



Sommaire

- 1 Objectifs
- 2 Fonctionnement
- 3 Les paramètres du système
- 4 Les paramètres de l'échelon / rampe
- 5 Génération de l'échelon $x(n)=\text{constante}$, ou une rampe $x(n)=n$
- 6 Définition de la fonction Sys2All()
- 7 Calcul des réponses du système
- 8 Affichage de l'entrée $x(n)$ et les sorties
- 9 Le programme Arduino

Objectifs

1. Implémentation d'un système du 2nd ordre : $y(n) = \text{> } \{x(n-1), y(n-1), y(n-2)\}$
2. Définition de la nouvelle fonction générique Sys2All()
3. La réponse indicielle d'un système en 2nd ordre
4. La réponse à une rampe d'un système en 2nd ordre
5. Etc.

Fonctionnement

On va aborder dans ce tuto l'analyse de plusieurs systèmes du second ordre en boucle ouverte (BO). On va particulièrement utiliser la réponse indicielle et la réponse à une rampe. On fera appel à la nouvelle Sys2All() qui permet de numériser (discrétiser) un système du second ordre caractérisé par sa pulsation propre ω_n et le coefficient d'amortissement ζ .

On verra dans la suite de la vidéo l'effet du ζ et ω_n sur la rapidité du système ainsi son dépassement.



Les paramètres du système

```
#define Fn 10.00
#define Zeta 0.7
#define K 1.0
#define T_ms 2
```

Les paramètres de l'échelon / rampe

```
#define A_step 10.0 // Amplitude
#define c_step 500 // Période de l'échelon = 2*c_step*T_ms
unsigned long c=0; // Compteur (période)
bool Step=false;
```

Génération de l'échelon $x(n)=\text{constante}$, ou une rampe $x(n)=n$

```
c++; c=c%c_step;
if(!c)
{
    Step=!Step;
```



```
c=0;

}

//x_nn=A_step*(double)Step; // Réponse à un échelon x(n)=cte

x_nn=(double)c;           // Réponse à une rampe x(n)=n
```

Définition de la fonction Sys2All()

La fonction Sys2All() prend en entrée les paramètres du système et le signal de l'entrée. Puis, elle renvoie la sortie du système (voir le tuto précédent pour plus des détails).

```
double Sys2All(double x_nn, double *x, double *y, double zeta, double wn, double k,
double T)

{

// Paramètre du système

double a1=2.0*zeta/wn;

double a2=1.0/(wn*wn);

const double b0=(a1/(2.0*T))+(a2/(T*T));

const double b1=-2.0*a2/(T*T);

const double b2=(-1.0*a1/(2.0*T))+(a2/(T*T));
```



```
const double b[3]={b0,b1,b2};

// Variables de l'entrée et la sortie

double y_nn=0.0;

// Calcul de la nouvelle sortie

y_nn= -(y[0]*(1.0+b[1]))-(y[1]*b[2])+(k*x[0]); // y[1]: y(n-2), y[0]: y(n-1)
y_nn/=b[0];

// Mise à jour de la sortie

y[1]=y[0];
y[0]=y_nn;

// Mise à jour de la sortie

x[0]=x_nn;

// Renvoi du résultat

return y_nn;
}
```



Calcul des réponses du système

```
Assev_Sys2ndAll | Arduino 1.8.10
Fichier Edition Croquis Outils Aide
Assev_Sys2ndAll
70 }
71 //x_nn=A_step*(double)Step; // Réponse à un
72 x_nn=(double)c; // Réponse à une r
73
74 // Calcul des sorties
75 y_nn1=Sys2All(x_nn, x1, y1, 0.1, Wn, K, T_s)
76 y_nn2=Sys2All(x_nn, x2, y2, 0.7, Wn, K, T_s)
77 y_nn3=Sys2All(x_nn, x3, y3, 1.0, Wn, K, T_s)
78 y_nn4=Sys2All(x_nn, x3, y3, 3.0, Wn, K, T_s)
79
80 // Affichage x(n), y1(n), y2(n)
81 Serial.print(x_nn); Serial.print(",");
82 Serial.print(y_nn1); Serial.print(",");
83 Serial.print(y_nn2); Serial.print(",");
84 Serial.print(y_nn3); Serial.print(",");
85 Serial.print(y_nn4); Serial.print(",");
86 Serial.print("\n");
87 }
88
89 // Le croquis utilise 4516 octets (1%) de l'espace de stockage de programmes. Le maximum est de 253
90 // Les variables globales utilisent 287 octets (3%) de mémoire dynamique, ce qui laisse 7905 octets
```

```
y_nn1=Sys2All(x_nn, x1, y1, 0.1, Wn, K, T_s);
y_nn2=Sys2All(x_nn, x2, y2, 0.7, Wn, K, T_s);
y_nn3=Sys2All(x_nn, x3, y3, 1.0, Wn, K, T_s);
y_nn4=Sys2All(x_nn, x3, y3, 3.0, Wn, K, T_s);
```

