MATX を用いた H_∞ 制御のシミュレーション

三平 満司,　古賀雅伸,　細川 敦邦

東京工業大学　情報環境学専攻

1　はじめに

本文では図 1のようなモデルの制御を行なう。ただし、設計は図 2に示すような低次元化されたモデルについて行なう。具体的には図 3のように実際の制御対象をノミナルプラントと加法的モデル化誤差 Δ の和とし、パフォーマンスとロバスト性の混合感度問題を解く。今回はまず与えられた値に対する設計およびシミュレーションにより全体の流れを掴み (Case1)、その後で実プラントのパラメータに変更を加えたものに対して任意の設計 (Case2) を行なう。

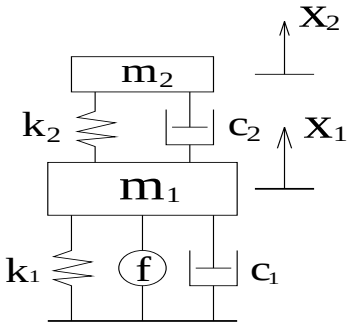


図 1: 制御対象

2　プラント

ノミナルモデル図 2の運動方程式は次式。

$$m_1\ddot{x}_1 + c_1\dot{x}_1 + k_1 x_1 = f$$

ただし x_1 は m_1 のつりあいの位置からの変位である。ノミナルプラントの状態方程式は状態を ${x_n}^T = [x_1\ \dot{x}_1]^T$、出力を x_1 とすると、以下のようになる。ただし、今回 Case1 で用いたパラメータの値は右の表の通りである。

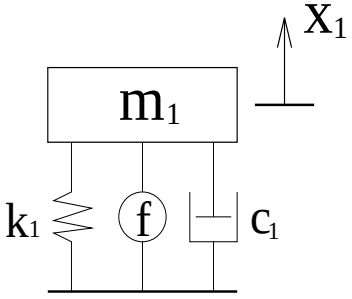


図 2: ノミナルモデル

$$\begin{aligned} \dot{x}_n &= A_n x_n + B_n u \\ y_p &= C_n x_n \end{aligned}$$

$$A_n = \begin{bmatrix} 0 & 1 \\ -\frac{k_1}{m_1} & -\frac{c_1}{m_1} \end{bmatrix} \tag{1}$$

$$B_n = \begin{bmatrix} 0 \\ \frac{1}{m_1} \end{bmatrix} \tag{2}$$

$$C_n = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{3}$$

このシステムを P_n とする。なお、実プラントの状態空間表現は次式である。(設計には使わないがシミュレーションに用いる。)

$$\begin{aligned} \dot{x}_p &= A_p x_p + B_p u \\ y_p &= C_p x_p \end{aligned}$$

	ノミナル	実プラント
m_1	10.0	0.1
k_1	1.0	1.0
c_1	3.0	3.0
m_2	—	10.0
k_2	—	100.0
c_2	—	0.01

ただし、$x_p^T = [x_1\ \dot{x}_1\ x_2\ \dot{x}_2]^T$ で、各行列はつぎの通り。

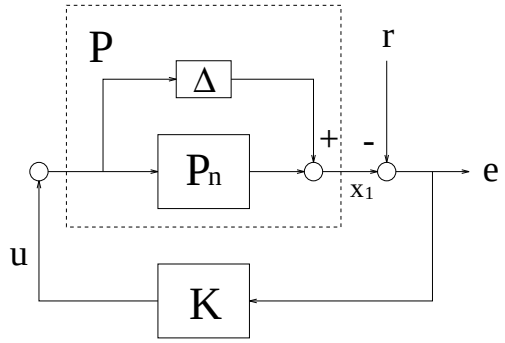


図 3: 閉ループ系

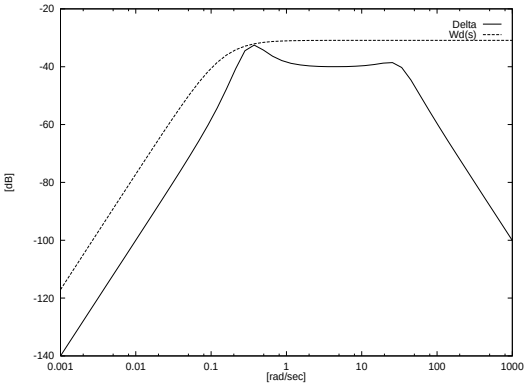


図 4: 誤差 Δ の周波数応答

$$A_p = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{k_1+k_2}{m_1} & -\frac{c_1+c_2}{m_1} & \frac{k_2}{m_1} & \frac{c_2}{m_1} \\ 0 & 0 & 0 & 1 \\ \frac{k_2}{m_2} & \frac{c_2}{m_2} & -\frac{k_2}{m_2} & -\frac{c_2}{m_2} \end{bmatrix} \tag{4}$$

