Fórmulas de análisis de estabilidad/prestaciones robustas ante incertidumbre estructurada: robstab, wcgain, robgain

Presentación en vídeo: http://personales.upv.es/asala/YT/V/margesc.html

Este código funcionó correctamente con Matlab **R2020a**

Objetivo: comprender el significado de las distintas cotas de pequeña ganancia (estandard, no escalado) y de las cotas que, usando peq. ganancia escalado, valor singular estructurado y otras, permite obtener el Robust Control Toolbox de Matlab (menos conservativas) con los comandos `robstab, wcgain, robgain`.

Tabla de Contenidos

1. Modelado y estimación de error de modelado

Consideremos el proceso:

```
s=tf('s');
G=minreal(ss([6.7/(s+1)^2  4/(0.8*s+1)^2;
              -13/(s+1)^2 -2.7/(0.5*s+1)^2]));
size(G)
```

```
State-space model with 2 outputs, 2 inputs, and 6 states.
```

Consideremos la posibilidad de que el proceso real esté en determinado intervalo de incertidumbre alrededor de él:

```
rng(4343)
rndinte=@(low,up) low+rand()*(up-low);
Ntests=60;
mxnugap=0;
for i=1:Ntests
   tau1=rndinte(0.9,1.1);
```

```
    Greal{i}=minreal(ss([rndinte(6.5,6.9)/(tau1*s+1)^2  rndinte(3.8,4.2)/(rndinte(0.75,0
          -rndinte(12,14)/(tau1*s+1)^2 -rndinte(2.5,2.9)/(rndinte(0.45,0.55)*s+1)^2] ...
          +randn(2)*.09*s/(s+10)),[],false);
       errormod{i}=G-Greal{i};
       [~,nug]=gapmetric(G,Greal{i});
       mxnugap=max(mxnugap,nug);
end
mxnugap
```

```
mxnugap = 0.2801
```

1.1 Uso de UCOVER para obtener una cota del error de modelado aditivo

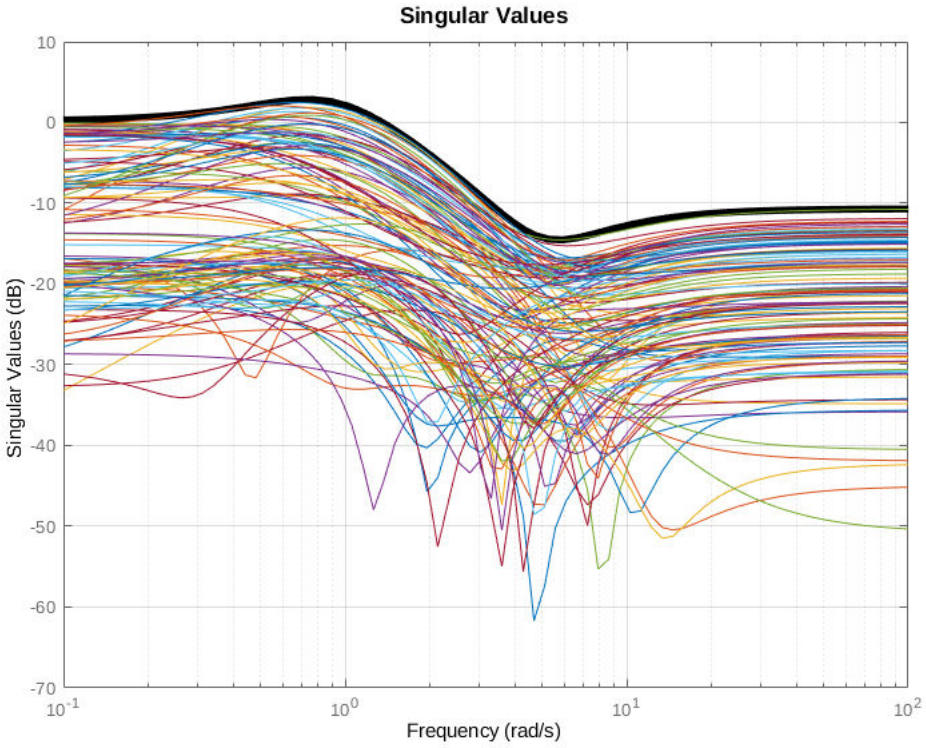
```
w=logspace(-1,2,80);
Parr=stack(1,Greal{:}); mPf=frd(Parr,w); %frequency-response data model
[Pr,Info_ucover]=ucover(mPf,G,3,'additive');%acotarla con un tamaño función de frec. de
Pr
```

```
Pr =

  Uncertain continuous-time state-space model with 2 outputs, 2 inputs, 12 states.
  The model uncertainty consists of the following blocks:
    mPf_AddDelta: Uncertain 2x2 LTI, peak gain = 1, 1 occurrences

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

