
Các Lệnh MatLab
Trong Điều Khiển Tự Động
Visit: www.vietsupport.com
cover by thavali07

GIỚI THIỆU LÝ THUYẾT ĐIỀU KHIỂN TỰ ĐỘNG

Điều khiển tự động đóng vai trò quan trọng trong sự phát triển của khoa học và kỹ thuật. Lĩnh vực này hữu hiệu khắp nơi từ hệ thống phi thuyền không gian, hệ thống điều khiển tên lửa, máy bay không người lái, người máy, tay máy trong các quy trình sản xuất hiện đại, và ngay cả trong đời sống hàng ngày: điều khiển nhiệt độ, độ ẩm...

Phát minh đầu tiên khởi đầu cho việc phát triển của lĩng vực điều khiển tự động là bộ điều tốc ly tâm để điều chỉnh nhiệt độ máy hơi nước của Jame Watt năm 1874. Các công trình đáng chú ý trong bước đầu phát triển lý thuyết điều khiển là của các nhà khoa học Minorsky, Hazen, Nyquist...năm 1922. Minorky thực hiện hệ thống điều khiển tự động các con tàu và chứng minh tính ổn định của hệ thống có thể được xác định từ phương trình vi phân mô tả hệ thống. Năm 1932, Nyquist đã đưa ra một nguyên tắc tương đối đơn giản để xác định tính ổn định của hệ thống vòng kìn dựa trên cơ sở đáp ứng vòng hở đối với các tính hiệu vào hình sin ở trạng thái xác lập. Năm 1934, Hazen đã giới thiệu thuật ngữ điều chỉnh cơ tự động (servo mechanism) cho những hệ thống điều khiển định vị vàà thảo luận đến việc thiết kế hệ thống relay điều chỉnh động cơ với ngõ vào tín hiệu thay đổi.

Trong suốt thập niên 40 của thế kỷ 20 phương pháp đáp ứng tần số đã giúp cjo các kỹ sư thiết kế các hệ thống vòng kín tuyến tính thỏa các yêu cầu chất lượng điều khiển. Từ cuối thập niên 40 cho đến đầu thập niên 50 phương pháp quỹ đạo nghiệm của Evan được phát triển khá toàn vẹn.

Phương pháp quỹ đạo nghiệm và đáp ứng tần số được xem là cốt lõi của lý thuyết điều khiển cổ điển cho phép ta thiết kế được những hệ thống ổn định và thỏa các chỉ tiêu chất lượng điều khiển. Những hệ thống này được chấp nhận nhưng chưa phải là tối ưu, hoàn thiện nhất. Cho tới cuối thập niên 50 của thế kỷ 20 việc thiết kế một hay nhiều hệ thống dần dần được chuyển qua việc thiết kế một hệ thống tối ưu với ý nghĩa đầy đủ hơn.

Khi các máy móc hiện đại ngày càng phức tạp hơn với nhioều tín hiệu vào và ra thì việc mô tả hệ thống điều khiển hiện đại này đòi hỏi một lượng rất lớn các phương trình. Lý thuyết điều khiển cổ điển liên quan các hệ thống một ngõ vào và một ngõ ra trở nên bất lực để phân tích các hệ thống nhiều đầu vào, nhiều đầu ra. Kể từ khoảng năm 1960 trở đi nhờ máy tính sốcho phép ta phân tích các hệ thống phức tạp trong miền thời gian, lý thuyết điều khiển hiện đại phát triển để đối phó với sự phức tạp của các hệ thống hiện đại. Lý thuyết điều khiển hiện đại dựa trên phân tích trong miền thới gian và tổng hợp dùng các biến trạng thái, cho phép giải các bài toán điều khiển có các yêu cầu chặt chẽ về độ chính xác, trọng lượng và giá thành của các hệ thống trong lĩnh vực kỹ nghệ không gian và quân sự.

Sự phát triển gần đây của lý thuyết điều khiển hiện đại là trong nhiều lĩnh vực điểu khiển tối ưu của các hệ thống ngẫu nhiên và tiền định. Hiện nay máy vi tính ngày càng rẽ, gọn nhưng khả năng xử lý lại rất mạnh nên nó được dùng như là một phần tử trong các hệ thống điều khiển. Những áp dụng gần đây của lý thuyết điều khiển hiện đại vào ngay cả những ngành kỹ thuật như: sinh học, y học, kinh tế, kinh tế xã hội.

I. NHỮNG KHÁI NIỆM CƠ BẢN

1. <u>Điều khiển học (Cybernctics)</u>:

Là khoa học nghiên cứu những quá trình điều khiển và truyền thông máy móc, sinh vật và kinh tế. Điều khiển học mang đặc trưng tổng quát và được phân chia thành nhiều lĩnh vực khác nhau như: toán điều khiển, điều khiển học kỹ thuật, điều khiển học sinh vật (phỏng sinh vật: bionics), điều khiển học kinh tế.

2. <u>Lý thuyết điều khiển tự động</u>:

Là cơ sở lý thuyết của điều khiển học kỹ thuật. Điều khiển tự động là thuật ngữ chỉ quá trình điều khiển một đối tượng trong kỹ thuật mà không có sự tham gia của con người (automatic) nó ngược lại với quá trình điều khiển bằng tay (manual).

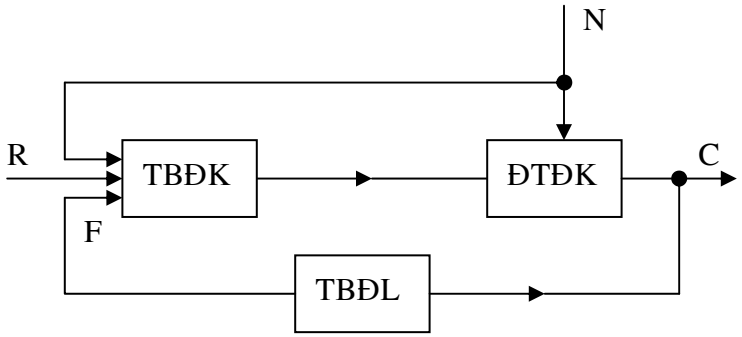
3. <u>Hệ thống điều khiển tự động</u>:

Một hệ thống điều khiển tự động bao gồm 3 phần chủ yếu:

Thiết bị điều khiển (TBĐK).

- Đối tượng điều khiển (ĐTĐK).
- Thiết bị đo lường.

Hình 1.1 là sơ đồ khối của hệ thống điều khiển tự động.



Hình 1.1

Trong đó:

C: tín hiệu cần điều khiển, thường gọi là tín hiệu ra (output).

U: tín hiệu điều khiển.

R: tín hiệu chủ đạo, chuẩn, tham chiếu (reference) thường gọi là tín hiệu vào (input).

N: tín hiệu nhiễu tác động từ bên ngoài vào hệ thống.

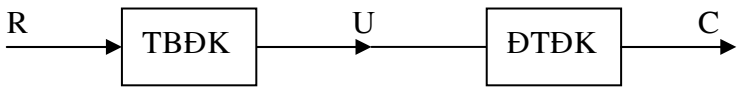
F: tín hiệu hồi tiếp, phản hồi (feedback).

4. <u>Hệ thống điều khiển kín (closed loop control system)</u>:

Là hệ htống điều khiển có phản hồi (feeback) nghĩa là tín hiệu ra được đo lường và đưa về thiết bị điều khiển. Tín hiệu hồi tiếp phối hợp với tín hiệu vào để tạo ra tín hiệu điều khiển. Hình 1.1 chính là sơ đồ của hệ thống kín. Cơ sở lý thuyết để nghiên cứu hệ thống kín chính là lý thuyết điều khiển tự động.

5. <u>Hệ thống điều khiển hở</u>:

Đối với hệ thống hở, khâu đo lường không được dùng đến. Mọi sự thay đổi của tín hiệu ra không được phản hồi về thiết bị điều khiển. Sơ đồ hình 1.2 là hệ thống điều khiển hở.



Hình 1.2: Hệ thống điều khiển hở

Cơ sở lý thuyết để nghiên cứu hệ thống hở là lý thuyết về relay và lý thuyết ôtômát hữu hạn.

II. PHÂN LOẠI HỆ THỐNG ĐIỀU KHIỂN TỰ ĐỘNG

Hệ thống điều khiển có thể phân loại bằng nhiều cách khác nhau. Sau đây là một số phương pháp phân loại:

1. Hệ tuyến tính và phi tuyến:

Có thể nói hầu hết các hệ thống vật lý đều là hệ phi tuyến, có nghĩa là trong hệ thống có ít nhất một phần tử là phần tử phi tuyến (quan hệ vào ra là quan hệ phi tuyến). Tuy nhiên, nếu phạm vi thay đổi của các biến hệ thống không lớn, hệ thống có thể được tuyến tính hóa trong phạm vi biến thiên của các biến tương đối nhỏ. Đối với hệ tuyến tính, phương pháp xếp chồng có thể được áp dụng.

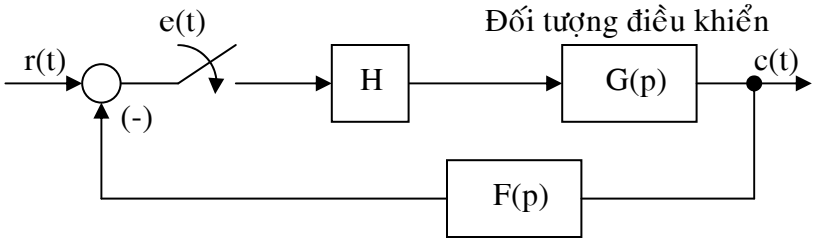
2. Hệ bất biến và biến thiên theo thời gian:

Hệ bất biến theo thời gian (hệ dừng) là hệ thống có các tham số không đổi (theo thời gian). Đáp ứng của các hệ này không phụ thuộc vào thời điểm mà tín hiệu vào được đặt vào hệ thống điều khiển phi thuyền không gian, với khối lượng giảm theo thời gian do tiêu thụ năng lượng trong khi bay.

3. Hệ liên tục và gián đoạn theo thời gian:

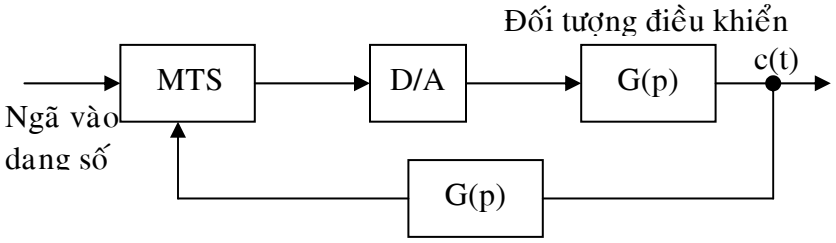
Trong hệ liên tục theo thìi gian, tất cả các biến là hàm liên tục theo thời gian. Công cụ phân tích hệ thống liên tục là phép biến đổi Laplace hay Fourier. Tronh khi đó, hệ gián đoạn là hệ thống có ít nhất một tín hiệu là hàm gián đoạn theo thời gian. Người ta phân biệt hệ thống gián đoạn gồm:

- Hệ thống xung: là hệ thống mà trong đó có một phần tử xung (khóa đóng ngắt) hay là tín hiệu được lấy mẫu (sample) và giữ (hold). (Hình 1.3)



Hình 1.3: Hệ thống điều khiển xung.

- Hệ thống số: là hệ thống gián đoạn trong đó tín hiệu được mã hóa thanh logic 1, 0. Đó là các hệ thống có các khâu biến đổi tương tự / số (A/D), số/ tương tự (D/A) và để kết nối kết nối tín hiệu với máy tính số. (Hình 1.4)



Hình 1.4: Hệ thống điều khiển số

Công cụ để phân tích hệ thống gián đoạn là phép biến đổi Laplace, Fourier gián đoạn hay phép biến đổi Z.

4. <u>Hệ đơn biến và đa biến</u>:

Hệ đơn biến là hệ chỉ có một ngõ vào và một ngõ ra. Công cụ để phân tích và tổng hợp hệ đơn biến là lý thuyết điều khiển cổ điển. Ví dụ: hệ điều khiển định vị (vị trí).

Hệ đa biến là hệ có nhiều ngõ vào và nhiều ngõ ra. Công cụ để phân tích và tổng hợp hệ đa biến là lý thuyết điều khiển hiện đại dựa trên cơ sở biểu diễn hệ trong không gian trạng thái. Ví dụ: hệ điều khiển quá trình (Process Control System) có thể gồm có điều khiển nhiệt độ và áp suất.

5. <u>Hệ thống thích nghi và hệ thống không thích nghi</u>:

Hệ thống thích nghi là hệ htống hoạt động theo nguyên tắc tự chỉnh định, trong đó hệ thống tự phát hiện những thay đổi của các tham số do ảnh hưởng của môi trường bên ngoài và thực hiện việc điều chỉnh tham số để đạt được chỉ tiêu tối ưu được đề ra.

6. <u>Hệ xác định (deterministic) và hệ ngẫu nhiên (stochastic)</u>:

Một hệ thống điều khiển là xác định khi đáp ứng đối với một ngõ vào nhất định có thể được biết trước (predictable) và có thể lặp lại được (repeatable). Nếu không thỏa mãn 2 điều kiện trên, hệ thống điều khiển là ngẫu nhiên.

III. NHIỆM VỤ CỦA LÝ THUYẾT ĐIỀU KHIỂN TỰ ĐỘNG

Để khảo sát và thiết kế một hệ thống điều khiển tự động người ta thực hiện các bước sau:

a) Dựa trên các yêu cầu thực tiễn, các mô hình vật lý ta xây dựng mô hình toán học dựa trên các quy luật, hiện tượng, quan hệ của các đối tượng vật lý. Mô hình toán học của hệ thống được xây dựng từ các mô hình toán học của các phần tử riêng lẻ.

b) Dựa trên lý thuyết ổn định, ta khảo sát tính ổn định của hệ thống. Nếu hệ thống không ổn định ta thay đổi đặc tính của hệ thống bằng cách đưa vào một khâu bổ chính (compensation) hay thay đổi thay đổi tham số của hệ để hệ thành ổn định.

c) Khảo sát chất lượng của hệ theo các chỉ tiêu đề ra ban đầu. Nếu hệ không đạt chỉ tiêu chất lượng ban đầu, ta thực hiện bổ chính hệ thống.

d) Mô phỏng hệ thống trên máy tính để kiểm tra lại thiết kế.

e) Thực hiện mô hình mẫu (prototype) và kiểm tra thiết kế bằng thực nghiệm.

f) Tinh chỉnh lại thiết kế để tối ưu hóa chỉ tiêu chất lượng và hạ thấp giá thành nếu có yêu cầu.

g) Xây dựng hệ thống thực tế.

TẬP LỆNH CƠ BẢN CỦA MATLAB

I. LỆNH CƠ BẢN

Chú ý: Các lệnh đều viết bằng chữ thường, nhưng vì tác giả muốn viết hoa để người xem tiện theo dõi.

1. Lệnh ANS

a) Công dụng: **(Purpose)**

Là biến chứa kết quả mặc định.

b) Giải thích: **(Description)**

Khi thực hiện một lệnh nào đó mà chưa có biến chứa kết quả, thì MATLAB lấy biến Ans làm biến chứa kết quả đó.

c) Ví dụ: **(Examples)**

```
2-1

ans = 1
```

2. Lệnh CLOCK

a) Công dụng: **(Purpose)**

Thông báo ngày giờ hiện tại.

b) Cú pháp:**(Syntax)**

```
c = clock
```

c) Giải thích: **(Description)**

Để thông báo dễ đọc ta dùng hàm fix.

d) Ví dụ: **(Examples)**

```
c = clock

c =

    1.0e+003*

    2.0010   0.0040   0.0200   0.0030   0.0420   0.0501

c = fix(clock)

c =   2001        4       20     3       43      3
```

3. Lệnh COMPUTER

a) Công dụng: **(Purpose)**

Cho biết hệ điều hành của máy vi tính đang sử dụng Matlab.

b) Cú pháp: **(Syntax)**

computer

[c,m] = computer

c) Giải thích: **(Description)**

c: chứa thông báo hệ điều hành của máy.

m: số phần tử của ma trận lớn nhất mà máy có thể làm việc được với Matlab.

d) Ví dụ: **(Examples)**

» [c,m]=computer

c =

PCWIN

m =

2.1475e+009

4. Lệnh DATE

a) Công dụng: **(Purpose)**

Thông báo ngày tháng năm hiện tại

b) Cú pháp: **(Syntax)**

s = date

c) Ví dụ:

» s=date

s =

20-Apr-2001

5. Lệnh CD

a) Công dụng:

Chuyển đổi thư mục làm việc.

b) Cú pháp:

cd

cd diretory

cd ..

c) Giải thích:

cd: cho biết thư nục hiện hành.

diretory: đường dẫn đến thư mục muốn làm việc.

cd .. chuyển đến thư mục cấp cao hơn một bậc.

6. Lệnh CLC

a) Công dụng:

Xóa cửa sổ lệnh.

b) Cú pháp:

clc

c) Ví dụ:

clc, for i: 25, home, A = rand(5), end.

7. Lệnh CLEAR

a) Công dụng:

Xóa các đề mục trong bộ nhớ.

b) Cú pháp:

clear

clear name

clear name1 name2 name3

clear functions

clear variables

clear mex

clear global

clear all

c) Giải thích:

clear: xóa tất cả các biến khỏi vùng làm việc.

clear name: xóa các biến hay hàm được chỉ ra trong name.

clear functions: xóa tất cả các hàm trong bộ nhớ.

clear variables: xóa tất cả các biến ra khỏi bộ nhớ.

clear mex: xóa tất cả các tập tin .mex ra khỏi bộ nhớ.

clear: xóa tất cả các biến chung.

clear all: xóa tất cả các biến, hàm, và các tập tin .mex khỏi bộ nhớ. Lệnh này làm cho bộ nhớ trống hoàn toàn.

8. Lệnh DELETE

a) Công dụng:

Xóa tập tin và đối tượng đồ họa.

b) Cú pháp:

delete filename

delete (n)

c) Giải thích:

file name: tên tập tin cần xóa.

n: biến chứa đối tượng đồ họa cần xóa. Nếu đối tượng là một cửa sổ thì cửa sổ sẽ đóng lại và bị xóa.

9. Lệnh DEMO

a) Công dụng:

Chạy chương trình mặc định của Matlab.

b) Cú pháp:

demo

c) Giải thích:

demo: là chương trình có sẵn trong trong Matlab, chương trình này minh họa một số chức năng của Matlab.

10. Lệnh DIARY

a) Công dụng:

Lưu vùng thành file trên đĩa.

b) Cú pháp:

diary filename

c) Giải thích:

filename: tên của tập tin.

11. Lệnh DIR

a) Công dụng:

Liệt kê các tập tin và thư mục.

b) Cú pháp:

dir

dir name

c) Giải thích:

dir: liệt kê các tập tin và thư mục có trong thư mục hiện hành.

dir name: đường dẫn đến thư mục cần liệt kê.

12. lệnh DISP

a) Công dụng:

Trình bày nội dung của biến (x) ra màn hình

b) Cú pháp:

disp (x)

c) giải thích:

x: là tên của ma trận hay là tên của biến chứa chuỗi ký tự, nếu trình bày trực tiếp chuỗi ký tự thì chuỗi ký tự được đặt trong dấu ‘’

d) Ví dụ:

```
» num=('Matlab')

num =

Matlab

» disp(num)
Matlab
» num=[2 0 0 1]

num =

   2    0    0    1

» disp(num)
```

2 0 0 1

» num='PHAM QUOC TRUONG'

num =

PHAM QUOC TRUONG

13. Lệnh ECHO

a) Công dụng:

Hiển thị hay không hiển thị dòng lệnh đang thi hành trong file *.m.

b) Cú pháp:

echo on

echo off

c) Giải thích:

on: hiển thị dòng lệnh.

off: không hiển thị dòng lệnh.

14. Lệnh FORMAT

a) Công dụng:

Định dạng kiểu hiển thị của các con số.

Cú pháp	Giải thích	Ví dụ
Format short	Hiển thị 4 con số sau dấu chấm	3.1416
Format long	Hiển thị 14 con số sau dấu chấm	3.14159265358979
Format rat	Hiển thị dạng phân số của phần nguyên nhỏ nhất	355/133
Format +	Hiển thị số dương hay âm	+

15. Lệnh HELP

a) Công dụng:

hướng dẫn cách sử dụng các lệnh trong Matlab.

b) Cú pháp:

help

help topic

c) Giải thích:

help: hiển thị vắn tắt các mục hướng dẫn.

topic: tên lệnh cần được hướng dẫn.

16. Lệnh HOME

a) Công dụng:

Đem con trỏ về đầu vùng làm việc.

b) Cú pháp:

home

17. Lệnh LENGTH

a) Công dụng:

Tính chiều dài của vectơ.

b) Cú pháp:

l = length (x)

c) Giải thích:

l: biến chứa chiều dài vectơ.

d) Ví dụ:

tính chiều dài của vectơ x.

x = [0 1 2 3 4 5 6 7 8 9]

l = length (x)

l = 10

```
» x=[01 09 77,20 04 2001 ]


x =


     1     9    77    20     4  2001


» l=length(x)
```

```
l =


     6
```

18. Lệnh LOAD

a) Công dụng:

Nạp file từ đĩa vào vùng làm việc.

b) Cú pháp:

load

load filename

load filename

load finame.extension

c) Giải thích:

load: nạp file matlap.mat

load filename: nạp file filename.mat

load filename.extension: nạp file filename.extension

Tập tin này phải là tập tin dạng ma trận có nghĩa là số cột của hàng dưới phải bằng số cột của hàng trên. Kết quả ta được một ma trận có số cột và hàng chính là số cột và hàng của tập tin văn bản trên.

19. Lệnh LOOKFOR

a) Công dụng:

Hiển thị tất cả các lệnh có liên quan đến topic.

b) Cú pháp:

lookfor topic

c) Giải thích:

topic: tên lệnh cần được hướng dẫn.

20. Lệnh PACK

a) Công dụng:

Sắp xếp lại bộ nhớ trong vùng làm việc.

b) Cú pháp:

pack

pack filename

c) Giải thích:

Nếu như khi sử dụng Matlap máy tính xuất hiện thông báo "Out of memory" thì lệnh pack có thể tìm thấy một số vùng nhớ còn trống mà không cần phải xóa bớt các biến.

Lệnh pack giải phóng không gian bộ nhớ cần thiết bằng cách nén thông tin trong vùng nhớ xuống cực tiểu. Vì Matlab quản lý bộ nhớ bằng phương pháp xếp chồng nên các đoạn chương trình Matlab có thể làm cho vùng nhớ bị phân mảnh. Do đó sẽ có nhiều vùng nhớ còn trống nhưng không đủ để chứa các biến lớn mới.

Lệnh pack sẽ thực hiện:

+ lưu tất cả các biến lên đĩa trong một tập tin tạm thời là pack.tmp.

+ xóa tất cả các biến và hàm có trong bộ nhớ.

+ lấy lại các biến từ tập tin pack.tmp.

+ xóa tập tin tạm thời pack.tmp.

kết quả là trong vùng nhớ các biến được gộp lại hoặc nén lại tối đa nên không bị lãng phí bộ nhớ.

Pack.finame cho phép chọn tên tập tin tạm thời để chứa các biến. Nếu không chỉ ra tên tập tin tạm thời thì Matlab tự lấy tên tập tin đó là pack.tmp.

Nếu đã dùng lệnh pack mà máy vẫn còn báo thiếu bộ nhớ thì bắt buộc phải xóa bớt các biến trong vùng nhớ đi.

21. Lệnh PATH

a) Công dụng:

Tạo đường dẫn, liệt kê tất cả các đường dẫn đang có.

b) Cú pháp:

path

p = path

path (p)

c) Giải thích:

path: liệt kê tất cả các dường dẫn đang có.

p: biến chứa đường dẫn.

path (p): đặt đường dẫn mới.

d) Ví dụ:

đặt đường dẫn đến thư mục c:\lvtn\matlab

p = 'd:\DA\matlab';

path (p);

22. Lệnh QUIT

a) Công dụng:

Thoát khỏi Matlab.

b) Cú pháp:

quit

23. Lệnh SIZE

a) Công dụng:

Cho biết số dòng và số cột của một ma trận.

b) Cú pháp:

d = size (x)

[m,n] = size (x)

m = size (x,1)

n = size (x,2)

c) Giải thích:

x: tên ma trận.

d: tên vectơ có 2 phần tử, phần tử thứ nhất là số dòng, phần tử còn lại là số cột.

m,n: biến m chứa số dòng, biến n chứa số cột

d) Ví dụ:

ta có ma trận a

x = 1 2 3 4

5 6 6 8

» x=[1 2 3 4,5 6 7 8]

x =

1 2 3 4 5 6 7 8

Các bạn chú ý về cách nhập 1 ma trận:

» x=[1 2 3 4;5 6 7 8]

x =

```
    1     2     3     4
    5     6     7     8

» d=size(x)

d =

    2     4

» m=size(x,1)

m =

    2

» n=size(x,2)

n =

    4

» [m,n]=size(x)

m =

    2


n =
```

4

24. Lệnh TYPE

a) Công dụng:

Hiển thị nội dung của tập tin.

b) Cú pháp:

type filename

c) Giải thích:

filename: tên file cần hiển thị nội dung.

Lệnh này trình bày tập tin được chỉ ra.

25. Lệnh WHAT

a) Công dụng:

Liệt kê các tập tin *.m, *.mat, *.mex.

b) Cú pháp:

what

what dirname

c) Giải thích:

what: liệt kê tên các tập tin .m, .mat, .mex có trong thư mục hiện hành.

dirname: tên thư mục cần liệt kê.

26. Lệnh WHICH

a) Công dụng:

Xác định chức năng của funname là hàm của Matlab hay tập tin.

b) Cú pháp:

which funname

c) Giải thích:

funname: là tên lệnh trong Matlab hay tên tập tin

d) Ví dụ:

which inv

inv is a build-in function

which f

c:\matlab\bin\f.m

27. Lệnh WHO, WHOS

a) Công dụng:

Thông tin về biến đang có trong bộ nhớ.

b) Cú pháp:

who

whos

who global

whos global

c) Giải thích:

who: liệt kê tất cả các tên biến đang tồn tại trong bộ nhớ.

whos: liệt kê tên biến, kích thước, số phần tử và xét các phần ảo có khác 0 không.

who global và whos: liệt kê các biến trong vùng làm việc chung.

II. CÁC TOÁN TỬ VÀ KÝ TỰ ĐẶC BIỆT

1. Các toán tử số học (Arithmetic Operators):

Toán tử	Công dụng
+	Cộng ma trận hoặc đại lượng vô hướng (các ma trận phải có cùng kích thước).
-	Trừ ma trận hoặc đại lượng vô hướng (các ma trận phải có cùng kích thước).
*	Nhân ma trận hoặc đại lượng vô hướng (ma trận 1 phải có số cột bằng số hàng của ma trận 2).
.*	Nhân từng phần tử của 2 ma trận hoặc 2 đại lượng vô hướng (các ma trận phải có cùng kích thước).
\	Thực hiện chia ngược ma trận hoặc các đại lượng vô hướng (A\B tương đương với inv (A)*B).
.\	Thực hiện chia ngược từng phần tử của 2 ma trận hoặc 2 đại lượng vô hướng (các ma trận phải có cùng kích thước).
/	Thực hiện chia thuận 2 ma trận hoặc đại lượng vô hướng (A/B tương đương với A*inv(B)).
./	Thực hiện chia thuận từng phần tử của ma trận này cho ma trận kia (các ma trận phải có cùng kích thước).
^	Lũy thừa ma trận hoặc các đại lượng vô hướng.
.^	Lũy thừa từng phần tử ma trận hoặc đại lượng vô hướng (các ma trận phải có cùng kích thước).

* ví dụ:

Phép tính ma trận		Phép tính mảng	
x	1 2 3	y	4 5 6
x’	1 2 3	y’	4 5 6
x + y	5 6 7	x – y	-3 -3 -3

x + 2	3 4 5	x – 2	-3 -3 -3
x * y	phép toán sai	x. * y	4 10 18
x'* y	32	x'.* y	phép toán sai
x * y'	4 5 6 8 10 12 12 15 18	x. * y'	phép toán sai
x * 2	2 4 6	x.* 2	2 4 6
x \ y	16/7	x.\ y	4 5/2 2
2 \ x	1/2 1 3/2	2./ x	2 1 2/3
x / y	0 0 1/6 0 0 1/3 0 0 1/2	x./ y	1/4 2/5 1/2
x / 2	1/2 1 3/2	x./ 2	1/2 1 3/2
x ^ y	phép toán sai	x.^ y	1/2 32 729
x ^ 2	phép toán sai	x.^ 2	1 4

	9
2 ^ x phép toán sai	2.^ x 2 4 8

2.. <u>Toán tử quan hệ (Relational Operators)</u>:

Toán tử	Công dụng
<	So sánh nhỏ hơn.
>	So sánh lớn hơn.
>=	So sánh lớn hơn hoặc bằng.
<=	So sánh nhỏ hơn hoặc bằng.
= =	So sánh bằng nhau cả phần thực và phần ảo.
-=	So sánh bằng nhau phần ảo.

a) <u>Giải thích</u>:

Các toán tử quan hệ thực hiện so sánh từng thành phần của 2 ma trận. Chúng tạo ra một ma trận có cùng kích thước với 2 ma trận so sánh với các phần tử là 1 nếu phép so sánh là đúng

và là 0 nếu phép so sánh là sai.

Phép so sánh có chế độ ưu tiên sau phép toán số học nhưng trên phép toán logic.

b) <u>Ví dụ</u>:

thực hiện phép so sánh sau:

» x=5 % đầu tiên ta nhập x=5

x =

5

» x>=[1 2 3;4 5 6;7 8 9] %so sánh trực tiếp x (x là 5) với ma trận

```
ans =                    % rõ ràng các phầ tử 1,2,3,4,5 đều <= 5

    1     1     1
    1     1     0
    0     0     0

» x=5

x =

    5

» A=[1 2 3;4 5 6;7 8 9]    % ta đặt ma trận A

A =

    1     2     3
    4     5     6
    7     8     9

» x>=A

ans =

    1     1     1
    1     1     0
    0     0     0

» x=A          % dòng lệnh này tức là cho x= ma trận A

x =
```

```
    1    2    3
    4    5    6
    7    8    9

» x==A          % so sánh x và A

ans =                % tất cả các phần tử đều đúng

    1    1    1
    1    1    1
    1    1    1

» x=5          % cho lại x=5

x =

    5

» x==A    % so sánh x = A

ans =

    0    0    0
    0    1    0        % chỉ duy nhất phần tử  5=x (vì x=5)
    0    0    0

» x<A

ans =
```

0 0 0

0 0 1

1 1 1

3. Toán tử logig (Logical Operators):

Toán tử	Công dụng
&	Thực hiện phép toán logic AND.
|	Thực hiện phép toán logic OR.
~	Thực hiện phép toán logic NOT.

a) Giải thích:

Kết quả của phép toán là 1 nếu phép logic là đúng và là 0 nếu phép logic là sai.

