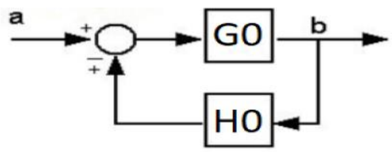
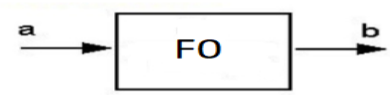
Moises Romero Hernandez

Tarea # 2, Segúndo cuatrimestre 2018

-Primeramente se obtuvo la funcion de transferencia resultante del sistema inicial.



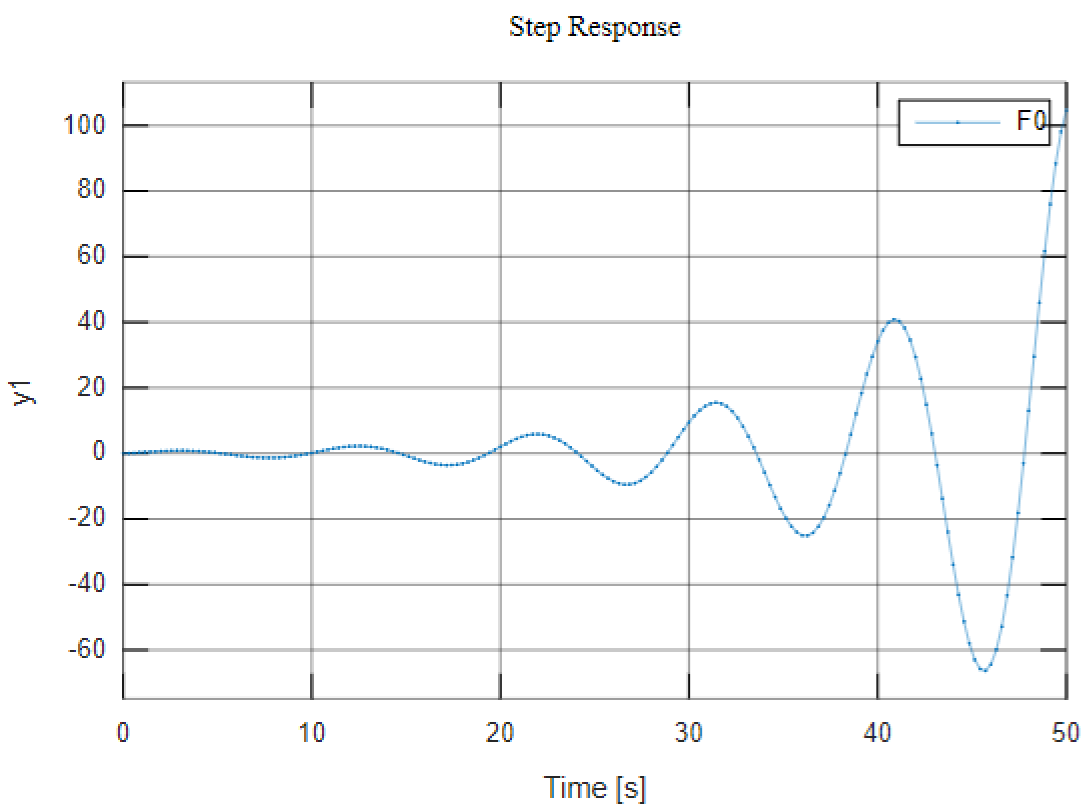
-La cual da como resultado la funcion F0



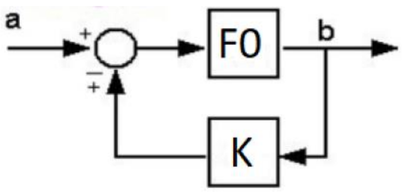
Se demuestra acontinuacion.

```
octave:1> numerador_qs=1
numerador_qs =  1
octave:2> denominador_ps=[1 2 0]
denominador_ps =

   1   2   0

octave:3> G0=tf(numerador_qs, denominador_ps)

Transfer function 'G0' from input 'u1' to output ...

             1
 y1:  ---------
      s^2 + 2 s

Continuous-time model.
octave:4> numerador_ms=1
numerador_ms =  1
octave:5> denominador_ns=[0 1 0]
denominador_ns =

   0   1   0

octave:6> H0=tf(numerador_ms, denominador_ns)

Transfer function 'H0' from input 'u1' to output ...

      1
 y1:  -
      s

Continuous-time model.
octave:7> F0=feedback(G0, H0)

Transfer function 'F0' from input 'u1' to output ...

             s
 y1:  ---------------
      s^3 + 2 s^2 + 1

Continuous-time model.
```

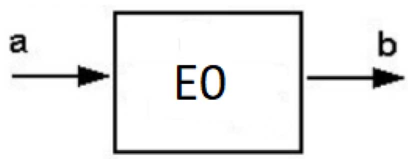
```
octave:1> F0=tf([0 1 0], [1 2 0 1])

Transfer function 'F0' from input 'u1' to output ...

                s
 y1:  ---------------
      s^3 + 2 s^2 + 1

Continuous-time model.
octave:2> [Z, P, K]=tf2zp(F0)
Z = 0
P =

  -2.20557 + 0.00000i
   0.10278 + 0.66546i
   0.10278 - 0.66546i

K =  1

octave:3> step(F0)
```



