

## 実験 I. 知的・機能工学システム実験

### 1. I-cd: 制御系設計

#### 背景：

様々な工業分野において商品の開発・製品化サイクルは年を追って短くなっているが、これにはコンピュータソフト支援環境の発達と充実が大きく貢献しており、制御システムも例外ではない。例えば航空機のフライトコントローラの基礎設計には従来年単位の時間を要したが最近では月単位程度に短縮されている。この様に、制御技術者はソフト支援環境を操りながら制御系の設計や評価を行う必要がある。

#### 目的：

本項目では垂直離着陸機 (VTOL) のピッチ角オートパイロットを例に、世界で広く利用されているソフト支援環境のうち MATLAB/Simulink<sup>1</sup>を取り上げてその基礎的な利用法を学ぶ。特に、根軌跡法によるピッチ角制御系の設計と評価を行い、このような支援環境を前提とした方法にある程度習熟し、今後の実験や卒業研究などの役に立たせる。

#### 注意：

これらのソフトは制御系設計を行う技術者を支援する環境であり、技術者の制御系に対する理解を不要にするものではない。基礎理論の理解なしでソフトを使用しても無闇に時間を費やすだけに終わることが多く、これでは本末転倒である。また使用するソフトで得られた結果を正しく評価することや、結果の妥当性などを検討することは今の所は設計者の仕事であり、そのためにも基礎理論の理解が必要である。更に、使用するソフトの限界や有効性を理解するにもある程度理論が必要である。この他に、理論を押さえなければ理論の限界をソフトで補えることも多くある。

#### 仮定：

予備知識として伝達関数や状態方程式による動的システムの表現法と基礎的なアナログ制御の概念は既に持っているものとする。ここでは理論に深く立ち入らないので、各自必要に応じて制御のノート等を読み返して理論の復習をしておくこと。なお、プログラムを保存するためにディスクケットを各自で用意すること。

#### 予定：

- 第1週目は二人一組のグループを作り、数値計算とそのグラフ化に便利な MATLAB の基本的使用法を学ぶ。毎週終了時に簡単な口頭試問を受ける。
- 第2週目はシミュレーションに便利な Simulink の基本的使用法を習得する。
- 第3週目は制御系設計用のツール集 (Control Toolbox) の中で根軌跡法に基づく rltool を利用する。
- 第4週目は根軌跡法による VTOL 機ピッチ角制御システムの設計とシミュレーションを行う。一週間後にこのプロジェクトを完了させ、グループレポートを提出する。

<sup>1</sup> MATLAB と Simulink は米国 The Math Works, Inc.の登録商標。

## 第1週目：MATLABの基本的な使用法

この実験手引き書を理解して読みながら MATLAB を試してみよう。その際 M ファイルと呼ばれるプログラムを作成することになるが、これはプロジェクトにも役立つと思われるので保存しておくこと。一通り手順を終了したら教官或いは TA による簡単な質疑応答を受け、パスしたらめでたく終了である。尚、本日の内容は単に monkey-see monkey-do をするだけであれば高校生にでも出来ることである。内容の理解に努めよう。MATLAB には非常に多くの機能があるが、ここでは VTOL 機の運動を計算してプロットすることに限定する<sup>2</sup>

### 1. MATLAB の開始

ウィンドウズ内の MATLAB のアイコンをダブルクリックするか、スタート+プログラム+MATLAB から MATLAB を開始すると、MATLAB コマンドウィンドウ内にメッセージと MATLAB プロンプト<sup>2</sup>が表われる(図 1.1)。