Phép logic có chế độ ưu tiên thấp nhất so với phép toán số học và phép toán so sánh.

b) Ví dụ:

Khi thực hiện phép toán 3>4 & 1+ thì máy tính sẽ thực hiện 1+2 được 3, sau đó tới 3>4 được 0 rồi thực hiện 0 & 3 và cuối cùng ta được kết qủa là 0.

4. Ký tự đặc biệt (Special Characters):

Ký hiệu	Công dụng
[]	Khai báo vector hoặc ma trận.
()	Thực hiện phép toán ưu tiên, khai báo các biến và các chỉ số của vector.
=	Thực hiện phép gán.
‘	Chuyển vị ma trận tìm lượng liên hiệp của số phức.
.	Điểm chấm thập phân.
,	Phân biệt các phần tử của ma trận và các đối số trong dòng lệnh.
;	Ngăn cách giữa các hàng khi khai báo ma trận.
%	Thông báo dòng chú thích.
!	Mở cửa sổ MS – DOS.

5. dấu ':'

a) Công dụng:

Tạo vector hoặc ma trận phụ và lặp đi lặp lại các giá trị.

b) Giải thích:

Khai báo	Công dụng
j : k	Tạo ra chuỗi j, j+1, j+2,...., k-1, k
j : i : k	Tạo ra chuỗi j, j+i, j+2I,....,k-i, k
A(: , j)	Chỉ cột thứ j của ma trận A
A(i , :)	Chỉ hàng thứ i của ma trận
A(: , :)	Chỉ toàn bộ ma trận A
A(j , k)	Chỉ phần tử A(j), A(j+1)...A(k)
A(: , j , k)	Chỉ các phần tử A(:, j), A(:, j+1)...A(:, k)
A(:)	Chỉ tất cả các thành phần của ma trận A

c) Ví dụ:

khi khai báo D = 1 : 10

ta được kết quả:

D = 1 2 3 4 5 6 7 8 9 10

còn khi khai báo D = 0 : 2 :10

thì ta được kết quả:

D = 0 2 4 6 8 10

III. CÁC HÀM LOGIC (LOGICAL FUNCTION)

1. Lệnh ALL

a) Công dụng:

Kiểm tra vector hay ma trận có giá trị 0 hay không.

b) Cú pháp:

y = all(x)

c) Giải thích:

y: biến chứa kết quả

x: tên vedtor hay ma trận

y = 1 khi tất cả các phần tử khác 0

y = 0 khi có 1 phần tử bằng 0

d) Ví dụ:

```
» a=[1 2 3]

a =

    1     2     3

» y=all(a)

y =

    1

» a=[1 0 3]

a =

    1     0     3

» y=all(a)
```

```
y =

    0

» a=[1 2 3;4 0 6;7 8 9]

a =

    1    2    3
    4    0    6
    7    8    9

» y=all(a)

y =

    1    0    1

» a=[1 2 0;0 3 5;2 6 8]

a =

    1    2    0
    0    3    5
    2    6    8

» y=all(a)

y =
```

0 1 0

2. Lệnh ANY

a) Công dụng:

Kiểm tra vector hay ma trận có giá trị khác 0 hay không.

b) Cú pháp:

y = any(x)

c) Giải thích:

y: biến chứa kết quả.

x: tên vector, hay ma trận.

y = 1 khi có 1 phần tử khác 0.

y = 0 khi có 1 phần tử bằng 0.

d) Ví dụ:

```
» a=[1 2 3];
» y=any(a)

y =

    1

» b=[1 0 3 0];
» y=any(b)

y =

    1

» c=[1 2 0 4;0 2 0 4;1 2 3 4;3 4 5 6]

c =

    1    2    0    4
```

```
    0     2     0     4
    1     2     3     4
    3     4     5     6

» y=any(c)

y =

    1     1     1     1

» d=[0 0 0 0;0 1 3 0]

d =

    0     0     0     0
    0     1     3     0

» y=any(d)

y =

    0     1     1     0
```

3. Lệnh EXIST

a) Công dụng:

Kiểm tra biến hay file có tồn tại hay không.

b) Cú pháp:

e = exist('item')

c) Giải thích:

item: là tên file hay tên biến.

e: biến chứa giá trị trả về.

e	Y nghĩa

0	item không tồn tại trong vùng làm việc
1	item là biến đang tồn tại trong vùng làm việc
2	item đang tồn tại trên đĩa (chỉ kiểm tra trong thư mục hiện hành)
3	item là MEX-file
4	item là file được dịch từ phần mềm Simulink
5	item là hàm của Matlab

d) Ví dụ:

e = exist('dir')

e = 5

4. Lệnh FIND

a) Công dụng:

Tìm phần tử trong vector hay ma trận theo yêu cầu.

b) Cú pháp:

k = find(x)

[i,j] = find(x)

[i,j,s] = find(x)

c) Giải thích:

k: chỉ vị trí của phần tử cần tìm trong vector.

i,j: chỉ số hàng và số cột tương ứng của phần tử cần tìm.

s: chứa giá trị của phần tử cần tìm.

x: tên vector, ma trận hay là yêu cầu đề ra. Nếu không nêu ra yêu cầu thì mặc nhiên là tìm các phần tử khác 0.

d) Ví dụ:

» x=[1 8 0 2 3 0]

x =

1 8 0 2 3 0

» k=find(x)

```
k =

    1     2     4     5

» k=[3 6]

k =

    3     6

» a=[5 0 0;8 0 3]

a =

    5     0     0
    8     0     3

» [i,j,k]=find(a)

i =

    1
    2
    2


j =

    1
    1
```

```
    3


k =


    5
    8
    3
```

IV. NHÓM LỆNH LẬP TRÌNH TRONG MATLAB

1. Lệnh EVAL

a) Công dụng:

Chuyển đổi chuỗi ký tự thành biểu thức.

b) Cú pháp:

kq = eval(‘string’)

c) Giải thích:

kq: biến chứa kết quả.

Nếu ‘string’ là các ký số thì chuyển thành những con số.

Nếu ‘string’ là câu lệnh thì chuyển thành các lệnh thi hành được.

d) Ví dụ:

```
» a='199999999';
» eval(a)+1


ans =


  200000000
```

2. Lệnh FOR

a) Công dụng:

Dùng để thực hiện 1 công việc cần lặp đi lặp lại theo một quy luật, với số bước lặp xác định trước.

b) Cú pháp:

for biến điều khiển = giá trị đầu : giá trị cuối,

thực hiện công việc;

end

c) Giải thích:

Công việc chính là các lệnh cần thi hành, có thể có nhiều lệnh, kết thúc lệnh phải có dấu;

d) Ví dụ:

In ra màn hình 5 dòng 'PHAM QUOC TRUONG chao cac ban'.

```
for i = 1:5,
    disp('PHAM QUOC TRUONG chao cac ban');
end
```

```
PHAM QUOC TRUONG chao cac ban
PHAM QUOC TRUONG chao cac ban
PHAM QUOC TRUONG chao cac ban
PHAM QUOC TRUONG chao cac ban
PHAM QUOC TRUONG chao cac ban
```

3. Lệnh FUNCTION

a) Công dụng:

Tạo thêm hàm mới.

b) Cú pháp:

```
function s = n(x)
```

c) Giải thích:

s: tên biến chứa giá trị trả về sau khi thi hành hàm.

n: tên gợi nhớ.

d) Ví dụ: (ở phần lập trong M.file)

4. Lệnh INPUT

a) Công dụng:

Dùng để nhập vào 1 giá trị.

b) Cú pháp:

tên biến = input (‘promt’)

tên biến = input (‘promt’, ‘s’)

c) Giải thích:

tên biến, là nơi lưu giá trị ngập vào.

‘promt’: chuỗi ký tự muốn nhập vào.

‘s’: cho biết giá trị nhập vào là nhiều ký tự.

d) Ví dụ1:

x = input(‘nhập giá trị của biến x: ’)

nhập giá trị của biến x: 5

x = 5

e) Ví dụ2:

trả_lời = input(‘bạn có muốn tiếp tục không ? ’,’s’)

bạn có muốn tiếp tục không ? không

trả_lời = không

5. Lệnh IF ...ELSEIF ...ELSE

a) Công dụng:

Thực hiện lệnh khi thỏa điều kiện.

b) Cú pháp:

```
if biểu thức luận lý 1
    thực hiện công việc 1;
elseif biểu thức luận lý 2
    thực hiện công việc 2;
else
    thực hiện công việc 3;
end
```

c) Giải thích:

Khi biểu thức luận ký 1 đúng thì thực hiện công việc 1 tương tự cho biểu thức luận lý 2. Nếu cả hai biểu thức sai thì thực hiện công việc sau lệnh else.

Biểu thức luận lý là các phép so sánh ==, <, >, <=, >=

công việc chính là các lệnh cần thi hành, có thể có nhiều lệnh, kết thúc lệnh phải có dấu ;

d) Ví dụ:

Viết chương trình nhập vào 2 số và so sánh hai số đó.

```
a = input('Nhập a: ');
b = input('Nhập b: ');
if a > b
    disp('a lớn hơn b');
elseif a ==b
    disp('a bằng b');
else
    disp('a nhỏ hơn b');
end
```

```
nhập a: 4
nhập b: 5
a nhỏ hơn b
```

6. Lệnh MENU

a) Công dụng:

Tạo menu để chọn chức năng.

b) Cú pháp:

tên biến = menu ('Tên menu','chức năng1','chức năng2', ... , 'chức năng n')

c) Giải thích:

tên menu: là tiêu đề của menu.

tên biến: là nơi cất giá trị nhận được sau khi chọn chức năng của menu.

Chức năng 1, 2, ...,n:khi chọn chức năng nào thì tên biến có giá trị là số thứ tự của chức năng đó.

d) Ví dụ:

```
k = menu('Choose a color', 'Red', 'Blue', 'Green')
```

```
---- Choose a color ----
    1) Red
    2) Blue
    3) Green
```

7. Lệnh PAUSE

a) Công dụng:

Dừng chương trình theo ý muốn.

b) Cú pháp:

pause on

pause off

pause (n)

c) Giải thích:

pause on: dừng chương trình, và chờ nhấn 1 phím bất kỳ (trừ các phím điều khiển) chương trình thực hiện tiếp.

pause off: tắt chức năng pause.

pause (n): dừng chương trình tại n giây.

d) Ví dụ:

```
for n = 1 : 3;
    disp('Press any key to continue...')
    pause
end
```

```
Press any key to continue...
Press any key to continue...
Press any key to continue...
```

8. Lệnh WHILE

a) Công dụng:

Dùng để thực hiện 1 công việc cần lặp đi lặp lại theo một quy luật, với số bước lặp không xác định, phụ thuộc vào biểu thức luận lý.

b) Cú pháp:

```
while biểu thức luận lý
    thực hiện công việc;
end
```

c) Giải thích:

Biểu thức luận lý là các phép so sánh = =, <, >, <=, >=

Công việc chính là các lệnh cần thi hành, có thể có nhiều lệnh, kết thúc lệnh phải có dấu ;

Khi thực hiện xong công việc thì quay lên kiểm tra lại biểu thức luận lý, nếu vẫn còn đúng thì tiếp tục thực hiện, nếu sai thì kết thúc.

d) Ví dụ:

tính tổng A = 1+1/2+1/3+...+1/n

n = input('nhập vào số n ');

a = 0; i = 1

while i <= n

a = a + 1/i

i = i + 1;

end

disp('ket qua');

disp(a);

nhap vao so n 3

ket qua

1.8333

B1(BT4a): Viết chương trình nhập vào một số n(n>=0)

với các trường hợp sau:

a) Nếu n<0 thì in thông báo bạn nhập sai
b) Nếu n>0 và lẽ thì tính tổng s1=1+3+5+...+n,n là số lẽ.
c) Nếu n>0 và chẵn thì s2=2+4+6+...+n,n chẵn.
d) Nếu n=0 dừng chương trình lại.

```
% BT4a: Viet chuong trinh nhap vao mot so n(n>=0)
%  voi cac truong hop sau:
%  a) Neu n<0 thi in thong bao ban nhap sai
%  b) Neu n>0 va le thi tinh tong s1=1+3+5+...+n,n la so le.
%  c) Neu n>0 va chan thi s2=2+4+6+...+n,n chan.
%  d) Neu n=0 dung chuong trinh lai.
n=input('nhap n= ');           %nhap so n
du=rem(n,2);                   %kiem tra n la le hay chan
                               %neu n le du=1, n chan du=0
if n<0
   fprintf('Ban nhap sai')     %xuat ra thong bao
end
if (n>0) & (du==1)             %neu n>0 va le
   i=1;                        %gan i=1;
   s1=1;                       %gan tong s1=1
   while i<n                   %thuc hien vong lap
       i=i+2;                  %tang i len 2 sau moi lan lap
       s1=s1+i;                %tinh tong s1 voi gia tri i moi
   end
   s1                 %in ra ket qua sau khi ket thuc vong lap
end
```

```
if (n>0) & (du==0)
   i=0;
   s2=0;
   while i<n
      i=i+2;
      s2=s2+i;
   end
   s2
 end
 if n==0                %neu n=0
      break             %lenh ket thuc
 end
```

Khi chạy chương trình:

```
» nhap n= 5

s1 =

     9


» BT4a
nhap n= 4

s2 =

     6

» BT4a
nhap n= -6
Ban nhap sai» BT4a
nhap n= 0
»
```

V. TẬP LỆNH XỬ LÝ CHUỖI

1. Lệnh ABS

a) Công dụng:

Tạo vector đơn có giá trị của mỗi phần tử là số thứ tự tương ứng với ký tự trong bảng mả ASCII.

Lấy trị tuyệt đối của một số âm.

b) Cú pháp:

n = ABS(s)

x = ABS(a)

c) Giải thích:

n: tên vector.

s: chuỗi ký tự, hoặc là tên biến chứa chuỗi ký tự.

a: số âm, hoặc là tên biến chứa số âm.

x: trị tuyệt đối của a.

d) Ví dụ:

```
» n=abs('PHAM QUOC TRUONG')


n =


 Columns 1 through 12


   80    72    65    77    32    81    85    79    67    32    84    82


 Columns 13 through 16


   85    79    78    71


» m=abs('MATLAB')


m =
```

```
   77    65    84    76    65    66

» U=abs('abc')

U =

   97    98    99
» T=abs(-1)

T =

    1
```

2. Lệnh BLANKS

a) Công dụng:

Tạo khoảng trắng giữa hai hay nhiều chuỗi ký tự theo mong muốn.

b) Cú pháp:

[S1 BLANKS(b1) S2 BLANKS(b2) ...BLANKS(bn) Sn]

c) Giải thích:

S1, S2, ...Sn: các chuỗi ký tự.

b1, b2: số khoảng trắng.

d) Ví dụ:

In 4 chuỗi 'Khao sat','ứng dụng', 'MATLAB', 'trong điều khiển tự động' ra màn hình với khoảng cách lần lượt giữa 4 chuỗi là: 2,4,3

```
» S=['Khao sat'blanks(2) 'ung dung'blanks(4) 'MATLAB'blanks(3) 'trong dieu khien tu dong']

S =

Khao sat  ung dung    MATLAB   trong dieu khien tu dong
```

3. Lệnh DEC2HEX

a) Công dụng:

Đổi con số của hệ 10 sang hệ 16.

b) Cú pháp:

s = dec2hex(n)

c) Giải thích:

s: biến chứa chuỗi ký số của hệ 16

n: con số nguyên hệ 10.

d) Ví dụ:

s = dec2hex(10)

s = 'A'

4. Lệnh HEX2DEC

a) Công dụng:

Đổi chuỗi ký số của hệ 16 sang con số của hệ 10.

b) Cú pháp:

n = hex2dec('s')

c) Giải thích:

n: con số của hệ 10.

s: chuỗi ký số hệ 16.

d) Ví dụ:

n = hex2dec('A')

n = 10

5. Lệnh INT2STR

a) Công dụng:

Chuyển số nguyên sang dạng chuỗi.

Chuyển các ký tự trong một chuỗi sang số thứ tự tương ứng trong bảng mã ASCII.

b) Cú pháp:

kq = INT

c) Giải thích:

kq: biến STR(n)chứa kết quả.

n: tên biến cần chuyển.

Nếu n là số nguyên thì kq là chuỗi ký số.

Nếu n là chuỗi ký tự thì kq là số tương ứng trong bảng mã ASCII

d) Ví dụ:

```
» n='MATLAB'

n =

MATLAB

» t=int2str(n)

t =

77  65  84  76  65  66

» n=2001

n =

     2001

» t=int2str(n)

t =

2001
```

6. Lệnh ISSTR

a) Công dụng:

Kiểm tra nội dung biến có phải là chuỗi ký tự không.

b) Cú pháp:

kq = isstr(n)

c) Giải thích:

kq: biến chứa kết quả.

n: tên biến cần kiểm tra.

kq = 1 nếu n là chuỗi ký tự.

0 nếu n không là chuỗi ký tự.

d) Ví dụ:

```
» n='MATLAB';
» kq=isstr(n)

kq =

    1
» m=[1 2 3 4];
» kq=isstr(m)

kq =

    0
```

7. Lệnh LOWER

a) Công dụng:

Cho ra chuỗi ký tự viết thường.

b) Cú pháp:

b = lower(s)

c) Giải thích:

b: biến chứa kết quả.

s: tên biến chứa chuỗi ký tự hay chuỗi ký tự.

d) Ví dụ:

```
» a='DO AN cua pHAm quOC TRuOnG';
» b=lower(a)

b =
```

do an cua pham quoc truong

8. Lệnh NUM2STR

a) Công dụng:

Chuyễn số thực sang dạng chuỗi.

Chuyển các ký tự trong một chuỗi sang số thứ tự tương ứng trong bảng mã ASCII.

b) Cú pháp:

kq = num2tr(n)

c) Giải thích:

kq: biến chứa kết quả.

n: tên biến cần chuyển.

Nếu n là số thực thì kq là số tương ứng trong bảng mã ASCII.

d) Ví dụ:

» n=3.1416;

» kq=num2str(n)

kq =

3.1416

9. Lệnh SETSTR

a) Công dụng:

Cho ra ký tự tương ứng với số thứ tự trong bảng mã ASCII.

b) Cú pháp:

x = Set Str(n)

c) Giải thích:

x: biến chứa ký tự tương ứng (thuộc bảng mã ASCII).

n: số nguyên ($0 \le n \le 255$).

d) Ví dụ:

Tìm ký tự có số thứ tự là 65 trong bảng mã ASCII.

» kt=setstr(65)

kt =

A

10. Lệnh STR2MAT

a) Công dụng:

Tạo ma trận có các phần tử dạng chuỗi.

b) Cú pháp:

s = str2mat('s1', 's2', ...)

c) Giải thích:

s: tên ma trận kết quả.

s1, s2: chuỗi ký tự.

d) Ví dụ:

s = str2mat('mat', 'lab')

s =

mat

lab

11. Lệnh STR2NUM

a) Công dụng:

Chuyển chuỗi (dạng số) sang số thực.

b) Cú pháp:

n = str2num(s)

c) Giải thích:

s: chuỗi dạng số.

n: số thực.

d) Ví dụ:

n = str2num('456456')

n = 456456

12. Lệnh STRCMP

a) Công dụng:

So sánh 2 chuỗi ký tự.

b) Cú pháp:

l = strcmp(s1, s2)

c) Giải thích:

l: biến chứa kết quả.

s1, s2: chuỗi cần so sánh.

d) Ví dụ:

a = 'MatLab WoRkS'

b = 'MatLab WoRkS'

strcmp(a,b)

ans = 1

13. Lệnh UPPER

a) Công dụng:

Cho ra chuỗi viết hoa.

b) Cú pháp:

b = upper

c) Giải thích:

b: biến chứa kết quả.

s: tên biến chứa chuỗi ký tự.

d) Ví dụ:

a = 'MaTlab WORks'

b = upper(a)

b = MATLAB

b = upper('MaTlab WORks')

b= MATLAB WORKS

VI. CÁC HÀM GIAO TIẾP

1. Lệnh FCLOSE

a) Công dụng:

Đóng file đang mở sau khi truy xuất xong.

b) Cú pháp:

fclose(fid)

c) Giải thích:

fid: tên biến trỏ đến file đang mở.

2. Lệnh FOPEN

a) Công dụng:

Mở file hoặc truy xuất dữ liệu của file đang mở.

b) Cú pháp:

fid = fopen('fn')

fid = fopen('fn', 'p')

c) Giải thích:

fid: tên biến trỏ đến file đang mở.

fn: tên file (có thể đặt đường dẫn).

Tham số p có các định dạng sau:

'r': chỉ đọc.

'r+': đọc và ghi.

'w': xóa tất cả nội dung của file hoặc tạo 1 file mới và mở file đó để ghi.

'w+': xóa tất cả nội dung của file hoặc tạo 1 file mới và mở file đó để ghi và đọc.

3. Lệnh FPRINTF

a) Công dụng:

Ghi đoạn dữ liệu thành file.

b) Cú pháp:

fprintf(fid, f)

c) Giải thích:

fid: tên biến trỏ đến file cần ghi.

f: các tham số để định dạng.

d) Ví dụ:

Tạo file exp.txt có nội dung:

```
x = 0:2:10;
y = [x, x/2];
fid = fopen('exp.txt', 'w');
fprintf(fid, '%d', [2, inf]);
```

Gán file exp.txt và biến a để xem nội dung:

```
fid = fopen('exp.txt')
a = fscanf(fid, '%d', [2,inf]);
```

```
disp(a);
fclose(fid);
```

Kết quả

```
0    2    4    6    8    10
0    1    2    3    4    5
```

4. Lệnh FREAD

a) Công dụng:

Đọc dữ liệu dạng nhị phân từ file.

b) Cú pháp:

```
[a, c] =fscanf(fid)
[a, c] = fscanf(fid,s)
```

c) Giải thích:

a: tên biến chứa dữ liệc được đọc vào.

c: số phần tử được đọc vào.

fid: tên biến trỏ đến file cần đọc.

s: kích thước dữ liệu đọc vào.

s được định dạng bởi các thông số:

n: chỉ đọc n phần tử vào cột vector a.

inf: đọc đến hết file.

[m,n]: chỉ đọc vào m cột và n hàng, n có thể bằng inf còn m thì không.

d) Ví dụ1:

file vd.txt có nội dung:

```
A B C
1  2  3
```

```
fid = fopen(vd.txt');
[a,c] = fread(fid);
disp(a);
disp(c);
```

```
a =
    65
    32
```

```
        66
        32
        67
        13
        10
        49
        32
        50
        32
        51
    c =
        12
```

e) Ví dụ2

```
fid = fopen('vd1.txt');
[a,c] = fread(fid, 4);
disp(a);
disp(c);
    a=
        65
        32
        66
        32
    c =
        4
```

f) Ví dụ 3:

file vd3.txt có nội dung

```
    ABCDE
    FGHIJ
    KLMNO
fid = fopen('vd3.txt');
[a,c] = fread(fid, [7, inf]);
```

```
disp(a);
disp(c);
    a =
        65   70   75
        66   71   76
        67   72   76
        68   73   78
        69   74   79
        13   13   13
        10   10   10
    c =
        21
    a'=
        65   66   67    68   69  13   10
        70   71   72   73   74   13   10
        75   76   77   78   79   13   10
```

5. Lệnh FWRITE

a) Công dụng:

Ghi đoạn dữ liệu dạng nhị phân thành file.

b) Cú pháp:

fwrite (fid,a)

c) Giải thích:

fid: tên biến trỏ đến file cần ghi.

a: tên biến chứa dữ liệu.

d) Ví dụ:

Ghi đoạn dữ liệu của biến a thành file a.txt

```
a = [65  66  67]
fid = fopen('a.txt', 'w');
fwrite(fid, '%');
fwite(fid,a);
```

Gán file a.txt vào biến b để xem nội dung

```
fid = fopen('a.txt');
b = fscanf(fid, '%');
disp(b);
fclose(fid);
```

Kết quả

```
b = ABC
```

6. Lệnh SPRINTF

a) Công dụng:

Hiển thị thông tin lên màn hình.

b) Cú pháp:

```
s = sprintf('ts',ds)
```

c) Giải thích:

s: biến chứa chuỗi số hiển thị trên màn hình.

ts: các tham số định dạng.

ds: danh sách các đối số.

Tham số định dạng thuộc 1 trong 2 kiểu sau:

(1) Chuỗi ký tự: chuỗi này sẽ được hiển thị lên màn hình giống hệt như được viết trong câu lệnh.

(2) Chuỗi các tham số định dạng: các chuỗi này sẽ không được hiển thị lên màn hình, nhưng tác dụng điều khiển việc chuyển đổi và cách hiển thị các đối số được đưa ra trong danh sách các đối số.

Ví dụ các tham số định dạng:

1) %d: đối số là số nguyên được viết dưới dạng thập phân.

```
s = sprintf('Đây là số: %d',-24)
s = Đây là số: -2
```

2) %u: đối số là số nguyên được viết dưới dạng thập phân không dấu.

```
s = sprintf('Đây là số: %u',24)
s = Đây là số: 24
```

3) %o: đối số là số nguyên được viết dưới dạng cơ số 8 không dấu.

```
s = sprintf('Đây là số: %o',9)
s = Đây là số: 11
```

4) %x: đối số là số nguyên được viết dưới dạng cơ số 16.

s = sprintf('Đây là số: %x',255)

s = Đây là số:ff

5) %f: đối số là số nguyên được viết dưới dạng cp số 10.

s = sprintf('Đây là số: %f',2550

s = Đây là số: 255.000000

Để định dạng phần thập phân thì thêm vào con số chứa số thập phân cần lấy.

s = sprintf('Đây là số: %.3f', 2.5568)

s = Đây là số: 2.557

6) %c: đối số là 1 ký tự riêng đặc biệt.

s = sprintf('Đây là chữ: %c','M')

s = Đây là chữ: M

7)%s: đối số là chuỗi ký tự.

s = sprintf('Đây là chuỗi: %s', 'Matlab')

s = Đây là chuỗi: Matlab

8. Lệnh SSCANF

a) Công dụng:

Đọc chuỗi ký tự và định dạng lại chuỗi ký tự đó.

b) Cú pháp:

[a,count] = sscanf(s, 'format', size)

c) Giải thích:

a: tên biến chứa chuỗi ký tự sau khi được định dạng.

count: đếm số phần tử được đọc vào.

size: kích thước sẽ được đọc vào.

format: phần định dạng giống như lệnh sprintf.

d) Ví dụ:

s = '3.12 1.2 0.23 2.56';

[a, count] = sscanf(s, '%f',3)

a =

3.1200

1.2000

0.2300

```
count =

     3
```

VII. CÁC HÀM TOÁN HỌC CƠ BẢN

1. **Một số hàm lượng giác:**

a) Cú pháp:

kq = hlg(x)

b) Giải thích:

kq: tên biến chứa kết quả.

x: đơn vị radian.

hlg: tên hàm lượng giác.

Tên hàm lượng giác	Giải thích
sin	Tính giá trị sine
cos	Tính giá trị cosine
tan	Tính giá trị tangent
asin	Nghịch đảo của sine
atan	Nghịch đảo của tangent
sinh	Tính giá trị hyperbolic sine
cosh	Tính giá trị hyperbolic cosine
tanh	Tính gía trị hyperbolic tangent

2. **Lệnh ANGLE**

a) Công dụng:

Tính góc pha của số phức.

b) Cú pháp:

p = angle(z)

c) Giải thích:

p: tên biến chứa kết quả, đơn vị radians

z: số phức

d) Ví dụ:

z = i-3j

z = 0 – 2.0000i

p = angle(z)

p = -1.5708

3. **Lệnh CEIL**

a) Công dụng:

Làm tròn số về phía số nguyên lớn hơn.

b) Cú pháp:

y = ceil(x)

c) Giải thích:

y: số sau khi được làm tròn.

x: số cần được làm tròn.

d) Ví dụ:

x = -1.9000 -0.2000 3.4000 5.6000 7.0000

y = ceil(x)

y = -1 0 4 6 7

4. **Lệnh CONJ**

a) Công dụng:

Tính lượng liên hiệp của số phức.

b) Cú pháp:

y = conj(z)

c) Giải thích:

y: tên biến chứa lượng liên hiệp

z: số phức

d) Ví dụ:

z = -3i + 2j

z = 0 – 1.0000i

y = conj(z)

y = 0 + 1.0000i

5. **Lệnh EXP**

a) Công dụng:

Tính giá trị e^x.

b) Cú pháp:

y = exp(x)

c) Ví dụ:

y = exp(x)

y = 20.0855

6. **Lệnh FIX**

a) Công dụng:

Làm tròn số về phía zero.

b) Cú pháp:

y = fix(x)

c) Giải thích:

y: số sau khi được làm tròn.

x: số cần được làm tròn.

d) Ví dụ:

x = -1.9000 -0.2000 3.4000 5.6000 7.0000

y = fix(x)

y = -1 0 3 5 7

7. **Lệnh FLOOR**

a) Công dụng:

Làm tròn số về phía số nguyên nhỏ hơn.

b) Cú pháp:

y = floor(x)

c) Giải thích:

y: số sau khi được làm tròn .

x: số cần được làm tròn

d) Ví dụ:

x = -1.9000 -0.2000 3.4000 5.6000 7.0000

y = floor(x)

y = -2 -1 3 5 7

8. **Lệnh IMAG**

a) Công dụng:

Lấy phần ảo của số phức.

b) Cú pháp:

y = imag(z)

c) Ví dụ:

y = imag(2 + 3j)

y = 3

9. **Lệnh LOG**

a) Công dụng:

Tìm logarithm cơ số e.

b) Cú pháp:

y = log(x)

d) Ví dụ:

y = log(2.718)

y = 0.9999

10. **Lệnh LOG2**

a) Công dụng:

Tìm logarithm cơ số 2.

b) Cú pháp:

y = log2(x)

d) Ví dụ:

y = log2(2)

y = 1

11. **Lệnh LOG10**

a) Công dụng:

Tìm logarithm cơ số 10.

b) Cú pháp:

y = log10(x)

d) Ví dụ:

y = log10(10)

y = 1

12. **Lệnh REAL**

a) Công dụng:

Lấy phần thực của số phức.

b) Cú pháp:

y = real(z)

d) Ví dụ:

y = real(1 + 3j)

y = 2

13. Lệnh REM

a) Công dụng:

Cho phần dư của phép chia.

b) Cú pháp:

r = rem(a,b)

c) Giải thích:

r: biến chứa kết quả

a, b: số chia và số bị chia

d) Ví dụ:

r = rem(16, 3)

r = 1

14. Lệnh ROUND

a) Công dụng:

Làm tròn số sao cho gần số nguyên nhất.

b) Cú pháp:

y = round(x)

c) Ví dụ:

x = -1.9000 -0.2000 3.4000 5.6000 7.0000

y = round(x)

y= -2 0 3 6 7

Bảng so sánh của các phép làm tròn số

X	-1.9000	-0.2000	3.4000	5.6000	7.0000
ceil(x)	-1	0	4	6	7
floor(x)	-2	-1	3	5	7

fix(x)	-1	0	3	5	7
round(x)	-2	0	3	6	7

15. **Lệnh SIGN**

a) Công dụng:

Xét dấu số thực.

b) Cú pháp:

y = sign(x)

c) Giải thích:

x: số thực cần xét dấu.

y: kết quả trả về.

y	x
0	số 0
1	số dương
-1	số âm

d) Ví dụ:

x = 2 0 -3 0.5

y = sugn(x)

y= 1 0 -1 1

16. **Lệnh SQRT**

a) Công dụng:

Tính căn bậc hai.

b) Cú pháp:

y = sqrt(x)

c) Ví dụ:

x = 4

y = sqrt(x)

y = 2

VIII. TẬP LỆNH THAO TÁC TRÊN MA TRẬN

1. Cộng, trừ, nhân, chia từng phần tử của ma trận với hằng số

a) Cú pháp:

Ma trận kết quả = ma trận [+] [-] [.] [/] hằng số.

b) Ví dụ:

```
a =
     1     2     3
     4     5     6
     7     8     9
```

Cộng ma trận a với 2 kết quả là ma trận b

```
b = a + 2
b =
     3     4     5
     6     7     8
     9    10    11
```

tương tự cho các phép tính trừ, nhân và chia.