Por lo observado anteriormente podemos observar que la funcion F0 es inestable debido a la ubicación de sus polos, Por lo que se toma la dicision de retroalimentar F0 con una ganancia K, esto con el fin de lograr estabilizar el sistema lo que conlleva ubicar sus polos en valores menores a 0



Su resultado lo denominamos E0



$$E0 = \frac{s}{s^3 + 2s^2 + \mathrm{ks} + 1}$$

Se utilizo el criterio de Routh para encontrar el rango de K dentro del cual la funcion es estable.

```
octave:1> syms k
OctSymPy v2.6.0: this is free software without warranty, see source.
Initializing communication with SymPy using a popen2() pipe.
Some output from the Python subprocess (pid 16799) might appear next.
Python 2.7.5 (default, Aug  4 2017, 00:39:18)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-16)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> >>>
OctSymPy: Communication established.  SymPy v1.1.2.dev.
octave:2> criterioRouth([1 2 k 1])
ans = (sym 4×3 matrix)

  ⎡ 3              ⎤
  ⎢s      1       k⎥
  ⎢                ⎥
  ⎢ 2              ⎥
  ⎢s      2       1⎥
  ⎢                ⎥
  ⎢s   k - 1/2    0⎥
  ⎢                ⎥
  ⎣1      1       0⎦
```

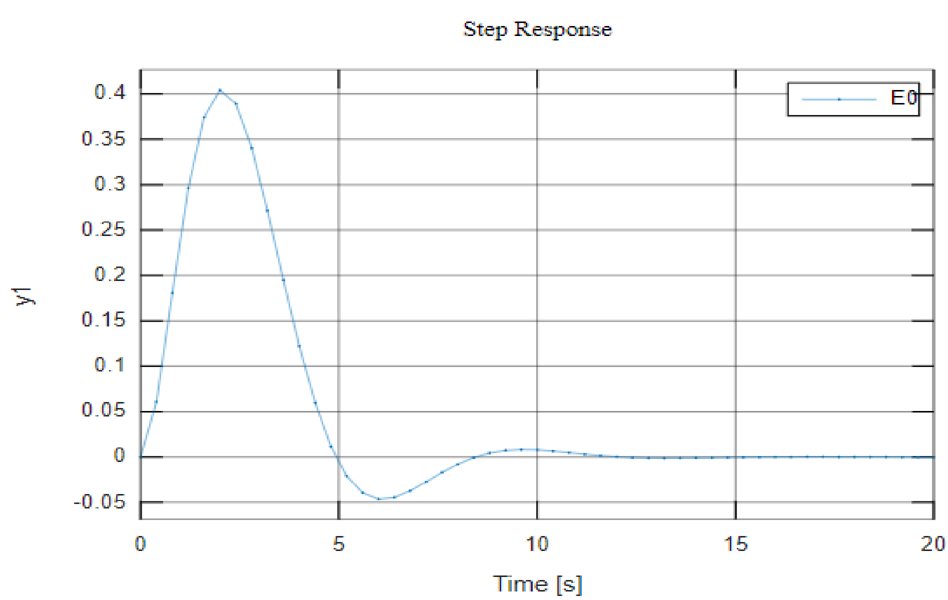
Por lo que la condición para que la función sea estable debe de ser:

$$k > \frac{1}{2}$$

Entonces comprobamos utilizado valores inferiores y superiores a ½, donde vemos que esta condición se cumple y con valores superiores a ½ la función es estable y con valores inferiores a ½ es inestable.

-Con K=2

```
octave:5> F0=tf([0 1 0], [1 2 0 1])

Transfer function 'F0' from input 'u1' to output ...

              s
 y1:  ---------------
      s^3 + 2 s^2 + 1

Continuous-time model.
octave:6> k=2
k =  2
octave:7> E0=feedback(F0, k)

Transfer function 'E0' from input 'u1' to output ...

                 s
 y1:  ---------------------
      s^3 + 2 s^2 + 2 s + 1

Continuous-time model.
octave:8> [Z, P, K]=tf2zp(E0)
Z = 0
P =

  -1.00000 + 0.00000i
  -0.50000 + 0.86603i
  -0.50000 - 0.86603i

K =  1

octave:9> step(E0)
```



-Con K=0.1

```
octave:10> k=0.1
k =  0.10000
octave:11> E0=feedback(F0, k)

Transfer function 'E0' from input 'u1' to output ...

                  s
 y1:  ----------------------
      s^3 + 2 s^2 + 0.1 s + 1

Continuous-time model.
octave:12> [Z, P, K]=tf2zp(E0)
Z = 0
P =

  -2.16683 + 0.00000i
   0.08342 + 0.67420i
   0.08342 - 0.67420i

K =  1
octave:13> step(E0)
```



```
octave:13> step(E0)
```