$$B_p = \begin{bmatrix} 0 \\ \frac{1}{m_1} \\ 0 \\ 0 \end{bmatrix} \tag{5}$$

$$C_p = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \tag{6}$$

このシステムを P とする。

Case1 においてはこれらの値ははそれぞれ以下のようになる。

$$A_n = \begin{bmatrix} 0 & 1 \\ -0.1 & -0.3 \end{bmatrix} \tag{7}$$

$$B_n = \begin{bmatrix} 0 \\ 0.1 \end{bmatrix} \tag{8}$$

$$C_n = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{9}$$

$$A_p = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1010 & -30.1 & 1000 & 0.1 \\ 0 & 0 & 0 & 1 \\ 10 & 0.001 & -10 & -0.001 \end{bmatrix} \tag{10}$$

$$B_p = \begin{bmatrix} 0 \\ 10 \\ 0 \\ 0 \end{bmatrix} \tag{11}$$

$$C_p = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \tag{12}$$

3 設計 (Case1)

制御目的は x_1 の任意の位置 r への追従とする。その場合目標値 r から r と変位 x_1 との差までの伝達関数を G_{er} とすると、そのノルムを小さくする設計を行なえばよい。 Case1 では偏差 $e = x_1 - r$ にかける重み $W_s(s)$ は以下のようにした。

$$W_s(s) = \frac{1}{s + 0.01} \tag{13}$$

また、加法的誤差 $\Delta = P - P_n$ のボード線図は図 4のようになるので Δ をおおう関数重みとして $W_d(s)$ を次式のように設定した。

$$W_d(s) = \frac{1.4s^2}{(1 + 7s)^2} \tag{14}$$

これらを考慮するとコントローラの設計は閉ループ系を内部安定にし、かつ以下の式を満足させる問題に帰着される。

$$\left\| \begin{array}{c} G_{er}(s)W_s(s) \\ G_{ur}(s)W_d(s) \end{array} \right\|_\infty < 1$$

すなわち

$$\begin{aligned} \|G_{er}(s)W_s(s)\|_\infty &< 1 \\ \|G_{ur}(s)W_d(s)\|_\infty &< 1 \end{aligned}$$

ここで $G_{ur}(s)$ は目標値 r から制御入力 u までの伝達関数で、この場合 Δ の出力から Δ の入力までの伝達関数に一致する。

これらの重み関数の状態空間表現は以下のように設定する。

- $W_s(s)$

$$\begin{aligned} \dot{x}_s &= A_s x_s + B_s e \\ z_1 &= C_s x_s + D_s e \end{aligned} \tag{15}$$

- $W_d(s)$

$$\begin{aligned} \dot{x}_d &= A_d x_d + B_d u \\ z_2 &= C_d x_d + D_d u \end{aligned} \tag{16}$$

一般に伝達関数 $G(s) = \dfrac{b_{n-1}s^{n-1} + \cdots + b_0}{s^n + a_{n-1}s^{n-1} + \cdots + a_0} + d$ の状態空間表現は以下のように書くことができる。

$$\frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 1 \\ -a_0 & -a_1 & \cdots\cdots & & -a_{n-1} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} u$$

$$y = \begin{bmatrix} b_0 & b_1 & \cdots\cdots & b_{n-1} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} + du$$

例題 1. $\dfrac{K}{Ts+1}$ の実現

$$\frac{K}{Ts+1} = \frac{\square}{s+\square}$$

$$\begin{aligned} \therefore \frac{d}{dt}x &= \square x + \square u \\ y &= \square x + \square u \end{aligned}$$

例題 2. $\dfrac{Ks^2}{(Ts+1)^2}$ の実現

$$\frac{Ks^2}{(Ts+1)^2} = \frac{Ks^2}{T^2s^2+2Ts+1} = \frac{\square s^{\ 2}}{s^2+\square s+\square}$$

$$= \frac{\square s+\square}{s^2+\square s+\square} + \square$$

よって

$$\begin{aligned} \frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} &= \begin{bmatrix} 0 & 1 \\ \square & \square \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \\ y &= \begin{bmatrix} \square & \square \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \square u \end{aligned}$$