2. Lệnh DET

a) Công dụng:

Dùng để tính định thức của ma trận.

b) Ví dụ:

Tính định thức của ma trận a

```
a =
     1     4
     5     6
det(a)
    ans = -8
```

3. Lệnh DIAG

a) Công dụng:

Tạo ma trận mới và xử lý đường chéo theo quy ước.

b) Cú pháp:

v = diag(x)

v = diag(x,k)

c) Giải thích:

x: là vector có n phần tử.

v: là ma trận được tạo ra từ x theo quy tắc: số hàng bằng số cột và các phần tử của x nằm trên đường chéo của v.

k: tham số định dạng cho v, số hàng và cột của v = n + abs(k).

Nếu k = 0 đường chéo của v chính là các phần tử của x

Nếu k > 0 các phần tử của x nằm phía trên đường chéo v

Nếu k < 0 các phần tử của x nằm phía dưới đường chéo v

d) Ví dụ:

x = 2 1 9 5 4

v = diag(x)

v =

2 0 0 0 0

0 1 0 0 0

0 0 9 0 0

0 0 0 0 4

v = diag(x,2)

v =

0 0 2 0 0 0 0

0 0 0 1 0 0 0

0 0 0 0 9 0 0

0 0 0 0 0 5 0

0 0 0 0 0 0 4

0 0 0 0 0 0 0

0 0 0 0 0 0 0

v = diag(x,0)

v =

2 0 0 0 0

0 1 0 0 0

```
     0     0     9     0     0
     0     0     0     5     0
     0     0     0     0     4
```

```
v = diag(x,-2)
```

```
v =
     0     0     0     0     0     0     0
     0     0     0     0     0     0     0
     2     0     0     0     0     0     0
     0     1     0     0     0     0     0
     0     0     9     0     0     0     0
     0     0     0     5     0     0     0
     0     0     0     0     4     0     0
```

4. **Lệnh EYE**

a) Công dụng:

Tạo ma trận đơn vị.

b) Cú pháp:

```
y = eye(n)
y = eye(n,m)
```

c) Giải thích:

n: tạo ma trận có n hàng, n cột.

m, n: tạo ma trận có m hàng, n cột.

d) Ví dụ:

```
y = eye(3)
y =
     1     0     0
     0     1     0
     0     0     1
```

```
y = eye(3,5)
y =
     1     0     0     0     0
     0     1     0     0     0
```

0 0 1 0 0

5. **Lệnh FLIPLR**

a) Công dụng:

Chuyển các phần tử của các ma trận theo thứ tự cột ngược lại.

b) Cú pháp:

b = fliplr(a)

c) Giải thích:

b: tên ma trận được chuyển đổi.

a: tên ma trận cần chuyển đổi.

d) Ví dụ:

a =

0 1 2 3 4

5 6 7 8 9

b = fliplr(a)

4 3 2 1 0

9 8 7 6 5

6. Lệnh FLIPUD

a) Công dụng:

Chuyển các phần tử của ma trận theo thứ tự hàng ngược lại.

b) Cú pháp:

b = flipud(a)

c) Giải thích:

b: tên ma trận được chuyển đổi.

a: tên ma trận cần chuyển đổi.

d) Ví dụ:

a =

1 4

2 5

3 6

b = flipud(a)

b =

3 6

2 5

1 4

7. Lệnh INV

a) Công dụng:

Tìm ma trận nghịch đảo.

b) Cú pháp:

Ma trận nghịch đảo = inv (ma trận)

c) Ví dụ:

Tìm ma trận nghịch đảo của a.

a =

1 2 0

2 5 -1

4 10 -1

b = inv(a)

b =

5 2 -2

-2 -1 1

0 -2 1

8. Lệnh tạo ma trận

a) Công dụng:

Dùng để tạo 1 ma trận gồm có n hàng và m cột.

b) Cú pháp:

Tên ma trận = [a_{11} a_{12}...a_{1m} ; a_{21} a_{22}... a_{2m} ;...;...]

c) Giải thích:

a_{11}, a_{12}, a_{1m} là các giá trị tại hàng 1 cột 1 đến các giá trị tại hàng 1 cột m, có n dấu (;) là có n hàng.

d) Ví dụ:

Tạo ma trận gồm 3 hàng và 3 cột với giá trị là

1 2 3

4 5 6

```
     1     0     0
a = [1  2  3; 4  5  6; 1  0  0]
   a =  1     2     3
        4     5     6
        1     0     0
```

9. Lệnh tạo vector đơn

a) Công dụng:

Lệnh này dùng để tạo 1 vector đơn gồm có n phần tử.

b) Cú pháp 1:

Tên vector = [pt1 pt2 pt3 ...ptn]

c) Giải thích:

pt1 pt2 ...ptn: là các số thực.

d) Ví dụ:

Tạo vector a gồm có 4 phần tử, với các giá trị là:1, 3, 7, 4

```
a = [1     3     7     4]
    a =
         1     3     7     4
```

e) Cú pháp 2:

Tên vector = gtđ:csc:gtkt

f) Giải thích:

gtđ: là giá trị bắt đầu của vector.

csc: cấp số cộng.

gtkt: giá trị kết thúc.

g) Ví dụ:

Tạo vector a có giá trị bắt đầu 0.2, giá trị kết thúc pi/2 (= 1.5708), cấp số cộng 0,3.

```
a = 0.2;0.3;pi/2
    a =
        0.2000   0.5000   0.8000   1.1000   1.4000
```

10. Lệnh LINSPACE

a) Công dụng:

Tạo vector có giá trị ngẫu nhiên giới hạn trong khoảng định trước.

b) Cú pháp:

y = linspace(x1, x2)

y = linspace(x1, x2, n)

c) Giải thích:

y: tên của vector.

x1, x2: giới hạn giá trị lớn nhất và nhỏ nhất của vector y.

n: số phần tử của vector y.

Nếu không có giá trị n thì mặc định n = 100.

d) Ví dụ:

y = linspace(1, 10, 7)

y = 1.0000 2.5000 4.0000 5.5000 7.0000 8.5000 10.0000

11. Ma trận chuyển vị

a) Công dụng:

Ma trận chuyển vị = ma trận đang có.

b) Cú pháp:

Tạo 1 ma trận chuyển vị từ 1 ma trận đang có.

c) Ví dụ:

a =

1 2 3

4 5 6

7 8 9

ma trận chuyển vị b

b = a’

b =

4 7

5 8

6 9

12. Lệnh MAGIC

a) Công dụng:

Tạo 1 ma trận vuông có tổng của các phần tử trong 1 hàng, 1 cột hoặc trên đường chéo bằng nhau.

b) Cú pháp:

Tên ma trận = magic(n)

c) Giải thích:

n: kích thước ma trận.

Giá trị của mỗi phần tử trong ma trận là một dãy số nguyên liên tục từ 1 đến 2^n.

Tổng các hàng, cột và các đường chéo đều bằng nhau.

d) Ví dụ:

```
tmt = magic(3)
    tmt =
        8    1    6
        3    5    7
        4    9    2
```

13. Nhân ma trận

a) Công dụng:

Ma trận kết quả = ma trận 1* ma trận 2.

b) Ví dụ:

Ta có 2 ma trận a và b như trên và c là ma trận kết quả

```
c = a*b
    c =
        14   32   50
        32   77  122
        50  122 194
```

14. Lệnh ONES

a) Công dụng:

Tạo ma trận mà giá trị của các phần tử là 1.

b) Cú pháp:

y = ones(n)

y = ones(m,n)

c) Giải thích:

y = tên ma trận.

n: tạo ma trận có n hàng

m, n: tạo ma trận có m hàng, n cột.

d) Ví dụ:

```
y = ones(3)
    y =
        1    1    1
        1    1    1
        1    1    1
y = ones(3,5)
    y =
        1    1    1    1    1
        1    1    1    1    1
        1    1    1    1    1
```

15. Lệnh PASCAL

a) Công dụng:

Tạo ma trận theo quy luận tam giác Pascal.

b) Cú pháp:

pascal (n)

c) Giải thích:

n: là số hàng (cột)

d) Ví dụ:

```
pascal(4)
    ans =
        1    1    1    1
        1    2    3    4
        1    3    6   10
        1    4   10   20
```

16. Lệnh RAND

a) Công dụng:

Tạo ma trận mà kết mà giá trị của các phần tử là ngẫu nhiên.

b) Cú pháp:

y = rand(n)
y = rand(m,n)

c) Giải thích:

y: tên ma trận.

n: tạo ma trận có n hàng, n cột.

m, n: tạo ma trận có m hàng, n cột.

Giá trị của các phần tử nằm trong khoảng [0 1]

d) Ví dụ:

```
y = rand(3)

    y =

        0.9340   0.0920   0.7012
        0.8462   0.6539   0.7622
        0.5269   0.4160   0.7622

y = rand(3,5)

    y =

        0.2625   0.3282   0.9910   0.9826   0.6515
        0.0475   0.6326   0.3653   0.7227   0.0727
        0.7361   0.7564   0.2470   0.7534   0.6316
```

17. Lệnh RESHAPE

a) Công dụng:

Định dạng lại kích thước ma trận.

b) Cú pháp:

b = reshape(a,m,n)

c) Giải thích:

b: ma trận được định dạng lại.

a: ma trận cần được định dạng.

m, n: số hàng và số cột của b.

Ma trận a phải có số phần tử là: m*n.

d) Ví dụ:

a =

```
    1   4   7   10
    2   5   8   11
    3   6   9   12
b = reshape(a,2,6)
    b =
        1   3   5   7   9   11
        2   4   6   8   10  12
```

18. Lệnh ROT90

a) Công dụng:

Xoay ma trận 90^0.

b) Cú pháp:

```
b = rot90(a)
```

c) Giải thích:

b: ma trận đã được xoay 90^0

a: ma trận cần xoay.

d) Ví dụ:

```
a =
    1   2   3
    4   5   6
    7   8   9
b = rot90(a)
    b =
        3   6   9
        2   5   8
        1   4   7
```

19. Lệnh TRACE

a) Công dụng:

Tính tổng các phần tử của đường chéo ma trận.

b) Cú pháp:

```
d = trace(a)
```

c) Giải thích:

d: biến chứa kết quả.

a: tên ma trận.

d) Ví dụ:

```
a =
    2    8    3
    4    7    1
    6    9    2
d = trace(a)
    d = 11
```

20. Lệnh TRIL

a) Công dụng:

Lấy phân nửa dưới ma trận theo hình.

b) Cú pháp:

```
I = tril(x)
I = tril(x,k)
```

c) Giải thích:

I: tên ma trận kết quả.

k: tham số.

Nếu k = o lấy từ đường chéo trở xuống.

Nếu k = n lấy từ đường chéo trở lên n đơn vị.

Nếu k = -n lấy từ đường chéo trở xuống n đơn vị.

d) Ví dụ:

```
a =
     5    9   13
     6   10   14
     7   11   15
     8   12   16
i = tril(a)
i =
     1    0    0    0
     2    6    0    0
```

```
     3     7    11    15
     4     8    12    16
i = tril(a,0)
i =
     1     0     0     0
     2     6     0     0
     3     7    11     0
     4     8    12    16
i = tril(a,1)
i =
     1     5     0     0
     2     6    10     0
     3     7    11    15
     4     8    12    16
i = tril(a,-1)
i =
     0     0     0     0
     2     0     0     0
     3     7     0     0
     4     8    12     0
```

21. Lệnh TRIU

a) Công dụng:

Lấy phân nửa trên ma trận theo hình tam giác.

b) Cú pháp:

I = triu(x)

I = triu(x,k)

c) Giải thích:

I: tên ma trận kết qủa.

k: tham số

Nếu k = 0 lấy từ đường chéo trở lên.

Nếu k = n lấy từ đường chéo trở xuống n đơn vị.

Nếu k = -n lấy từ đường chéo trở lên n đơn vị.

d) Ví dụ:

```
a =
     1     5     9    13
     2     6    10    14
     3     7    11    15
     4     8    12    16
I = triu(a)
I =
     1     5     9    13
     0     6    10    14
     0     0    11    15
     0     0     0    16
I = triu(a,0)
I =
     1     5     9    13
     0     6    10    14
     0     0    11    15
     0     0     0    16
I = triu(a,-1)
I =
     1     5     9    13
     2     6    10    14
     0     7    11    15
     0     0    12    16
I = triu(a,1)
I =  0     5     9    13
     0     0    10    14
     0     0     0    15
     0     0     0     0
```

22. Lệnh ZEROS

a) Công dụng:

Tạo ma trận mà giá trị của các phần tử

b) Cú pháp:

y = zeros(n)

y = zeros(m,n)

c) Giải thích:

y: tên ma trận.

n: tạo ma trận có n hàng và n cột.

m, n: tạo ma trận có m hàng, n cột.

d) Ví dụ:

```
y = zeros(3)
y =
     0     0     0
     0     0     0
     0     0     0
y = zeros(3,7)
y =
     0     0     0     0     0     0     0
     0     0     0     0     0     0     0
     0     0     0     0     0     0     0
```

IX. CÁC PHÉP TÍNH ĐẠI SỐ

1. Lệnh CONV

a) Công dụng:

Nhân hai đa thức.

b) Cú pháp:

c = conv(a,b)

c) Giải thích:

a,b: đa thức

c: tích số của a,b

Cách khai báo: sắp xếp biến theo thứ tự giảm dần của lũy thừa.

d) Ví dụ:

Nhân hai đa thức $(3x^2+4x+5).(2x^3-3x^2+2)$

```
a = [0 3  4  5]
    a = 0   3   4   5
b = [2 -3  0  2]
    b =2  -3   0   2
c = conv(a,b)
    c = 0    6  -1   -2   -9    8    10
```

2. Lệnh CUMPROD

a) Công dụng:

Nhân dồn các phần tử.

b) Cú pháp:

```
cp = cumprod (a)
```

c) Giải thích:

cp: biến chứa kết qủa

a: tên của ma trận hay vector.

d) Ví dụ:

```
b =  1    9    3    4
cp =cumprod(b)
cp = 1    9    27   108
    a =
         1    3    5
         9    1    2
         4    2    1
cp = cumprod(a)
    cp =1    3    5
              9    3    10
              36   6    10
```

3. Lệnh CUMSUM

a) Công dụng:

Cộng dồn các phần tử.

b) Cú pháp:

cs = cumprod(a)

c) Giải thích:

cs: biến chứa kết quả.

a: là tên của ma trận hay vector.

d) Ví dụ:

b = 1 10 1 2 5

cs = cumsum(b)

cs =1 11 12 14 19

a=

1 3 5

9 1 2

4 2 1

cs = cumsum(a)

cs =

1 3 5

10 4 7

14 6 8

4. Lệnh DECONV

a) Công dụng:

Chia hai đa thức.

b) Cú pháp:

[q,r] =deconv(a,b)

c) Giải thích:

a,b: đa thức.

q: thương số của a, b.

r: số dư.

Cách khai báo: sắp xếp biến theo thứ tự giảm dần của lũy thừa.

d) Ví dụ:

Chia 2 đa thức $(2x^2+3x+6)/(2x+3)$

```
a = [2     3  6]
b = [2   3]
[q,r] = deconv (a,b)
    q = 1     0
    r =  0     0     6
```

5. Lệnh EXPM

a) Công dụng:

Tính e^x

b) Cú pháp:

```
kq = expm(x)
```

c) Giải thích:

kq: biếnchứa kết qủa.

d) Ví dụ:

```
kq = expm(3)
    kq = 20.0855
```

6. Lệnh FMIN

a) Công dụng:

Tìm giá trị nhỏ nhất của hàm số.

b) Cú pháp:

```
x = fmin('fuction',x1,x2)
```

c) Giải thích:

x: biến chứa kết quả.

fuction: tên hàm số.

x1, x2: khoảng khảo sát.

d) Ví dụ:

Tìm giá trị nhỏ nhất của hàm số: x3-2x-5 trong khoảng [0 2]

```
x =fmin('x.^3-2*x-5',0,2);
    x = 0.8165
y = f(x)
    y = -6.0887
```

7. Lệnh FPLOT

a) Công dụng:

Vẽ đồ thị của hàm số.

b) Cú pháp:

fplot('fun',[xmin,xmax]

c) Giải thích:

fun: tên hàm số.

xmin, xmax: xác định khoảng cần vẽ.

d) Ví dụ:

fplot('x.^3-2*x-5',[0,2]);

grid;

8. Lệnh FZERO

a) Công dụng:

Tìm điểm 0 của hàm số.

b) Cú pháp:

fzero('fun',x0)

c) Giải thích:

Điểm 0 của hàm số là điểm (0,x), đây cũng chính là nghiệm của hàm số. Nếu hàm số có nhiều nghiệm thì sẽ tìm được nghiệm gần giá trị x0.

fun: tên hàm số.

c) Ví dụ:

Tìm giá trị 0 của hàm số: x^2-5x+3.

Trước tiên ta khai báo hàm số f trong tập tin f.m: (xem thêm lệnh function)

function y = f(x);

y = x.^2-5*x+3;

Sau đó, tạo tập tin gt0.m:

x = 0:10;

% Giá trị x0 = 0

z = fzero('f',0);

sprinf('z = %3f',z)

z = 0.382

% Giá trị x0 = 2

```
z = fzero('f',2);
sprintf('z = %.3f',z)
    z = 2.618
% Vẽ đồ thị hàm số minh họa:
z = fzero('f',0);
fplot('f',[0,5];
grid;
hold on;
plot(z,0,'o');
hold off
```

9. Lệnh MAX

a) Công dụng:

Tìm giá trị lớn nhất.

b) Cú pháp:

```
m = max(x)
[m,i] = max(x) v = max(x,y)
```

c) Giải thích:

x,y,v:tên vector.

m: giá trị lớn nhất.

i: vị trí của m.

Nếu x là ma trận tìm ra giá trị lớn nhất của mỗi cột.

d) Ví dụ:

```
x = 3    5    2    1    4
m= max(x)
    m = 5
[m,i] = max(x)
    m =5
    i =2
y =  1    6    8    -5    3
v =max(x,y)
```

```
v = 3     6     8     1     4
b =
     3     6     2
     1     7     9
     2     8     1
m = max(b)
     m = 3     8     9
[m,i] = max(b)
     m= 3     8     9
     i = 1     3     2
a =
     0     3     6
     7     1     1
     4     6     8
v = max(a,b)
v =
     3     6     6
     7     7     9
     4     8     8
```

10. Lệnh MEAN

a) Công dụng:

Tìm giá trị trung bình.

b) Cú pháp:

Mô hình = mean(a)

c) Giải thích:

m: biến chứa kết qủa.

a: tên vector hay ma trận cần tính giá trị trung bình.

Nếu a là ma trận thì tính giá trị trung bình của mỗi cột.

d) Ví dụ:

```
b = 1     10   1     2     5
m = mean(b)
```

```
    m = 3.8000
a =
    1    3    5
    9    1    2
    4    2    1
m = mean(a)
    m = 4.6667    2.0000    2.6667
```

11. Lệnh MIN

a) Công dụng:

Tím giá trị nhỏ nhất

b) Cú pháp:

```
m = min(x)
[m,i] = min(x)
v = min(x,y)
```

c) Giải thích:

x,y,v: tên vector.

m: là giá trị lớn nhất.

i: là vị trí của m.

Nêú x là ma trận tìm ra giá trị nhỏ nhất trong mỗi cột.

d) Ví dụ:

```
x = 3    5    2    1    4
m = min(x)
    m = 1
    i =4
y =1    6    8   -5    3
v = min(x,y)
    v = 1    5    2   -5   3
b =
    3    6    2
    1    7    9
    2    8    1
```

```
m = min(b)
    m = 1    6    1
    i = 2    1    3
a =
    0    3    6
    7    1    1
    4    6    8
v = min(a,b)
    v =
        0    3    2
        1    1    1
        2    6    1
```

12. Lệnh PROD

a) Công dụng:

Nhân các phần tử.

b) Cú pháp:

```
p = prod(x)
```

c) Giải thích:

p: biến chứa kết quả.

x: tên ma trận hay dãy số.

Nếu là ma trận nhân từng phần tử cuả mỗi cột.

d) Ví dụ:

```
a = 2    3    4    5
p = prod(a)
    p = 20
b =
    2    2    3
    5    6    4
    7    5    4
p =prot(b)
    p =70    60   48
```

13. Lệnh ROOTS

a) Công dụng:

Tìm nghiệm của đa thức.

b) Cú pháp:

r = roots(p)

c) Giải thích:

r: biến chứa kết quả.

p: tên biểu thức.

d) Ví dụ:

Tìm nghiệm cuả phương trình: x^2-1 =0

p = [1 0 -1]

r = roots(p);

disp(r)

-1.0000

1.0000

14. Lệnh SORT

a) Công dụng:

Sắp xếp mảng hay ma trận theo thứ tự tăng dần.

b) Cú pháp:

kq = sort(x)

[kq,i] = sort(x)

c) Giải thích:

kq: biến chứa kết quả.

i: số thứ tự cuả phần tử trước khi sắp xếp.

Nếu x là ma trận thì sắp xếp theo thứ tự tăng dần của từng cột.

d) Ví dụ:

a = 2 8 5 6 -3 9

kq = sort(a)

kq = -3 2 5 6 8 9

[kq,i] = sort(a)

kq = -3 2 5 6 8 9

```
        i =   5    1    3    4    2    6
    b =
        3    4   -4
        2   -3    5
        1    6    2
    kq =sort(b)
        kq =
            1   -3   -4
            2    4    2
            3    6    5
    [kq,i] = sort(b)
        kq =
            1   -3   -4
            2    1    2
            3    6    5
        i =
            3    2    1
            2    1    3
            1    3    2
```

15. Lệnh SUM

a) Công dụng:

Tính tổng của các phần tử.

b) Cú pháp:

s = sum(x)

c) Giải thích:

s: là biến chứa kết quả.

x: là tên ma trận.

Nếu x là ma trận thì s là tổng của các cột.

d) Ví dụ:

a = 2 8 5 6 -3 9

s = sum(a)

```
    s = 27
b =
    3    4    -4
    2    -3   5
    1    6    2
s = sum(b)
    s = 6     7     3
```

X. TẬP LỆNH ĐỒ HỌA

1. Lệnh AXES

a) Công dụng:

Đặt các trục tọa độ tại vị trí định trước.

b) Cú pháp:

axes('propertyname', propertyvalue ...)

c) Giải thích:

Tương ứng với một propertyname đi kèm với 1 propertyvalue.

1. 'position',[left, bottom, width, height]: định vị trí và kích thước của trục.

   left: khoảng cách từ mép trái cửa sổ đến trục đứng.

   bottom: khoảng cách từ mép dưới cửa sổ đến trục ngang.

   width: chiều dài của trục ngang.

   height: chiều cao trục đứng.

   Ghi chú:

   Luôn lấy điểm [0,0] làm gốc tọa độ.

   Trục ngang và trục đứng có giá trị trong khoảng [0 1] và chia theo tỷ lệ thích hợp

   */ Ví dụ:

   ```
   axes('position',[.1  .1  .8  .6])
   ```

2. 'xlim', [min,max]: định giá trị nhỏ nhất và lớn nhất trên trục x.

   */ Ví dụ:

   ```
   axes('xlim', [2  5])
   ```

3. 'ylim', [min,max]: định giá trị nhỏ nhất và lớn nhất trên trục y.

   */ Ví dụ:

axes('ylim', [2 5])

định giá trị trên cả hai trục

axes('xlim', [min,max], 'ylim',[min,max])

2. Lệnh AXIS

a) Công dụng:

Chia lại trục tọa độ.

b) Cú pháp:

axis([xmin xmax ymin ymax])

axis([xmin xmax ymin ymax zmin zmax])

axis on

axis off

c) Giải thích:

xmin, ymin, zmin: là giá trị nhỏ nhất của các trục x, y, z.

xmax, ymax, zmax: là giá trị lớn nhất của các trục x, y, z.

on: cho hiển thị trục tọa độ.

off: không cho hiển thị trục tọa độ.

3. Lệnh BAR

a) Công dụng:

Vẽ đồ thị dạng cột.

b) Cú pháp:

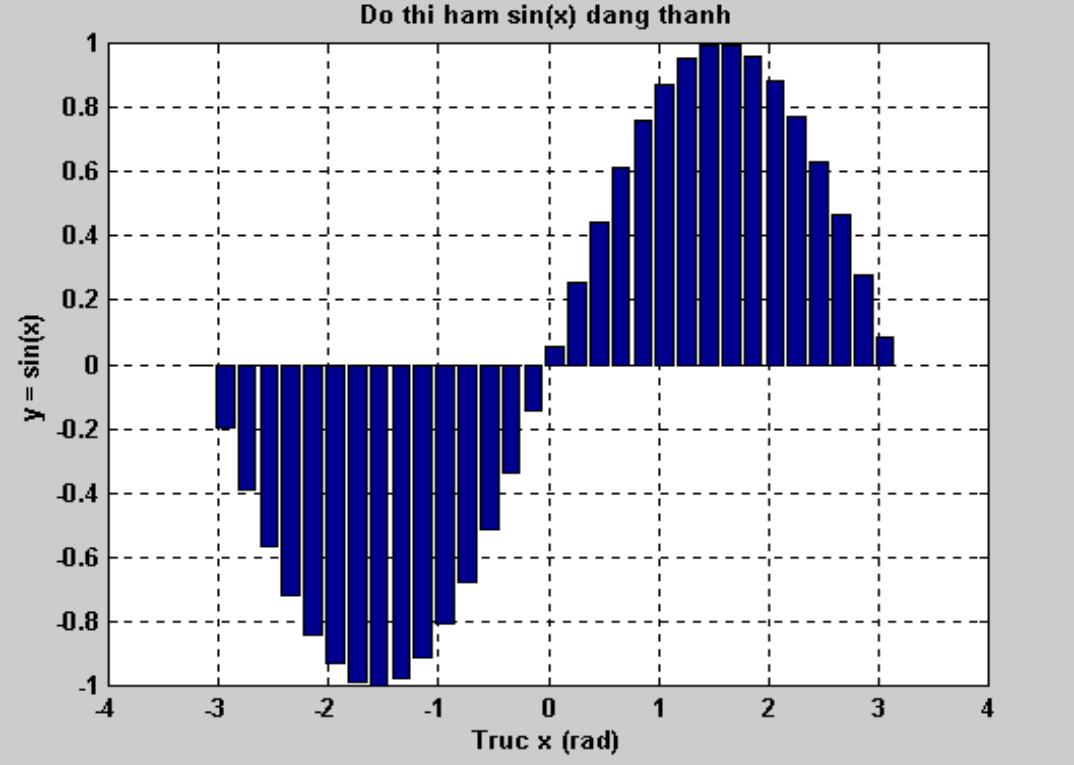
bar(x,y)

c) Giải thích:

Vẽ giá trị x theo giá trị y.

d) Ví dụ:

```
x = -pi:0.2:pi;
bar(x,sin(x));
grid on
title('Do thi ham sin(x) dang thanh')
xlabel('truc x (rad)')
ylabel('y = sin(x)')
```



4. Lệnh CLA

a) Công dụng:

Xóa tất cả các đối tượng như: đường đồ thị, tên đồ thị...nhưng không xóa trục tọa độ.

b) Cú pháp:

cla

5. Lệnh CLF

a) Công dụng:

Xóa hình ảnh (đồ thị) hiện tại.

b) Cú pháp:

clf

6. Lệnh CLOSE

a) Công dụng:

Đóng hình ảnh (đồ thị) hiện tại.

b) Cú pháp:

close

7. Lệnh COLORMAP

a) Công dụng:

Tạo màu sắc cho đồ thị trong không gian 3 chiều.

b) Cú pháp:

colormap(map)

colormap('default')

c) Giải thích:

Colormap là sự trộn lẫn của 3 màu cơ bản: red, green, blue. Tùy theo tỷ lệ của 3 màu cơ bản mà cho ra các màu sắc khác nhau.

'default': màu có được là màu mặc định.

map: biến chứa các thông số sau:

Map	màu có được
Bone	gray + blue
Cool	cyan + magenta
Flag	red + white + blue + black
Gray	gray
Hot	black + red + yellow + white
Pink	pink

8. Lệnh FIGURE

a) Công dụng:

Tạo mới hình ảnh (đồ thị).

b) Cú pháp:

figure

9. Lệnh GCA

a) Công dụng:

Tạo các đặc tính cho trục.

b) Cú pháp:

h = gca

c) Giải thích:

h: là biến gán cho lệnh cga.

Các đặc tính của trục gồm có:

Cú pháp	Giải thích
Set(gca,'XScale','log',	Định đơn vị trên trục tọa độ: trục x có

'Yscale','linear')	đơn vị là log và trục y có đơn vị tuyến tính.
Set(gca,'Xgrid','on','YGrid', 'nomal')	Tạo lưới cho đồ thị: trục x có tạo lưới và trục y không tạo lưới.
Set(gca,'XDir','reverse', 'YDir','normal')	Đổi trục tọa độ: đổi trục x về phía đối diện, trục y giữ nguyên.
Set(gca,'XColor','red', 'Ycolor','yellow')	Đặt màu cho lưới đồ thị: đặt lưới trục x màu đỏ, lưới trục y màu vàng. Gồm có các màu: yellow, magenta, cyan, red, green, blue, white, black.

10. Lệnh GRID

a) Công dụng:

Tạo lưới tọa độ.

b) Cú pháp:

grid on

grid off

c) Giải thích:

on: hiển thị lưới tọa độ.

off: không hiển thị lưới tọa độ.