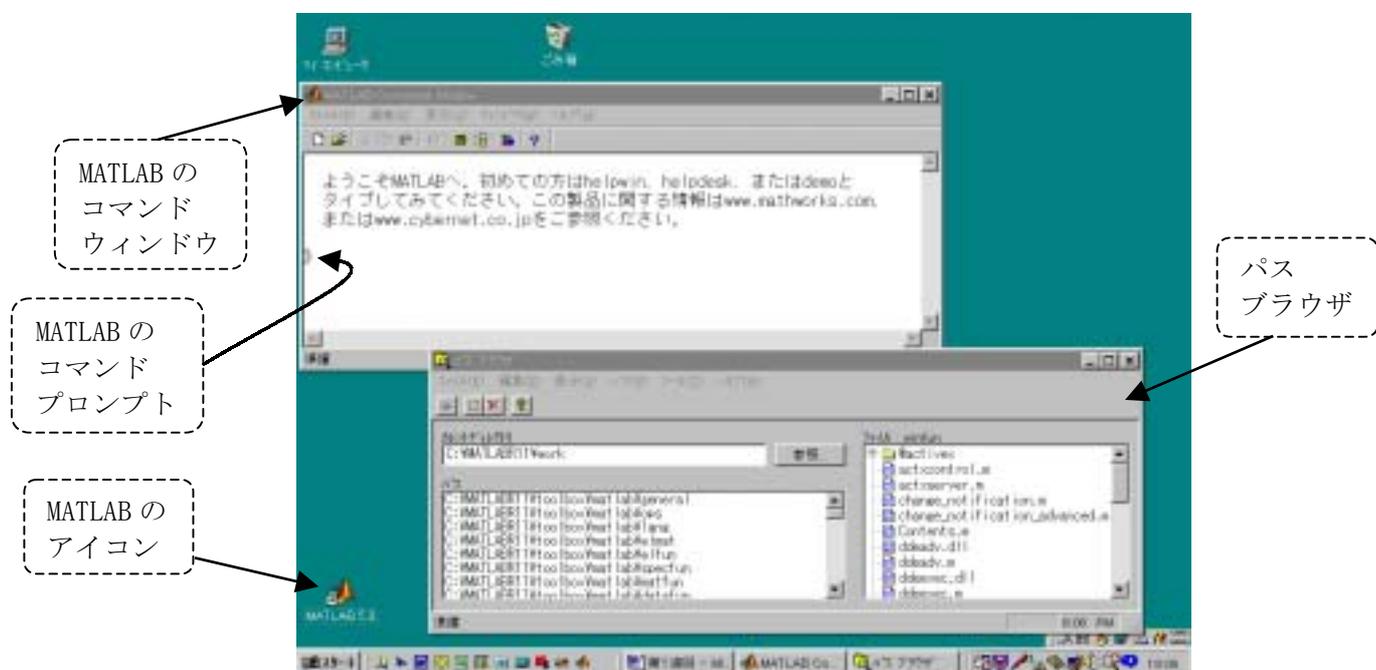


図 1.1 MATLAB コマンドウィンドウとパスブラウザ  
(ターミナルによって見え方が異なる)

これから色々ファイルを作成するので、それを保存する場所を決める。

1. **L204 のパソコンを使用する場合。**この場合は MATLABR11/WORK/にファイルが保存されるように初期設定されているので、何もしなくてよい(この設定は変更しないこと)。またハードディスク内のプログラムファイル等は一切変更しないように留意すること。
2. **L504 のターミナルを使用する場合。**L204 は施錠されているので時間外にプロジェクト等で MATLAB を使用するには L504 のターミナルを利用することになる。各自の z ドライブに MATLAB ファイル保存用のディレクトリを作成し、MATLAB コマンドウィンドウの ファイル+パス の設定によりパスブラウザを開く。参照ボタンを押してその z ドライブをカレントディレクトリ

<sup>2</sup> **注意：**本実験で使用する環境にはツールボックスが多く備わっており、他の実験室や個人の持つ環境では利用できないコマンドも存在する。またこれら膨大な量のコマンドのうち本項目では一部しか使用しないので、今後は必要に応じて各自で更に高度な使用法を習得して欲しい。(オンラインマニュアルやヘルプ機能を利用すればかなりのことは出来るようになる。)

(使用する現在のディレクトリ)に指定して閉じる。

以上で MATLAB 使用の準備が出来た。

## 2. 航空機の運動方程式

機体を剛体と考えると(柔軟性を無視して)ニュートンの運動法則を適用すると、航空機は一般には縦横の相互干渉を持つ幾つかの非線形常微分方程式で表される。しかし、定常飛行状態からの微小擾乱のみを考える場合には縦横の干渉がない線形常微分方程式で近似できる(高迎角飛行やクイックロールなどの場合はダメ)。垂直離着陸機(VTOL機、図1.2)は垂直推力によりホバーしたり、巡航速度で前進したり、これらの間の遷移状態での飛行などが可能な航空機である。しかし低速では空気力が小さいため一般に不安定であり、パイロットの負担を軽減するために安定性増大装置(SAS)が必要である。以下ではVTOL機の線形方程式に基づいた簡単なピッチ角制御オートパイロットを考よう。

空中で静止飛行をするホバー状態にある航空機の運動は通常、上下運動と姿勢及び前進運動とに分離してモデル化するが、ここでは姿勢及び前進運動のみを考える。この運動は安定軸系と呼ばれる座標軸を用いると



図 2.1 垂直離着陸機 (VTOL 機) の例

$$\begin{aligned}\dot{v}(t) + g\theta(t) - X_v v(t) &= X_\delta \delta(t) \\ \ddot{\theta}(t) - M_q \dot{\theta}(t) - M_v v(t) &= M_\delta \delta(t) \\ q(t) &= \dot{\theta}(t)\end{aligned}\tag{1-1}$$

と表すことができる<sup>3</sup>。ここで  $v(t)$  は機体  $x$  軸(前向きが正)方向の速度(m/s)、 $\theta(t)$  はピッチ角(機首上げが正方向)(rad)、 $q(t)$  はピッチレート(rad/s)、 $\delta(t)$  はエレベータ角(昇降舵下げが正の角度)に相当する入力(rad)であるが、これらは全て定常飛行状態からの変化分である。また  $g$  は重力加速度(32.2ft/s<sup>2</sup>)であり、 $X_v$  はドラッグダンピング(-0.019/s)、 $X_\delta$  はピッチドラッグ(32.2ft/s<sup>2</sup>)、 $M_q$  はピッチレートダンピング(-0.58/s)、 $M_v$  は速度ダンピング(0.004/ft/s)、 $M_\delta$  はピッチ効率(-6.0/s<sup>2</sup>)と呼ばれる係数である。これらの係数は飛行条件により変化するが、上の値は低高度ホバーリング時のものである。これらパラメータの数値を MATLAB に入力しよう。