```
figure
sigmaplot(Info_ucover.W1,'k'), grid on
lines = findobj(gcf,'Type','Line');
lines(2).LineWidth=4;
if(length(lines)>2) lines(3).LineWidth=4; end
hold on
sigma(errormod{:},w)
hold off
```




```
tamanyoerrormodeladoad=minreal(Info_ucover.W1);zpk(tamanyoerrormodeladoad)
```

```
3 states removed.

ans =

  0.29257 (s+0.6053) (s^2 + 4.744s + 25.09)
  -----------------------------------------
      (s+5.237) (s^2 + 1.215s + 0.8297)

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

Nota: la acotación del error con "ucover" no sería estrictamente necesaria en teoría. Podría generase una planta incierta con estructura de incertidumbre con parámetros tipo "ureal" o subsitemas tipo "ultidyn" (aunque si hay muchos de ellos, podría haber problemas numéricos, y se hace no convexo el diseño de reguladores).

En este caso, se ha mantenido la acotación de "ucover" para comprobar que los resultados coinciden con los de pequeña ganancia "clásico", no escalado, ante incertidumbre aditiva (sólo los de estabilidad robusta).

2. Diseño de alternativas de control

2.1 Control PID Multibucle (pidtune)

```
dcg=dcgain(G);
rga=dcg.*inv(dcg')
```

```
rga = 2×2
   -0.5335    1.5335
```

```
    1.5335   -0.5335
```

```
Pid1=pidtune(G(1,2),'PIDF');
Pid2=pidtune(G(2,1),'PIDF');
Kpid=[0 Pid2;Pid1 0];
```

Comprobamos estabilidad robusta (peq. gan **no** escalado).

Esto debe ser menos de 1:

```
pruebapeqganPID=norm(minreal(tamanyoerrormodeladoad*feedback(Kpid,G)),"inf")
```

```
pruebapeqganPID = 5.9625
```

Pero es casi 6! El tamaño del error de modelado debería escalarse por un factor:

```
margenpeqganPID=1/norm(minreal(tamanyoerrormodeladoad*feedback(Kpid,G)),"inf")
```

```
margenpeqganPID = 0.1677
```

La incertidumbre debería ser del 16% de la que es para que se cumpliera la condición de pequeña ganancia.

En cuanto a incertidumbre en factores coprimos:

```
ncfmargin(G,Kpid)
```

```
ans = 0.0490
```

Este diseño NO asegura estabilidad robusta en las bolas aditiva/ncf de incertidumbre.

2.2 Diseño Glover-McFarlane Loop Shaping (ncfsyn)

Seleccionemos primero los pesos de la planta ponderada:

```
W1=eye(2)*1/(0.2*s+1); %high freq cut, input weight
W2=diag([2.85 2.85])*(20*s+15)/(20*s+1); %low freq boost, output weight
Gshaped=minreal(ss(W2*G*W1));
```

Evaluemos una cota de incertidumbre entre las plantas reales/nominal ponderadas:

```
for i=1:Ntests
      [~,nugapmW{i}]=gapmetric(Gshaped,W2*Greal{i}*W1); %error ncf ponderado
end
cotanugapW=max([nugapmW{:}])
```

```
cotanugapW = 0.3071
```

```
[Kgmf,~,GAM,Info]=ncfsyn(G,W1,W2);
```

Comprobemos estabilidad robusta (peq. ganancia NO escalado):

Con plantas ponderadas incerditumbre ncf: **SÍ**

```
1/GAM
```

```
ans = 0.3071
```