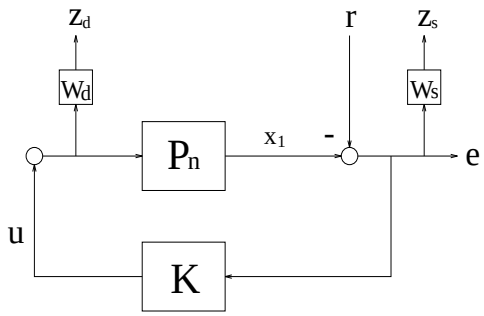


図 5: 一般化プラント

これらを踏まえると一般化プラントは図 5となり、その状態空間表現は次式となる。

$$\frac{d}{dt}x = Ax + B_1 r + B_2 u$$

$$\begin{bmatrix} z_s \\ z_d \end{bmatrix} = C_1 x + D_{11} r + D_{12} u$$

$$y = C_2 x + D_{21} r$$

ここで z は評価出力で、y はフィードバック出力。すなわち偏差 e をフィードバックしている。また、$x^T = [{x_n}^T, {x_s}^T, {x_d}^T]^T$ で、各行列は (1),(2),(3), (15),(16) 式の行列を用いると次ページの様に表される (例題 3)。

この一般化プラントを用いて設計した H_∞ コントローラのシミュレーション結果を図 6に示す。

具体的には x_1 の目標値を $r = 3.0$ として目標値応答特性をシミュレーションした。図 6では実プラントの応答が若干遅れているものの、ほぼノミナルと同様の応答を示している事がわかる。

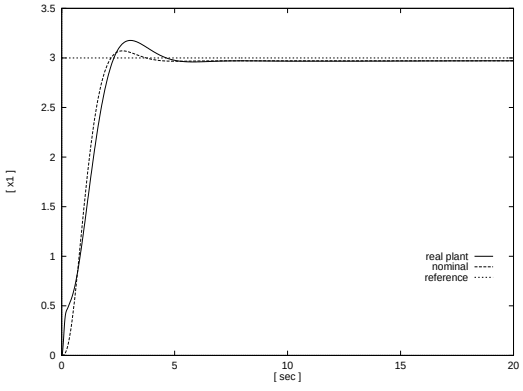


図 6: ステップ応答

ただし、図 6では若干の定常偏差が残っている事がわ

例題 **3.** 一般化プラントの状態空間表現

$$P_n: \begin{array}{rcl} \dot{x}_n &=& A_n x_n + B_n u \\ y_p &=& C_n x_n \end{array}$$

$$W_s: \begin{array}{rcl} \dot{x}_s &=& A_s x_s + B_s e \\ z_s &=& C_s x_s + D_s e \end{array}$$

$$W_d: \begin{array}{rcl} \dot{x}_d &=& A_d x_d + B_d u \\ z_d &=& C_d x_d + D_d u \end{array}$$

$$e = y_p - r$$

より

$$\frac{d}{dt}\begin{bmatrix} x_n \\ x_s \\ x_d \end{bmatrix} = \begin{bmatrix} \cdots & \vdots & \cdots \\ \cdots & \vdots & \cdots \\ \cdots & \vdots & \cdots \end{bmatrix}\begin{bmatrix} x_n \\ x_s \\ x_d \end{bmatrix} + \begin{bmatrix} \cdots \\ \cdots \\ \cdots \end{bmatrix} r + \begin{bmatrix} \cdots \\ \cdots \\ \cdots \end{bmatrix} u$$

$$\begin{bmatrix} z_s \\ z_d \end{bmatrix} = \begin{bmatrix} \cdots & \vdots & \cdots \\ \cdots & \vdots & \cdots \end{bmatrix}\begin{bmatrix} x_n \\ x_s \\ x_d \end{bmatrix} + \begin{bmatrix} \cdots \\ \cdots \end{bmatrix} r + \begin{bmatrix} \cdots \\ \cdots \end{bmatrix} u$$

$$y = e = \begin{bmatrix} \cdots & \vdots & \cdots \end{bmatrix}\begin{bmatrix} x_n \\ x_s \\ x_d \end{bmatrix} + \begin{bmatrix} \cdots \end{bmatrix} r$$