MATLAB コマンドウィンドウ内では電卓の様に数値計算が行える。例えば  $\sqrt{2} \times \pi^2$  の計算は

```
» sqrt(2)*pi^2 リターン
```

と入力すればよく、計算結果は

```
ans =  
13.9577
```

とスクリーン上に表示される。四則演算は+-\*/、べき乗は^である。»help sqrt とすると平方根演算 sqrt に関する情報が表示される。ans もヘルプで調べれば分かるが、これは直前の計算結果を記憶している変数である。

【このカッコ記号の中の内容はいつか時間がある時に試してみるとよい事項である。

<sup>3</sup> 実験当日に教官(或いは TA)が軸系を図示する。

help 以外に lookfor も便利である。(ヘルプにはヘルプデスクとヘルプウィンドウもある。) スカラー関数には sqrt のほかに三角関数(sin,cos,tan)、指数関数(exp)、対数関数(log,log2,log10)などがある。また MATLAB にはデモンストレーションがあるので、いつかコマンドウィンドウで  $\gg$ demo と入力して一通り見てみよう。】

さて、計算が複雑になるとコマンドラインで毎回式を書くことは面倒であるから、これを M ファイル (拡張子が .M のファイル) としてセーブすることが出来る。以後そのファイル名を拡張子なしでプロンプト  $\gg$ 後にタイプしてリターンキーを押せば、ファイル内のコマンドをすべて実行してくれる。まずコマンドウィンドウにおいて、ファイル + 新規作成 + M-file として MATLAB エディター/ディバッガーを呼び、次の様にタイプしてみよう。(図 1.3)

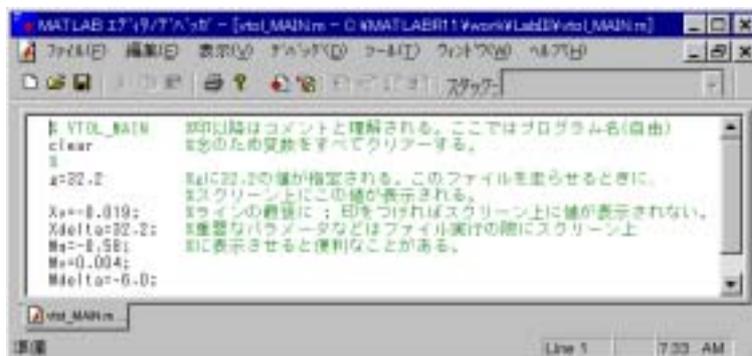


図 2.2 MATLAB エディター/ディバッガー

% VTOL_MAIN	%プログラム名。%印以降はコメントと理解される。
clear	%念のため変数をすべてクリアーする。
%	
g=32.2	%g に 32.2 の値が指定され、スクリーン上に値が表示される。
Xv=-0.019;	%ラインの最後に ; 印をつければこのファイルを走らせる
Xdelta=32.2;	%ときにスクリーン上に値が表示されない。
Mq=-0.58;	%重要なパラメータなどはファイル実行の際にスクリーン上
Mv=0.004;	%に表示させると便利なおことがある。
Mdelta=-6.0;	

上の内容をエディター内に打ち込み、ファイル名を例えば VTOL\_MAIN として必ず自分の z ドライブに保管すること。こうしておく次回からは  $\gg$ VTOL\_MAIN と入力するだけでこれらの変数に値が指定される。VTOL\_MAIN は拡張子が .M でなければならないが、このエディターでは自動的にそう保存される。もし他のエディターを用いる場合は必ず拡張子を .M とすること。

これ以降は新しいコマンド等が出て来たらその都度コマンドラインあるいはファイルに打ち込んで試してみること。

変数名とその値は MATLAB ワークスペースに保管されているが、どの様な変数が使われているかは  $\gg$ who 或いは  $\gg$ whos とコマンド入力すると表示される。変数全部を消去したい時は clear を使用する。また clear nnn とすると nnn という変数のみが消去される。一回消去された変数は再度指定しなければ元に戻らない。

### 3. 伝達関数

初期値をすべて零として (1) 式をラプラス変換すると

$$\begin{aligned}
 sV(s) + g\Theta(s) - X_v V(s) &= X_\delta \Delta(s) \\
 s^2\Theta(s) - M_q s\Theta(s) - M_v V(s) &= M_\delta \Delta(s) \\
 Q(s) &= s\Theta(s)
 \end{aligned}
 \tag{1-2}$$

となる。ここで入力をエレベータ角  $\Delta$ 、出力をピッチ角  $\Theta$  とするシステムの伝達関数は、(2) 第 1 と 2 式より

$$G(s) = \frac{\Theta(s)}{\Delta(s)} = \frac{M_\delta s + (M_v X_\delta - M_\delta X_v)}{s^3 - (M_q + X_v)s^2 + X_v M_q s + M_v g} \quad (1-3)$$

