Diseño de control robusto ante incertidumbre pasiva mediante μ-síntesis en un circuito eléctrico

Presentación en vídeo:

http://personales.upv.es/asala/YT/V/paselml1.html ,

http://personales.upv.es/asala/YT/V/paselml2.html .

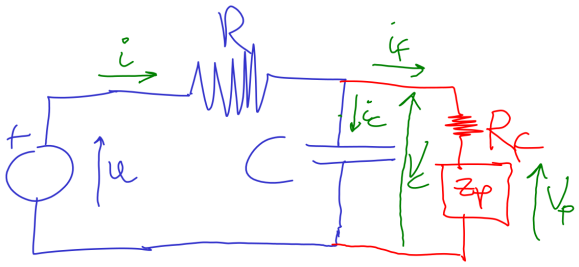
El presente código ejecutó sin errores en Matlab **R2020a**

Objetivos: Modelar un sistema con una impedancia incierta. Incorporar a un modelo de la Robust Control Toolbox el conocimiento específico de que dicha impedancia es pasiva. Diseñar un controlador que obtenga las máximas prestaciones posibles en bucle cerrado (ancho banda error) dada esa incertidumbre.

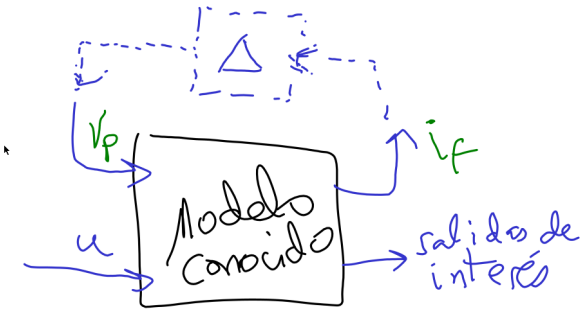
Tabla de Contenidos

Modelado físico

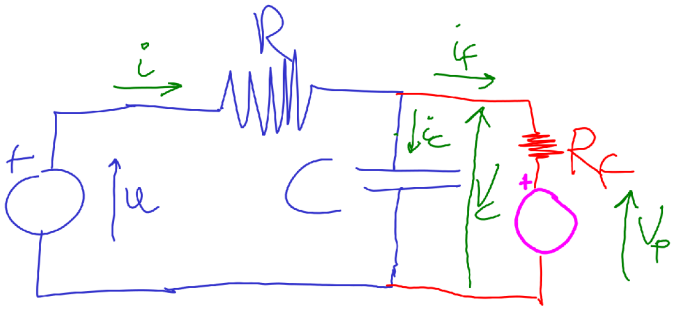
Consideremos el circuito RC siguiente (modelo "nominal" en azul) sujeto a una impedancia de fugas (rojo) que tiene un componente resistivo R_f y posiblemente otros componentes pasivos z_p no modelados ($z_p \equiv \Delta$) .



La incertidumbre debe ser modelada "fuera" de las ecuaciones conocidas:



Ello equivale a escribir las ecuaciones del "modelo conocido" como si V_p fuera una fuente de tensión ideal V_p arbitraria, a falta de añadir (en ecuaciones "fuera" de este modelo) la relación entre V_p e i_f (que no podemos dado que la impedancia es desconocida).




```
R=20;R_f=50;C=1e-3;
syms u i i_f V_p V_c i_f i_c dV_c real
Modelo_conocido=[u==R*i+V_c; dV_c==i_c/C; ...
    i==i_c+i_f; V_c==R_f*i_f+V_p]
```

```
Modelo_conocido =
```

$$\begin{pmatrix} u = V_c + 20\,i \\ \mathrm{dV}_c = 1000\,i_c \\ i = i_c + i_f \\ V_c = V_p + 50\,i_f \end{pmatrix}$$

*Obviamente, falta la ecuación $V_p = z_p \cdot i_f$, que, desaforturnadamente, no podemos incorporar porque z_p es desconocido.