かる。これは閉ループ系がサーボ系になっていないのが原因である。

一般に 1 型のサーボ系を構成する、つまりステップ応答の定常偏差の残らないコントローラを構成するためには、内部モデル原理よりコントローラに積分特性を持たせれば良い事が知られている。H_∞ 制御理論では感度重み $W_s(s)$ に積分特性を持たせればコントローラに積分特性を持たせる事が出来るが、この場合一般化プラントが可安定でなくなるため、H_∞ 標準問題に帰着することが出来ない。そこで以下に示すような方法でサーボ系を構成する。

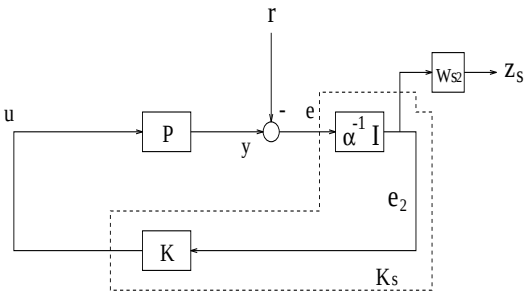


図 7: サーボ系設計の為のブロック線図

4 サーボ系の設計

図 7の様なブロック線図について標準 H_∞ 問題を解く。ここで、

$$\alpha(s) = \frac{s}{s+a}, \qquad a > 0$$

$$W_s(s) = \frac{1}{s+a}$$

とすると一般化プラントは可安定となり、標準問題が可解になる。

例題 4.　サーボ系設計の場合の一般化プラントとコントローラ

$$W_{s2}: \begin{array}{rcl} \dot{x}_s &=& A_{s2}x_s + B_{s2}e_2 \\ z_s &=& C_{s2}x_s + D_{s2}e_2 \end{array}$$

$$\alpha^{-1}: \begin{array}{rcl} \dot{x}_\alpha &=& A_\alpha x_\alpha + B_\alpha e \\ e_2 &=& C_\alpha x_\alpha + D_\alpha e \end{array}$$

とすると一般化プラントは次式。

$$\frac{d}{dt}\begin{bmatrix} x_n \\ x_s \\ x_d \\ x_\alpha \end{bmatrix} = \left[\begin{array}{c:c:c:c} & & & \\ \hdashline & & & \\ \hdashline & & & \end{array}\right]\begin{bmatrix} x_n \\ x_s \\ x_d \\ x_\alpha \end{bmatrix} + \begin{bmatrix} \\ \\ \\ \end{bmatrix} r + \begin{bmatrix} \\ \\ \\ \end{bmatrix} u$$

$$\begin{bmatrix} z_s \\ z_d \end{bmatrix} = \left[\begin{array}{c:c:c:c} & & & \\ \hdashline & & & \end{array}\right]\begin{bmatrix} x_n \\ x_s \\ x_d \\ x_\alpha \end{bmatrix} + \begin{bmatrix} \\ \end{bmatrix} r + \begin{bmatrix} \\ \end{bmatrix} u$$

$$y = e = \left[\begin{array}{c:c:c:c} & & & \end{array}\right]\begin{bmatrix} x_n \\ x_s \\ x_d \\ x_\alpha \end{bmatrix} + \begin{bmatrix} \end{bmatrix} r$$

この時、得られたコントローラを

$$K: \begin{array}{rcl} \dot{x}_c &=& A_c x_c + B_c e e \\ u &=& C_c x_c \end{array}$$

とすると、$Ks = K/\alpha$ より Ks は次式。

$$\frac{d}{dt}\begin{bmatrix} x_c \\ x_\alpha \end{bmatrix} = \left[\begin{array}{c:c} & \\ \hdashline & \end{array}\right]\begin{bmatrix} x_c \\ x_\alpha \end{bmatrix} + \begin{bmatrix} \\ \end{bmatrix} e$$

$$u = \left[\begin{array}{c:c} & \end{array}\right]\begin{bmatrix} x_c \\ x_\alpha \end{bmatrix}$$

さらにコントローラ K を

$$Ks(s) = K(s)/\alpha(s)$$

とすれば、Ks に原点極をもたせることができる。(例題 4)

これは

$$W_s(s) = \frac{1}{s}$$

として普通の H_∞ 制御問題を解いているのと等価である。

5 MATX を用いたシミュレーションの流れ

次に MATX による前述の一般化プラントを用いた H_∞ コントローラの設計およびシミュレーションの手順を述べる。今回のシミュレーションのために作成したプログラムを以下に示す。

plant.mm
weight.mm
cont.mm
close.mm
pflsim.mm

実行は “matx” と打ち込むことで MATX のインタラクティブモードにし、そこで例えば

load “plant.mm”

