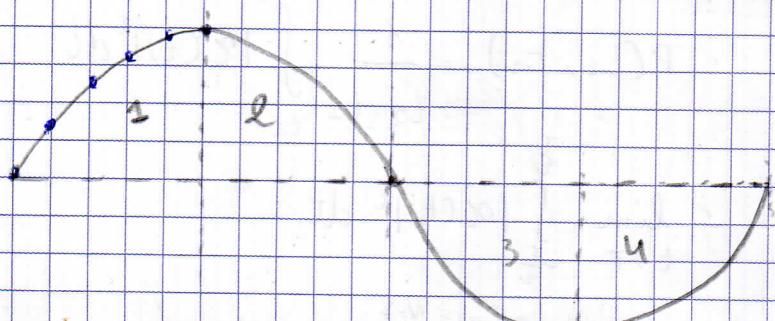
$$\sin(x) \simeq \sum_{n=0}^{+\infty} \frac{(-1)^n}{(2n+1)!} = x$$

(lim x)

$$\simeq x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} - \dots$$

• Look-up table

est de créer un tableau `psw` qui contient les valeurs d'une période "ou moins de que "un période" pour l'utiliser ultérieurement



$$3 = -1$$

L2-1

Est la  
mémoire de 1

$$A = [0,25; 0,46; 0,75; 0,79]$$

## Equation aux différences :

$$y(n) = A y(n-1) + B y(n-2) + C x(n-1)$$

pour  $n=2$        $y(1) = A y(0) + B y(-1) + C ; \quad x(0)=1$

$m > 2 \quad y(0)=0 ; \quad y(-1)=0$

$$y(n) = A y(n-1) - y(n-2)$$

pour  $f = 1,5 \text{ kHz}$

$$A = 2 \cos(\omega T) \rightarrow 0,765 \rightarrow A \cdot 2^{14} = 18,540$$

$$y_1 = C = 0,024 \rightarrow C \cdot 2^{14} = 15,137$$

$$y_2 = A y_1 = 0,707 \rightarrow y_2 \cdot 2^{14} = 11,585$$

2025-04-07

## Produit de convolution :

$$y = a(n) * b(n) ; \quad y(k) = \sum_{n=0}^{N-1} a(n) \cdot b(k-n)$$

$$= \sum_{n=0}^{N-1} a(n) \cdot b(m) ; \quad m = k-n$$

$$\rightarrow a(n) * b(n) = b(n) * a(n)$$

$$\rightarrow a(n) * a(n) + a(n) * y(n) = a(n) * (x(n) + y(n))$$

$$\rightarrow a(t-t_0) * b(t) = a(t) * b(t-t_0)$$

$$\rightarrow \delta(t) a(t) = a(t)$$

$$a(n) : n \in \underbrace{\{0, 1, \dots, N-1\}}_{N \text{ elements}} ; \quad b(m) : m \in \underbrace{\{0, 1, 2, \dots, M-1\}}_{M \text{ elements}}$$

$$y(k) : k \in \underbrace{\{0, 1, \dots, K-1\}}_{K \text{ elements}} ; \quad a(n) * b(m) = y(k)$$

$$K = N + M - 1$$

$$\text{Ex0: } a = [1, 2, 3]$$

$$N=3$$

$$b = [1, 2, 3]$$

$$M=3$$

$$y = ?$$

$$K=?$$

$$K=N+M-1=5$$

$$y(k) = \sum_{n=0}^{M-1} a(n) \cdot b(k-n)$$

$$y(0) = a(0) \cdot b(0) + \cancel{a(1) \cdot b(-1)}^0 + \cancel{a(2) \cdot b(-2)}^0 = 1$$