De las cuatro ecuaciones, suponiendo conocidos estado V_c y entradas u y V_p, podemos despejar el resto de señales:

```
sol=solve(Modelo_conocido,i,i_f,dV_c,i_c);
sol.i
```

```
ans =
```

$$\frac{u}{20}-\frac{V_c}{20}$$

```
sol.i_f
```

```
ans =
```

$$\frac{V_c}{50}-\frac{V_p}{50}$$

```
sol.dV_c %esto será la ecuación de estado
```

```
ans =
```
$20\,V_p-70\,V_c+50\,u$

```
sol.i_c
```

```
ans =
```

$$\frac{V_p}{50}-\frac{7\,V_c}{100}+\frac{u}{20}$$

```
estados=V_c;
entradas_generalizadas=[V_p,u];
```

La ecuación de estado del "modelo conocido" será determinada por las matrices:

```
A=eval(jacobian(sol.dV_c,estados))
```

```
A = -70
```

```
B=eval(jacobian(sol.dV_c,entradas_generalizadas))
```

```
B = 1×2
    20    50
```

La ecuación de salida hacia la incertidumbre (i_f) será:

```
C_f=eval(jacobian(sol.i_f,estados))
```

```
C_f = 0.0200
```

```
D_f=eval(jacobian(sol.i_f,entradas_generalizadas))
```

```
D_f = 1×2
   -0.0200         0
```

Como la variable controlada de interés será la tensión del condensador, la extraeremos con:

```
Cperf=1;Dperf=[0 0];
```

Y formamos la ecuación de salida completa:

```
C=[C_f;Cperf];
D=[D_f;Dperf];
```

Con lo que construimos la parte "conocida" del modelo:

```
sys=ss(A,B,C,D);
sys.InputName={'V_p','u'};sys.OutputName={'i_f','V_c'};
```

Ejemplos sustituyendo valores de impedancia

Un modelo extremo donde $z_p = 0$ (cortocircuito) sería:

```
sysconF1=lft(0,sys);
zpk(sysconF1)
```

```
ans =

  From input "u" to output "V_c":
    50
  ------
  (s+70)

Continuous-time zero/pole/gain model.
```

Otro modelo extremo con $z_p \to \infty$ (circuito abierto, sin fugas, esto sería el modelo "nominal" posiblemente) es:

```
syssinFugas=lft(2e6,sys);
zpk(syssinFugas)
```

```
ans =

  From input "u" to output "V_c":
    50
  ------
  (s+50)

Continuous-time zero/pole/gain model.
```

Una carga inductiva en z_p se modelaría con:

```
L=8e-1;s=tf('s');
sysconL=lft(L*s+1e-2,sys);
zpk(sysconL)
```

```
ans =

  From input "u" to output "V_c":
      50 (s+62.51)
  ---------------------
  (s^2 + 112.5s + 4376)

Continuous-time zero/pole/gain model.
```

```
eig(sysconL)
```

```
ans = 2×1 complex
 -56.2563 +34.7974i
 -56.2563 -34.7974i
```

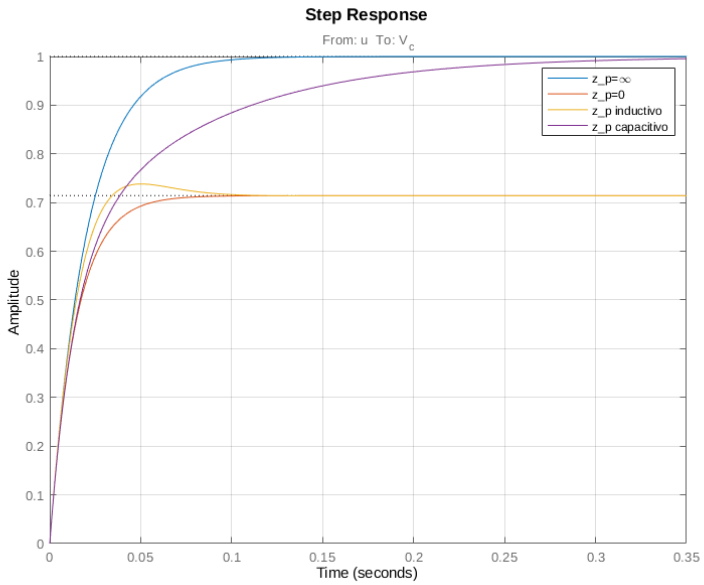
Una carga capacitiva pura en z_p se modelaría con:

```
C2=1e-3;
sysconC=lft(1/(C2*s),sys);
zpk(sysconC)
```