11. Lệnh PLOT

a) Công dụng:

Vẽ đồ thị tuyến tính trong không gian 2 chiều.

b) Cú pháp:

plot(x,y)

plot(x,y,'linetype')

c) Giải thích:

x,y: vẽ giá trị x theo giá trị y.

linetype: kiểu phần tử tạo nên nét vẽ bao gồm 3 thành phần:

- Thành phần thứ nhất là các ký tự chỉ màu sắc:

Ký tự	Màu
y	Vàng

m	Đỏ tươi
c	Lơ
r	Đỏ
g	Lục
b	Lam
w	Trắng
k	Đen

- Thành phần thứ hai là các ký tự chỉ nét vẽ của đồ thị:

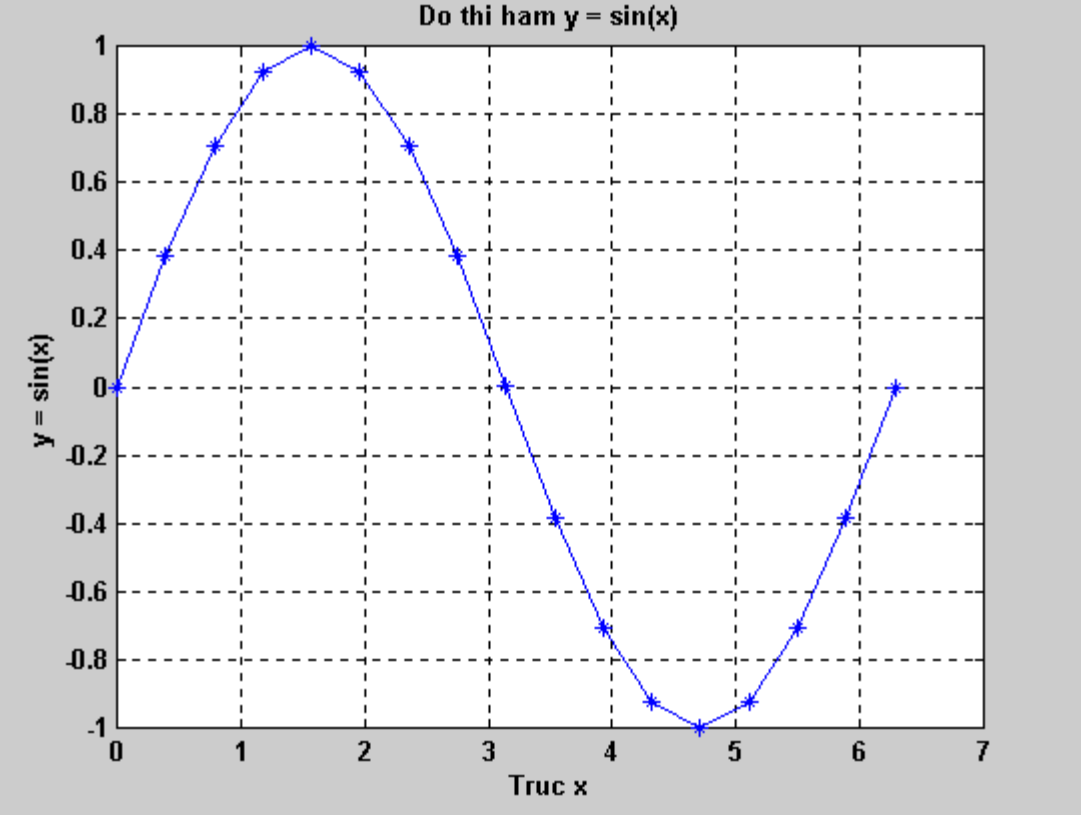
Ký tự	Loại nét vẽ
-	Đường liền nét
:	Đường chấm chấm
-.	Đường gạch chấm
--	Đường nét đứt đoạn

- Thành phần thứ ba là các ký tự chỉ loại điểm đánh dấu gồm:., o, x, +, *

d) Ví dụ:

Vẽ đồ thị hàm y = sin(x) với đồ thị màu lam, đường liền nét và đánh dấu các điểm được chọn bằng dấu *, trục x thay đổi từ 0 tới 2π, mỗi bước thay đổi là π/8

```
x = 0:pi/8:2*pi;
y = sin(x);
plot(x,y, 'b-* ')
ylabel('y = sin(x)')
xlabel('Truc x')
title('Do thi ham y = sin(x)')
grid on
```



12. Lệnh SUBPLOT

a) Công dụng:

Tạo các trục trong một phần của cửa sổ đồ họa.

b) Cú pháp:

subplot(m,n,p)

subplot(mnp)

c) Giải thích:

subplot(m,n,p) hoặc subplot(mnp)thành cửa sổ đồ họa thành m×n vùng để vẽ nhiều đồ thị trên cùng một cửa sổ.

m: số hàng được chia.

n: số cột được chia

p: số thứ tự vùng chọn để vẽ đồ thị.

Nếu khai báo p > m×n thì sẽ xuất hiện một thông báo lỗi.

d) Ví dụ:

Chia cửa sổ đồ họa thành 2×3 vùng và hiển thị trục của cả 6 vùng.

subplot(231)

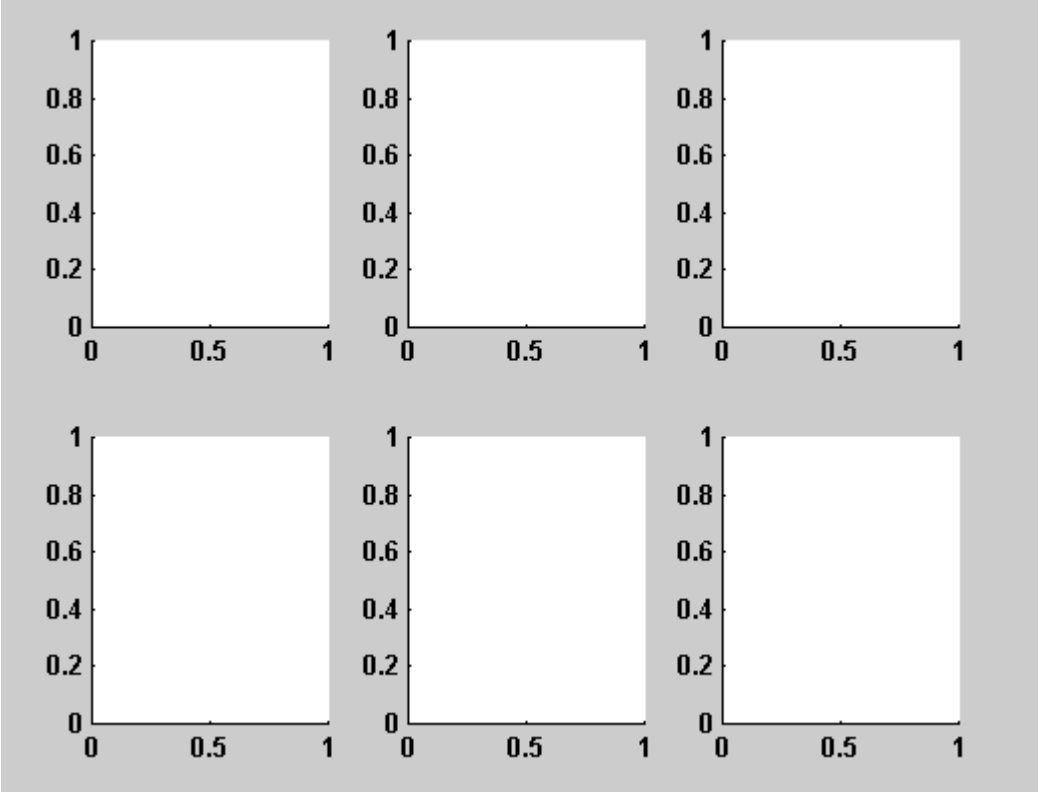
subplot(232)

subplot(233)

subplot(234)

subplot(235)

subplot(236)



13. Lệnh SEMILOGX, SEMILOGY

a) Công dụng:

Vẽ đồ thị theo logarith.

b) Cú pháp:

semylogx(x,y)

semylogx(x,y,'linetype')

semylogy(x,y)

semylogy(x,y,'linetype')

c) Giải thích:

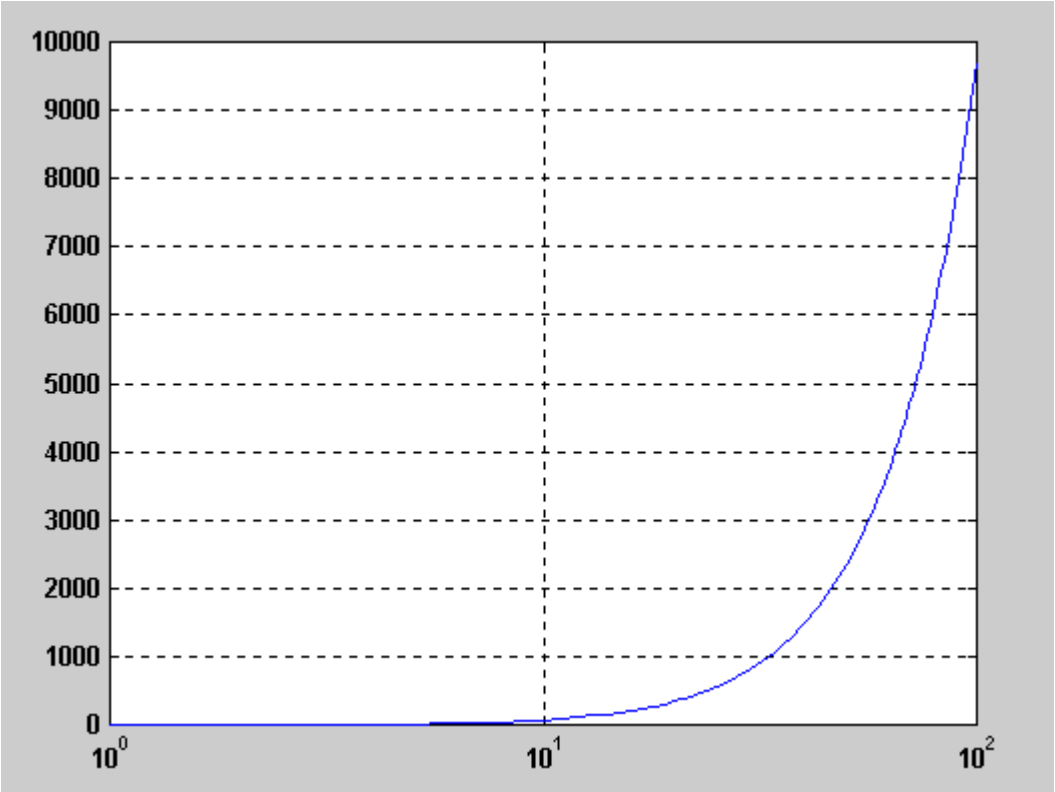
semylogx và semylogy giống như lệnh plot nhưng chỉ khác một điều là lệnh này vẽ đồ thị theo trục logarith. Do đó, ta có thể sử dụng tất cả các loại 'linetype' của lệnh plot.

d) Ví dụ:

Vẽ đồ thị hàm $y = x^2 - 3x + 2$ theo trục logarith của x.

x = 0:100;

```
y = x.^2-3*x+2;
semylogx(x,y,'b')
grid on
```



14. Lệnh POLAR

a) Công dụng:

Vẽ đồ thị trong hệ trục tọa độ cực.

b) Cú pháp:

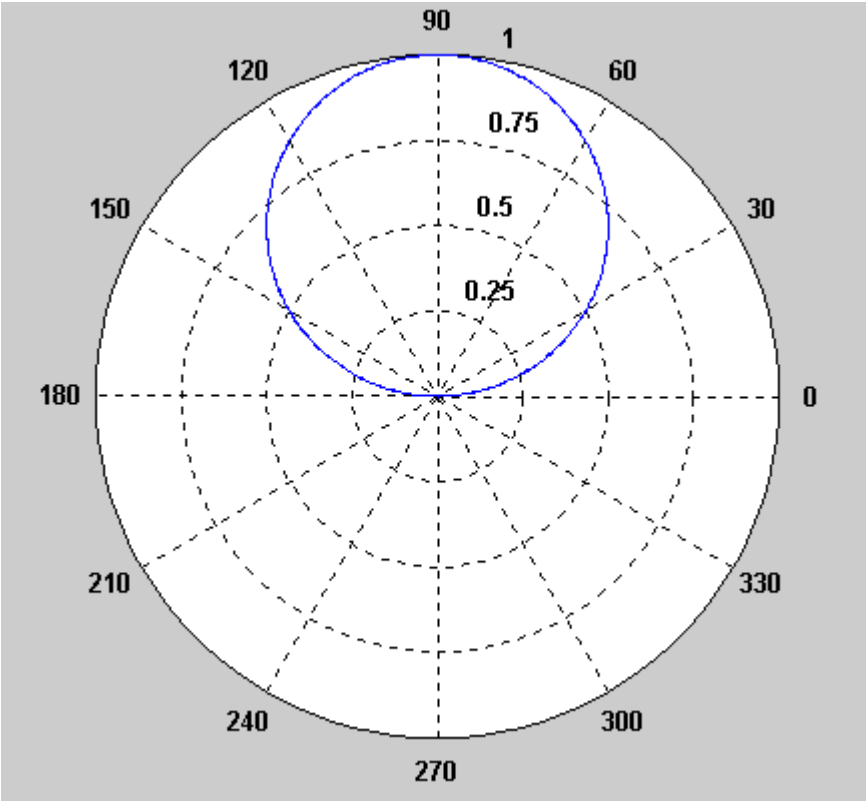
```
polar(theta,rno)
```

c) Giải thích:

Vẽ giá trị x theo giá trị y.

d) Ví dụ:

```
t = -pi:0.01:pi;
polar(t, sin(t))
```



15. Lệnh SET

a) Công dụng:

Thiết lập các đặc tính chất cho đối tượng nào đó.

b) Cú pháp:

set(h, ‘propertyname’, propertyvalue,...)

c) Giải thích:

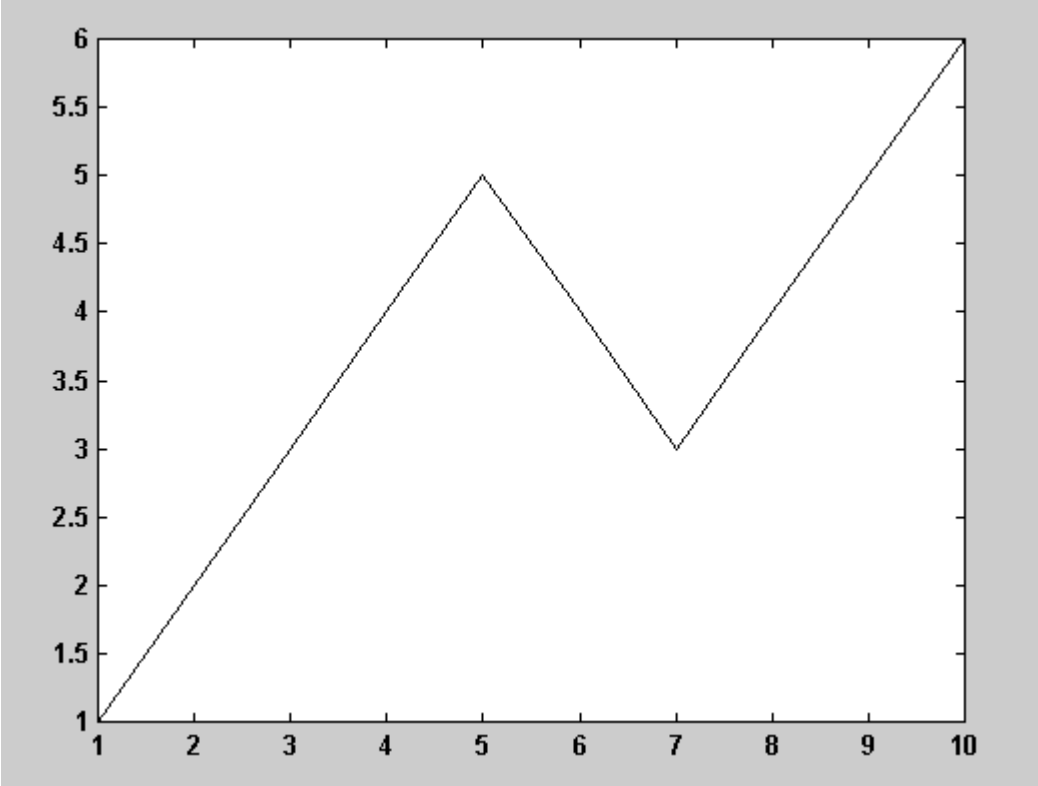
h: biến chứa đối tượng.

PropertyName và PropertyValue được cho trong bảng sau:

Cú pháp	PropertyName	PropertyValue	Giải thích
Set(h,’Marker’,’+’)	Marker	-, --, :, -., o, ×, +, *	Chọn kiểu phần tử
Set(h,’LineWidth’,1)	LineWidth	1, 2, 3,...	Độ dày nét vẽ
Set(h,’MarkerSize’,9)	MarkerSize	1, 2, 3,...	Kích thước các điểm tạo nên h
Set(h,’color’,’cyan’)	Color	yellow,magenta, red,green,blue, cyan,white,black	Chọn màu cho đối tượng h

d) Ví dụ:

```
a = [1  2  3  4  5  4  3  4  5  6];
h = plot(a)
set(h,'color','black')
```



16. Lệnh STAIRS

a) Công dụng:

Vẽ đồ thị dạng bậc thang.

b) Cú pháp:

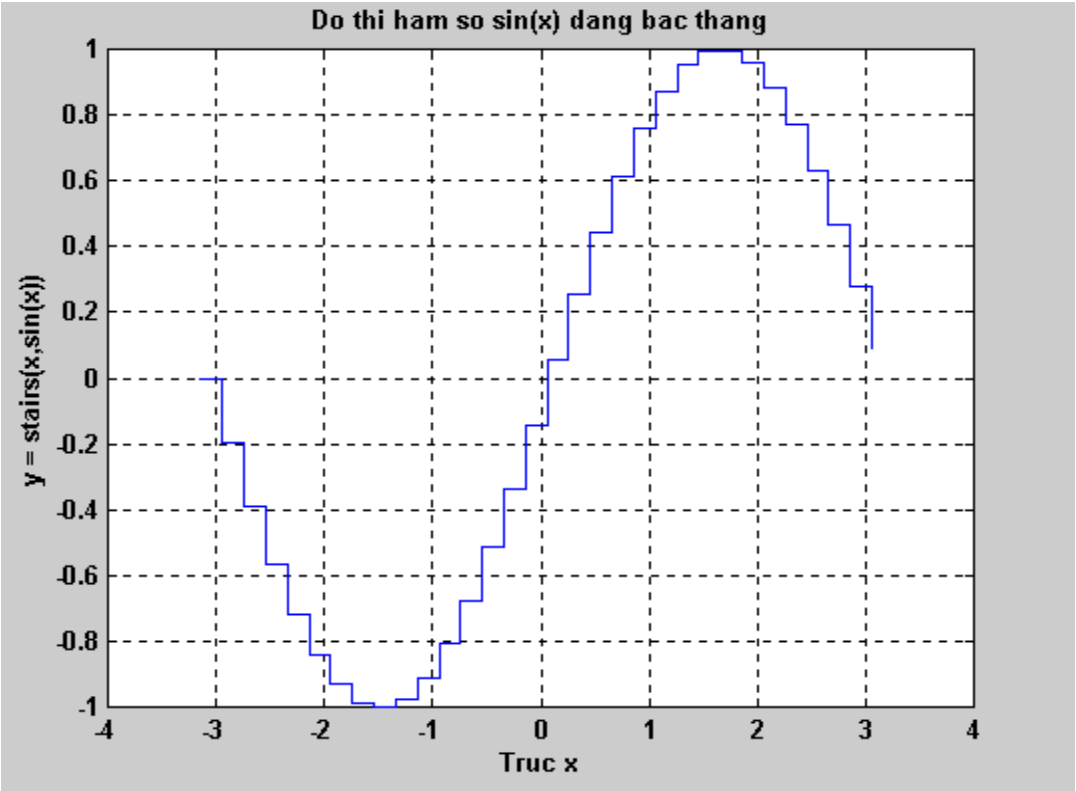
```
stairs(x,y)
```

c) Giải thích:

Vẽ giá trị x theo giá trị y.

d) Ví dụ:

```
x = -pi:0.2:pi;
stairs(x,sin(x))
xlabeL('Truc x')
ylabel('y = stairs(x,sin(x)')
grid on
```



17. Lệnh TITLE

a) Công dụng:

Đặt tiêu đề cho đồ thị.

b) Cú pháp:

title('text')

c) Giải thích:

text: tên tiêu đề.

18. Lệnh XLABEL, YLABEL, ZLABEL

a) Công dụng:

Đặt tên cho trục X, Y, Z.

b) Cú pháp:

xlabel('nx')

ylabel('ny')

zlabel('nz')

c) Giải thích:

nx, ny, nz: tên trục x, y, z

19. Lệnh WHITEBG

a) Công dụng:

Thay đổi màu nền của cửa sổ đồ họa.

b) Cú pháp:

whitebg

whitebg('color')

c) Giải thích:

whitebg chuyển đổi qua lại màu nền cửa sổ đồ họa giữa trắng và đen.

whitebg('color') chuyển màu nền cửa sổ đồ họa thành màu của biến color.

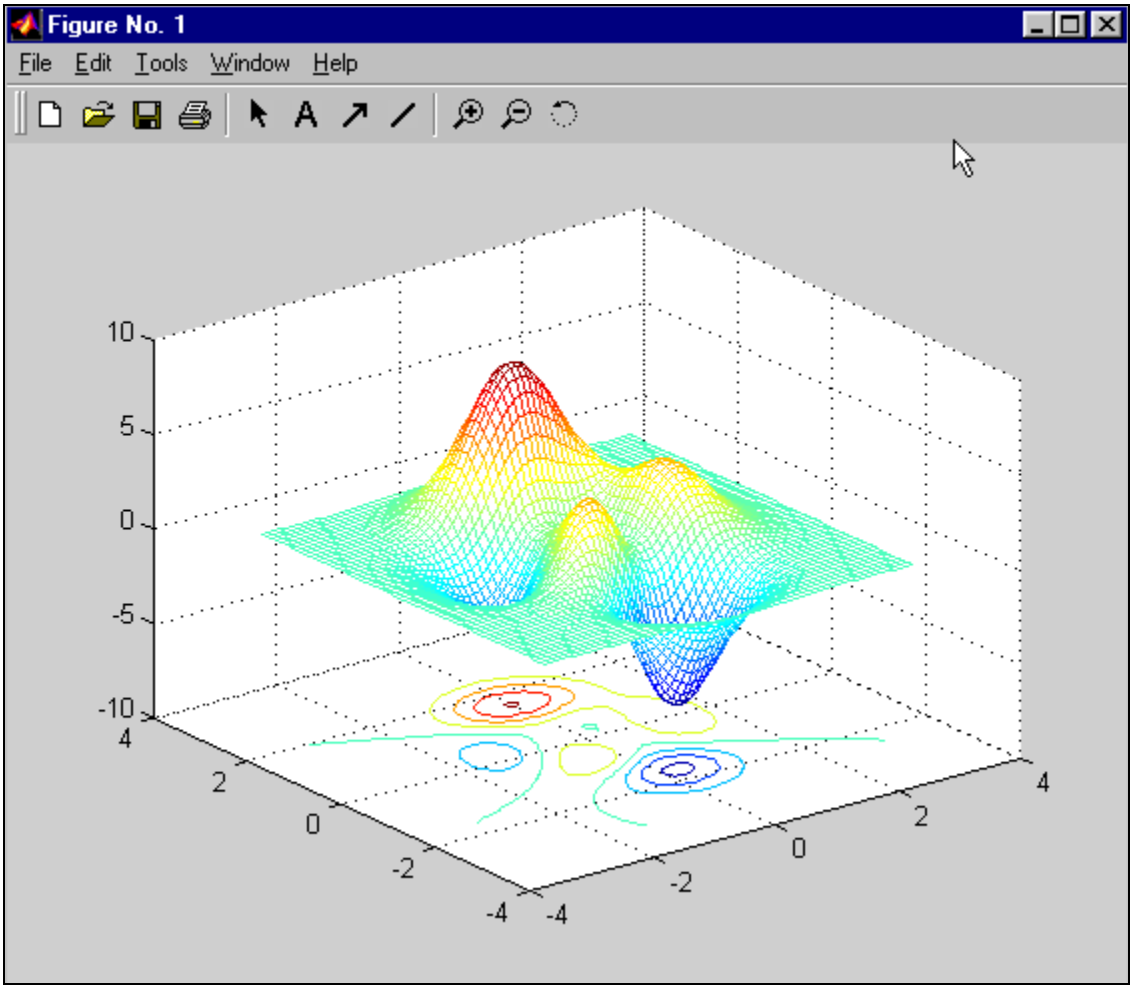
color có thể là các màu: yellow (vàng), magenta (đỏ tươi), cyan (lơ), red (đỏ), green (lục), blue (lam), white (trắng), black (đen).

BT3c: được viết trong **BT3c.m**. Bài tập này tổng hợp từ các sách **'The Student Edition of MATLAB', 'The MATLAB 5. Handboox', 'Ứng dụng MATLAB trong điều khiển tự động'**

```
%BT3c: VE QUA DIA CAU
[x,y]=meshgrid(-3:0.1:3);
z=peaks(x,y);
meshc(x,y,z)
pause

k=5;
n=2^k-1;
[x,y,z]=sphere(n);
c=hadamard(2^k);
surf(x,y,z,c);
colormap([1 1 0;0 1 1])
pause

t=0:pi/10:2*pi;
[x,y,z]=cylinder(2+cos(t));
surf(x,y,z)
pause

[x,y,z]=cylinder(1:10);
surfnorm(x,y,z)
pause

[x,y,z]=meshgrid(-2:.2:2,-2:.2:2,-2:.2:2);
v=x.*exp(-x.^2-y.^2-z.^2);
slice(v,[5 15 21],21,[1 10],21)
pause

[X,Y]=meshgrid(-3:.5:3);
Z=peaks(X,Y);
[XI,YI]=meshgrid(-3:.25:3);
ZI=interp2(X,Y,Z,XI,YI);
mesh(X,Y,Z), hold, mesh(XI,YI,ZI+15)
hold off
axis([-3 3 -3 3 -5 20])
pause
```

```
syms x y
ezsurf(real(atan(x+i*y)))

[x,y]=meshdom(-12:.6:12,-12:.6:12);
r=sqrt(x.^2+y.^2);
z=bessel(0,r);
m=[-45 60];
mesh(z,m)
```

Khi chạy chương trình ta lần lượt có kết quả:



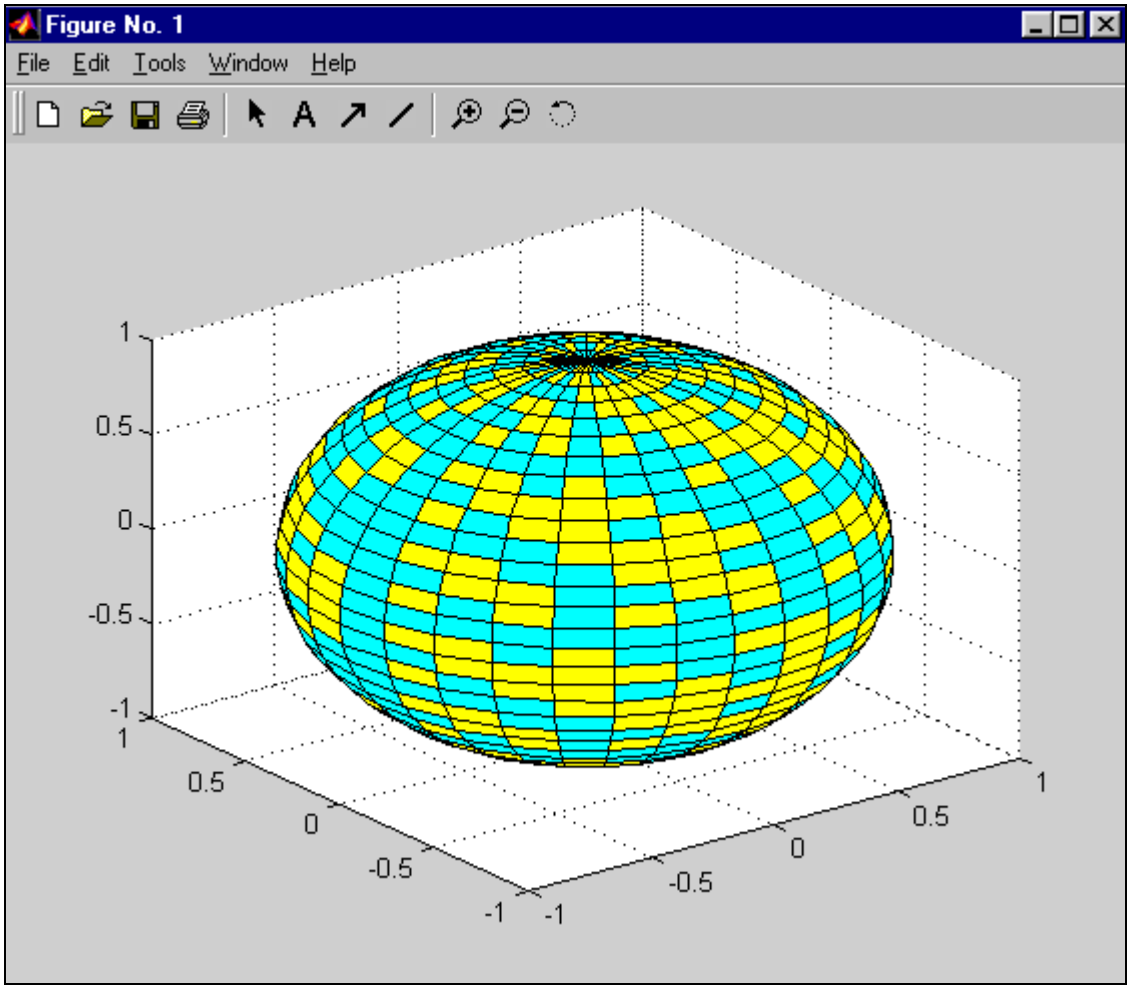


Figure No. 1
File Edit Tools Window Help
1
0.5
0
-0.5
-1
1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1