と求まる。この伝達関数を MATLAB で使えるようにするために新しく VTOL\_TF.M を作成しよう。まず MATLAB での多項式はその係数から成る配列として定義される。例えば  $s^4 + 2s^3 + 3s + 4$  という多項式  $p$  を定義するには  $\gg p=[1 \ 2 \ 0 \ 3 \ 4];$  と高次の項の係数から順番に並べ、スペース（あるいはタブやコンマ）で区切る。伝達関数には分母と分子に多項式があるから、例えば

```
% VTOL Transfer Function      %VTOL_TF のコメント。≫help vtol_tf とすると
% This file requires          %このファイル頭部のコメントが表示される。内容
% vtol_main.m to run.        %を忘れてしまった時など、中身を調べるのに便利
% (by N. Hori, May, 2000)     %である。
VTOL_MAIN                     %vtol の係数データをこのファイルから生成する。
                               %あるいは VTOL_MAIN.M の内容をそのままここへ
                               %入れてもよい。
num=[Mdelta Mv*Xdelta-Mdelta*Xv]; %伝達関数の分子多項式の係数からなる
                               %1x2 ベクトル。
den=[1 -Mq-Xv Xv*Mq Mv*g];    %伝達関数の分母多項式係数からなる 1x4 ベクトル。
G=tf(num,den)                 %tf は伝達関数を生成するコマンド。分子は第一
                               %要素、分母は第二要素である。変数に関しては
                               %大文字と小文字は区別されるので、この G と g は
                               %混同されない。
```

とエディターに打って VTOL\_TF.M とセーブすればよい。その後は  $\gg$ VTOL\_TF とするだけで VTOL\_MAIN も実行され VTOL 機の伝達関数が指定される。ただし、VTOL\_MAIN は VTOL\_TF と同じディレクトリになければならない点に注意されたい（パスを指定すればそうでなくてもよいが）。なお、zpk(G)は G の分母分子多項式を因数分解した形で表現し、極や零点が分かりやすい。また多項式 den の根は roots(den)で求まる。

#### 4. 時間応答

VTOL 機の伝達関数より時間応答を調べることができる。時間応答は後に Simulink の所でも扱うので、ここではステップ入力とインパルス入力に対するシステムの応答計算とそれらのプロットの仕方について触れておく。

まず、単位ステップ応答は  $\gg$ step(G)とすると自動的に Figure ウィンドウにプロットされるので試してみる（図 1.4 左）。

以下、プログラムを変更する場合は、元の部分は消さずに%マークを付けてコメントとして残しておくこと。実験終了後にそのファイルを見せる様に教官あるいは TA から指示されることもある。

もし自分でステップ応答をプロットしたければ（例えばタイトルを変更したいなどの理由で）、新しいファイルを作成するか、先のファイルの最後に次を足せばよい。

```
clf                             %プロットを始める前に、念のため図用ウィンドウ
                               %をクリアしておく。
[ys,ts]=step(G);               %時間ベクトル ts が自動的に作成され、ステップ
                               %応答 ys が計算される。
plot(ts,ys)                    %横軸が ts、縦軸が ys のプロットが描かれる。
title('Step Response of Uncontrolled VTOL') %図のタイトルが挿入される。
xlabel('Time (s)')              %図の横軸にラベルが挿入される。
```

```
ylabel('Pitch Angle (rad)')
grid
```

%図の縦軸にラベルが挿入される。  
%プロットにグリッドを入れる。

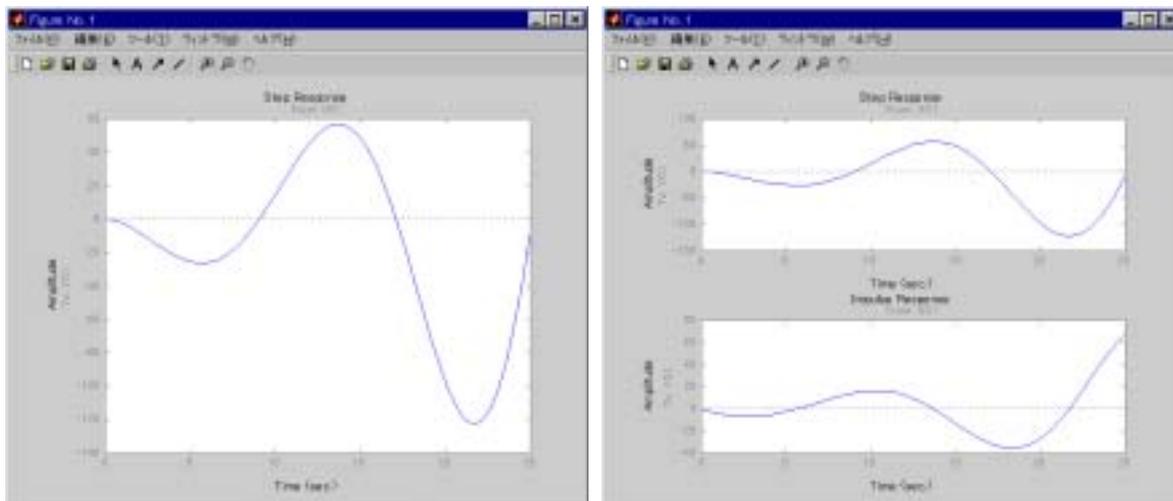


図 4.1 プロット(左)とサブプロット(右)の例

【軸の操作には `axis`、図中にテキストを挿入するには `text` というコマンドがあるので、時間がある時にヘルプで調べてみよ。】

インパルス応答は `impulse(G)` によりプロットできる。もしステップとインパルスを別々にスクリーンいっぱいに表示したければ、間にポーズを入れて

```
step(G)           %G のステップ応答を表示。
pause            %何かのキーを押すまでそのままの状態を保持する。
impulse(G)       %インパルス応答を表示。
```