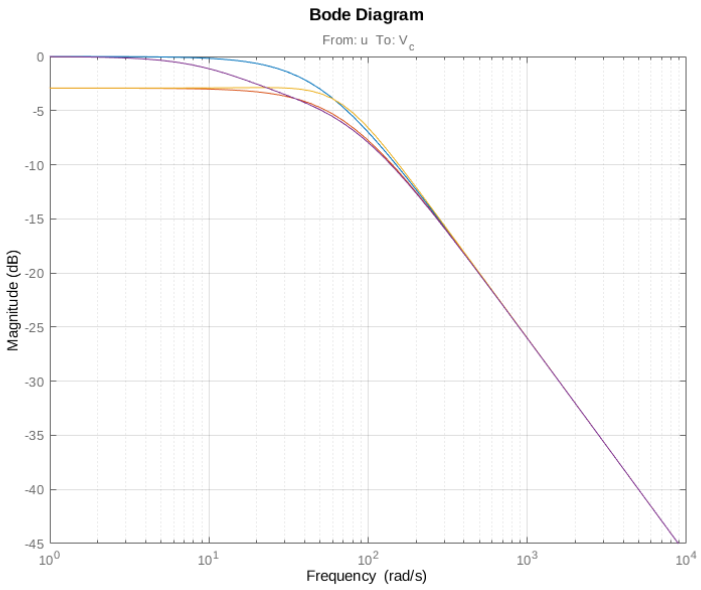
```
ans =

  From input "u" to output "V_c":
       50 (s+20)
  -------------------
  (s+12.98) (s+77.02)

Continuous-time zero/pole/gain model.
```

Sus respuestas son:

```
step(syssinFugas,sysconF1,sysconL, sysconC), grid on
legend('z_p=\infty','z_p=0','z_p inductivo','z_p capacitivo')
```




```
bodemag(syssinFugas,sysconF1,sysconL,sysconC), grid on
```



Modelado de incertidumbre con Robust Control Toolbox

```
delta=ultidyn('imped_fuga_z_p',[1 1],'Type','PositiveReal','Bound',0)
```

```
delta =
  Uncertain LTI dynamics "imped_fuga_z_p" with 1 outputs, 1 inputs, and positive real bound of 0.
```

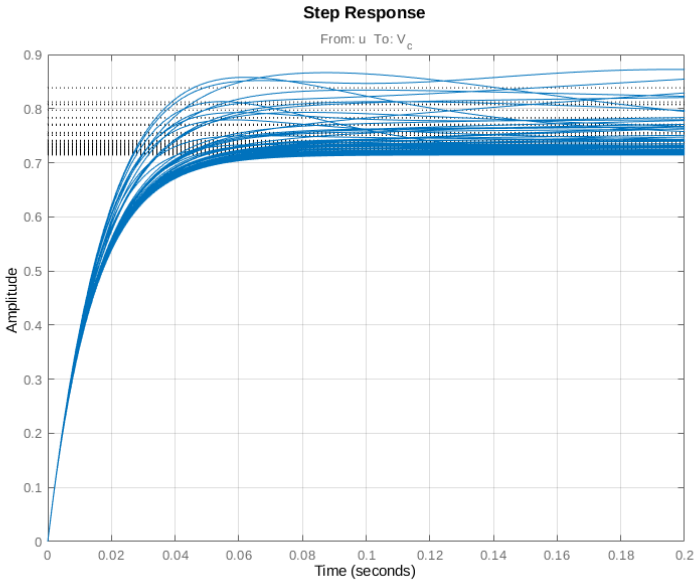
```
delta.NominalValue %el valor nominal es una resistencia de 1 Ohm
```

```
ans =

  D =
       u1
   y1   1

Name: imped_fuga_z_p
Static gain.
```

```
sysconFugas=lft(delta,sys)
```

```
sysconFugas =

  Uncertain continuous-time state-space model with 1 outputs, 1 inputs, 1 states.
  The model uncertainty consists of the following blocks:
    imped_fuga_z_p: Uncertain 1x1 LTI, positive real bound = 0, 1 occurrences

Type "sysconFugas.NominalValue" to see the nominal value, "get(sysconFugas)" to see all properties, and "s
```

```
Plantas=usample(sysconFugas,200);
step(Plantas,.2), grid on
```



