# (Solution) Constructing the smallest destabilizing perturbation (at Plant input)

A nominal plant and controller are given. There are several tasks to complete, culminating in construction of the smallest input-multiplicative plant perturbation that leads to instability.

UC Berkeley, ME C231B/EECS C220C, Spring 2017

## Contents



## Plant, controller specification

2-input, 2-output plant and controller are specified in state-space form.

```
Ap = [ -0.2  10; -10  -0.2];
Bp = eye(2);
Cp = [1 8;-10 1];
P = ss(Ap,Bp,Cp,0);

Ac = blkdiag(zeros(2,2),-62.83,[-15.68 11;-11 -15.68],-14.33);
Bc = [0.5343   0.2178;...
     -0.2174   0.5255;...
      3.019   -3.903;...
      0.1649    2.265;...
     -1.308    -2.36;...
     -2.334  0.06362];
Cc = [ -4   0  -1.369  -0.02616  1.771  -3.681;...
         0 -4  -1.467  -3.616   -1.955  -0.6843];
C = ss(Ac,sqrt(3)*Bc,sqrt(3)*Cc,0);
```

## Verify closed-loop system is stable

Form all closed-loop maps with LOOPSENS

```
help loopsens
```

```
--- help for DynamicSystem/loopsens ---

 LOOPSENS  Sensitivity functions of plant-controller feedback loop.

    SF = LOOPSENS(P,C) analyzes the multivariable negative feedback loop
    with plant P and controller C:

                      u
             du --->O----->[ P ]-----+---> yP
                  - |                |
                    |             y  |
             uC <---+<-----[ C ]-----O<--- dy

    For "2-dof" architectures, C should only include the portion of the
    controller in the feedback path.

    SF is a structure containing the following sensitivity functions:
        Si   Sensitivity at the plant input
        Ti   Complementary sensitivity at the plant input (I-Si)
        Li   Open-loop transfer at the plant inputs (CP)
        So   Sensitivity at the plant output
        To   Complementary sensitivity at the plant output (I-So)
        Lo   Open-loop transfer at the plant outputs (PC)
       PSi   Closed-loop transfer from plant input to plant output
       CSo   Closed-loop transfer from controller input to controller
             output.

    See also LOOPMARGIN, ROBSTAB, ROBGAIN, WCSENS, WCMARGIN.
```

```
H = loopsens(P,C)
```

```
H =

  struct with fields:

        Si: [2×2 ss]
        Ti: [2×2 ss]
        Li: [2×2 ss]
        So: [2×2 ss]
        To: [2×2 ss]
        Lo: [2×2 ss]
```

```
        PSi: [2×2 ss]
        CSo: [2×2 ss]
      Poles: [8×1 double]
     Stable: 1
```

## Compute Hinf norm of Ti

Ti is complementary sensitivity at plant input

```
[TiNorm,freq] = norm(H.Ti,inf,0.001)
```

```
TiNorm =

    1.5604


freq =

   17.1975
```

## Compute Frequency Response Matrix of Ti at frequency where peak occurs

```
M = freqresp(H.Ti,freq)
```

```
M =

   0.1189 - 0.7672i  -0.5812 - 0.4673i
   0.8871 + 0.1348i  -0.1957 - 0.7533i
```

Find the smallest plant input-multiplicative uncertainty (a constant, complex matrix of dimension 2-by-2) which causes instability

```
[U,S,V] = svd(M);
Delta = -1/S(1,1)*V(:,1)*U(:,1)';
[norm(Delta)  1/TiNorm]
```

```
ans =

    0.6409    0.6409
```

## Perturbed closed-loop system

Form perturbed plant, using input-multiplicative form and the perturbation matrix.

```
Ptilde = P*(eye(size(P,2))+Delta);
```

## Verify Instability of perturbed closed-loop system

Compute perturbed closed-loop system using `loopsens`, and check poles

```
pole(feedback(Ptilde,C))
```

```
ans =

 -65.2650 + 1.0946i
 -25.5940 + 1.7140i
  -3.7787 -17.6834i
  -0.0000 +17.1975i
  -8.2364 - 9.4348i
  -1.2554 - 5.2177i
  -4.0102 + 8.0959i
  -0.7802 + 4.2339i
```

## Connection between unstable pole, and frequency where peak of Ti occurs

Show connection between the two frequencies

```
freq
```

```
freq =

   17.1975
```