とすればよい。スクリーンを幾つかに区切って同時に表示することもできる(図 1.4 右)。例えばスクリーンを上下二つ (2x1) に分け、上 (1 番目) にステップ応答、下 (2 番目) にインパルス応答を表示するには

```
subplot(211),step(G)   %上にGのステップ応答
subplot(212),impulse(G) %下にインパルス応答
```

とすればよい。また、もし両方を一つのグラフにプロットしたければ、

```
[yi,ti]=impulse(G)    %インパルス応答 yi を計算する。
plot(ts,ys,ti,yi)     %ys と yi をプロットする、
legend('Step Response','Impulse Response') %どちらがどのプロットに対応
%するかを図中に示す。
```

とする。また、各線に特徴を付けることもできる。例えば `ys` を赤 (r) の点線 (:) で、`yi` を青 (b) の実線 (-) に x 印 (x) を添えて書きたい場合は

```
plot(ts,ys,'r:',ti,yi,'b-x') %初めの3要素がステップ応答用、その次の3要素が
%インパルス応答用。
```

とすればよい。他の色、線のタイプ、マーカーなどについては `help plot` で知ることができる。

二つを別々にプロットすることも出来る。例えば図ウィンドウを上下には 1 個に、左右には 2 個の部分に分割(1x2)し、その左側 (1 番目) に `ys` を、右側 (2 番目) に `yi` をプロットした

い場合は

```
subplot(121),plot(ts,ys) %左側にステップ応答をプロット。(1x2の1番目)
title ? %?部にプロットのタイトルやラベルを各自で入れよ。
subplot(122),plot(ti,yi) %右側にインパルス応答をプロット。(1x2の2番目)
title ? %?部にプロットのタイトルやラベルを各自で入れよ。
```

とすればよい。画面は 9x9 まで分割できるが小さくなり過ぎるので、せいぜい 4x4 位までにしておいた方が無難であろう。

さて、上のプログラムでは 1 ラジアン (57.3 度) という大きなステップ入力に対しては数十 rad というピッチ角が発生している。しかしこのモデルが有効であるのは入力出力ともせいぜい 30 度程度であるから本当の応答とはかけ離れている (実際には墜落してしまう)。このように、シミュレーション結果を見る場合には使用しているモデルの限界 (定常状態からの微小擾乱) を常に認識しておく必要がある。なお、線形系では単位が rad でも度でも応答波形は同じであり、その単位もラジアンとも度とも自由に解釈できる。例えばもし入力が 1 rad の単位ステップであると考えれば、出力単位も rad であり、1 度の単位ステップと考えれば、出力単位は度である。よって、例えば 10 度のステップ入力を加えた場合のステップ応答が欲しければ、

```
[ys,ts]=step(G); %入力を1度と解釈すると、ysの単位は度である。
ys=ys*10; %1度ステップに対する応答を10倍すれば線形系では
%10度ステップに対する応答と同じになる。
```

とすればよい。(非線形系ではこうすると大きな誤差を招く。)

**step** は自動的に時間ベクトルを作成して応答を計算してしまうが、これを自分で設定したい場合は例えば

```
ts=0:0.1:1; %0から1秒まで刻み0.1秒で時間(行)ベクトルtsを生成する。
%なおts=t0:t1; とするとt0からt1まで自動的に刻み幅1となる。
[ys,ts]=step(G,ts) %上のtsを用いて単位ステップ応答を計算する。
ys=ys*10; %10度ステップ応答に変更
```

とすると、0 から 1 秒までの応答がプロットされる。これによれば 10 度のステップ舵を取ると 1 秒後にはピッチは -25 度 (機首下げ) 程になることが分かる。

## 5. 周波数応答

時間応答以外にも伝達関数からはボード線図、ナイキスト線図、根軌跡、極零配置などを書いて、安定性その他の特性を調べることもできる。ボード線図は

```
bode(G) %Gのボード線図。≫help bode で使用法を参照のこと。
```

で得られ、振幅比はデシベル ( $20\log_{10} x$ )、位相差は度である。

【ナイキスト線図、根軌跡、極零配置は夫々

```
nyquist(G) %ナイキスト線図。≫help nyquist で使用法を参照。
rlocus(G) %根軌跡。≫help rlocus で使用法を参照。
pzmap(G) %極零配置。≫help pzmap で使用法を参照。この他に
%pole(G)、zero(G)もある。】
```

上の **bode(G)** で表示されるボード線図とは違った周波数帯でのプロットも可能である。例えば 0.0001rad から 100rad までの周波数範囲における周波数応答を計算するには

```
bode(G,{0.0001,100})
```