Obviamente, el sistema es estable dado que la red RC ya era estrictamente pasiva, y la interconexión física de sistemas pasivos es pasiva.

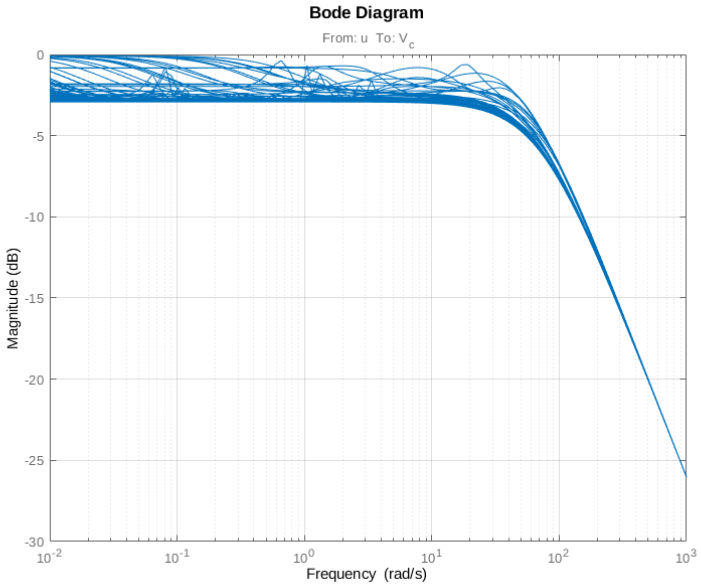
```
robstab(sysconFugas)
```

```
ans = struct with fields:
           LowerBound: 1.0269
           UpperBound: 1.0290
    CriticalFrequency: 0
```

```
wcgain(sysconFugas)
```

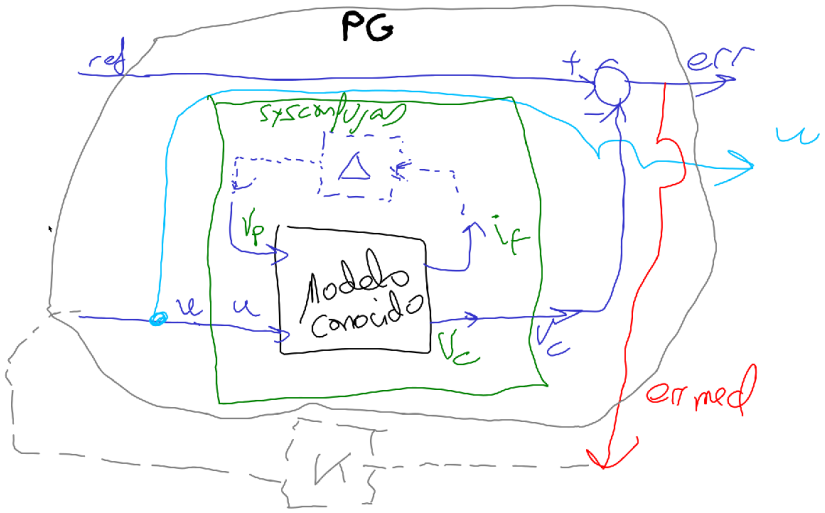
```
ans = struct with fields:
           LowerBound: 1.0000
           UpperBound: 1.0021
    CriticalFrequency: 0
```

```
bodemag(Plantas,logspace(-2,3,100)), grid on
```