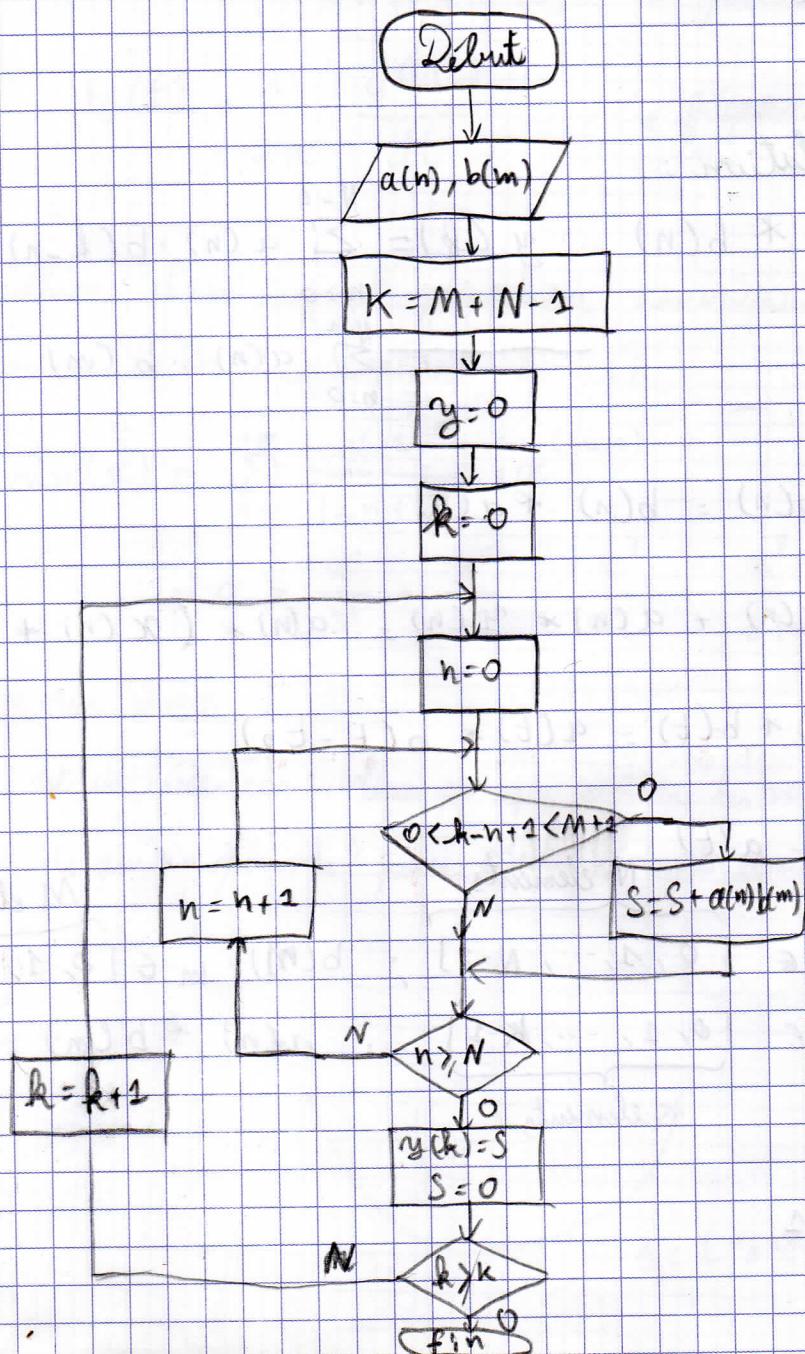
$$y(1) = a(0) \cdot b(1) + a(1) \cdot b(0) + \cancel{a(2) \cdot b(-1)}^0 = 4$$

$$y(2) = a(0) \cdot b(2) + a(1) \cdot b(1) + a(2) \cdot b(0) = 10$$

$$y(3) = \cancel{a(0) \cdot b(3)}^0 + a(1) \cdot b(2) + a(2) \cdot b(1) = 12$$

$$y(4) = a(0) \cdot \cancel{b(4)}^0 + a(1) \cdot \cancel{b(3)}^0 + a(2) \cdot b(2) = 9$$

$$y = [1 \ 4 \ 10 \ 12 \ 9]$$



Mathlab:

```
a = 1:3;  
b = 1:3;  
y = zeros(1, 5);
```

```
k = 1;  
for k = 1:5
```

```
S = 0;
```

```
for n = 1:3
```

```
if (k - n + 1 > 0 && k - n + 1 < 4)
```

```
S = S + a(n) * b(k - n + 1);
```

```
end;
```

```
y(k) = S;
```

```
end
```

```
end,
```

conv.c :

```
#include <stdio.h>
```

```
void conv ( int a[], int b[], int y[], int K, int N, int M ) {
```

```
int i, j, S;
```

```
for ( i=0 ; i<K ; i++ ) {
```

```
S=0;
```

```
for ( j=0 ; j<N ; j++ ) {
```

```
if ( i-j+1 > 0 && i-j+1 < M+1 ) {
```

```
S = S + a[j] * b[i-j];  
}
```

```
y[i]=S;
```

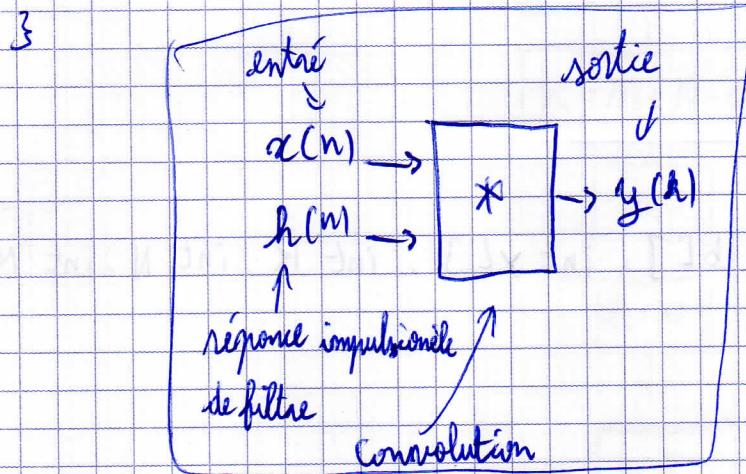
```
}
```

```
}
```

```
}
```

main.c :

```
Int main () {  
    int k=5, N=3, M=3;  
    int a [] = {1, 2, 3};  
    int b [] = {1, 2, 3};  
    int s [] = {0, 0, 0, 0, 0};  
  
    // appelle la fonction conv  
    conv (a, b, y, k, N, M);  
  
    // affichage  
    for (i=0, i<k, i++) {  
        printf ("y = %.d \n", y[i]);  
    }  
    return 0;
```



$$\Rightarrow y(k) = \sum_{n=0}^{N-1} h(n) x(k-n)$$

$h(n) \Rightarrow$  coefficient de filtre

filtre R I F  
e n o i  
p p n i  
o u i s  
n L s  
e S i  
e n e  
L e

pour obtenir implémenter le filtre, on doit l'implémenter en Z

$$H(z) = \sum_{n=0}^{N-1} h(n) z^{-n}$$