とすればよい。上では計算に使用される周波数の数が自動的に決定され(上の場合には **49**)、時にはプロットが粗いこともある。もっと多くの点で計算したければ、周波数ベクトルを次のように発生すればよい。例えば、**0.0001rad** から **100rad** までの範囲で **100** ポイントにおける周波数応答を計算するには

```
w=logspace(-4,2,100); %周波数 10-4rad/s から 102 rad/s まで、対数尺上で
                        %100 分割したベクトル w を作成。
bode(G,w)              %上の w を用いてボード線図を書く。
```

とすればよい。

【ゲイン余裕、位相余裕とゲイン交差周波数、位相交差周波数を調べるには **margin** というコマンドがある。なお対数尺の **logspace** に対し、線形尺では **linspace** がある。例えば **1** から **10** までを **25** 等分してベクトルを作るには **linspace(1,10,25)** とすればよい。

ボード線図をすべて自分でプロットしたい場合には、**freqresp** で周波数応答を計算し、**semilogy** で y 軸片対数プロットあるいは **loglog** で両対数プロットができる。この他には **hold**, **zoom** が、また 2 次元プロットに関するコマンドとして **semilogx**, **plotyy**, などや、3 次元プロットには **plot3**, **mesh**, **contour3**, **surf** などがある。】

## 6. 状態方程式

運動方程式(1)において状態変数を例えば

$$x_1(t) = v(t), \quad x_2(t) = \theta(t), \quad x_3(t) = q(t) \quad (1-4)$$

と定義すると、

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \end{bmatrix} = \begin{bmatrix} X_v & -g & 0 \\ 0 & 0 & 1 \\ M_v & 0 & M_q \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix} + \begin{bmatrix} X_\delta \\ 0 \\ M_\delta \end{bmatrix} \delta(t) \quad (1-5)$$

$$\theta(t) = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix} \quad (1-6)$$

という状態方程式が得られる。これを MATLAB で作るには、

```
%VTOL_SSF      %VTOL_SSF. M のコメント。
vtol_main;     %係数 Xv などを vtol_main.m から生成する。
A=[Xv -g 0;    %行の要素の区切にはスペース、タブ、コンマを使用。
  0 0 1;       %改行はセミコロンやリターンキーで。
  Mv 0 Mq];    %MATLAB エディターではリターンキーを押すと自動的に
b=[Xdelta; 0; Mdelta]; %インデントする。
c=[0 1 0];     %最後のセミコロンで印字を抑える。
d=0;           %直達項が無い場合も 0 と指定すること。
vtol=ss(A,b,c,d) %ss は A, b, c, d から成る状態方程式を作成する。
```

とすればよい。この後は伝達関数の場合と同様に時間応答 `step(vtol), impulse(vtol)` や周波数応答 `bode(vtol)` を求めることが出来るのですべて試してみることに。

さて、伝達関数と違って状態方程式は唯一でない。相似変換、正準形、その他は `cannon, ctrbf, obsvf` などのヘルプを参照にされたい。また、先に求めた伝達関数  $G(\text{num}, \text{den})$  からコントローラ正準形の状態方程式を作成するには

```
[A,b,c,d]=tf2ss(num,den) %G からコントローラ正準形の A, b, c, d を求める。
```

で、逆に状態方程式から伝達関数を求めるには

```
[num,den]=ss2tf(A,b,c,d) %A, b, c, d から伝達関数の分子分母の係数を求める。
```

が利用できる。残念ながら一般の伝達関数表示法やコントローラ正準形、オブサーバ正準形ではシステムの次数が余り高くなると（場合によるが **20** 次程度以上）数値計算的に好ましくなくなる。このような場合には **1** 次か **2** 次の要素に並列あるいは直列分解した伝達関数や、対角形あるいはジョルダン形の状態方程式が好ましい。

MATLAB は行列演算に優れており、以下で最も基本的な演算をいくつか示す。これらの例はファイルを作成してもよいが、コマンドラインから入力して一つ一つ見た方がよいであろう。

```
% Sample Program
%
eye(2)           %2x2 の単位対角行列。eye(2,2)の短縮形。
zeros(3)        %全要素が零の 3x3 行列。zeros (2,3)で 2x3 の零行列となる。
ones(4)         %全要素が 1 の 4x4 行列。ones(4, 4)の短縮形。
%
AA=[1 -2; -3 4]; %以下の例のためのデータ。
BB=[2 3; -4 -1];
%
AA^3            %AA の 3 乗
AA'            %AA の転置行列
inv(AA)        %AA の逆行列
AA+BB          %普通（要素ごとの）足し算
AA*BB          %普通の行列積
AA.*BB         %対応する位置にある要素同士の積（上の積と比較せよ）
AA/BB          %AA を BB で割る、つまり AA×inv(BB)
AA\BB          %BB を AA で割る、つまり inv(AA)×BB
%
eig(AA)        %AA の固有値。
trace(BB)      %BB のトレース。
rank(AA)       %AA のランク。
det(BB)        %BB の行列式。
%
expm(AA)       %行列 AA の指数行列。
exp(AA)        %行列 AA の要素ごとの指数からなる行列(上と比較せよ)
```