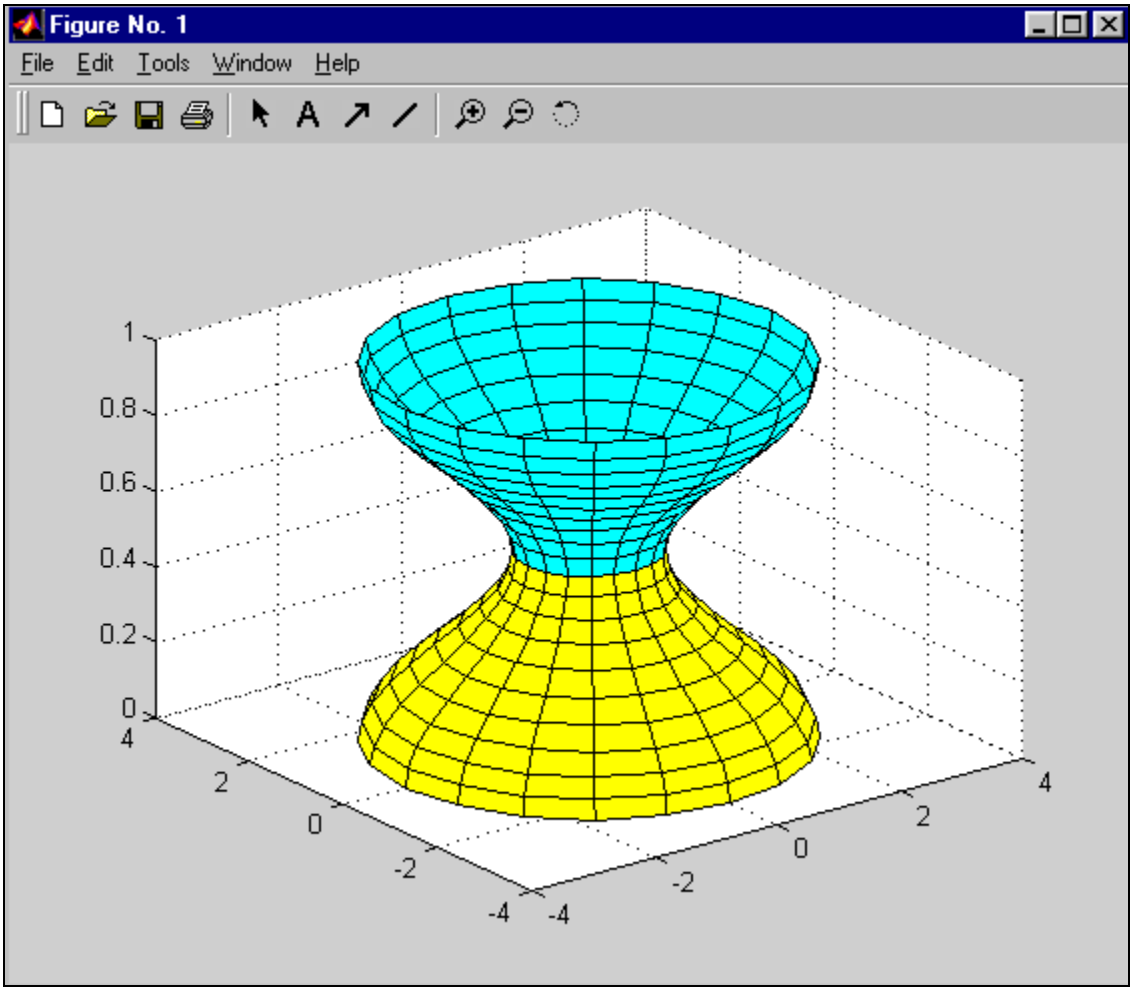


Figure No. 1
File Edit Tools Window Help
1
0.8
0.6
0.4
0.2
0
4
2
0
-2
-4
-4
-2
0
2
4

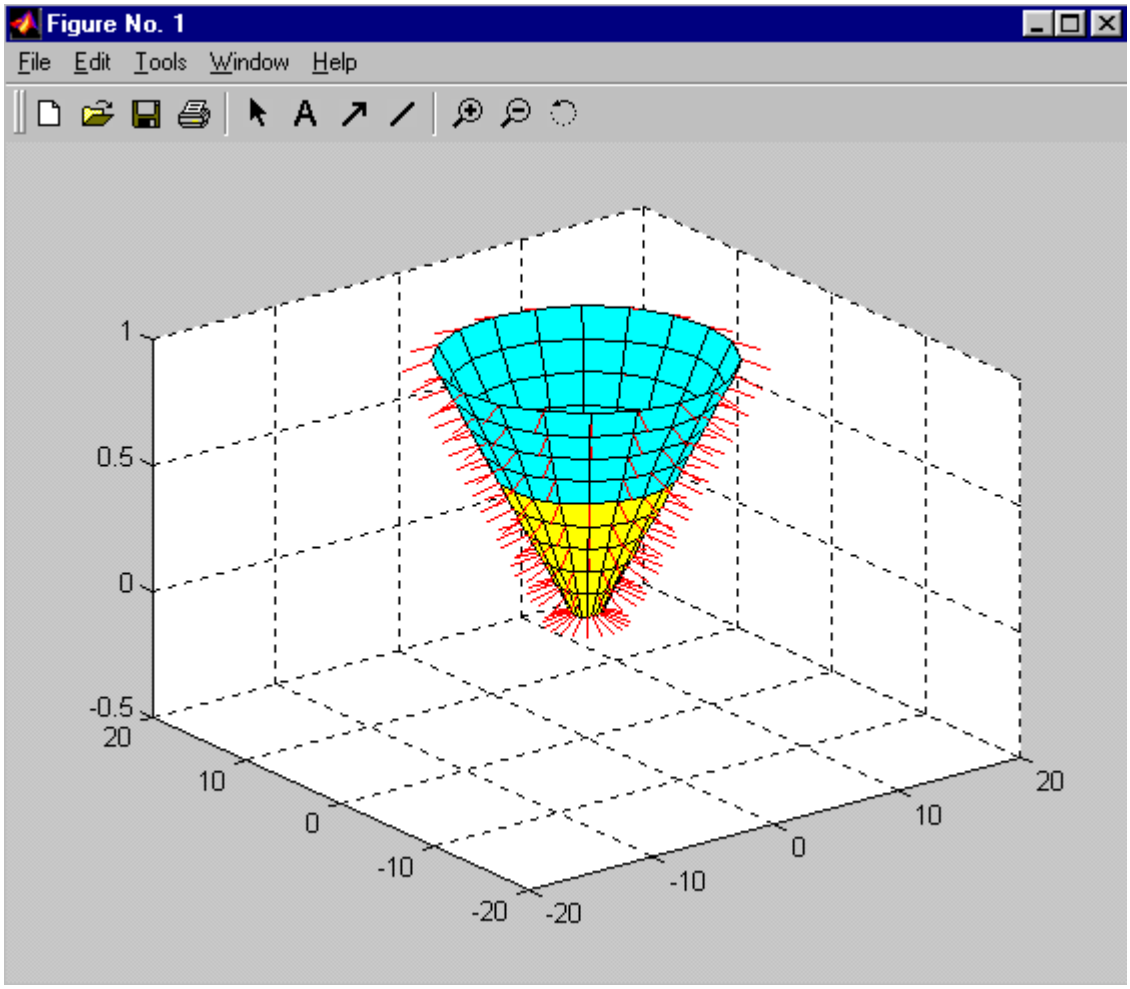


Figure No. 1
File Edit Tools Window Help
1
0.5
0
-0.5
20
10
0
-10
-20
-20
-10
0
10
20

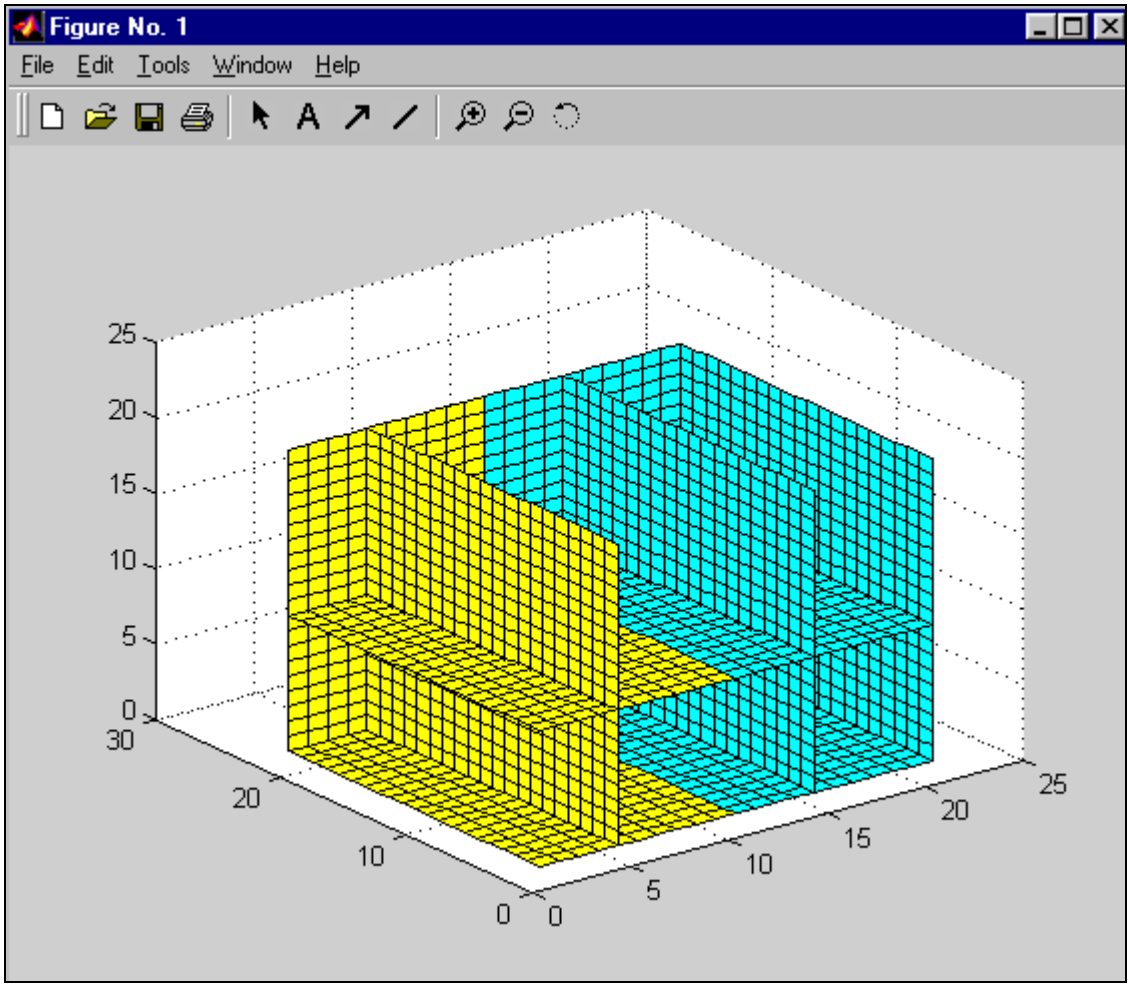


Figure No. 1
File Edit Tools Window Help
25
20
15
10
5
0
30
20
10
0
0
5
10
15
20
25

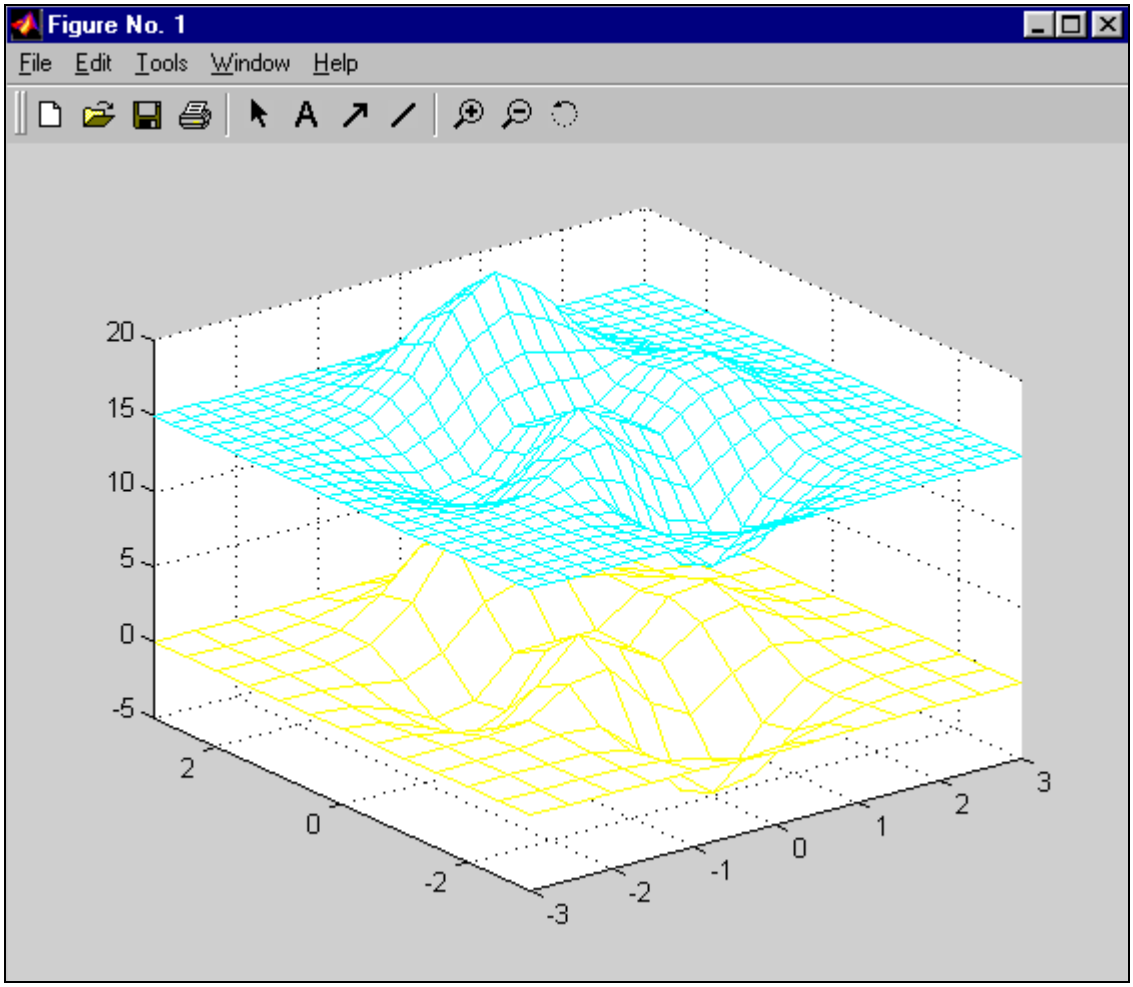


Figure No. 1
File Edit Tools Window Help
20
15
10
5
0
-5
2
0
-2
-3
-2
-1
0
1
2
3

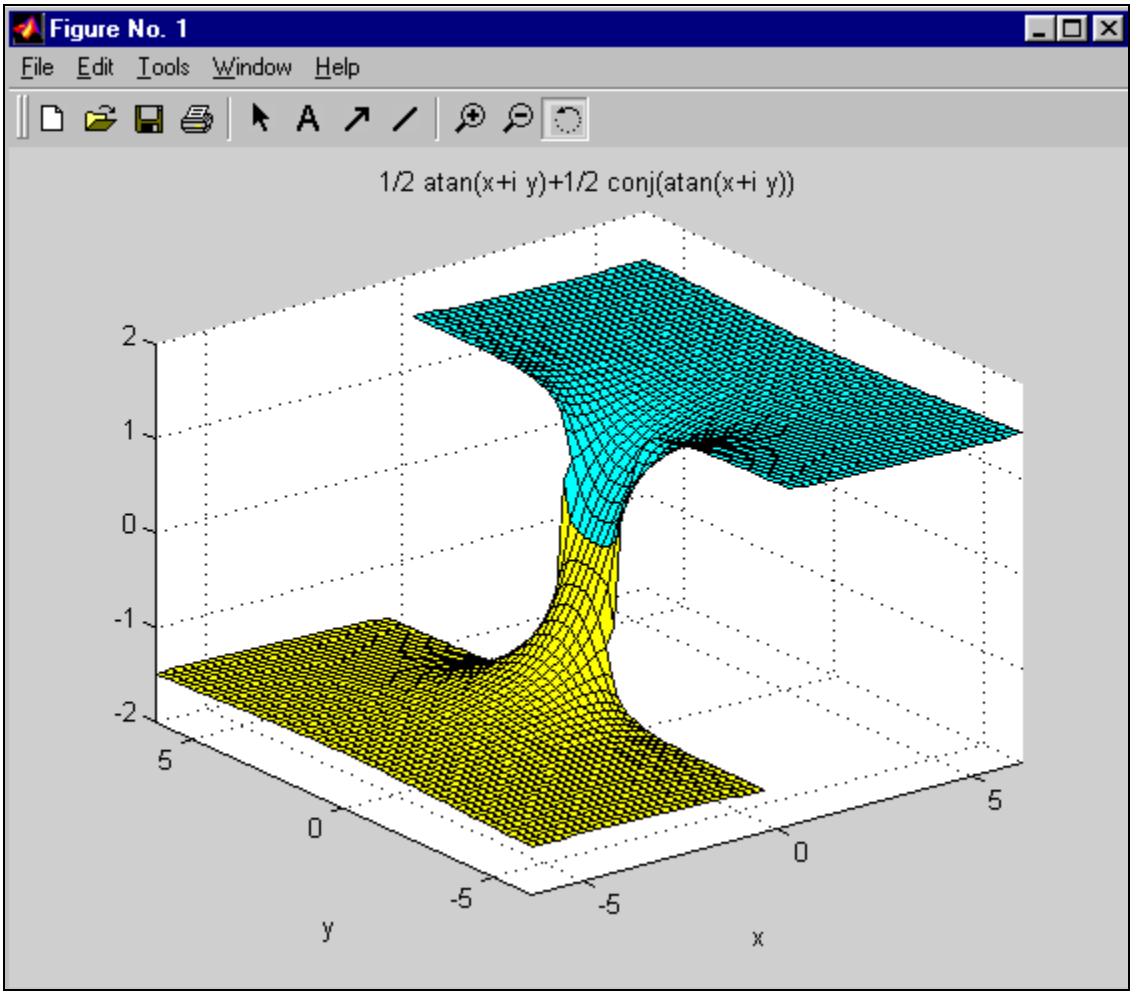
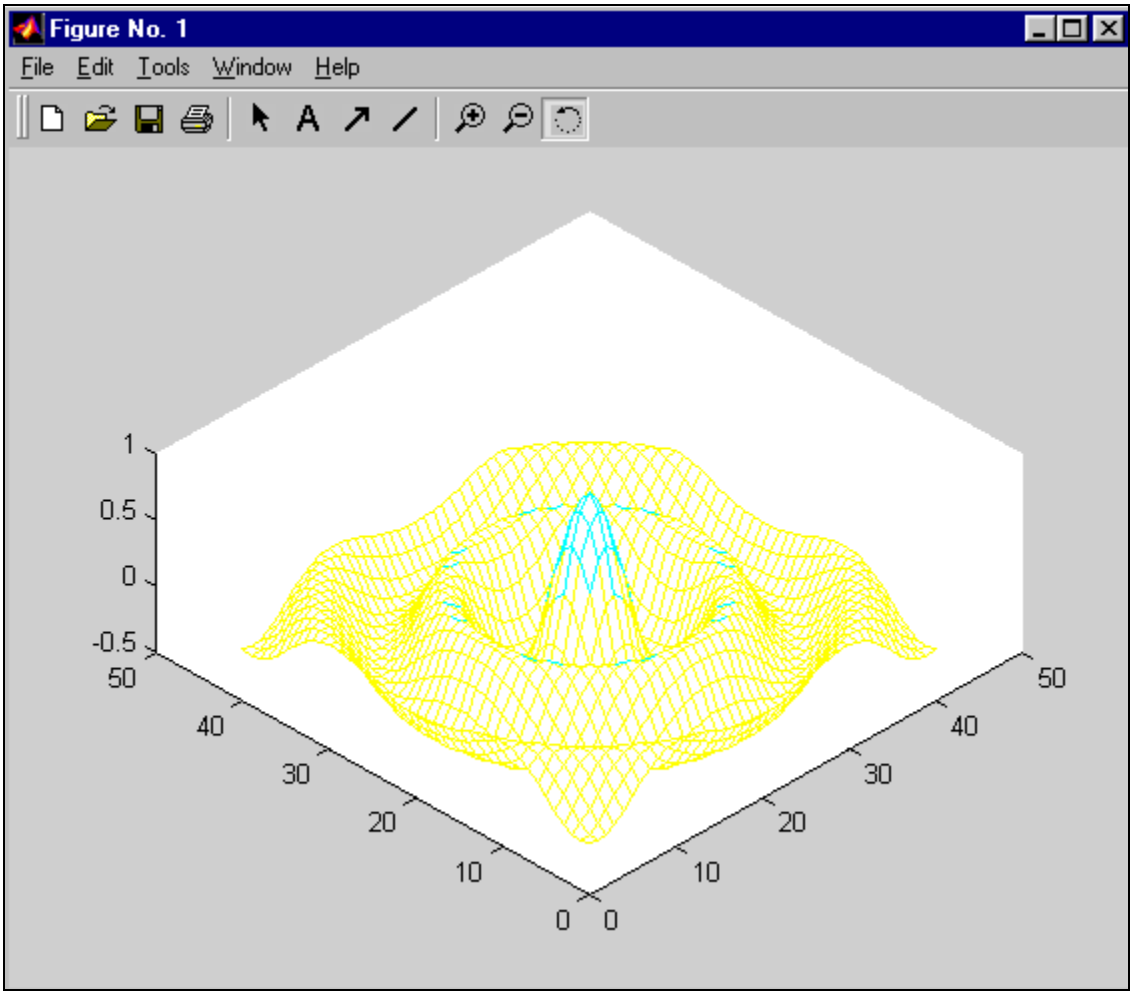


Figure No. 1
File Edit Tools Window Help
1/2 atan(x+i y)+1/2 conj(atan(x+i y))
2
1
0
-1
-2
5
0
-5
y
5
0
-5
x



Cũng là hàm **bessel** nhưng ta khảo sát riêng 1 bài:

```
%hm bessel
[x,y]=meshdom(-12:.6:12,-12:.6:12);
r=sqrt(x.^2+y.^2);
z=bessel(0,r);
m=[-45 60];
mesh(z,m)
```

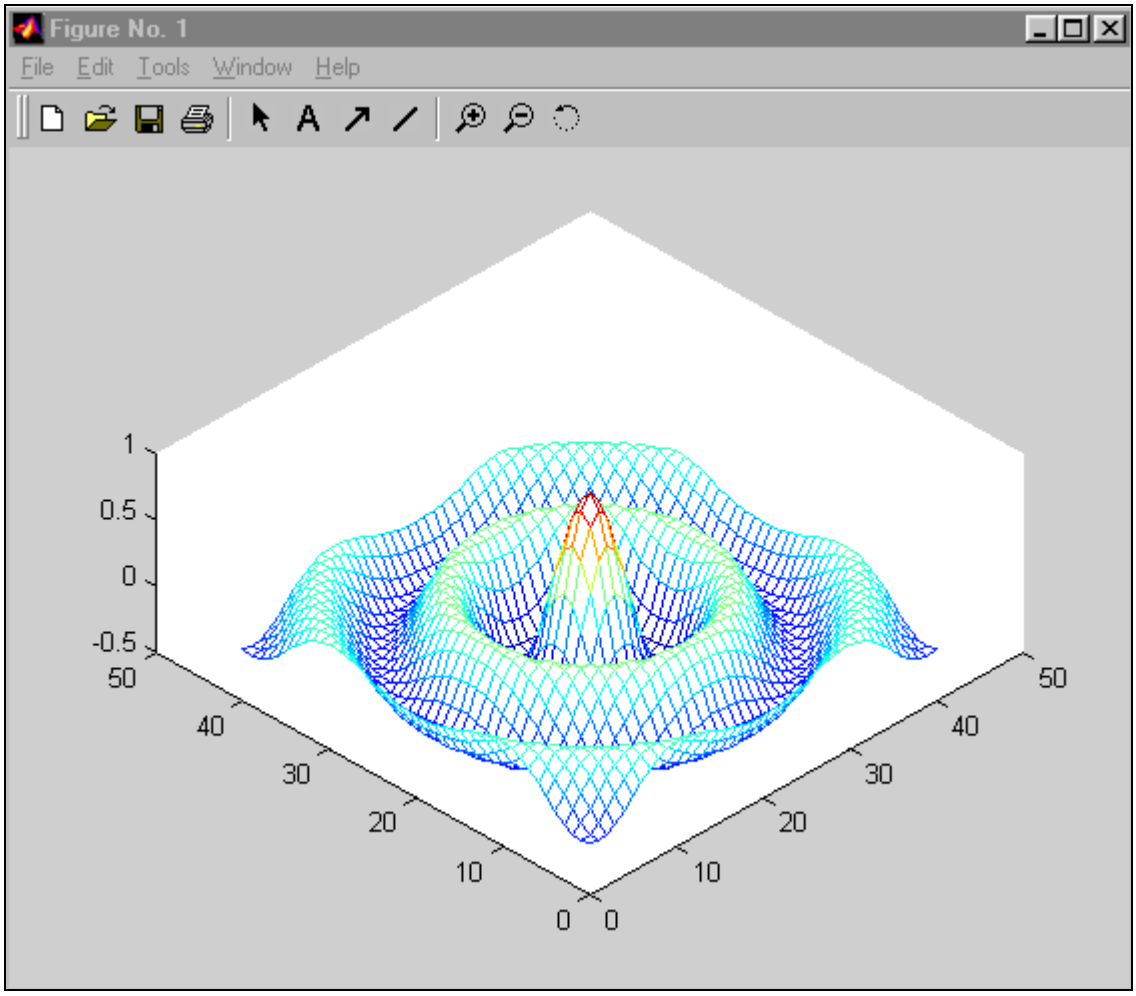


Figure No. 1
File Edit Tools Window Help
1
0.5
0
-0.5
50
40
30
20
10
0
0
10
20
30
40
50

NHÓM LỆNH VỀ ĐẶC ĐIỂM MÔ HÌNH

(Model Properties)

1. Lệnh COVAR, DCOVAR

a) Công dụng: (**Purpose)**

Tìm đáp ứng hiệp phương sai đối với nhiễu trắng (white noise).

b) Cú pháp: (**Syntax)**

[P,Q]= covar(a,b,c,d,w)

P = covar(num,den,w)

[P, Q]= dcovar(a,b,c,d,w)

P = dcovar(num,den,w)

c) Giải thích: (**Description)**

Covar tính các ngõ ra cố định và đáp ứng hiệp phương sai trạng thái của một hệ thống đối với các ngõ vàonhiễu trắng Gaussian với cường độ w:

$$E[w(t)w(\tau)']= w\delta(t -\tau)$$

[P,Q]= covar(a,b,c,d,w) tìm đáp ứng hiệp phương sai của hệ không gian trạng thái liên tục.

$$\dot{x} = Ax + Bu$$

$$y = Cx + Du$$

đối với nhiễu trắng với cường độ w từ tất cả các ngõ vào tới tất cả trạng thái và ngõ ra:

$$P = E[yy']$$

$$Q = E[xx']$$

Hệ thống phải ổn định và ma trận D phải là zero.

P = covar(num,den,w) tìm đáp ứng hiệp phương sai ngõ ra hệ SIMO của hàm truyền đa thức

$$G(s)= num(s)/den(s)$$

trong đó num và den chứa các hệ số đa thức theo chiều giảm dần số mũ của s, wlà cường độ nhiễu ngõ vào.

Để tìm đáp ứng hiệp phương sai của hệ gián đoạn ta dùng lệnh dcovar thay cho covar.

d) Ví dụ 1: (**Exemple**)

Tìm đáp ứng hiệp phương sai do nhiễu trắng Gaussian của hệ SISO với cường độ w=2 có hàm truyền:

$$H(s) = \frac{5s+1}{s^2+2s+3}$$

num = [5 1];

den = [1 2 3];

P = covar(num,den,2)

Ta được: P = 12.6667

2. Lệnh CTRB, OBSV

a) Công dụng:

Tạo ma trận có thể điều khiển và có thể quan sát.

b) Cú pháp:

co = ctrb(a,b)

ob = obsv(a,c)

c) Giải thích:

co = ctrb(a,b) tạo ma trận có thể điều khiển C_0 = [B ABA^2B ..... A^{n-1}B] cho hệ không gian trạng thái ob = obsv(a,c) tạo ma trận có thể quan sát O_b cho hệ không gian trạng thái.

$$O_b = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n-1} \end{bmatrix}$$

Hệ thống có thể điều khiển được nếu hạng của ma trận Co là n và có thể quan sát được nếu hạng của ma trận Ob là n.

d) Ví dụ:

Dùng lệnh ctrb và obsv để kiểm tra hệ thống (a,b,c,d) có thể điều khiển được hay có thể quan sát được hay không:

% Nhập hàm truyền và xác định không gian trạng thái:

num = [2 3];

den = [1 4 7];

[a,b,c,d]= tf2ss(num,den)

% Xác định ma trận có thể điều khiển và ma trận có thể quan sát:

co = ctrb(a,b)

ob = obsv(a,c)

% số trạng thái không thể điều khiển được:

```
    unco = length(a) – rank(co)
% số trạng thái không thể quan sát được:
    unob = length(a) – rank(ob)
```

Cuối cùng ta được kết quả:

```
a =
    -4    -7
     1     0
b =
     1
     0
c =
     2     3
d =    0
co =
     1    -4
     0     1
unco =    0
ob =
     2     3
    -5   -14
unob =  0
```

3. Lệnh DAMP, DDAMP

a) Công dụng:

Tìm tần số tự nhiên (Natural Frequencies) và hệ số tắt dần (Damping Factors).

b) Cú pháp:

```
[wn,Z]= damp(a)
mag= ddamp(a)
[mag,Wn,Z]= ddamp(a,Ts)
```

c) Giải thích:

Damp và ddamp tính tần số tự nhiên và hệ số tắt dần. Nếu bỏ các đối số bên trái trong các lệnh này thì ta nhận được một bảng các giá trị riêng, tỉ lệ tắt dần và tần số tự nhiên trên màn hình.

[wn,Z]= damp(a) tạo ra vector cột Wn và Z chứa các tần số tự nhiên wn, hệ số tắt dần của các giá trị riêng liên tục (Continous eigenvalues) được tính từ a. Biến a có thể là một trong các dạng sau:

+ Nếu a là ma trận vuông thì a được xem như là ma trận không gian trạng thái A.

+ Nếu a là vector hàng thì nó được xem như là vector chứa các hệ số đa thức của hàm truyền.

+ Nếu a là vector cột thì a chứa các nghiệm.

Mag = damp(a) tạo ra vector cột mag chứa biên độ các giá trị riêng gián đoạn được tính từ a. a có thể là một trong các dạng được nói đến ở trên.