Con la bola de incertidumbre aditiva "tamanyoerrormodeladoad": **NO**

```
margenpeqgan_gmf=1/norm(minreal(tamanyoerrormodeladoad*feedback(-Kgmf,G)),"inf")
```

```
1 state removed.
margenpeqgan_gmf = 0.5338
```

2.3. Diseño MIXED SENSITIVITY

```
bw=.858; %esto con fit orden 3 resulta GAM aprox. 1
plantillaerror=makeweight(0.02,bw,2.5);
Werr_mxs=1/plantillaerror*eye(2);
tamanyoerrorrelativo=0.05; %que tolere un 5% de error sea  cual sea el filtro W1
[Kmxs,~,GAM,~]=mixsyn(G,Werr_mxs,tamanyoerrormodeladoad,tamanyoerrorrelativo);
GAM
```

```
GAM = 0.9995
```

```
margenpeqgan_mxs=1/norm(minreal(tamanyoerrormodeladoad*feedback(Kmxs,G)),"inf")
```

```
12 states removed.
margenpeqgan_mxs = 1.1268
```

3. Comparación de respuesta temporal (nominal/robusta)

El objeto con el que trabajaremos será el resultado de Ucover.

Construimos planta generalizada, sólamente deseamos prestaciones de error (limitación de u en mixsyn era por robustez... podríamos poner si queremos, por temas saturación o límite explícito de ancho de banda).

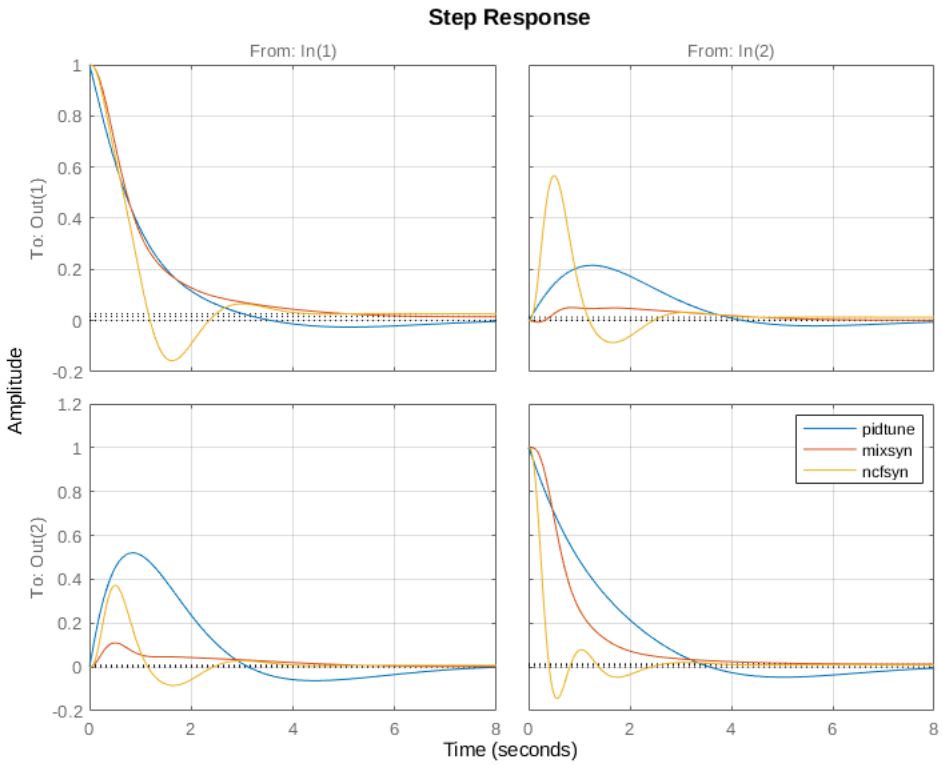
```
PGsinU=[[eye(2);eye(2)]  -[eye(2);eye(2)]*Pr];
CLpid=minreal(lft(PGsinU,Kpid));
CLmixsyn=minreal(lft(PGsinU,Kmxs));
```

```
6 states removed.
```