さて、 $AA \times \text{inv}(AA)$  は理論的には単位行列であるが、

```
AA*inv(AA)
```

とすると零であるはずの (2, 1) 要素に負のサインが見える。これは誤差のため完全に零となっていないからである。≫ `format long e` とすると計算結果はスクリーン上では 15 桁の浮動小数で表

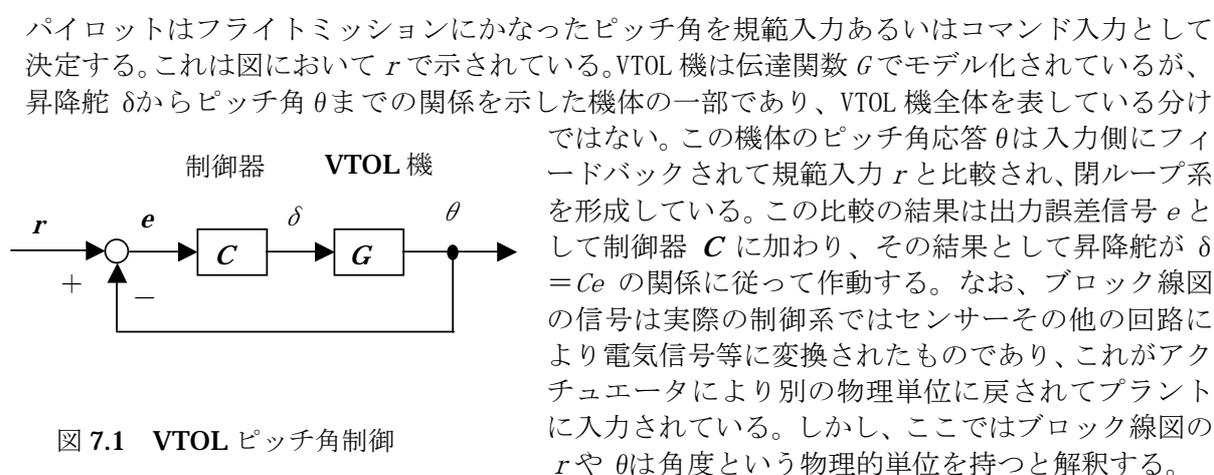
示されるようになり、 $-8.881784197001252e-016$  であることが分かる。なおデフォルトは `format short` (5桁の浮動小数表示) であるが、どれもスクリーン表示の方法であり、計算結果に影響は無い。行列計算にすぐれた MATLAB でさえごく僅な計算誤差はあり、過信は禁物である。例えば、単位正弦波は  $\pm 1$  以内の値を持つが、MATLAB バージョン 5.2 までは

```
x=0.9999;
sin((1/(1-x)^6));
```

の計算をすると何と 10 の 24 乗になっていた。(現在のバージョン 5.3 では改良されたが。)

## 7. 閉ループ制御系

これまでに機体固有の応答を見てきたが、不安定であるからその操縦はパイロットにとって負担である。よって適切に制御装置を付加して安定性の増大や操縦性の向上をはかりたい。そのためパイロットの意思は制御器を通して機体に伝えられ、制御器は適切に操舵をしてパイロットを補助する。図 7.1 は一番基本的なフィードバック制御系の構造である。



### (1) 比例制御

比例制御は図 7.1 において、制御器が比例ゲイン  $K_p$  の場合である。出力誤差信号  $e$  がこの制御器に加わると、昇降舵は  $\delta = K_p \cdot e$  の関係により作動する。この比例制御では、制御入力  $\delta$  は現時刻の応答誤差  $e$  の大きさのみに依存し、過去の誤差の大きさや、現在どれ位の速さで誤差が変化しているかなどは考慮しない。今現在の誤差が大きければ舵角は大きく、逆に今現在の誤差が小さければ舵角も小さい。

角度の向きを定義する慣習上、正の昇降舵角は負のピッチ角を発生する。このため、VTOL の伝達関数を見ても分かるように、分子に負の符号がある。この不都合を除く方法には二つある。一つは正のピッチ角を発生する場合には常に負の規範入力を加えることにすることである。もう一つは規範入力は正とし、それからピッチ角を引いて誤差をとるために、図の負フィードバックを正フィードバックにする方法である。ここでは二番目の方法を用いてこの閉ループ系の伝達関数を求め、そのステップ応答を見てみよう。M ファイルを

```
% Proportional Control of VTOL
kP=0.1; % 比例ゲインを仮に 0.1 としておく。
T=feedback(kP*G,1,+1) % kP*G は順方向(左から右)のパス伝達関数、
% 1 は逆方向(右から左)のパス伝達関数。+1
% は正フィードバック、-1 は負フィードバック
% を表す。
step(G,T); % 開ループ系 G と閉ループ系 T のステップ応答。
```

とする。Gは不安定であり閉ループ系の応答が読み取りにくいので、上の最後の行を

```
subplot(121),step(G);    %開ループ系 Gのステップ応答。
subplot(122),step(T);    %閉ループ系 Tのステップ応答。
```

とした方がよいであろう。

比例ゲイン  $k_P$  を変化させると、どの様に閉ループ系のステップ応答が影響を受けるかを調べる。このとき、毎回プログラムを変更しなくて済むように先の  $k_P=0.1$ ;のラインを次で置き換えると便利である。

```
disp('Enter Proportional Gain') %disp でテキストをスクリーンに表示する。
kP=input('kP=')                %input で kPの値をキーボードから指定する。
```