[mag,Wn,Z]= ddamp(a,Ts) tạo ra các vector mag, Wn và Z chứa các biên độ, tần số tự nhiên trong mặt phẳng s tương ứng và hệ số tắt dần của các giá trị riêng của a. Ts là thời gian lấy mẫu. Hệ số tắt dần và tần số tự nhiên trong mặt phẳng s tương ứng của các giá trị riêng gián đoạn λ là:

$$\omega_n = \left|\frac{\log \lambda}{Ts}\right| \qquad \zeta = -\cos(\angle \log \lambda)$$

d) Ví dụ: (Trích từ trang 11-52 sách **'Control System Toolbox'**)

Tính và hiển thị các giá trị riêng, tần số tự nhiên và hệ số tắt dần của hàm truyền liên tục sau:

$$H(s) = \frac{2s^2 + 5s + 1}{s^2 + 2s + 3}$$

```
num =   [2   5   1];
den =   [1   2   3];
damp(den)
```

Eigenvalue	Damping	Freq.(rad/sec)
-1.0000 + 1.4142i	0.5774	1.7321
-1.0000 + 1.4142i	0.5774	1.7321

Tính và hiển thị các giá trị riêng, biên độ, tần số và hệ số tắt dần trong mặt phẳng s tương ứng của hàm truyền gián đoạn với thời gian lấy mẫu Ts = 0.1:

$$H(z) = \frac{2z^2 - 3.4z + 1.5}{z^2 - 1.6s + 0.8}$$

```
num = [2   -3.4   1.5]
den =  [1   -1.6   0.8]
ddamp(den,0.1)
```

Eigenvalue	Magnitude	Equiv.Damping	Equiv.Freq (rad/sec)
0.8000 + 0.4000i	0.8944	0.2340	4.7688
0.8000 – 0.4000i	0.8944	0.2340	4.7688

4. Lệnh DCGAIN, DDCGAIN

a) Công dụng:

Tìm độ lợi trạng thái xác lập của hệ thống.

b) Cú pháp:

k = dcgain(a,b,c,d)
k = dcgain(num,den)
k = ddcgain(a,b,c,d)
k = ddcgain(num,den)

c) Giải thích:

dcgain dùng để tính độ lợi trạng thái xác lập (DC hay tần số thấp) của hệ thống.

k = dcgain(a,b,c,d) tính độ lợi trạng thái xác lập của hệ không gian trạng thái liên tục:

$$\dot{x} = Ax + Bu$$

$$y = Cx + Du$$

từ tất cả các ngõ vào tới tất cả các ngõ ra:

$$K = -CA^{-1} + D$$

k = dcgain(num,den) tính độ lợi trạng thái xác lập của hàm truyền đa thức:

$$G(s) = \frac{num(s)}{den(s)}$$

trong đó num và den chứa các hệ số đa thức theo thứ tự giảm dần số mũ của s:

$$K = \left.\frac{num(s)}{den(s)}\right|_{s=0}$$

Để tính độ lợi DC của hệ gián đoạn ta dùng lệnh ddcgain thay cho lệnh dcgain. Đối với hệ không gian trạng thái xác lập, ma trận độ lợi DC là:

$$K = C(I - A)^{-1} + D$$

Và đối với hàm truyền gián đoạn, t độ LợI DC là:

$$K = \left.\frac{num(z)}{den(z)}\right|_{z=1}$$

d) Ví dụ 1:

Tính độ lợi DC của hệ thống có hàm truyền:

$$H(s) = \frac{2s^2 + 5s + 1}{s^2 + 2s + 3}$$

```
num = [ 2     5     1];
den = [1      2     3];
k = dcgain(num,den)
     k = 0.3333
```

Ví dụ 2: Tính độ lợi DC của hệ không gian trạng thái MIMO:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -0.5572 & -0.7814 \\ 0.7814 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 & 0.5397 \\ 0 & -0.2231 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

$$\begin{bmatrix} y \\ z \end{bmatrix} = \begin{bmatrix} 1.9691 & 6.4493 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

```
a = [-0.5572   -0.7814 ;  0.7814    0];
b = [1     0.5379 ;  0    -0.2231];
c = [1.9691   6.4493 ; 1   0];
d = [0     0 ; 0       0];
k = dcgain(a,b,c,d)
      k =
           8.2466   3.6861
              0      0.2855
```

5. <u>Lệnh GRAM, DGRAM</u>

a) <u>Công dụng</u>:

Đánh giá khả năng điều khiển và khả năng quan sát.

b) <u>Cú pháp</u>:

Gc = gram(a,b)

Go = gram(a',c')

Gc = dgram(a,b)

Go = dgram(a',c')

c) <u>Giải thích</u>:

gram tính toán khả năng điều khiển và khả năng quan sát. Sự đánh giá này có thể được dùng để nghiên cứu đặc tính điều khiển và đặc tính quan sát của các hệ không gian trạng thái và giảm bậc mô hình.

gram(a,b) tạo ra sự đánh giá khả năng điều khiển G_c:

$$G_c = \int_0^\infty e^{A\tau} BB' e^{A'\tau} d\tau$$

đó là một ma trận đối xứng; hơn nữa, nếu ma trận có hạng đủ (bằng kích thước của ma trận đánh giá) thì hệ thống có thể điều khiển được.

Go = gram(a',c') tạo ra sự đánh giá khả năng quan sát G_o:

$$G_o = \int_0^\infty e^{A\tau} CC' e^{A'\tau} d\tau$$

Nếu ma trận đánh giá có hạng đủ thì hệ thống có thể quan sát được.

dgram dùng cho các hệ thống gián đoạn.

d) <u>Ví dụ</u>:

Xác định khả năng điều khiển của hệ không gian trạng thái ở ví dụ về lệnh dcgrain

```
a = [-0.5572   -0.7814  ; 0.7814    0];
b = [1     0.5379  ;  0   -0.2231];
c = [1.9691   6.4439 ;  1    0];
```

```
d = [0    0  ;  0    0];
Gc = gram(a,b)
```

Ta nhận được ma trận:

```
Gc =
     1.2016   -0.0318
    -0.0318    1.0708
```

Tìm hạng ma trận bằng lệnh:

```
r = rank(Gc)
```

ta được r = 2 và bằng kích thước của ma trận đánh giá. Vậy hệ thống này có thể điều khiển được.

6. Lệnh DSORT, ESORT

a) Công dụng:

Sắp xếp các giá trị riêng theo thứ tự phần thực hoặc biên độ số phức.

b) Cú pháp:

```
s = dsort(p)
[s,ndx] = dsort(p)
s = esort(p)
[s,ndx] = esort(p)
```

c) Giải thích:

s = esort(p) xếp các giá trị riêng phức trong vector p theo thứ tự giảm dần của phần thực. Đối với các giá trị riêng liên tục, các giá trị riêng không ổn định xuất hiện trước.

s = dsort(p) xếp các gí trị riêng phức trong vector p theo thứ tự giảm dần của biên độ. Đối với các giá trị riêng gián đoạn, cá giá trị riêng không ổn định xuất hiện trước.

[s,ndx] = dsort(p) hay [s,ndx] = esort(p) cũng tạo ra vector ndx chứa các chỉ số dùng theo thứ tự.

d) Ví dụ:

Xếp các phần tử của vector p = [2+3j -3+j 1-9j 3-7j 5+2j 6-j] theo thứ tự giảm dần của phân thực và độ lớn số phức.

```
p = [2+3j     -3+j  1-9j  3-7j  5+2j  6-j]
% Xếp theo thứ tự giảm dần của độ lớn số phức:
    s = dsort(h)
        s =
            1.0000 + 9.0000j
            3.0000 + 7.0000j
            6.0000 + 1.0000j
            5.0000 – 2.0000j
            2.0000 + 3.0000j
           -3.0000 + 1.0000j
% Xếp theo thứ tự giảm dần của phần thực:
    s' = esort(h)
            6.0000 + 1.0000j
            5.0000 – 2.0000j
            3.0000 + 7.0000j
```

2.0000 – 3.0000j
1.0000 + 9.0000j
-3.0000 – 1.0000j

7. Lệnh EIG

a) Công dụng:

Tìm các giá trị riêng và các vector riêng của hệ thống.

b) Cú pháp:

E = eig(X)
[V,D] = eig(X)
[V,D] = eig(X)
[V,D] = eig(X,'nobalance')
E = eig(A,B)
[V,D] = eig(A,B)

c) Giải thích:

E = eig(X) là một vector chứa các giá trị riêng của ma trận vuông X.

[V,D] = eig(X) tạo ra một ma trận đường chéo D của các giá trị riêng và ma trận đủ với các cột là các vector riêng tương ứng để cho X*V = V*D.

[V,D] = eig(X,'nobalance') giống như [V,D] = eig(X) nhưng bỏ qua sự cân bằng. Cách này đôi khi cho kết quả chính xác hơn.

E = eig(A,B) là vector chứa các giá trị riêng phổ biến của các ma trận vuông A và B.

[V,D] = eig(A,B) tạo ra ma trận đường chéo D của các giá trị riêng phổ biến và các ma trận đủ V với các cột là các vector riêng tương ứng để cho A*V = B*V*D.

d) Ví dụ:

Cho X = [2+3j -3+j 1-9j ; 3-7j 5+2j 6-j ; 0+7j 6-8j 2+5j]. tìm các giá trị riêng của X.

```
X = [2+3j    -3+j    1-9j ; 3-7j    5+2j   6-j ; 0+7j   6-8j   2+5j];
[V,D] = eig(X)
    V =
        0.4158 + 0.3442j      0.5455 + 0.4929j      0.4344 – 0.2255j
       -0.3275 + 0.3580j      0.1837 – 0.2659j      0.5974 + 0.1368j
        0.1209 – 0.6772j     -0.5243 + 0.2831j      0.4954 + 0.3734j
    D =
       -9.3743 + 4.7955j            0                     0
                0                   9.2099 + 0.2831j      0
                0                   0                     9.1644 – 2.2542j
```

8. Lệnh PRINTSYS

a) Công dụng:

In ra các tham số của hệ thống tuyến tính

b) Cú pháp:

printsys(a,b,c,d)
printsys(a,b,c,d,ulabels,ylabels,xlabels)
printsys(num,den,'s')

printsys(num,den,'z')

c) Giải thích:

printsys in các tham số của hệ không gian trạng thái và hàm truyền theo dạng đặc biệt. Đối với hệ không gian trạng thái, các ngõ vào, ngõ ra và trạng tháicủa hệ được đặt tên và hàm truyền được hiển thị dưới dạng tỷ số của hai đa thức.

printsys(a,b,c,d) in ra hệ không gian trạng thái (a,b,c,d) với tên tham số ở phía trên và phía bên trái của ma trận hệ thống.

printsys(a,b,c,d,ulabels,ylabels,xlabels) in ra hệ không gian trạng thái với tên tham số được chỉ định bởi các vector ulabels, ylabels và xlabels. ulabels, ylabels và xlabels chứa tên ngõ vào, ngõ ra và trạng thái của hệ thống.

printsys(num,den,'s') hoặc printsys(num,den,'z') in ra hàm truyền dưới dạng tỷ số của hai đa thức theo s hoặc z. Nếu biến của hàm truyền ('s' hoặc 'z') không được chỉ định thì phép biến đổi Laplace ('s') được thừa nhận.

d) Ví dụ:

Cho hệ không gian trạng thái sau:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 2 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u$$

$$y = \begin{bmatrix} 2 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + [1]\, u$$

In ra hệ không gian trạng thái với tên gọi các tham số mặc nhiên và với tên được chỉ định như sau: ngõ vào u là sensor, trạng thái x là alpha và beta, ngõ ra là angle.

```
% Khai báo hệ thống:
    a = [1    1 ; 2   -1];
    b = [1 ; 0];
    c = [2    4];
    d = 1;
% In theo tên mặc nhiên:
    printsys(a,b,c,d)
        a =
                x1          x2
        x1   1.00000    1.00000
        x2   2.00000   -1.00000
    b =
                u1
        x1    1.00000
        x2          0
    c =
                x1          x2
        y1    2.00000    4.00000
    d =
                u1
```

```
        y1     1.00000
% Chỉ định tên tham số:
    inputs = 'sensor';
    outputs = 'angle';
    states = 'alpha beta';
    states = 'alpha beta';
% In theo tên đã chỉ định:
    printsys(a,b,c,d,inputs,outputs,states)
a =
                    alpha         beta
        alpha     1.00000     1.00000
        beta      2.00000    -1.00000
    b =
                    sensor
        alpha     1.00000
         beta           0
    c =
                    alpha         beta
        angle     2.00000     4.00000
    d =
                    sensor
        angle     1.00000
```

9. Lệnh TZERO

a) Công dụng:

Tìm zero truyền đạt của hệ không gian trạng thái.

b) Cú pháp:

```
z = tzero(sys)
[z,gain] = tzero(sys)
z = tzero(a,b,c,d)
```

c) Giải thích:

z = tzero(sys) tìm các zero truyền đạt của hệ thống LTI trong sys.

[z,gain] = tzero(sys) tìm độ lợi hàm truyền nếu hệ thống là hệ SISO.

z = tzero(a,b,c,d) tìm zero truyền đạt của hệ không gian trạng thái:

$\dot{x} = Ax + Bu$ hoặc $x[n+1\} = Ax[n] + Bu[n]$

$y = Cx + Du$ $y[n] = Cx[n] + Du[n]$

d) Ví dụ:

Tìm zero truyền đạt của hệ không gian trạng thái sau:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 2 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u$$

$$y = \begin{bmatrix} 2 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \end{bmatrix} u$$

```
a = [1     1 ; 2    -1];
b = [1 ; 0];
c = [2     4];
d = 1;
z = tzero(a,b,c,d)
      z =
            -1.0000 + 2.4495j
            -1.0000 – 2.4495j
```

NHÓM LỆNH XÂY DỰNG MÔ HÌNH

(Model Building)

1. Lệnh APPEND

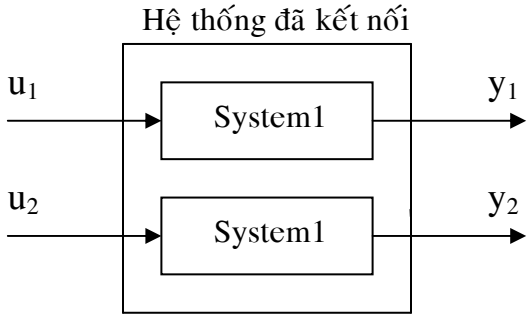
a) Công dụng:

Kết hợp động học 2 hệ thống không gian trạng thái.

b) Cú pháp:

[a,b,c,d] = append(a1,b1,c1,d1,a2,b2,c2,d2)

c) Giải thích:



Lệnh append kết nối động học 2 hệ thống không gian trạng thái tạo thành 1 hệ thống chung.

[a,b,c,d] = append(a1,b1,c1,d1,a2,b2,c2,d2) tạo ra hệ thống không gian trạng thái kết hợp bao gồm hệ thống 1 và hệ thống 2. Hệ thống nhận được là:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} B_1 & 0 \\ 0 & B_2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} C_1 & 0 \\ 0 & C_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

d) Ví dụ 1: Cho 2 hệ không gian trạng thái

$$\begin{cases} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 2 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u \\ y = \begin{bmatrix} 2 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + [1] u \end{cases} \quad \textbf{(Hệ I)}$$

$$\begin{cases} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 4 & 3 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u \\ y = \begin{bmatrix} 4 & -2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + [0] u \end{cases} \quad \textbf{(Hệ II)}$$

Kết nối 2 hệ không gian trạng thái trên để tạo ra một hệ không gian trạng thái kết hợp.

```
a1 = [1   1;2  -1];
b1 = [1; 0];
c1 = [2   4];
d1 = [1];
a2 = [4   3;1  0];
b2 = [1; 0];
c2 = [4   -2];
d2 = [0];
[a,b,c,d] = append(a1,b1,c1,d1,a2,b2,c2,d2)
```

```
a =
     1     1     0     0
     2    -1     0     0
     0     0     4     3
     0     0     1     0
b =
     1     0
     0     0
     0     1
     0     0
c =
     2     4     0     0
     0     0     4    -2
d =
     1     0
     0     0
```

Ví dụ 2: Trích từ **Ví dụ 3.12** sách **'Ứng dụng Matlab trong điều khiển tự động'** tác giả Nguyễn Văn giáp. Và được viết bởi file.m

```
%KET NOI HAI HE THONG SONG SONG
a=[1 2 3;4 5 6;7 8 9];
b=[3 4;4 5;7 9];
c=[0 0 1];
d=[0 0];
e=[1 9 3;4 5 6;7 8 7];
f=[2 4;4 6;7 9];
g=[0 1 1];
h=[0 0];
[A,B,C,D]= append(a,b,c,d,e,f,g,h)
```

Kết quả:

A =

1	2	3	0	0	0
4	5	6	0	0	0
7	8	9	0	0	0
0	0	0	1	9	3
0	0	0	4	5	6
0	0	0	7	8	7

B =

3	4	0	0
4	5	0	0
7	9	0	0
0	0	2	4
0	0	4	6
0	0	7	9

C =

0	0	1	0	0	0

0 0 0 0 1 1

D =

0 0 0 0

0 0 0 0

2. Lệnh AUSTATE

a) Công dụng:

Thêm vào hệ không gian trạng thái các ngõ ra.

b) Cú pháp:

[ab,bb,cb,db] = austate(a,b,c,d)

c) Giải thích:

[ab,bb,cb,db] = austate(a,b,c,d) tạo ra một hệ không gian trạng thái mới và số ngõ vào bằng số ngõ vào hệ ban đầu nhưng số ngõ ra nhiều hơn. Kết quả ta được hệ thống sau:

$$\dot{x} = Ax + Bu$$
$$\begin{bmatrix} y \\ x \end{bmatrix} = \begin{bmatrix} C \\ 1 \end{bmatrix} x + \begin{bmatrix} D \\ 0 \end{bmatrix} u \qquad (1.2)$$

d) Ví dụ:

Cho hệ không gian trạng thái có:

a = b = c = d =

4 5 3 2 1 3 1 2

6 7 6 1 2 4 3 4

Dùng lệnh:

[ab,bb,cb,db] = augstate(a,b,c,d) ta được hệ mới như hệ (1.2) có:

ab = bb =

1 2 4 5

3 4 6 7

cb = db =

1 3 3 2

2 4 6 1

1 0 0 0

0 1 0 0

3. Lệnh BLKBUILD, CONNECT

a) Công dụng:

Chuyển sơ đồ khối thành mô hình không gian trạng thái.

b) Cú pháp:

blkbuild

[aa,bb,cc,dd] = connect(a,b,c,d,Q,inputs,outputs)

c) Giải thích:

[aa,bb,cc,dd] = connect(a,b,c,d,Q,inputs,outputs) tạo ra các ma trận mô hình không gian trạng thái (ac,bc.cc,dc) của hệ thống trong sơ đồ khối, các ma trận (a,b,c,d) và ma trận Q (ma trận cho biết sự kết nối bên trong hệ thống). Vector inputs và outputs dùng để chọn các ngõ vào và ngõ ra sau cùng cho hệ thống (ac,bc,cc,dc).

Việc thực hiện xây dựng mô hình dùng lệnh connect được thực hiện qua các bước:

c.1) Xác định hàm truyền hay hệ thống không gian trạng thái: nhập các hệ số số của tử số và mẫu số mỗi hàm truyền sử dụng tên biến n1, n2, n3, ..., và d1, d2, d3,... hoặc nhập ma trận (A,B,C,D) sử dụng tên biến a1, b1, c1, d1; a2, b2, c2, d2; a3, b3, c3, d3,...

c.2) Xây dựng mô hình không gian trạng thái chưa nối: hình thành mô hình bao gồm tất cả hàm truyền chưa được kết nối. Điều này được thực hiện bằng cách lặp đi lặp lại lệnh append cho các khối không gian trạng thái hay **tf2ss** và **append** cho các khối hàm truyền. **tf2ss** có thể chuyển mỗi khối thành hệ không gian trạng thái nhỏ sau đó dùng lệnh **append** để tập hợp các khối nhỏ thành một mô hình hoàn chỉnh.

c.3) Chỉ ra các kết nối bên trong: xác định ma trận Q chỉ ra cách kết nối các khối của sơ đồ khối. Trong một hàng của ma trận Q thành phần đầu tiên là số ngõ vào. Những thành phần tiếp theo chỉ các ngõ được nối vào ngõ vào trên.

Ví dụ: nếu ngõ vào 7 nhận các ngõ vào khác từ ngõ ra 2, 15 và 6 trong đó ngõ vào âm thì hàng tương ứng trong Q là [7 2 -15 6].

c.4) Chọn ngõ vào và ngõ ra: tạo các vector inputs và outputs để chỉ ra ngõ vào và ngõ ra nào được duy trì làm ngõ vào và ngõ ra của hệ thống.

Ví dụ: nếu ngõ vào 1, 2 và 15 và ngõ ra 2 và 7 được duy trì thì inputs và outputs là:

inputs = [1 2 15]

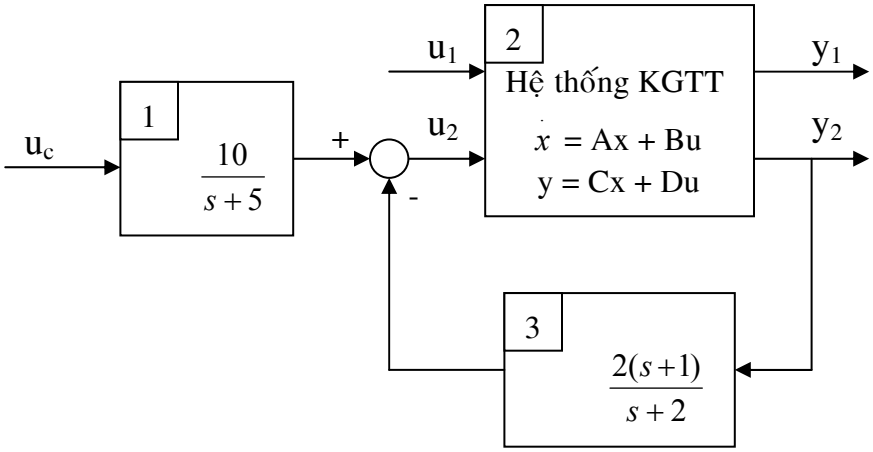
outputs = [2 7]

c.5) Kết nối bên trong: dùng lệnh:

[ac,bc,cc,dc] = connect(a,b,c,d,Q,inputs,outputs) lệnh này lấy thông tin trong ma trận Q tiến hành nối chéo các khối tạo thành hệ thống với các ngõ vào và các ngõ ra được chọn bởi biến inputs và outputs.

d) Ví dụ :

Xét sơ đồ khối của hệ MIMO (Mylti Input Milti Output) sau:



Để tạo ra mô hình không gian trạng thái của hệ thống này, ta sử dụng các lệnh sau:

```
% Khai báo hàm truyền khâu (1):
    n1 = 10;
    d1 = [1   5];
% Khai báo các ma trận của hệ không gian trạng thái (2):
    a2 = [1    2
          -5   3];
    b2 = [2   -4
          6    5];
    c2 = [-3   9
           0    4];
    d2 = [2    1
          -5   6];
% Khai báo hàm truyền khâu điều khiển (3):
    n3 = 2*[1    1];
    d3 = [1    2];
% Khai báo số khâu của sơ đồ khối:
```

```
nblocks = 3;
```

% Thực hiện các lệnh kết nối:

```
blkbuild;
```

% Khai báo ma trận điều khiển kết nối bên trong (Q):

```
Q = [3   1   -4
     4   3    0];
inputs = [1   2]
outputs = [2   3];
[ac,bc,cc,dc] = connect(a,b,c,d,Q,inputs,outputs)
```

Và ta được hệ thống có các ma trận ac, bc, cc, dc như sau:

```
ac =
   -5.0000        0         0         0
   -3.0769   1.0000    4.4615   -6.6154
    3.8462   -5.0000  -0.0769    0.7692
    4.6154        0    0.3077   -1.0769
bc =
    1.0000
         0   -1.0769
         0    9.8462
         0   -0.3846
cc =
    0.7692   -3.0000    8.3846    0.1538
    4.6154         0    0.3077    0.9231
dc =
    0    2.7692
    0   -0.3846
```

Hệ thống này có 2 ngõ vào là 1 và 2 và có 2 ngõ ra là 2 và 3.

4. Lệnh CLOOP

a) Công dụng:

Hình thành hệ thống không gian trạng thái vòng kín.

b) Cú pháp:

[ac,bc,cc,dc] = cloop(a,b,c,d,sign)

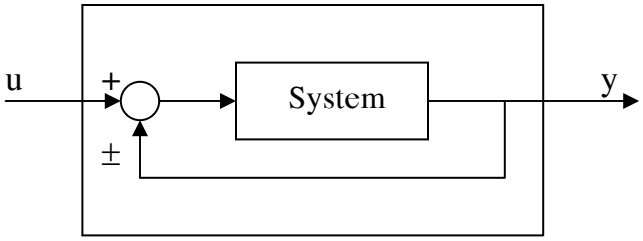
[ac,bc,cc,dc] = cloop(a,b,c,d,inputs,outputs)

[numc,denc] = cloop(num,den,sign)

c) Giải thích:

cloop tạo ra hệ thống vòng kín bằng cách hồi tiếp các ngõ ra và các ngõ vào của hệ thống. Tất cả các ngõ vào và ngõ ra của hệ vòng hở được giữ lại trong hệ vòng kín. cloop sử dụng được cho cả hệ liên tục và gián đoạn.

[ac,bc,cc,dc] = cloop(a,b,c,d,sign) tạo ra mô hình không gian trạng thái của hệ vòng kín bằng cách hồi tiếp tất cả ngõ ra tới tất cả các ngõ vào.



Hệ thống vòng kín

sign = 1: hồi tiếp dương.

sign = -1: hồi tiếp âm.

Nếu không có tham số sign thì xem như là hồi tiếp âm.

Kết quả ta được hệthống vòng kín:

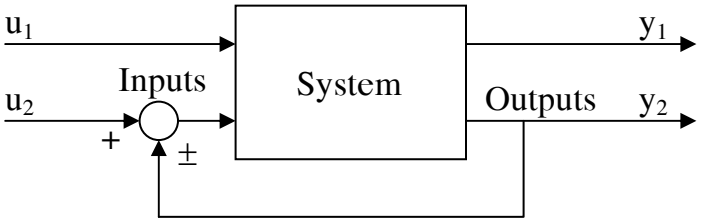
$$\dot{x} = \left[A \pm B(I \mp D)^{-1}\ C\right]x + \left[B(I \mp D)^{-1}\right]u$$
$$y = \left[C \pm D(I \mp D)^{-1}\ C\right]x + \left[D(I \mp C)^{-1}\right]u$$

trong đó dấu "-" ứng với hồi tiếp dương và dấu "+" ứng với hồi tiếp âm.

[numc,denc]= cloop(num,den,sign) thực hiện hồi tiếp đơn vị với dấu được cho bởi tham số sign để tạo ra hệ thống vòng kín có hàm truyền đa thức.

$$\frac{num(s)}{den(s)} = \frac{G(s)}{1 \mp G(s)} = \frac{num(s)}{den(s) \mp num(s)}$$

[ac,bc,cc,dc] = cloop(a,b,c,d,outputs,inputs) thực hiện hồi tiếp các ngõ ra được chỉ định trong vector outputs về ngõ vào được chỉ định rõ trong vector inputs để tạora mô hình không gian trạng thái của hệ vòng kín.



Hệ thống vòng kín

Vector outputs chứa chỉ số các ngõ ra nào được hồi tiếp về ngõ vào. Trong trường hợp này, hồi tiếp dương được sử dụng. Muốn chọn hồi tiếp âm, ta dùng tham số –inputs thay cho inputs.

d) Ví dụ:

Xét hệ không gian trạng thái (a,b,c,d) có 5 ngõ ra và 8 ngõ vào. Để hồi tiếp các ngõ ra 1, 3 và 5 về các ngõ vào 2, 8 và 7 và chọn hồi tiếp âm.

```
outputs = [1    3    5];
inputs = [2    8    7];
[ac,bc,cc,dc] = cloop(a,b,c,d,outputs,-inputs)
```

Cho hệ không gian trạng thái:

$$\dot{x} = Ax + \begin{bmatrix} B_1 & B_2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} x + \begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

Giả sử vòng kín được tạo ra bằng cách hồi tiếp ngõ ra y_2 về ngõ vào u_2 thì ta được hệ không gian trạng thái:

$$\dot{x} = \begin{bmatrix} A \pm B_2 E C_2 \end{bmatrix} x + \begin{bmatrix} B_1 \pm B_2 E D_{21} & B_2 E \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} C_1 \pm D_{12} E C_2 \\ C_2 \pm D_{22} E C_2 \end{bmatrix} x + \begin{bmatrix} D_{11} \pm D_{12} E D_{21} & D_{12} E \\ D_{21} \pm D_{22} E D_{21} & D_{22} E \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

trong đó $E = (I \mp D_2 D_1)^{-1}$ với I là ma trận đơn vị.

Các biểu thức trên đều đúng cho mô hình gián đoạn khi thay phép vi phân bằng phép sai phân và hàm truyền trong mặt phẳng z thay cho hàm truyền trong mặt phẳng s. Chú ý: ma trận $(I \mp D_2 D_1)^{-1}$ phải có thể nghịch đảo được.

5. Lệnh FEEDBACK

a) Công dụng:

Kết nối hồi tiếp hai hệ thống.

b) Cú pháp:

[a,b,c,d] = feedback(a1,b1,c1,d1,a2,b2,c2,d2)

[a,b,c,d] = feedback(a1,b1,c1,d1,a2,b2,c2,d2,sign)

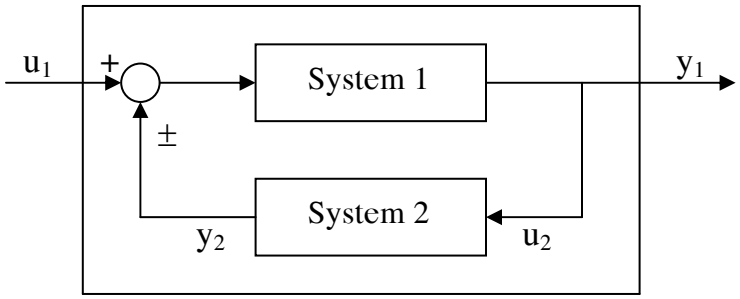
[a,b,c,d] = feedback(a1,b1,c1,d1,a2,b2,c2,d2, inputs1, outputs1)

[num,den] = feedback(num1,den1, num2,den2)

[num,den] = feedback(num1,den1, num2,den2,sign)

c) Giải thích:

[a,b,c,d] = feedback(a1,b1,c1,d1,a2,b2,c2,d2,sign) tạo ra hệ thống không gian trạng thái tổ hợp với kết nối hồi tiếp của hệ thống 1 và 2:



Hệ thống hồi tiếp

Hệ thống hồi tiếp được tạo ra bằng cách nối các ngõ ra của hệ thống 1 tới các ngõ vào của hệ thống 2 và các ngõ ra của hệ thống 2 tới các ngõ vào của hệ thống 1.

sign = 1: Hồi tiếp dương.

sign = -1: Hồi tiếp âm.

Nếu bỏ qua tham số sign thì lệnh sẽ hiểu là hồi tiếp âm.

Sau khi hồi tiếp ta thu được thống:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} A_1 \pm B_1 E D_2 C_1 & \pm B_1 E C_2 \\ B_2 C_1 \pm B_2 D_2 E D_2 C_1 & A \pm B_2 D_2 E C_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} B_1 (I \pm E D_2 D_1) \\ B_2 D_1 (I \pm E D_2 D_1) \end{bmatrix} u_1$$

$$y_1 = \begin{bmatrix} C_1 \pm D_1 E D_2 C_1 & \pm D_1 E C_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} D_1 (I \pm E D_2 D_1 \end{bmatrix} u_1$$

trong đó:

$E = (I \mp D_2 D_1)^{-1}$ với I là ma trận đơn vị, dấu "-" ứng với hồi tiếp dương và dấu "+" ứng với hồi tiếp âm.