と打ち込むことによりファイルをロードして

main

と打ち込めばよい。
プログラムはそれぞれ

1. plant.mm ···プラントの状態空間表現の設定
2. weight.mm···Δ の見積りと各重みの設定
3. cont.mm ···一般化プラントの構成と設計
4. close.mm ···周波数整形の確認
5. pflsim.mm···時間応答のシミュレーション

の順に行なう。

5.1 Case 1(デフォルトパラメータでの設計)

まず plant.mm を実行すると実プラントの各パラメータの設定を聞いてくるが、この場合は最初から設定されている値 (1 ページの表参照) のままでよい。
設定値を変える必要のない時は、ただリターンキーを押せばよい。

実行例 1

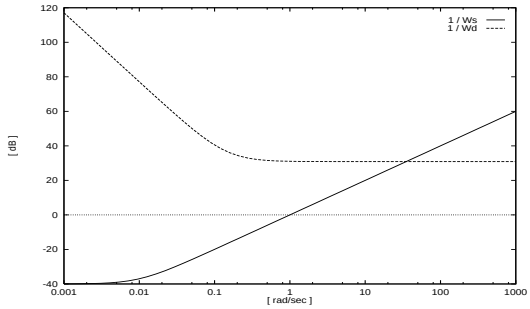
```
 MaTX (1) load"plant.mm"
 MaTX (2) main
m1 =   1.00000000E-01 ---->
k1 =   1.00000000E+00 ---->
c1 =   3.00000000E+00 ---->
m2 =   1.00000000E+01 ---->
k2 =   1.00000000E+02 ---->
c2 =   1.00000000E-02 ---->
```

その後、ノミナルおよび実プラントの状態空間表現の行列を表示してプログラムは終了される。
次に、 weight.mm を実行する。まず Case 1 か Case 2 かを聞いて来るので Case 1 を選択する (リターンキーを押す)。次にサーボ系かどうかを聞いて来るので”not” を選択する (リターンのみ)。すると各重み関数 $W_s(s), W_d(s)$ を表示した後それらのボード線図のグラフを表示する。

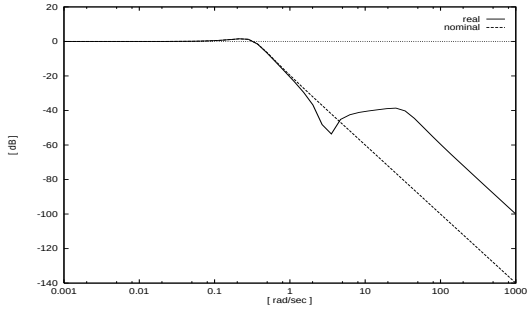
実行例 2

```
 MaTX (1) load"weight.mm"
 MaTX (2) main
 Case 1 or Case 2 ?

Case = 1 ---->

servo controller or not?
         (servo: q=1, not: q=2)

q = 2 ---->
         1
Ws = ----------
      s + 0.01

           1.4 s^2
Wd = -------------------
      49 s^2 + 14 s + 1

Case 1

1/Ws (Red) and 1/Wd (Green).

nominal(Red) and real plant(Green).

Delta(Red) and Wd(Green).
```

すなわち最初に各重みの逆数、次にノミナルおよび実プラントのボード線図、最後に加法的誤差 Δ と相補感度重み W_d が表示される (図 8参照)。
次に cont.mm により一般化プラントを構成し、コントローラを設計する。
ここでもサーボ系かどうか、 Case 1 か Case 2 かを聞いて来るのでそれぞれ “not”, “Case 2” を選択する。ここで H_∞ 制御問題が可解ならばそのままインタラクティブモードに戻るが、そうでない場合は Warning が表示されるので注意が必要である。