Mu-síntesis con Planta generalizada ante incertidumbre pasiva

Planta Generalizada para control de 1 grado de libertad (≡ rechazo de perturbación a la salida):

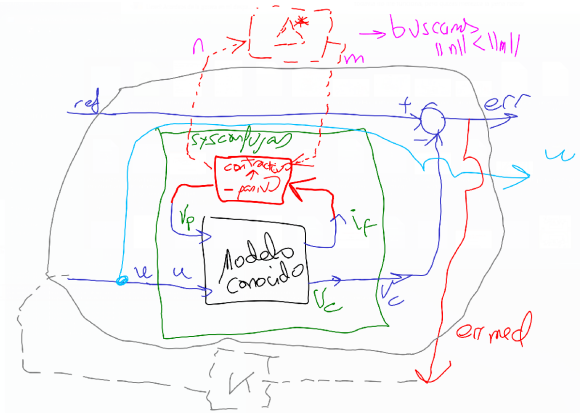


$$\begin{pmatrix} err \\ u \\ err_{medido} \end{pmatrix} = \begin{pmatrix} 1 & -sysconFugas \\ 0 & 1 \\ 1 & -sysconFugas \end{pmatrix} \begin{pmatrix} ref \\ u \end{pmatrix}$$

```
PG=[ [1;0;1]  [-1;0;-1]*sysconFugas+[0;1;0]];
PG.InputName={'ref','u'};
PG.OutputName={'error','u','errmed'};
PG
```

```
PG =

  Uncertain continuous-time state-space model with 3 outputs, 2 inputs, 1 states.
  The model uncertainty consists of the following blocks:
    imped_fuga_z_p: Uncertain 1x1 LTI, positive real bound = 0, 1 occurrences

Type "PG.NominalValue" to see the nominal value, "get(PG)" to see all properties, and "PG.Uncertainty" to
```

Internamente, plantea un problema de minimización de norma, usando la transformación a contractivo de forma transparente al usuario:



Construiremos una planta generalizada ponderada, requiriendo prestaciones en el error (2% error de posición, 2.5 pico resonancia máximo error, maximizar ancho de banda) y cota máxima de u = 2 * referencia.

```
bw=61.5; %subir o bajar hasta que musyn devuelva cota de 1.
Plantilla_err=makeweight(0.02,bw,2.5);
Werr=1/Plantilla_err;
CotaU=2;
Wu=1/CotaU;
PGPond=blkdiag(Werr,Wu,1)*PG;
[K,Clperf]=musyn(PGPond,1,1);Clperf
```

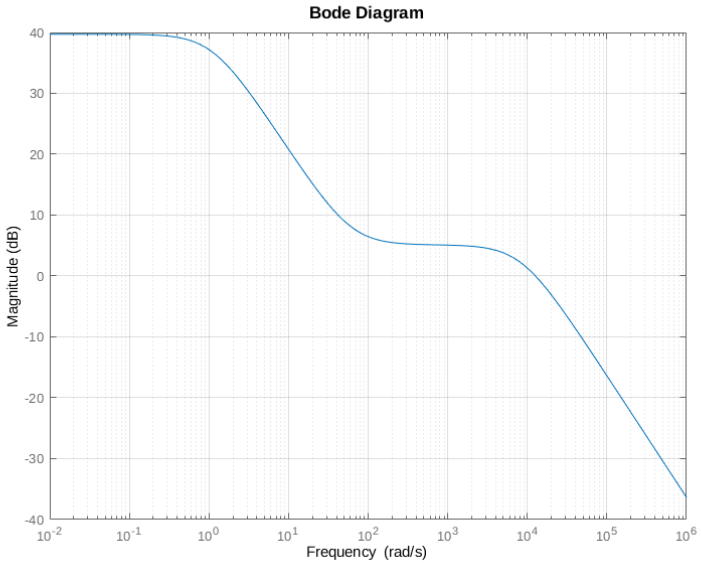
```
D-K ITERATION SUMMARY:
-----------------------------------------------------------------
                      Robust performance               Fit order
-----------------------------------------------------------------
  Iter          K Step        Peak MU        D Fit             D
    1           1.104         1.099          1.099             0
    2           1.006         1.005          1.005             0
    3          0.9973        0.9973          1.002             0
    4           1.001              1          1.001             0

Best achieved robust performance: 0.997
Clperf = 0.9973
```

```
zpk(K)
```