[num,den] = feedback(num1,den1, num2,den2,sign) tạo ra hàm truyền đa thức của hệ thống hồi tiếp.

sign = 1: Hồi tiếp dương.

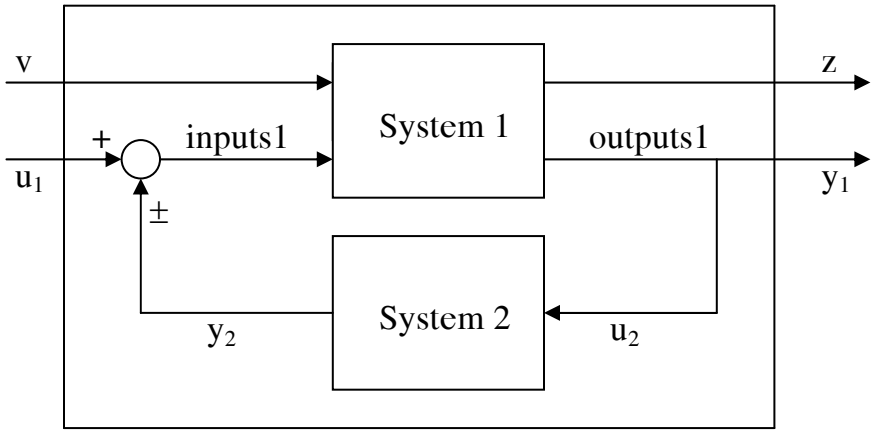
sign = -1: Hồi tiếp âm.

Nếu bỏ qua tham số sign thì lệnh sẽ hiểu là hồi tiếp âm.

Hàm truyền của hệ thống là:

$$\frac{num(s)}{den(s)} = \frac{G_1(s)}{1 \mp G_1(s)G_2(s)} = \frac{num_1(s)den_2(s)}{den_1(s)den_2(s) \mp num_1(s)num_2(s)}$$

[a,b,c,d] = feedback(a1,b1,c1,d1,a2,b2,c2,d2, inputs1, outputs1) tạo ra hệ thống hồi tiếp bằng cách hồi tiếp các ngõ ra trong outputs của hệ thống 2 tới các ngõ vào trong inputs của hệ thống 1.

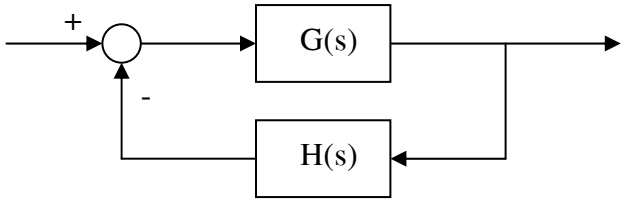


Hệ thống hồi tiếp

Vector inputs 1 chứa các chỉ số ngõ vào của hệ thống 1 và chỉ ra ngõ ra nào của hệ thống 1 được chọn hôi tiếp. Vector outputs1 chứa các chỉ số ngõ ra của hệ thống 1 và chỉ ra ngõ ra nào của hệ thống 1 được hồi tiếp về ngõ vào của hệ thống 2. Trong hệ thống này, hồi tiếp là hồi tiếp dương. Nếu muốn dùng hồi tiếp âm thì dùng tham số –inputs thay cho inputs1.

d) Ví dụ:

Kết nối khâu có hàm truyền $G(s) = \frac{2s^2 + 5s + 1}{s^2 + s + 3}$ với khâu hồi tiếp có hàm truyền $H(s) = \frac{5(s+2)}{s+10}$ theo dạng hồi tiếp âm như sau:



numg = [2 5 1];

deng = [1 2 3];

numh = [5 10];

denh = [1 10];

[num,den] = feedback(numg, deng, numh, denh);

Kết quả:

num =

2 25 51 10

den =

11 57 78 40

6. Lệnh PARALLEL

a) Công dụng:

Nối song song các hệ thống.

b) Cú pháp:

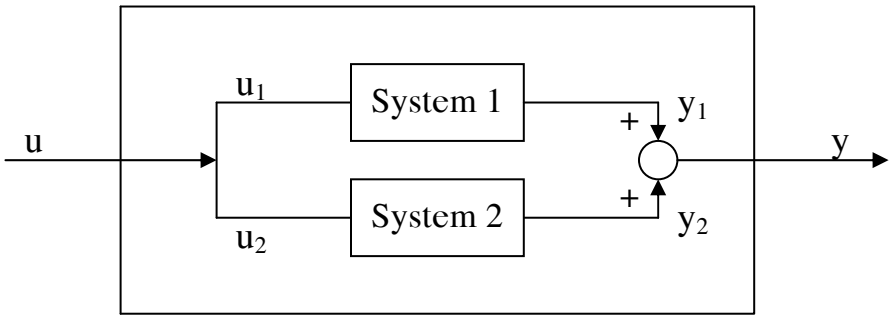
[a,b,c,d] = parallel(a1,b1,c1,d1,a2,b2,c2,d2)

[a,b,c,d] = parallel(a1,b1,c1,d1,a2,b2,c2,d2, in1, in2, out1, out2)

[num,den] = parallel(num1,den1, num2,den2)

c) Giải thích:

[a,b,c,d] = parallel(a1,b1,c1,d1,a2,b2,c2,d2) nối song song 2 hệ thống tạo thành hệ thống tổ hợp có ngõ ra là tổng các ngõ ra của 2 hệ thống $y = y_1 + y_2$ và các ngõ vào được nối lại với nhau.



Hệ thống song song

Cuối cùng, ta có hệ thống:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} u$$

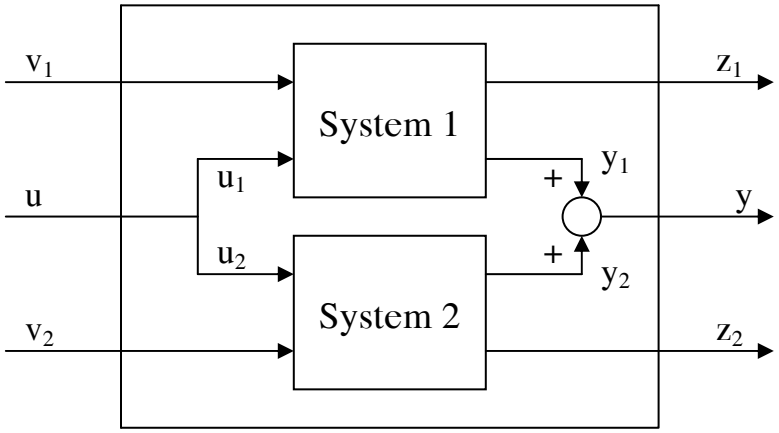
$$y = y_1 + y_2 = [C_1 + C_2] + [D_1 + D_2]u$$

[num,den] = parallel(num1,den1, num2,den2) tạo ra hàm truyền đa thức của hệ thống nối song song. num và den chứa các hệ số đa thức theo thứ tự giảm dần số mũ của s.

Kết quả ta có hàm truyền:

$$\frac{num(s)}{den(s)} = G_1(s) + G_2(s) = \frac{num_1(s)den_2(s) + num_2(s)den_1(s)}{den_1(s)den_2(s)}$$

[a,b,c,d] = parallel(a1,b1,c1,d1,a2,b2,c2,d2, in1, in2, out1, out2) nối song song 2 hệ thống để tạo thành một hệ thống tổ hợp. Các ngõ vào của hệ thống 1 được nối với các ngõ vào của hệ thống 2 và các ngõ ra của hệ thống 1 và 2 được cộng lại với nhau cho ra ngõ ra chung của hệ thống.



Hệ thống song song

Vector in1 chứa chỉ số các hệ thống vào của hệ thống 1 và chỉ ra ngõ vào nào nối với ngõ vào tương ứng của hệ thống 2 được chỉ ra trong vector in2. Tương tự, vector out1 chứa chỉ số các ngõ ra của hệ thống 1 và chỉ ra ngõ ra nào là ngõ ra tổng của các ngõ ra tương ứng của hệ thống 2 được chỉ ra trong vector out2.

Các ngõ vào của hệ thống song song bao gồm các ngõ vào được nối và các ngõ vào không nối. Tương tự, ngõ ra của hệ thống song song gồm các ngõ vào đã nối và các ngõ vào chưa nối của cả hai hệ thống.

Parallel sử dụng cho cả hệ thống liên tục và hệ thống gián đoạn.

d) Ví dụ:

Nối 2 khâu có hàm truyền G(s) và H(s) thành hệ thống song song:

$$G(s) = \frac{3}{s+4}$$

$$H(s) = \frac{2s+4}{s^2+2s+4}$$

numg = 3;

deng = [1 4];

numh = [2 4];

denh = [1 2 3];

[num,den] = parallel(numg, deng, numh, denh);

và ta được hệ thống song song có hàm truyền

G'(s) = num(s)/den(s) với các hệ số:

num = [0 5 18 25]

den = [1 6 11 12]

7. Lệnh SERIES

a) Công dụng:

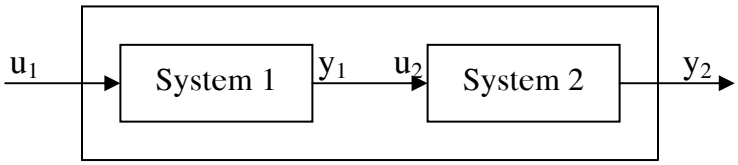
Nối nối tiếp hai hệ thống không gian trạng thái.

b) Cú pháp:

[a,b,c,d] = series(a1,b1,c1,d1,a2,b2,c2,d2)
[a,b,c,d] = series(a1,b1,c1,d1,a2,b2,c2,d2, outputs1, inputs2)
[num,den] = series(num1,den1, num2,den2)

c) Giải thích:

Lệnh [a,b,c,d] = series(a1,b1,c1,d1,a2,b2,c2,d2) nối các ngõ ra của hệ thống 1 với các ngõ vào của hệ thống 2, $u_2 = y_1$.



Hệ thống nối tiếp

Để được hệ thống:

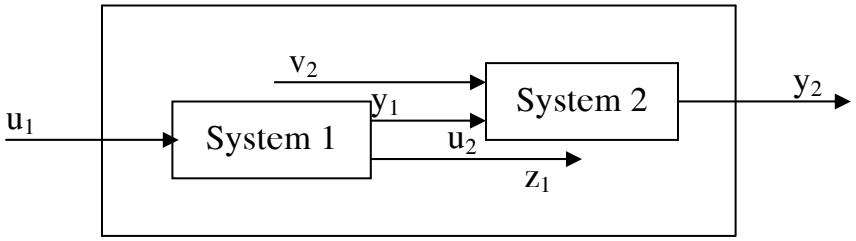
$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} A_1 & 0 \\ B_2C_1 & A_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2D_1 \end{bmatrix} u_1$$

$$y_2 = \begin{bmatrix} D_2C_1 & C_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} D_2D_1 \end{bmatrix} u_1$$

[num,den] = series(num1,den1, num2,den2) tạo ra hàm truyền đa thức của hệ thống nối tiếp. num và den chứa các hệ số đa thức theo chiều giảm dần số mũ của s. Hệ thống nối tiếp có hàm truyền như sau:

$$\frac{num(s)}{den(s)} = G_1(s)G_2(s) = \frac{num_1(s)num_2(s)}{den_1(s)den_2(s)}$$

[a,b,c,d] = series (a1,b1,c1,d1,a2,b2,c2,d2, outputs1, inputs2) nối nối tiếp 2 hệ thống 1 và 2 tạo thành hệ thống tổ hợp. Các ngõ ra được chỉ rõ của hệ thống 1 đượcnối nối tiếp với các ngõ vào được chỉ rõ của hệ thống 2:



Hệ thống nối tiếp

Vector output1 chứa các chỉ số ngõ ra của hệ thống 1 và chỉ ra ngõ ra nào của hệ thống 1 nối với các ngõ vào của hệ thống 2 được chỉ ra bởi vector inputs2.

Lệnh này có thể sử dụng cho hệ thống liên tục và hệ thống gián đoạn.

d) Ví dụ 1:

Kết nối 2 khâu có hàm truyền G(s) và H(s)

$$G(s) = \frac{3}{s+4} \quad , \quad H(s) = \frac{2s+4}{s^2+2s+3}$$

để tạo thành hệ thống nối tiếp. Ta thực hiện như sau:

num1 = 3;

den1 = [1 4];

num2 = [2 4];

den2 = [1 2 3];

[num,den] = series(num1,den1, num2,den2)

ta được kết quả:

num = [0 0 6 12]

den = [1 6 11 12]

Xét hệ thống không gian trạng thái (a1, b1, c1, d1) với 5 ngõ vào và 4 ngõ ra và một hệ thống khác (a2, b2, c2, d2) với 2 ngõ vào và 3 ngõ ra. Nối nối tiếp 2 hệ thống bằng cách nối các ngõ ra 2 và 4 của hệ thống 1 với các ngõ vào 1 và 2 của hệ thống 2:

outputs1 = [2 4];

inputs2 = [1 2];

[a,b,c,d] = series (a1,b1,c1,d1,a2,b2,c2,d2,...., outputs2, inputs1)

Ví dụ 2: Trích từ **Ví dụ 3.14** sách ... tác giả Nuyễn Văn Giáp

```
% KET NOI 2 HAM TRUYEN NOI TIEP
num1=[1 4];
den1=[1 4];
num2=[2 4];
den2=[2 4];
[num,den]=series(num1,den1,num2,den2)
```

Kết quả:

num =

2 12 16

den =

2 12 16

8. Lệnh SSDELETE

a) Công dụng:

Xóa các ngõ vào, ngõ ra, và các trạng thái của hệ thống không gian trạng thái.

b) Cú pháp:

[ar,br,cr,dr] = ssdelete(a,b,c,d,inputs,outputs)

[ar,br,cr,dr] = ssdelete(a,b,c,d,inputs,outputs,state)

c) Giải thích:

Cho hệ thống không gian trạng thái:

$$\dot{x} = Ax + \begin{bmatrix} B_1 & B_2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} x_1 + \begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

[ar,br,cr,dr] = ssdelete(a,b,c,d,inputs,outputs) xóa các ngõ vào và ngõ ra được chỉ định từ hệ thống không gian trạng thái (a,b,d,d). Vector inputs chứa chỉ số các ngõ vào của hệ thống và chỉ ra ngõ vào nào được xóa khỏi hệ thống không gian trạng thái. Tương tự, vector outputs chứa chỉ số các ngõ ra và chỉ ra ngõ ra nào được xóa khỏi hệ thống không gian trạng thái.

Cho hệ thống

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

[ar,br,cr,dr] = ssdelete(a,b,c,d,inputs,outputs,state) xóa các ngõ vào, ngõ ra, trạng thái ra khỏi hệ thống không gian trạng thái.

ssdelete sử dụng được cho hệ thống liên tục và gián đoạn.

d) Ví dụ:

Xóa ngõ vào 1, ngõ ra 2 và 3 ra khỏi hệ thống không gian trạng thái (a,b,c,d) với 2 ngõ vào và 3 ngõ ra và 3 trạng thái.

```
inputs = [1];
outputs = [2  3];
[ar,br,cr,dr] = ssdelete(a,b,c,d,inputs,outputs);
```

Cho hệ thống không gian trạng thái với 5 trạng thái, 2 ngõ vào và 3 ngõ ra hệ thống có bậc được giảm bằng cách xóa trạng thái 2 và 4 không đáp ứng tới các loại với giá trị riêng nhỏ.

[ar,br,cr,dr] = ssdelete(a,b,c,d,[],[].(2,4)

9. Lệnh SSSELECT

a) Công dụng:

Chọn hệ phụ (hệ con) từ hệ không gian trạng thái.

b) Cú pháp:

[ae,be,ce,de] = ssselect(a,b,c,d,inputs,outputs)

[ae,be,ce,de] = ssselect(a,b,c,d,inputs,outputs,states)

c) Giải thích:

Cho hệ không gian trạng thái:

$$\dot{x} = Ax + \begin{bmatrix} B_1 & B_2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} x + \begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

[ae,be,ce,de] = ssselect(a,b,c,d,inputs,outputs) tạo ra hệ thống phụ với các ngõ vào và ngõ ra được chỉ định trong 2 vector inputs và outputs.

[ae,be,ce,de] = ssselect(a,b,c,d,inputs,outputs,states) tạo ra hệ thống phụ với ngõ vào, ngõ ra và trạng thái được chỉ định trong các vector inputs, outputs, states.

ssselect được sử dụng cho cả hệ liên tục và gián đoạn.

d) Ví dụ:

Xét hệ không gian trạng thái (a,b,c,d) có 5 ngõ ra và 4 ngõ vào. Để chọn hệ thống phụ có ngõ vào 1, 2 và ngõ ra 2,3,4 ta thực hiện các lệnh:

```
inputs = [1    2];
outputs = [2    3    4];
[ae,be,ce,de] = ssselect(a,b,c,d,inputs,outputs)
```

10. Lệnh ESTIM, DESTIM

a) Công dụng:

Hình thành khâu quan sát.

b) Cú pháp:

```
[ae,be,ce,de] = estim(a,b,c,d,L)
[ae,be,ce,de] = estim(a,b,c,d,L,sensors,known)
[ae,be,ce,de] = destim(a,b,c,d,L)
[ae,be,ce,de] = destim(a,b,c,d,L,sensors,known)
```

c) Giải thích:

estim và destim tạo ra khâu quan sát Kalman cố định từ một hệ không gian trạng thái và ma trận độ lợi khâu quan sát L.

[ae,be,ce,de] = estim(a,b,c,d,L) tạo ra khâu quan sát trạng thái dựa trên hệ thống liên tục:

$$\dot{x} = Ax + Bu$$

$$y = Cx + Du$$

bằng cách xem tất cả các ngõ ra của khâu là các ngõ ra cảm biến. Khâu quan sát đạt được là:

$$\dot{\hat{x}} = [A - LC]\hat{x} + Ly$$

$$\begin{bmatrix} \hat{y} \\ \hat{x} \end{bmatrix} = \begin{bmatrix} C \\ I \end{bmatrix} \hat{x}$$

[ae,be,ce,de] = estim(a,b,c,d,L,sensors,known) tạo ra khâu quan sát trạng thái liên tục dùng các ngõ cảm biến được chỉ định trong vector sensors và các ngõ vào biết trước được chỉ định trong vector known. Các ngõ vào này bao hàm cả các ngõ vào khâu quan sát. Các ngõ vào biết trước là các ngõ vào của khâu không được dùng để thiết kế khâu quan sát như các ngõ vào điều khiển hay các lệnh bên ngoài.

[ae,be,ce,de] = destim(a,b,c,d,L) tạo ra khâu quan sát trạng thái của hệ gián đoạn:

$$x[n + 1] = Ax[n] + Bu[n]$$

$$y[n] = Cx[n] + Du[n]$$

bằng cách xem tất cả các ngõ ra là ngõ cảm biến. Ta có khâu quan sát của hệ thống là:

$$\bar{x}[n + 1] = [A - ALC]\bar{x}[n] + Aly[n]$$

$$\begin{bmatrix} \hat{y}[n] \\ \hat{x}[n] \end{bmatrix} = \begin{bmatrix} C - CLC \\ I - LC \end{bmatrix} \bar{x}[n] + \begin{bmatrix} CL \\ L \end{bmatrix} y[n]$$

[ae,be,ce,de] = destim(a,b,c,d,L,sensors,known) tạo ra khâu quan sát trạng thái gián đoạn sử dụng các ngõ vào cảm biến và ngõ vào biết trước được chỉ định trong vector sensors và known.

d) Ví dụ: (Trích từ trang 11-71 sách **'Control System Toolbox'**)

Xét hệ không gian trạng thái (a,b,c,d) có 7 ngõ ra và 4 ngõ vào. tạo khâu quan sát trạng thái khi ma trận độ lợi Kalman L được thiết kế sử dụng ngõ ra 4, 7 và 1 của khâu làm các cảm biến và ngõ vào 1, 4, 3 là các ngõ vào biết trước. Khâu quan sát trạng thái được tạo thành bằng cách sử dụng:

```
sensors = [4   7   1];
known = [1   4   3];
[ae,be,ce,de] = estim(a,b,c,d,L,sensors,known)
```

11. Lệnh REG, DREG

a) Công dụng:

Tạo khâu điều khiển.

b) Cú pháp:

[ac,bc,cc,dc] = reg(a,b,c,d,K,L)

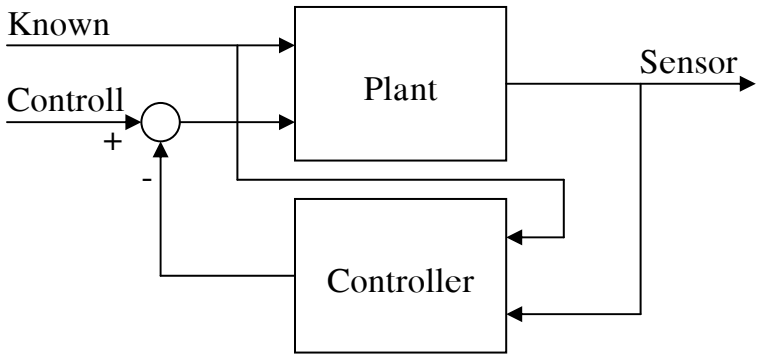
[ac,bc,cc,dc] = reg(a,b,c,d,K,L,sensors,known,controls)

[ac,bc,cc,dc] = dreg(a,b,c,d,K,L)

[ac,bc,cc,dc] = dreg(a,b,c,d,K,L,sensors,known,controls)

c) Giải thích:

reg và dreg tạo ra khâu điều khiển/ khâu quan sát từ một hệ không gian trạng thái, ma trận độ lợi hồi tiếp K và ma trận độ lợi khâu quan sát L.



Kết nối giữa khâu độ lợi và khâu điều khiển

[ac,bc,cc,dc] = reg(a,b,c,d,K,L) tạo ra khâu điều khiển/ khâu quan sát cho hệ liên tục:

$$\dot{x} = Ax + Bu$$

$$y = Cx + Du$$

bằng cách xem các ngõ vào của khâu là ngõ vào điều khiển và các ngõ ra là ngõ ra cảm biến. Kết quả ta có khâu điều khiển/ khâu quan sát:

$$\hat{\dot{x}} = [A - BK - LC + LDK]\hat{x} + Ly$$

$$\hat{u} = K\hat{x}$$

[ac,bc,cc,dc] = reg(a,b,c,d,K,L,sensors,known,controls) tạo ra khâu điều khiển/ khâu quan sát sử dụng các cảm biến được chỉ định trong vector sensors, ngõ vào biết trước được chỉ định bởi vector known và ngõ vào điều khiển được được chỉ định bởi vector controls.

[ac,bc,cc,dc] = dreg(a,b,c,d,K,L) tạo ra khâu điều khiển/ khâu quan sát cho hệ gián đoạn.

$$x[n + 1] = Ax[n] + Bu[n]$$

$$y[n] = Cx[n] + Du[n]$$

bằng cách xem tất cả các ngõ vào điều khiển và tất cả ngõ ra là ngõ ra cảm biến. Kết quả ta có khâu điều khiển/ khâu quan sát:

$$\bar{x}[n+1]=[A-ALC-(A-ALD)E(K-KLC)\bar{x}[n]]+[AL-(B-ALD)EKL]Y[n]]$$

$$\hat{u}[n] = [K-KLC+KLDE(K-KLC)\bar{x}[n]]+[KL+KLDEKL]Y[n]]$$

trong đó $E = (I - KLD)^{-1}$ với I là ma trận đơn vị.

[ac,bc,cc,dc] = dreg(a,b,c,d,K,L,sensors,known,controls) tạo ra khâu điều khiển/ khâu quan sát gián đoạn sử dụng các cảm biến, các ngõ vào biết trước và các ngõ vào điều khiển đã được chỉ định.

d) Ví dụ: (Trích từ trang 11-178 sách **'Control System Toollbox'**)

Xét hệ không gian trạng thái liên tục (a,b,c,d) có 7 ngõ ra và 4 ngõ vào. tạo khâu điều khiển/ khâu quan sát khi ma trận độ lợi hồi tiếp K và được thiết kế sử dụng ngõ vào 1, 2, 4 của khâu như ngõ vào điều khiển, ma trận dộ lợi Kalman L được thiết kế sử dụng ngõ ra 4, 7, 1 như các cảm biến và ngõ vào 3 của khâu là ngõ vào biết trước.

```
controls = [1, 2, 4];
sensors = [4, 7, 1];
known = [3];
[ac,bc,cc,dc] = reg(a,b,c,d,K,L,sensors,known,controls)
```

12. Lệnh RMODEL, DRMODEL

a) Công dụng:

Tạo ra mô hình ổn định ngẫu nhiên bậc n.

b) Cú pháp:

```
[a,b,c,d] = rmodel(n)
[a,b,c,d] = rmodel(n,p,m)
[num,den] = rmodel(n)
[num,den] = rmodel(n,p)
[a,b,c,d] = drmodel(n)
[a,b,c,d] = drmodel(n,p,m)
[num,den] = drmodel(n)
[num,den] = drmodel(n,p)
```

c) Giải thích:

[a,b,c,d] = rmodel(n) tạo ra mô hình không gian trạng thái ổn định ngẫu nhiên bậc n (a,b,c,d) có 1 ngõ vào và 1 ngõ ra.

[a,b,c,d] = rmodel(n,p,m) tạo ra mô hình ổn định ngẫu nhiên bậc n có m ngõ vào và p ngõ ra.

[num,den] = rmodel(n) tạo ra hàm truyền của mô hình ổn định ngẫu nhiên bậc n. num và den chứa các hệ số của hàm truyền đa thức theo chiều giảm dần số mũ của s.

[num,den] = rmodel(n,p) tạo ra mô hình SIMO (Singular Input Multi Outputs) ổn định ngẫu nhiên bậc n có 1 ngõ vào và m ngõ ra.

drmodel tạo ra các mô hình ổn định ngẫu nhiên gián đoạn.

d)Ví dụ: Trích từ trang 11-190 sách **'Control System Toolbox'**

Tạo mô hình ổn định ngẫu nhiên với 3 trạng thái(state), 2 inputs, 2 outputs:

sys=rss(3,2,2)

Kết quả:

a =

	x1	x2	x3
x1	-0.36837	0.20275	0.14925
x2	-0.23638	-0.64783	0.51501
x3	0.086654	-0.52916	-0.59924

b =

	u1	u2
x1	-0.1364	0
x2	0.11393	-0.095648
x3	0	-0.83235

c =

	x1	x2	x3
y1	0.29441	0	0
y2	0	1.6236	0.858

d =

	u1	u2
y1	1.254	-1.441

y2 0 0.57115

Continuous-time model.

13. Lệnh ORD2

a) Công dụng:

Tạo ra hệ bậc 2.

b) Cú pháp:

[a,b,c,d] = ord2(w,z)

[num,den] = ord2(wn,z)

c) Giải thích:

[a,b,c,d] = ord2(w,z) tạo ra sự mô tả không gian trạng thái (a,b,c,d) của hệ bậc 2.

$$H(s) = \frac{1}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

được cho bởi tần số tự nhiên wn và tỷ lệ tắt dần.

[num,den] = ord2(wn,z) tạo ra hàm truyền đa thức của hệ bậc 2.

d) Ví dụ: (Trích từ trang 11-163 sách **'Control System Toolbox'**)

Tìm hàm truyền của hệ bậc 2 có tỷ lệ tắt dần $\zeta = 0.4$ và tần số tự nhiên $\omega_n = 2.4$ rad/s.

[num,den] = ord2 (2.4, 0.4)

num = 1

den =

1.0000 1.9200 5.7600

Tức là ta có hàm truyền (transfer function):

$1/(s^2+1,92s+5,76)$

14. Lệnh PADE

a) Công dụng:

Tìm mô hình gần đúng của khâu trễ.

b) Cú pháp:

[a,b,c,d] = pade(T,n)

[num,den] = pade(T,n)

c) Giải thích:

pade tạo ra mô hình LTI bậc n gần đúng. Mô hình gần đúng pade được sử dụng để mô phỏng ảng hưởng của thời gian trễ như thời gian trễ tính toán trong phạm vi hệ liên tục. Phép biến đổi Laplace của thời gian trễ T giây là e^{-sT} có thể gần bằng hàm truyền với tử số và mẫu số bậc n.

$$e^{-sT} = 1 - sT + \frac{1}{2!}(sT)^2 \frac{1}{3!}(sT)^3 + ... \approx \frac{num(s)}{den(s)}$$

[a,b,c,d] = pade(T,n) tạo ra mô hình trạng thái SISO (Singular Input Singular Outputs) bậc n xấp xỉ thời gian trễ T giây.

[num,den] = pade(T,n) tạo ra hàm truyền đa thức gần thời gian trễ nhất. num và den chứa các hệ số đa thức theo chiều giảm dần số mũ của s.

d) Ví dụ 1:

Tìm hàm truyền và mô hình gần đúng khâu bậc 1 với thời gian trễ là 0.2 giây.