$W_s{}^{-1}, W_d{}^{-1}$ のボード線図



ノミナルと実プラントのボード線図

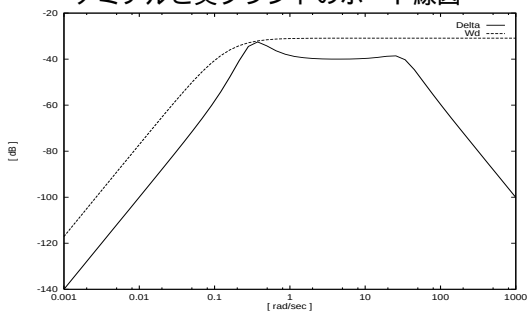
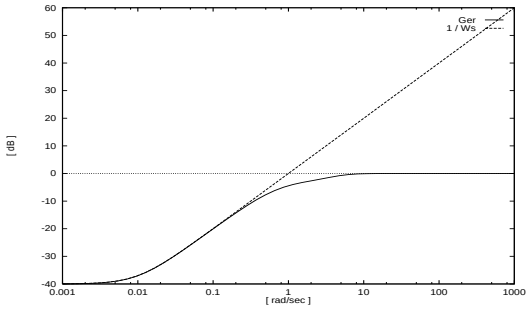
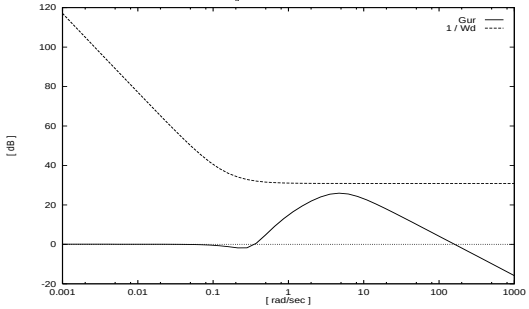


図 8: 加法的誤差 Δ と W_d のボード線図



G_{er} と $\frac{1}{W_s}$ のボード線図



G_{ur} と $\frac{1}{W_d}$ のボード線図

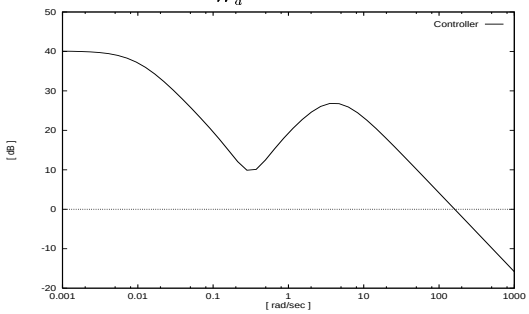


図 9: コントローラのボード線図

実行例 3

```
MaTX (1) load"cont.mm"
MaTX (2) main

servo controller or not?
       (servo: ii=1, not: ii=2)

ii = 2 ---->

Case 1  or Case 2?

Case = 1 ---->
```

次にこのコントローラにより周波数整形がなされているかを調べてみる。具体的には close.mm を実行すればよい。なお close.mm は Case 1 用のみのプログラムなので、サーボ系かどうかの選択だけでよい。

実行例 4

```
MaTX (1) load"close.mm"
MaTX (2) main

servo controller or not?
         (servo: q=1, not: q=2)

q = 2 ---->
  Bodeplot of 1/Ws(Red) and Ger(Green)

  Bodeplot of 1/Wd(Red) and Gur(Green)

  Bodeplot of Controller
```

結果は図 9の様に表示される。これにより閉ループ系がそれぞれ感度重み W_s, 相補感度重み W_d によりおさえ

られていることがわかる。

最後に pflsim.mm を実行することにより、時間応答のシミュレーションを行なう。

ここではまずプラントが Case 1 か Case 2 か? サーボ系かどうか? コントローラが Case 1 用か Case 2 用かを聞かれるので、それぞれ Case 1, not servo, Case 1 を選べばよい。すると、x_1 と u の時間応答のグラフが表示される (図 10参照)。

実行例 5

```
 MaTX (1) load"pflsim.mm"
 MaTX (2) main

 Is real plant Case 1 or Case 2?

Case = 1 ---->

 servo controller or not?
(servo: ii=1, not: ii=2)

ii = 2 ---->

 Controller is for Case1 or not?

Case = 1 ---->

 Red   is time response of nominal.
 Green is real plant's.
 Blue is reference.

 Red   is input in nominal plant.
 Green is input in real plant.
```

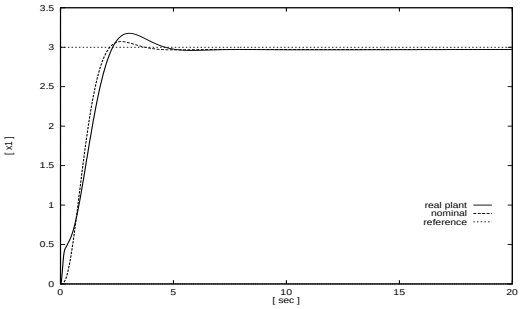
5.2 サーボ系 (Case 1)