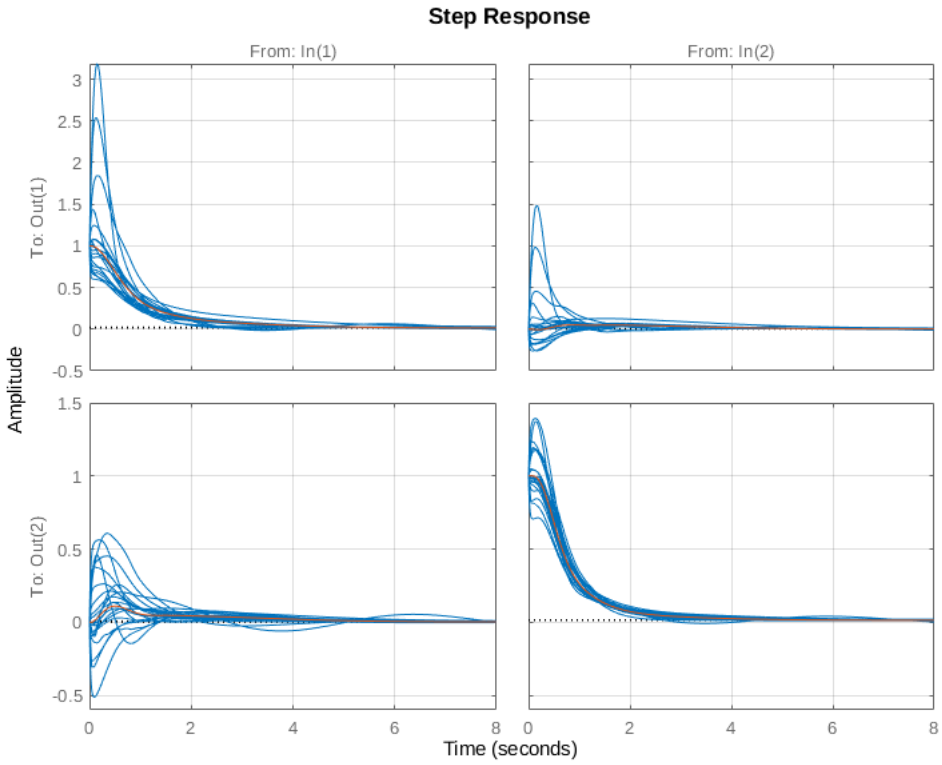
```
CLgmf=minreal(lft(PGsinU,-Kgmf));
```

```
1 state removed.
```

```
step(CLpid.NominalValue,CLmixsyn.NominalValue,CLgmf.NominalValue,8), grid on
legend('pidtune','mixsyn','ncfsyn')
```




```
step(CLmixsyn,CLmixsyn.NominalValue,8), grid on
```



*La respuesta temporal de PIDs y NCFsyn ante "Pr" (incert. aditiv) se hace inestable... no ante la bola de plantas el NCFsyn, porque tenemos el margen adecuado, pero sí ante el resultado de "ucover".

4.- Análisis de estabilidad/prestaciones robustas con tma. peq. ganancia escalado (y valor singular "estructurado")

Como ninguna de las técnicas asegura "prestaciones robustas" en teoría, vamos a intentar ver si se cumplen con unas prestaciones "relajadas": límites de error más grandes y más lentos.

```
%Werr era 1/makeweight(.02,0.858,2.5) el mixsyn
Werr_relax=1/makeweight(.025,0.55,4);
```

Objetivo de prestaciones: con el error por debajo de la inversa Werr_relax realmente me conformaría si fuera "prestaciones robustas" y no sólo "nominales"...

Ponderamos los bucles cerrados en los que la salida ya era, directamente, el error:

```
CLpid_pond=Werr_relax*CLpid;
CLmixsyn_pond=Werr_relax*CLmixsyn;
CLgmf_pond=Werr_relax*CLgmf;
```

Entonces, nuestro objetivo es que la norma infinito de esos **`CL____pond`** sea menor de 1, no sólo nominalmente, sino con toda la bola de incertidumbre aditiva del modelo "**`Pr`**".