Affichage de l'entrée $x(n)$ et les sorties

```
Serial.print(x_nn); Serial.print(",");
Serial.print(y_nn1); Serial.print(",");
Serial.print(y_nn2); Serial.print(",");
```



```
Serial.print(y_nn3); Serial.print(",");

Serial.println(y_nn4);
```

Le programme Arduino

```
/*
 * 1. Implémentation d'un système du 2nd ordre
 *   y(n)=> {x(n-1), y(n-1), y(n-2)}
 * 2. Définition de la fonction Générique Sys2All()
 * 3. La réponse indicielle d'un système en 2nd ordre
 * 4. La réponse à une rampe d'un système en 2nd ordre
 * 5. Etc.
 *
 *           -----
 *           -       -
 * x(n) ----- SYS2 ----- y1(n)
 *           -       -
 *           -----
 *           -----
 *           -       -
 * x(n) ----- SYS2 ----- y2(n)
 *           -       -
 *           -----
 *           -----
 *           -       -
 * x(n) ----- SYS2 ----- y3(n)
 *           -       -
 *           -----
 *           -----
 *           -       -
 * x(n) ----- SYS2 ----- y4(n)
 *           -       -
 *           -----
 */

#define Fn 10.00
```



Asservissement | Arduino #8: Réponse à une rampe d'un système du second ordre

```
#define Zeta 0.7
#define K 1.0
#define T_ms 2

#define A_step 10.0 // Amplitude
#define c_step 500 // Période de l'échelon = 2*c_step*T_ms

double Wn=2.0*PI*Fn;
double T_s=(double)T_ms/1000.0;

double x_nn=0.0;

double y_nn1=0.0;
double y_nn2=0.0;
double y_nn3=0.0;
double y_nn4=0.0;

double x1[2], y1[3]; // Sys2 1
double x2[2], y2[3]; // Sys2 2
double x3[2], y3[3]; // Sys2 3
double x4[2], y4[3]; // Sys2 4

// Paramètres de l'échelon
unsigned long c=0; // Compteur (période)
bool Step=false;

void setup()
{
  // Port série de la réponse du système
  Serial.begin(19200);
}

void loop()
{
  // Le signal échelon x(n) => [0, A_step]
  c++; c=c%c_step;
  if(!c)
  {
    Step=!Step;
    c=0;
  }
  //x_nn=A_step*(double)Step; // Réponse à un échelon x(n)=cte
  x_nn=(double)c; // Réponse à une rampe x(n)=n
```



Asservissement | Arduino #8: Réponse à une rampe d'un système du second ordre

```
// Calcul des sorties
y_nn1=Sys2All(x_nn, x1, y1, 0.1, Wn, K, T_s);
y_nn2=Sys2All(x_nn, x2, y2, 0.7, Wn, K, T_s);
y_nn3=Sys2All(x_nn, x3, y3, 1.0, Wn, K, T_s);
y_nn4=Sys2All(x_nn, x3, y3, 3.0, Wn, K, T_s);
// Affichage x(n), y1(n), y2(n)
Serial.print(x_nn); Serial.print(",");
Serial.print(y_nn1); Serial.print(",");
Serial.print(y_nn2); Serial.print(",");
Serial.print(y_nn3); Serial.print(",");
Serial.println(y_nn4);

// Période d'échantillonnage
delay(T_ms);
}

double Sys2All(double x_nn, double *x, double *y, double zeta, double wn, double k,
double T)
{
    // Paramètre du système
    double a1=2.0*zeta/wn;
    double a2=1.0/(wn*wn);

    const double b0=(a1/(2.0*T))+a2/(T*T);
    const double b1=-2.0*a2/(T*T);
    const double b2=(-1.0*a1/(2.0*T))+a2/(T*T);
    const double b[3]={b0,b1,b2};

    // Variables de l'entrée et la sortie
    double y_nn=0.0;
    // Calcul de la nouvelle sortie
    y_nn= -(y[0]*(1.0+b[1]))-(y[1]*b[2])+(k*x[0]); // y[1]: y(n-2), y[0]: y(n-1)
    y_nn/=b[0];
    // Mise à jour de la sortie
    y[1]=y[0];
    y[0]=y_nn;

    // Mise à jour de la sortie
    x[0]=x_nn;
    // Renvoi du résultat
    return y_nn;
}
```




Asservissement | Arduino #8: Réponse à une rampe d'un système du second ordre

```
}
```

[Accueil Asservissement avec Arduino](#)

Click to rate this post!

[Total: 2 Average: 5]