次にサーボ系の設計を行なう。

サーボ系の設計も大体の流れは前述のものとほとんど同じである。

プラントの状態空間表現は先程 plant.mm を実行したことにより既に出来ているので、 weight.mm から始めてよい。ここで先程と同様に Case 1 および servo を選択すると実行例 6 の様に α, W_{s2}, W_d を表示した後各ボード線図が描かれる。

実行例 6



時間応答 (x_1)

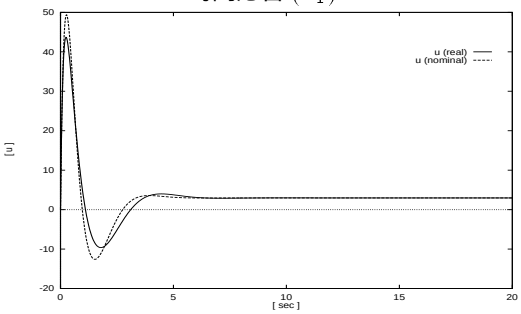


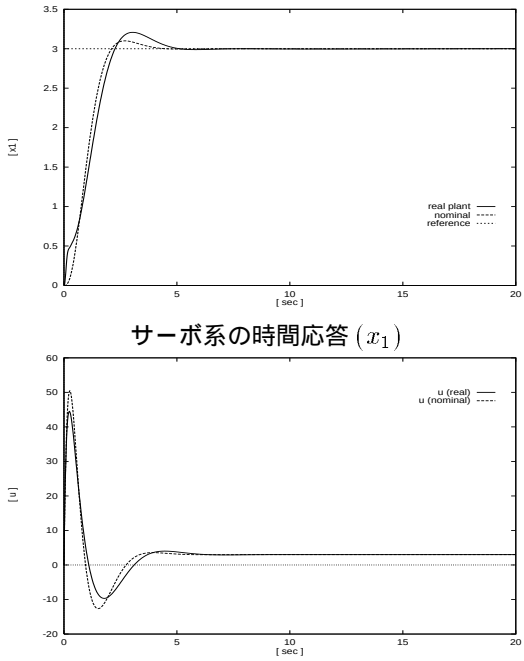
図 10: 時間応答 (u)

```
 MaTX (1) load"weight.mm"
 MaTX (2) main
 Case 1 or Case 2 ?
Case = 1 ---->

 servo controller or not?
(servo: q=1, not: q=2)

q = 2 ----> 1
              s
 alpha = -----------
          s + 0.001
            1
 Ws2 = -----------
        s + 0.001
            1.4 s^2
 Wd = -------------------
       49 s^2 + 14 s + 1

Case 1

1/Ws (Red) and 1/Wd (Green).

nominal(Red) and real plant(Green).

Delta(Red) and Wd(Green).
```

その後 cont.mm を実行して、ここでも Case 1 と servo を選択すればサーボ系用のコントローラが設計され、コントローラの極が表示される。これによりコントローラが原点極を持っている、即ち積分特性があることがわかる。

なお、このコントローラはファイル"K_sv.mx" として保存されているのでこのファイルが新たに出来ているはずである。(not servo の場合は"K.mx")

最後に pflsim.mm において、プラントおよびコントローラを Case 1 にして servo を選択すればサーボコントローラのシミュレーション結果が表示される (図 11参照)。この図からも定常偏差がなくなっていることが確認できる。



サーボ系の時間応答 (x_1)

図 11: サーボ系の時間応答 (u)

5.3 Case 2

次に実プラントのパラメータを変えて設計を行なってみる。実プラントのパラメータはそれぞれ以下のように変えて設計を行なう。

$$m_1 : \ 0.1 \ \longrightarrow \ 5.0$$
$$m_2 : \ 10.0 \ \longrightarrow \ 5.0$$
$$k_2 : 100.0 \longrightarrow 50.0$$

まず plant.mm において以下のように値を変更する。

実行例 7

```
  MaTX (1) load"plant.mm"
  MaTX (2) main
m1 =   1.00000000E-01 ----> 5
k1 =   1.00000000E+00 ---->
c1 =   3.00000000E+00 ---->
m2 =   1.00000000E+01 ----> 5
k2 =   1.00000000E+02 ----> 50
c2 =   1.00000000E-02 ---->
```