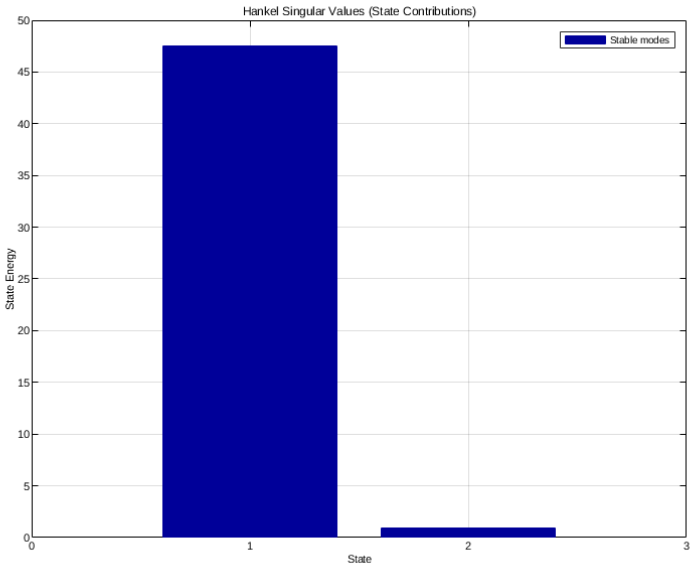
```
ans =

   15210 (s+60.52)
  ------------------
  (s+1.128) (s+8433)
```

```
Continuous-time zero/pole/gain model.
```

```
bodemag(K), grid on
```



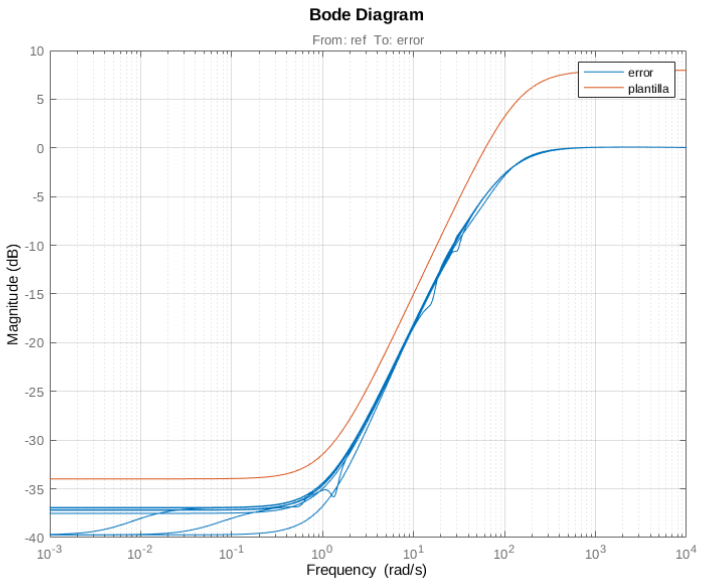

```
hsvd(K) %podría seguramente reducirse a un primer orden... es "casi" un PI.
```