## (Extra Credit!) Use `cnum2sys` to create real-dynamic uncertainty

Rather than accepting a complex matrix as a destabilizing element, use the command `cnum2sys` to convert the complex matrix uncertainty into a real-valued, linear, dynamic system that leads to instability. The Hinf norm of the dynamic system should match the maximum singular value of the original destabilizing complex matrix.

```
v1hat = cnum2sys(V(1,1),freq);
v2hat = cnum2sys(V(2,1),freq);
ubar1hat = cnum2sys(conj(U(1,1)),freq);
```

```
ubar2hat = cnum2sys(conj(U(2,1)),freq);
DeltaHat = -1/S(1,1)*[v1hat;v2hat]*[ubar1hat ubar2hat];
pole(DeltaHat)
[norm(DeltaHat,inf) norm(Delta)]
pole(feedback(P*(eye(2)+DeltaHat),C))
```

```
ans =

  -16.9840
  -11.5246
   -0.2719


ans =

    0.6409    0.6409


ans =

 -63.0739 + 0.0000i
 -22.3928 + 0.0000i
 -21.5166 + 4.7974i
 -21.5166 - 4.7974i
  -0.0000 +17.1975i
  -0.0000 -17.1975i
  -3.9585 + 7.9124i
  -3.9585 - 7.9124i
  -0.5597 + 5.0453i
  -0.5597 - 5.0453i
  -0.1642 + 0.0000i
```

## Frequency-dependent uncertainty

Adopt a frequency-dependent uncertainty model, using a scalar weighting function, `wu`. Take uncertainty model to be `P* (eye(2) + wu(s) Delta)`, using a first order `wu` with:

- DC gain of 0.35 (35% uncertainty at low frequency)
- gain of 1 at 40 rads/TimeUnit (100% uncertainty at 40)
- High-Frequency gain of 10 (1000% uncertainty at high frequency)

```
wu = makeweight(0.35,40,10);
```

Mimic/repeat procedure to find smallest destabilizing `Delta`, naming the perturbation `wDelta` in this case. Both **constant, complex** and **real, linear dynamic** perturbations can be constructed, following the same procedure, incorporating the weighting function `wu`. Start by computing the norm of the weighted complementary sensitivity function.

```
[wTiNorm,wfreq] = norm(wu*H.Ti,inf,0.001);
M = freqresp(wu*H.Ti,wfreq);
[U,S,V] = svd(M);
```

Constant, complex perturbation is constructed with the same steps, based on SVD

```
wDelta = -1/S(1,1)*V(:,1)*U(:,1)';
[norm(wDelta)  1/wTiNorm]
```

```
ans =

    1.1831    1.1831
```

Real, linear, dynamic perturbation is constructed in the same manner, using `cnum2sys` on the elements from the SVD construction.

```
v1hat = cnum2sys(V(1,1),wfreq);
v2hat = cnum2sys(V(2,1),wfreq);
ubar1hat = cnum2sys(conj(U(1,1)),wfreq);
ubar2hat = cnum2sys(conj(U(2,1)),wfreq);
wDeltaHat = -1/S(1,1)*[v1hat;v2hat]*[ubar1hat ubar2hat];
pole(wDeltaHat)
```

```
ans =

  -17.2301
   -4.9214
  -57.3897
```

The constructed perturbations are of the same magnitude.

```
[norm(wDeltaHat,inf) norm(wDelta)]
```

```
ans =

    1.1831    1.1831
```

Both result in closed-loop poles on the imaginary axis at the frequency associated with the peak value of the `wu*Ti`.

```
pole(feedback(P*(eye(2)+wu*wDelta),C))
```

```
ans =

   1.0e+02 *

  -4.2424 + 0.0031i
  -0.6984 - 0.0069i
  -0.0000 + 0.1785i
  -0.2422 - 0.0079i
  -0.0373 - 0.1810i
  -0.0402 + 0.0807i
  -0.0133 + 0.0435i
  -0.0127 - 0.0521i
  -0.0514 - 0.0580i
  -4.2487 + 0.0000i
```

```
pole(feedback(P*(eye(2)+wu*wDeltaHat),C))
```

```
ans =

   1.0e+02 *

  -4.2504 + 0.0000i
  -0.6370 + 0.1949i
  -0.6370 - 0.1949i
  -0.2808 + 0.0000i
  -0.1614 + 0.0000i
   0.0000 + 0.1785i
   0.0000 - 0.1785i
  -0.0679 + 0.0000i
  -0.0421 + 0.0799i
  -0.0421 - 0.0799i
  -0.0073 + 0.0489i
  -0.0073 - 0.0489i
  -4.2487 + 0.0000i
```

## Conclusions

Using the `svd` command, the smallest destabilizing uncertainty for an input-multiplicative uncertainty model is constructed. It is initially constructed as a complex matrix, but can also be constructed as a real, linear, dynamic system, using `cnum2sys`. If the input-multiplicative uncertainty model is frequency-dependent, the same process can be used, simply incorporating the weighting function

## Attribution

## File Information

```
disp(mfilename)
```

```
DestabilizingPerturbationSolution
```

## Lessons learned

loopsens provides a complete analysis result of a system's sensitivity properties comprised by a plant and a negative feedback controller. The output of the loopsens could be extracted to do other analysis and proof of some theories such as the small gain theory. The smallest plant input uncertainty causing the system to be unstable can be calculated by using the singular value decomposition, where we can obtain the delta, and det(I-M*delta) = 0, and the system is on the fringe of being unstable. Notice the M = TiNorm The instability of the system can also be visualized by probing the poles of the uncertainty perturbed system (where the perturbation can be a dynamic system as well as a complex matrix mimicing the dynamic system).

*Published with MATLAB® R2017b*