Diseño PID

Prestaciones nominales: Sí!

```
norm(CLpid_pond.NominalValue,"inf")
```

```
ans = 0.9809
```

Estabilidad robusta: NO, tolera sólo un 16% de la incert. aditiva

```
robstab(CLpid_pond)
```

```
ans = struct with fields:
           LowerBound: 0.1674
           UpperBound: 0.1677
    CriticalFrequency: Inf
```

```
margenpeqganPID
```

```
margenpeqganPID = 0.1677
```

coincide con el estimado por "pequeña ganancia" no escalado, porque la incert. en Pr es "aditiva" no estructurada.

Prestaciones robustas para la incertidumbre "tamaño completo":

```
wcgain(CLpid_pond) %no estabilidad robusta -> ganancia infinita peor caso
```

```
ans = struct with fields:
           LowerBound: Inf
           UpperBound: Inf
    CriticalFrequency: Inf
```

Prestaciones robustas para incertidumbre más pequeña?:

```
robgain(CLpid_pond,1)
```

```
ans = struct with fields:
           LowerBound: 0.0269
           UpperBound: 0.0269
    CriticalFrequency: 0.5960
```

Sólo una bola del 2.7% de la incert. original garantizaría estar error por debajo de W_relax.

Diseño MIXSYN

Prestaciones nominales: SÍ. (las tenía con el peso original más restrictivo)

```
norm(CLmixsyn_pond.NominalValue,"inf")
```

```
ans = 0.7124
```

Estabilidad robusta: Sí, tolera un 12% de incert. aditiva "extra"

```
robstab(CLmixsyn_pond)
```

```
ans = struct with fields:
           LowerBound: 1.1264
           UpperBound: 1.1287
    CriticalFrequency: 10.4493
```

```
margenpeqgan_mxs %coindice con "pequeña ganancia" no escalado, claro.
```

```
margenpeqgan_mxs = 1.1268
```

Prestaciones robustas para la incertidumbre "tamaño completo":

NO, incluso con las especificaciones "relajadas" a alguna frecuencia puede superarlas en un factor de 2.4

```
wcgain(CLmixsyn_pond)
```

```
ans = struct with fields:
           LowerBound: 2.3942
           UpperBound: 2.3994
    CriticalFrequency: 8.0025
```

Prestaciones robustas para incertidumbre más pequeña?:

```
robgain(CLmixsyn_pond,1)
```

```
ans = struct with fields:
```

```
            LowerBound: 0.6361
            UpperBound: 0.6374
    CriticalFrequency: 0
```

Sí, si la incertidumbre fuera de un 63% de la que es, las prestaciones "relajadas" se cumplirían.

Diseño NCFSYN

Prestaciones nominales: NO, el objetivo era otro.

```
norm(CLgmf_pond.NominalValue,"inf")
```

```
ans = 1.2633
```

Estabilidad robusta: NO, tolera sólo un 53% de incert. aditiva original (el "cover" del nugap es diferente)

```
robstab(CLgmf_pond)
```

```
ans = struct with fields:
            LowerBound: 0.5328
            UpperBound: 0.5339
    CriticalFrequency: 2.4380
```

```
margenpeqgan_gmf %coincide con "pequeña ganancia" no escalado.
```

```
margenpeqgan_gmf = 0.5338
```

Prestaciones robustas para la incertidumbre "tamaño completo":

NO, al no tener estabilidad robusta

```
wcgain(CLgmf_pond)
```

```
ans = struct with fields:
            LowerBound: Inf
            UpperBound: Inf
    CriticalFrequency: 1
```

Prestaciones robustas para incertidumbre más pequeña?:

No, al no tener ni siguiera prestaciones nominales.

```
robgain(CLgmf_pond,1)
```

```
ans = struct with fields:
            LowerBound: 0
            UpperBound: 0
    CriticalFrequency: NaN
```

Conclusiones

Para analizar las prestaciones de un bucle ante una incertidumbre estructurada, aparte de una posible "simulación exhaustiva", tenemos las herramientas **robstab**, **wcgain** y **robgain** que proveen de márgenes numéricos de ganancias o tamaños de incertidumbre para validar la robustez un diseño.