## (2) 位相進み補償 (比例・微分制御)

比例制御は現在の誤差に比例した(誤差の大小に基づいた)制御動作であったが、誤差の変化速度に比例した制御動作が微分制御である。これにより、現在の誤差が小さくても誤差変化の度合いが大きければ、操舵を大きくとることができる。微分動作は比例動作と共に利用され  $C(s) = k_p + k_D s$  と表せるが、純粋な微分動作を実行する装置は存在せず、これを近似した

$$C(s) = \frac{k_p + k_D s}{as + 1}, \quad a < k_D / k_p \quad (1-7)$$

を利用する。位相進み補償器は一般に

$$C(s) = \frac{k(s+z)}{s+p}, \quad p > z \quad (1-8)$$

と表され、この  $k$ (ゲイン)、 $z$ (零点  $s = -z$ )、 $p$ (極  $s = -p$ ) の3つのパラメータを決定することが必要となる。そのためには第3週の実験で使用する設計ツールが役に立つが、ここでは例えば  $k=1$ 、 $p=5$ 、と固定して  $z$ のみを変化させる場合に閉ループ系のステップ応答がどう変化するかを3Dプロットで可視化する。この場合のMファイルは

```
%Phase Lead Control of VTOL
k=1;                %Cのゲイン
p=5;                %Cの極(s=-5)
denc=[1 p];        %Cの分母
for l=1:10          %ループ開始
    t=linspace(0,50,100); %時間ベクトル作成
    z=i/10;          %Cの零点(s=-0.1~-1)
    numc=k*[1 z];    %Cの分子
    C=tf(numc,denc); %Cの伝達関数
    TL=feedback(C*G,1,+1); %閉ループ系の伝達関数
    [y(:,l),t]=step(TL,t); %制御下のVTOL機ステップ応答。
                        %yは2次元並列。各列は各零点の値に対する応答。
                        %:は1から最後まで全要素を表す。
end                 %ループ終了

%mesh(y)           %3Dメッシュプロット(後で試すこと)
ribbon(t,y,0.1)    %3Dリボンプロット。0.1はリボンの幅
xlabel('Zero x (-10)') %x軸ラベル
```

```
ylabel('Time')           % y 軸ラベル  
zlabel('Step Response') % z 軸ラベル  
view(60,40)             %視点変更(横 60 度、上 40 度)
```

でよいだろう。3Dプロットにより制御器の零点が閉ループ系のステップ応答に与える影響が見やすくなるので、他にも幾つかのゲイン  $k$  と極  $p$  の値についても調べておこう。(上のプログラムに `input` を加えてスクリーンから値を入力できるようにすると便利であろう。)

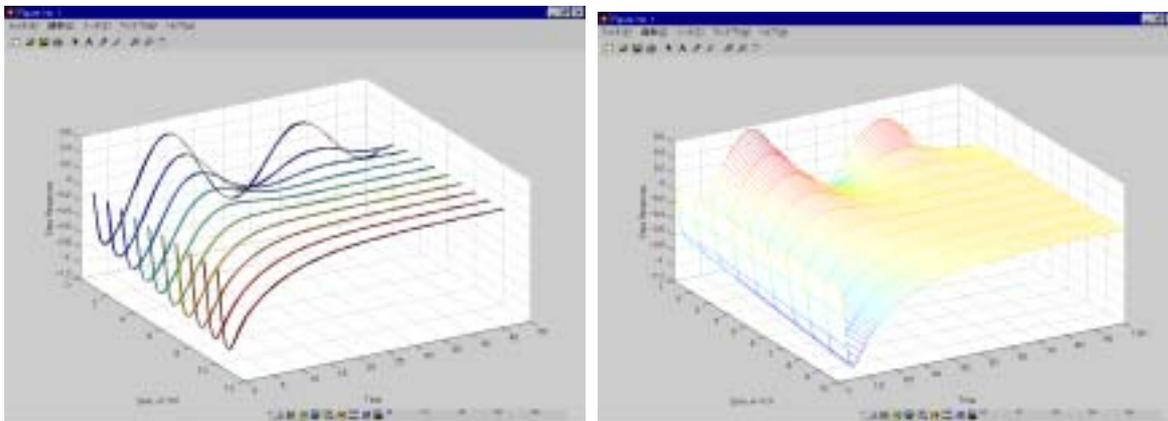


図 7.2 (a) リボンプロット

(b) メッシュプロット

## 8. 終了時：

一通り項目を終了したら教官あるいは TA にその旨を告げ、**口頭試問**を受ける。これにパスしたらメデタク終了となる。

各自が作成したファイルは Z ドライブあるいはディスクなど**に保管**してあることを確かめる。特に、3L204 で実験を行った場合は、各グループで作成したファイルをフロッピーなどにコピーして保管し、MATLABR11/WORK/のサブディレクトリーから消去すること。ファイルをセーブした後に正規の手続きによりからネットワークからログアウトあるいはパソコンをシャットダウンする。