次に weight.mm を実行し、 Case 2 を選択する。するとサーボ系かどうかの選択の後、パラメータ T, Kd, Ks の入力モードになる (実行例 8 参照)。これらはそれぞれ

$$W_d = \frac{K_d s^2}{(1+Ts)^2}$$

$$W_s = \frac{K_s}{s+0.01}$$

である。これらを変えることにより適当な重みを決める。具体的には各重みと Δ, W_d のボード線図が表示された後 W_d が適当かどうかを聞いてくるのでボード線図をみて判断すればよい。もう一度重みを決め直したい時にはここで no を選択すれば各パラメータの入力モードに戻ることが出来る。

実行例 8

```
 MaTX (1) load"weight.mm"
 MaTX (2) main
 Case 1 or Case 2 ?

Case = 1 ----> 2

 servo controller or not?
         (servo: q=1, not: q=2)
q = 2 ---->
T =   7.00000000E+00 ---->
Kd =   1.40000000E+00 ---->
Ks =   1.00000000E+00 ---->
         1
Ws = ----------
      s + 0.01
           1.4 s^2
Wd = --------------------
      49 s^2 + 14 s + 1
```

```
 Delta(Red) and Wd(Green).

 Is Wd ok? (yes: 1 or no:2)
yorno = 1 ---->
```

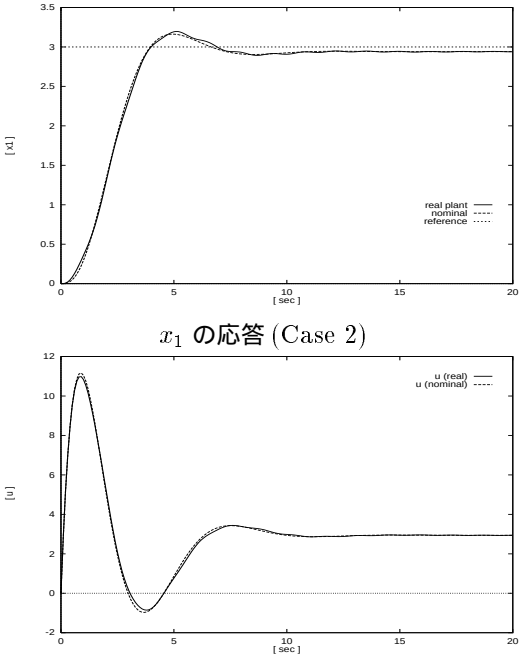
次に cont.mm によりコントローラを決定する。

これは Case 2 を選択し、あとは同様に行なえばよい。ただしここで Warning が出る時は H_∞ 制御問題が解けていないことになるので重みの設定からやりなおして、Warning が出ないように調整する必要がある。

時間シミュレーションについてもプラントとコントローラを Case 2 に選択し、あとはサーボ系かどうかを選択すればよい。

例として各パラメータを以下のようにして設計したコントローラ (サーボではない) によるシミュレーション結果を図 12に示す。

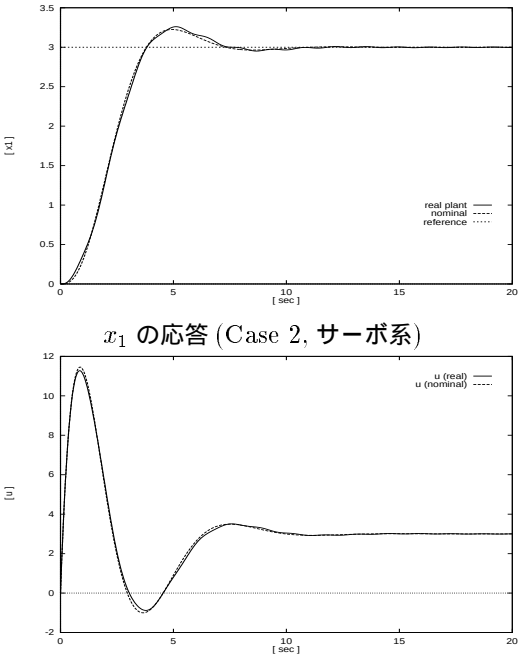
$$
\begin{aligned}
T &= \square \\
K_d &= \square \\
K_s &= \square
\end{aligned}
$$



x_1 の応答 (Case 2)

図 12: 制御入力 u の応答 (Case 2)

図 12を見る限りではやはり定常偏差が残っているため、ここでもサーボ系コントローラを設計する。



x_1 の応答 (Case 2, サーボ系)

図 13: 制御入力 u の応答 (Case 2, サーボ系)

用いたパラメータ (T, K_d, K_s) の値や設計法はほとんど同じでよい。

結果を図 13に示す。

ここでも定常偏差をなくすような設計が実現できていることがわかる。