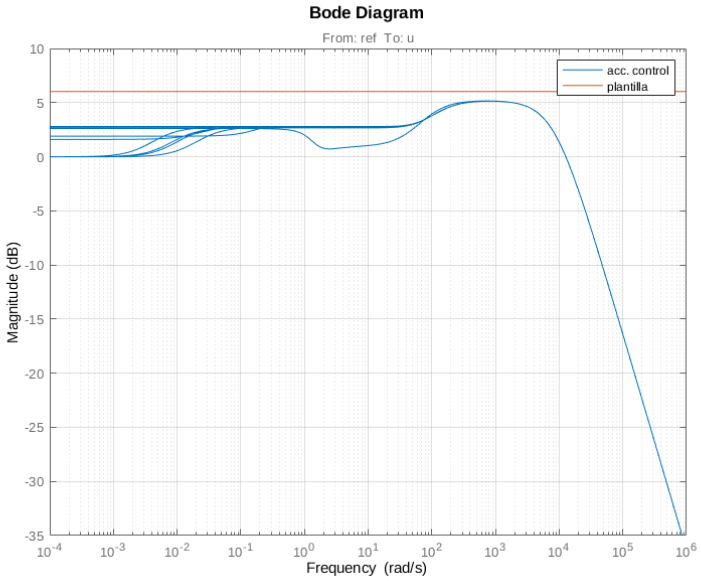

```
CL_noponderado=lft(PG,K);
CL_ponderado=lft(PGPond,K);
```

Evaluemos la respuesta en frecuencia de bucle cerrado:

```
bodemag(CL_noponderado(1,:),Plantilla_err), grid on, legend('error','plantilla')
```

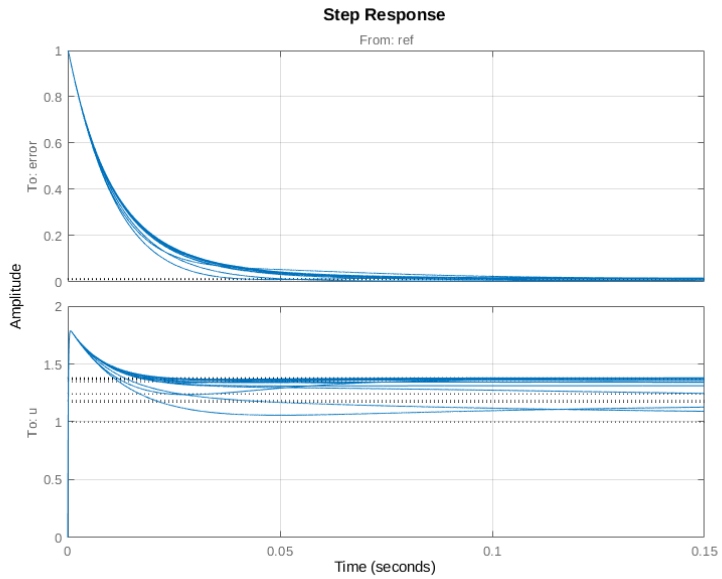



```
bodemag(CL_noponderado(2,:),tf(CotaU)), grid on, legend('acc. control','plantilla')
```



Y la respuesta temporal

```
step(CL_noponderado,.15), grid on
```




```
robstab(CL_noponderado)
```

```
ans = struct with fields:
            LowerBound: 1.0336
            UpperBound: 1.0357
    CriticalFrequency: 70.4711
```

```
wcgain(CL_ponderado)
```

```
ans = struct with fields:
            LowerBound: 0.9955
            UpperBound: 0.9971
    CriticalFrequency: 392.5727
```

```
robgain(CL_ponderado,1)
```

```
ans = struct with fields:
            LowerBound: 1.0027
            UpperBound: 1.0047
    CriticalFrequency: 45.2581
```

Conclusiones

Hemos modelado un sistema con incertidumbre pasiva. En la fase de modelado no importa cómo es la incertidumbre: se sustituye por una fuente de tensión ideal (entrada) y se despeja la intensidad que recorre dicha fuente como salida.

EL hecho de que sea "pasiva" se puede introducir en la Robust Control Toolbox. Internamente, la transforma a "contractiva" para sus operaciones y para μ-síntesis.