Ta thực hiện lệnh sau:

[num,den] = pade(0.2, 1)

ta được:

num =

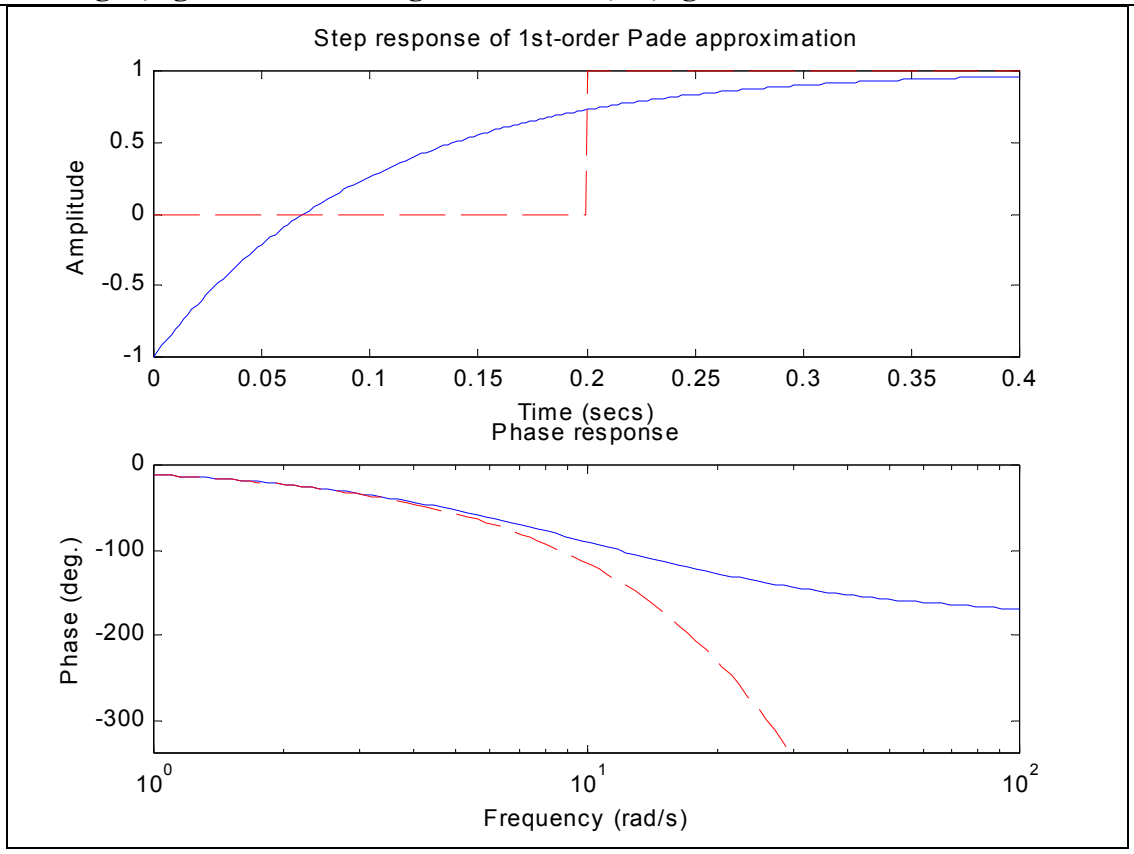
-0.0995 0.9950

den =

0.0995 0.9950

$$\text{tức là } H(s) = \frac{num(s)}{den(s)} = \frac{-0.0995s + 0.9950}{0.0995s + 0.9950}$$

Sau đó ta gõ tiếp ở ngoài dấu nhắc lệnh:
pade(0.2,1)
Ta có kết quả:



Ví dụ 2: Tìm hàm truyền mô hình gần đúng khâu bậc 3 với thời gian trễ là 0.1 giây. (Trích từ trang 11-166 sách **'Control System Tollbox')**

[num,den] = pade(0.1, 3)

pade(0.1,3)

Ta có kết quả:

num =

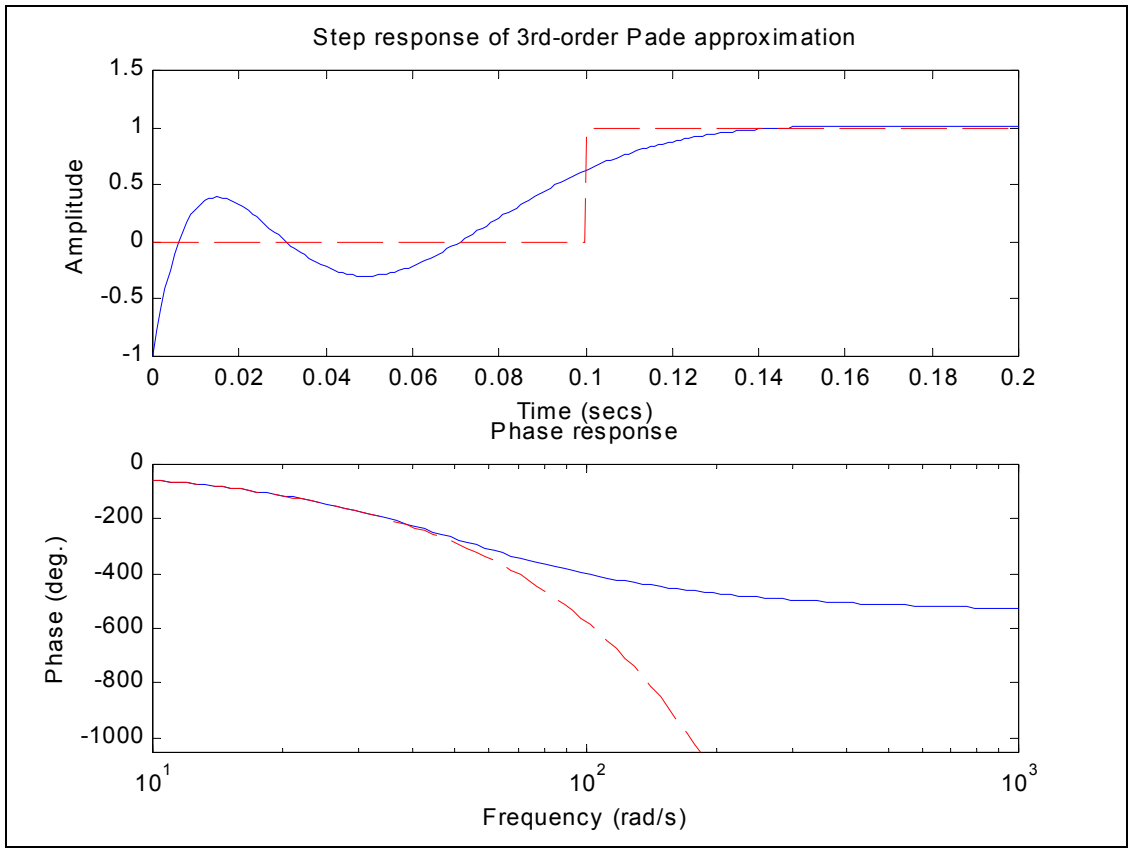
1.0e+005 *

-0.0000 0.0012 -0.0600 1.2000

den =

1.0e+005 *

0.0000 0.0012 0.0600 1.2000



CÁC BÀI TẬP

Bài 1: Trích từ **Ví dụ 3.13** sách … tác giả Nuyễn Văn Giáp

```
%Ket NOI 2 HE thong SONG SONG
a=[1 2 3;4 5 6;7 8 9];
b=[3 4;4 5;7 9];
c=[0 0 1];
d=[0 0];
e=[1 9 3;4 5 6;7 8 7];
f=[2 4;4 6;7 9];
g=[0 1 1];
h=[0 0];
[A,B,C,D]= parallel(a,b,c,d,e,f,g,h)
```

Kết quả:

A =

1	2	3	0	0	0
4	5	6	0	0	0
7	8	9	0	0	0
0	0	0	1	9	3
0	0	0	4	5	6
0	0	0	7	8	7

B =

3	4	0	0
4	5	0	0
7	9	0	0
0	0	2	4
0	0	4	6
0	0	7	9

C =

0	0	1	0	0	0
0	0	0	0	1	1

D =

0	0	0	0
0	0	0	0

A =

1	2	3	0	0	0
4	5	6	0	0	0
7	8	9	0	0	0
0	0	0	1	9	3
0	0	0	4	5	6
0	0	0	7	8	7

B =

3	4
4	5
7	9
2	4
4	6
7	9

C =

0 0 1 0 1 1

D =

0 0

Bài 2: Kết nối hai hàm truyền nối với số liệu nhập từ bàn phím (viết chương trình trong m_file)

```
%Bai tap tong quat ket noi 2 he thong noi tiep
%Cu phap SYS=series(SYS1,SYS2,OUTPUTS1,INPUTS2)
%Vi du ta se ket noi 2 ham truyen
num1=input(' Nhap num1= ');
den1=input('Nhap den1= ');
num2=input('Nhap num2= ');
den2=input('Nhap den2= ');
[num,den]=series(num1,den1,num2,den2)
```

Bài 3: (Trích trang 11-14 sách **Control System Toolbox**)

sys1=tf(1,[1 0])

Transfer function:

1

-

s

sys2=ss(1,2,3,4)

a =

	x1
x1	**1**

b =

	u1
x1	**2**

c =

	x1
y1	3

d =

	u1
y1	4

Continuous-time model.

sys=append(sys1,10,sys2)

a =

	x1	x2
x1	0	0
x2	0	1

b =

	u1	u2	u3
x1	1	0	0
x2	0	0	2

c =

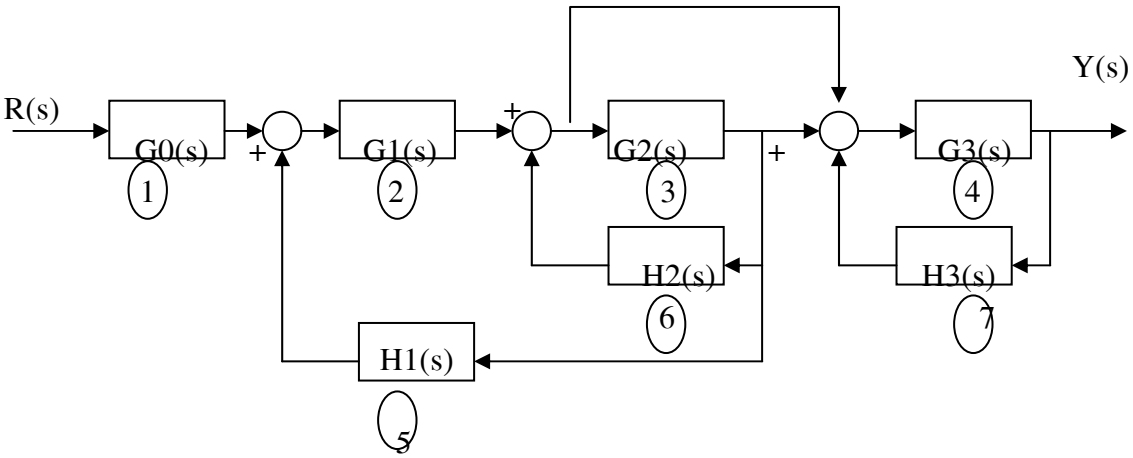
	x1	x2
y1	1	0
y2	0	0
y3	0	3

d =

	u1	u2	u3
y1	0	0	0
y2	0	10	0
y3	0	0	4

Continuous-time model.

Bài 4: một hệ thống biểu diễn như hình sau với
G0(s)=1;G1(s)=1/(s+1);G2(s)=1/(s+2);G3(s)=1/(s+3);
H1(s)=4;H2(s)=8;H3(s)=12.




```
n1=1; d1=1;
n2=1; d2=[1 1];
n3=1; d3=[1 2];
n4=1; d4=[1 3];
n5=4; d5=1;
n6=8; d6=1;
n7=12; d7=1;
nblocks=7;
blkbuild
q=[1  0  0  0  0
```

```
   2  1 -5  0  0
   3  2 -6  0  0
   4  2 -6  3 -7
   5  3  0  0  0
   6  3  0  0  0
   7  4  0  0  0];
input=1;
output=4;
[aa,bb,cc,dd]=connect(a,b,c,d,q,input,output);
[num,den]=ss2tf(aa,bb,cc,dd);
printsys(num,den,'s')
```

Giải thích:

Ta phải đánh số trong mỗi hệ thống phụ như hình trên. Bảy câu lệnh đầu tiên biểu diễn hàm truyền của bảy khối, qui định tên tương ứng với tử và mẫu là **n1,d1,n2,d2,...**trong trường hợp nếu cho dạng là kiểu biến trạng thái trong từng hệ thống phụ thì tên của chúng tương ứng là **a1,b1,c1,d1,a2,b2,c2,d2,...**

Đặt biến **nblock=7** (bằng với số của hệ thống phụ).

Sau đó là lệnh **blkbuild** dùng những biến của **nblock** để bắt đầu xây dựng hệ thống. Biến **blkbuild** chuyển đổi tất cả cách thức diễn tả hàm truyền của từng hệ thống phụ thành kiểu biến trạng thái như dùng lệnh **tf2ss** và đưa chúng vào một khối lớn của ma trận trang thái gọi là **a, b, c, d.**

Tạo ra ma trận **q** để nhận biết mối liên hệ giữa các hệ thống phụ. (Mỗi hàng của ma trận **q** tương ứng với một hệ thống phụ khác nhau. Phần tử đầu tiên trong hàng là số hệ thống nguồn,số còn lại chỉ khối kết nối giữa ngõ ra và ngõ vào của hệ thống phụ.)

Hàng thứ hai của ma trận **q** có phần tử đầu tương ứng với hệ thống phụ 2 (G1(s)). Bởi vì ngõ ra của hệ thống 1 và hệ thống 5 là ngõ vào của hệ thống 2,do đó hai phần tử kế tiếp trong hàng là 1 và –5, hai số 0 được thêm vào để cần thiết tạo ra để bảo đảm q là ma trận hình chữ nhật.

Sau khi tạo được ma trận **q** ta phải chỉ rõ khối ngõ vào (biểu diễn bởi biến **input**) và khối ngõ ra(biểu diễn bởi biến **output**).

Lệnh **connect** dùng để nối các kiểu biến trạng thái thu được từ việc thành lập ở trên. Sau đó ta chuyển qua dạng hàm truyền dùng lệnh **ss2tf** và in ra màn hình.

ta được kết quả như sau:

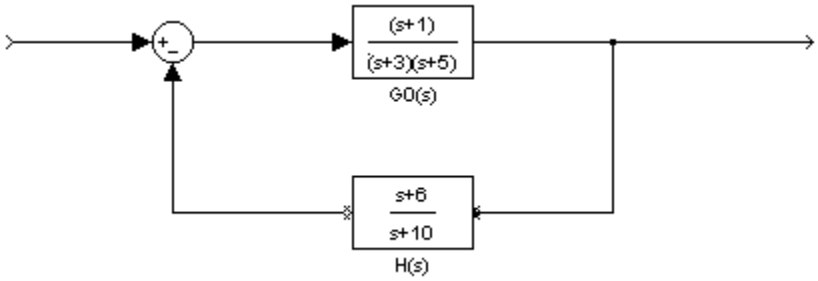
```
» Bài 4
State model [a,b,c,d] of the block diagram has 7 inputs and 7        outputs

num/den =

          1 s + 3
  -------------------------
```

```
  s^3 + 26 s^2 + 179 s + 210
```

Nhận xét: Khi phần tử phản hồi không thuộc loại phản hồi đơn vị trong hệ thống vòng kín, thì ta sử dụng lệnh feedback.

Bài 5: Cho hệ thống diễn tả trong hình sau có hàm truyền:

$$G_{c2}(s) = \frac{G_0(s)}{(1 + G_0(s)H(s))}$$



Hình : Sơ dồ hệ thống phản hồi

Chương trình tạo ra hàm truyền trên:

```
% Bài 5.m
% tao ra ham truyen
% voi he thong phan hoi khong phai la phan hoi don vi
tuG=[1 1]; % tao ra vecto cua tu ham G(s)
mauG=conv([1 3],[1 5]); % tao ra vecto cua mau ham G(s)
tuH=[1 6]; % tao ra vecto cua tu ham H(s)
mauH=[1 10]; % tao ra vecto cua mau ham H(s)
[tu,mau]=feedback(tuG,mauG,tuH,mauH);
printsys(tu,mau)
```

Kết quả:

```
» Bài5

num/den =

          s^2 + 11 s + 10
  --------------------------
  s^3 + 19 s^2 + 102 s + 156
```

NHÓM LỆNH VỀ RÚT GỌN MÔ HÌNH
(Model Reduction)

1. Lệnh BALREAL, DBALREAL

a) Công dụng:

Thực hiện cân bằng hệ khhông gian trạng thái.

b) Cú pháp:

[ab,bb,cb]= balreal(a,b,c)

[ab,bb,cb,g,T]= balreal(a,b,c)

[ab,bb,cb]= dbalreal(a,b,c)

[ab,bb,cb,g,T]= dbalreal(a,b,c)

c) Giải thích:

[ab,bb,cb]= balreal(a,b,c) thực hiện hệ cân bằng không gian trạng thái (a,b,c).

[ab,bb,cb,g,T]= balreal(a,b,c) chuyển đổi tương đương giữa a,b và c,a. Vectoe g chứa các phần tử nằm trên đường chéo của ma trận đánh giá sự thực hiện cân bằng. Ma trận T dùng trong phép biến đổi để chuyển (a,b,c) thành (ab,bb,cb). Ta phải nghịch đảo ma trận T nếu ta muốn sử dụng với lệnh ss2ss.

Nếu hệ thống được chuẩn hóa hoàn toàn thì vector g được dùng để giảm bậc của mô hình. Vì g phản ánh khả năng điều khiển và khả năng quan sát kết hợp của cac trạng thái riêng biệt. Những trạng thái này có g(I) nhỏ và có thể loại bỏ ra khỏi mô hình. Sự triệt tiêu các trạng thái nàyvẫn duy trì hầu hết các đặc tính vào ra quan trọng của hệ thống ban đầu.

Dbalreal được dùng cho các hệ thống gián đoạn.

d) Ví dụ:

Thực hiện hệ cân bằng bậc 4.

[ab,bb,cb,g,T]= balreal(a,b,c)

ta được:

g=

0.6342 0.4313 0.0812 0.0203

từ đó ta có thể loại bỏ 2 trạng thái thứ ba và thứ tư:

[ar,br,cr]= mored(ab,bb,cb,[3 4])

2. Lệnh MODRED, DMODRED

a) Công dụng:

Giảm trạng thái của mô hình.

b) Cú pháp:

[ar,br,cr,dr]= modred(a,b,c,d,elim)

[ar,br,cr,dr]= dmodred(a,b,c,d,elim)

c) Giải thích:

Lệnh modred và dmodred dùng để giảm bậc của mô hình không gian trạng thái trong khi vẫn duy trì các mối quan hệ vào ra ở trạng thái xác lập. Do đó, modred và dmodred rất hữuích khi loại bỏ các trạng thái tần số cao. Dùng lệnh ssdelete để loại bỏ các trạng thái tần số thấp, modred và dmodred thường dùng kết hợpvới lệnh balreal và dbalreal.

[ar,br,cr,dr]= modred(a,b,c,d,elim) giảm bậc các mô hình bằng cách loại bỏ các trạng thái được chỉ định trong vector elim. Cuối cùng ta được mô hình có số trạng thái ít hơn.

[ar,br,cr,dr]= dmodred(a,b,c,d,elim) được sử dụng cho hệ gián đoạn.

3. Lệnh MINREAL

a) Công dụng:

Thực hiện cực tiểu hóa cực-zero.

b) Cú pháp:

[am,bm,cm,dm]= minrael(a,b,c,d)

[am,bm,cm,dm]= minreal(a,b,c,d,tol)

[zm,pm]= minreal(z,p)

[zm,pm]= minreal(z,p,tol)

[numm,denm]= minreal(num,den)

[numm,denm]= minreal(num,den,tol)

c) Giải thích:

Thực hiện cực tiểu hóa là thực hiện loại bỏ các trạng thái dư thừa, không cần thiết. Đối với hàm truyền hay mô hình độ lợi cực-zero, điều này tương đương với việc bỏ các cặp cực-zero mà chúng khử lẫn nhau.

+ Đối với mô hình không gian trạng thái:

[am,bm,cm,dm]= minreal(a,b,c,d) thực hiện cực tiểu hóa hệ không gian trạng thái và hiển thị số trạng thái được loại bỏ. Số trạng thái này có liên quan tới hệ thống. Nếu loại bỏ quá nhiều hoặc quá ít thì sai sốn sẽ thay đổi.

[aam,bm,cm,dm]= minreal(a,b,c,d,tol) dùng sai số tol để chỉ định trạng thái nào bị loại bỏ. Nếu không dùng tham số tol thì giá trị mặc nhiên là:

tol= 10*max(size(a))*norm(a,1)*eps

+ Đối với mô hình độ lợi cực-zero:

[zm,pm]= minreal(z,p), trong đó z và p là các vector cột chứa cáccực và zero, dùng để khử các nghiệm chung lẫn theo biểu thức:

tol= 10*sprt(eps)*abs(z(I)).

[zm,pm]= minreal(z,p,tol) dùng sai số tol.

Đối với mô hình hàm truyền:

[numm,denm]= minreal(num,den), trong đó num và den là các vector hàng chứa các hệ số đa thức, dùng để khử các nghiệm chung của đa thức lẫn nhau theo biểu thức:

tol= 10*sqrt(eps)*abs(z(I))

[numm,denm]= minreal(num,den,tol) dùng sai số tol.

NHÓM LỆNH VỀ CHUYỂN ĐỔI MÔ HÌNH

(Model Conversion)

1. Lệnh C2D, C2DT

a) Công dụng:

Chuyển đổi mô hình từ liên tục sang gián đoạn.

b) Cú pháp:

[ad,bd] = c2d(a,b,Ts)

c) Giải thích:

c2d và c2dt chuyển mô hình không gian trạng thái từ liên tục sang gián đoạn thừa nhận khâu giữ bậc 0 ở ngỏ vào. c2dt cũng có khoảng thời gian trễ ở ngõ vào.

[ad, bd] = c2d(a,b,Ts) chuyển hệ không trạng thái liên tục x = Ax + Bu thành hệ gián đoạn: $x[n+1] = A_d x[n] + B_d u[n]$ thừa nhận ngõ vào điều khiển là bất biến từng đoạn bên ngoài thời gian lấy mẫu Ts.

[ad,bd,cd,dd] = c2dt(a,b,c,Ts,lambda) chuyển hệ không gian trạng thái liên tục với thời gian trễ thuần túy λ ở ngõ vào:

$$\dot{x}(t) = Ax(t) + Bu(t - \lambda)$$

$$y(t) = Cx(t)$$

thành hệ gián đoạn:

$$x[n+1] = A_d x[n] + B_d u[n]$$

$$y[n] = C_d x[n] + D_d u[n]$$

Ts là thời gian lấy mẫu và lambda là thời gian trễ ở ngõ vào. λ phải nằm trong khoảng $-Ts < \lambda < \infty$.

d) Ví dụ: (Trích từ trang 11-24 sách '**Control System Toolbox'**)

Cho hệ thống: $H(s) = (s-1)/(s^2 + 4s + 5)$

Với T_d=0,35, thời gian lấy mẫu Ts=0,1

```
» num=[1 -1];
» den=[1 4 5];
» H=tf(num,den,'inputdelay',0.35)
```

Kết quả:

Transfer function:

```
          s - 1
exp(-0.35*s) * -------------
        s^2 + 4 s + 5
```

```
» Hd=c2d(H,0.1,'foh')
```

Transfer function:

```
      0.0115 z^3 + 0.0456 z^2 - 0.0562 z - 0.009104
z^(-3) * ---------------------------------------------
            z^3 - 1.629 z^2 + 0.6703 z
```

Sampling time: 0.1

2. Lệnh C2DM

a) Công dụng:

Chuyển đổi hệ liên tục sang gián đoạn.

b) Cú pháp:

[ad,bd,cd,dd] = c2dm(a,b,c,d,Ts,'method')

[numd,dend] = c2dm(num,den,Ts,'method').

c) Giải thích:

[ad,bd,cd,dd] = c2dm(a,b,c,d,Ts,'method') chuyển đổi từ hệ không gian trạng thái liên tục (a,b,c,d) sang gián đoạn sử dụng phương pháp khai báo trong 'method'. 'method' có thể là:

+ 'zoh': chuyển sang hệ gián đoạn thừa nhận một khâu giữ bậc 0 ở ngõ vào, các ngõ vào điều khiển được xem như bất biến từng đoạn trong khoảng thời gian lấy mẫu Ts.

+ 'foh': chuyển sang hệ gián đoạn thừa nhận một khâu giữ bậc 1 ở ngõ vào.

+ 'tustin': chuyển sang hệ gián đoạn sử dụng pháp gần đúng song tuyến tính (Tusin) đối với đạo hàm.

+ 'prewarp': chuyển sang hệ gián đoạn sử dụng pháp gần đúng song tuyến tính (Tusin) với tần số lệch trước. Nếu thêm vào tham số Wc thì lệnh sẽ chỉ ra tần số tới hạn.

Ví dụ như c2dm(a,b,c,d,Ts,prewarp,Wc).

+ 'matched': chuyển hệ SISO sang gián đoạn sử dụng phương pháp cực zero hàm truyền phù hợp.

[numd, dend] = c2dm(num,den,Ts,'method') chuyển từ hàm truyền đa thức liên tục G(s) = num(s)/den(s) sang gián đoạn G(z) = num(z)/den(z) sử dụng phương pháp được khai báo trong 'method'.

Nếu bỏ qua các đối số bên trái thì:

c2dm(a,b,c,d,Ts,'method')

c2dm(num,den,Ts,'method')

sẽ vẽ ra 2 đồ thị của 2 đáp ứng với đường liền nét là đáp ứng liên tục còn đường đứt đoạn là đáp ứng gián đoạn.

d) Ví dụ:

Chuyển hệ không gian trạng thái liên tục:

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 2 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u$$

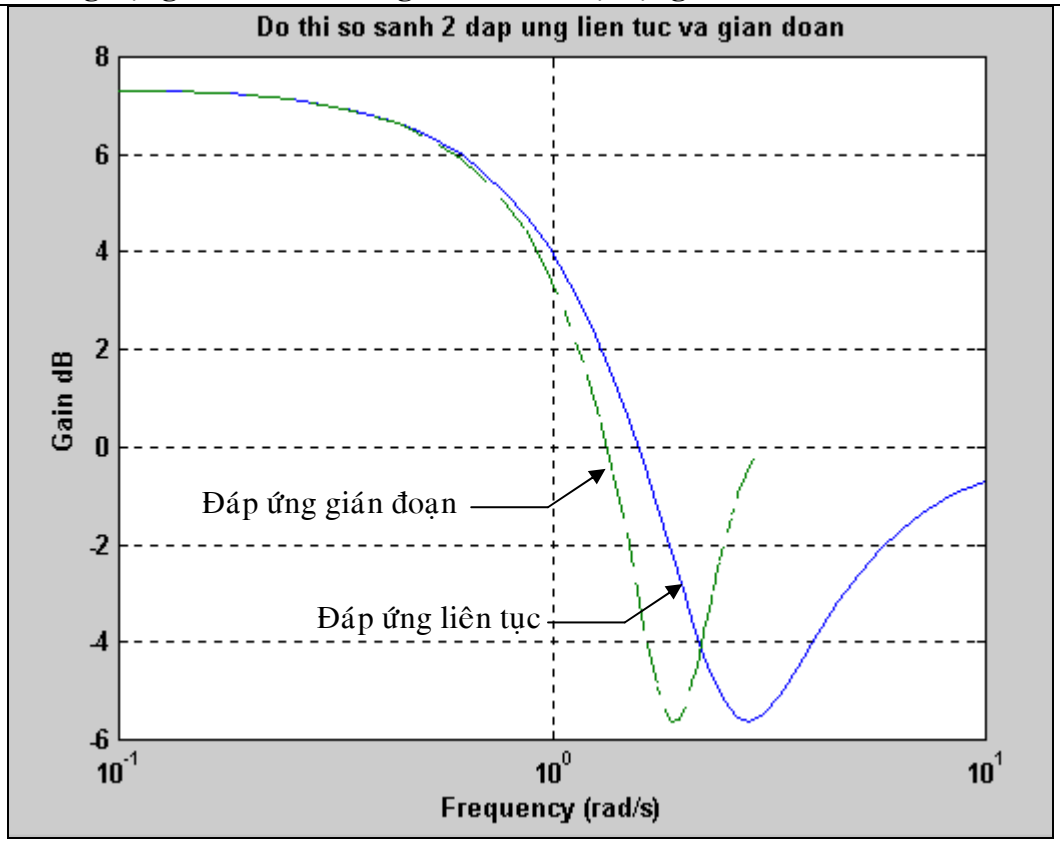
$$y = \begin{bmatrix} 2 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \end{bmatrix} u$$

thành hệ gián đoạn dùng phương pháp 'Tustin', vẽ 2 đồ thị đáp ứng so sánh.

```
a = [1  1; 2  -1];
b = [1; 0];
c = [2  4];
d = 1;
Ts = 1;
[ad,bd,cd,dd] = c2dm(a,b,c,d,Ts,'tustin')
c2dm(a,b,c,d,Ts,' tustin') %vẽ đồ thị so sánh
title ('Do thi so sanh 2 dap ung lien tuc va gian doan')
grid on
```

ta được đồ thị và các giá trị như sau:

```
ad =
    11   4
     8   3
bd =
     6
     4
cd =
    28    12
dd =
    15
```



3. Lệnh D2C

a) Công dụng:

Chuyển đổi mô hình từ gián đoạn sang liên tục.

b) Cú pháp:

[ad,bd] = c2d(a,b,Ts).

c) Giải thích:

d2c chuyển mô hình không gian trạng thái từ gián đoạn sang liên tục thừa nhận khâu giữ bậc 0 ở ngõ vào. C2DT cũng có một khoảng thời gian trễ ở ngõ vào.

[ad,bd] = c2d (a,b,Ts) chuyển hệ không gian trạng thái gián đoạn:

$$x[n+1] = Ax[n] + Bu[n]$$

thành hệ liên tục

$$\dot{x} = A_c x + B_c u$$

xem các ngõ vào điều khiển là bất biến từng đoạn trong khoảng thời gian lấy mẫu Ts.

4. Lệnh D2CM

a) Công dụng:

Chuyển đổi mô hình không gian trạng thái từ gián đoạn sang liên tục.

b) Cú pháp:

[ac,bc,cc,dc] = d2cm(a,b,c,d,Ts,'method')

[numc,denc] = d2cm(num,den,Ts,'method').

c) Giải thích:

[ac,bc,cc,dc] = d2cm(a,b,c,d,Ts,'method') chuyển đổi hệ không gian trạng thái từ gián đoạn sang liên tục sử dụng phương pháp được khai báo trong 'method'. 'method' có thể là:

+ 'zoh': chuyển sang hệ liên tục thừa nhận một khâu giữ bậc 0 ở ngõ vào, các ngõ vào điều khiển được xem như bất biến từng đoạn trong khoảng thời gian lấy mẫu Ts.

+ 'tustin': chuyển sang hệ liên tục sử dụng phương pháp gần đúng song tuyến tính (Tusin) đối với đạo hàm.

+ 'prewarp': chuyển sang hệ liên tục sử dụng pháp gần đúng song tuyến tính (Tusin) với tần số lệch trước. Nếu thêm vào tham số Wc thì lệnh sẽ chỉ ra tần số tới hạn.

Ví dụ như d2cm (a,b,c,d,Ts,prewarp,Wc).

+ 'matched': chuyển hệ SISO sang liên tục sử dụng phương pháp cực zero hàm truyền phù hợp.

[numc,denc] = d2cm(num,den,Ts,'method') chuyển từ hàm truyền đa thức gián đoạn G(z) = num(z)/den(z) sang liên tục G(s) = num(s)/den(s) sử dụng phương pháp được khai báo trong 'method'.

Nếu bỏ qua các đối số bên trái thì:

d2cm(a,b,c,d,Ts,'method')

d2cm(num,den,Ts,'method')

sẽ vẽ ra 2 đồ thị của 2 đáp ứng với đường liền nét là đáp ứng gián đoạn còn đường đứt đoạn là đáp ứng liên tục.

d) Ví dụ:

Chuyển hệ không gian trạng thái gián đoạn:

$$x[n+1] = Ax[n] + Bu[n]$$
$$y[n] = Cx[n] + Du[n]$$

với:

$$A = \begin{bmatrix} 11 & 4 \\ 8 & 3 \end{bmatrix}; \qquad B = \begin{bmatrix} 6 \\ 4 \end{bmatrix}; \qquad C = \begin{bmatrix} 28 & 12; \end{bmatrix} \qquad D = 15;$$

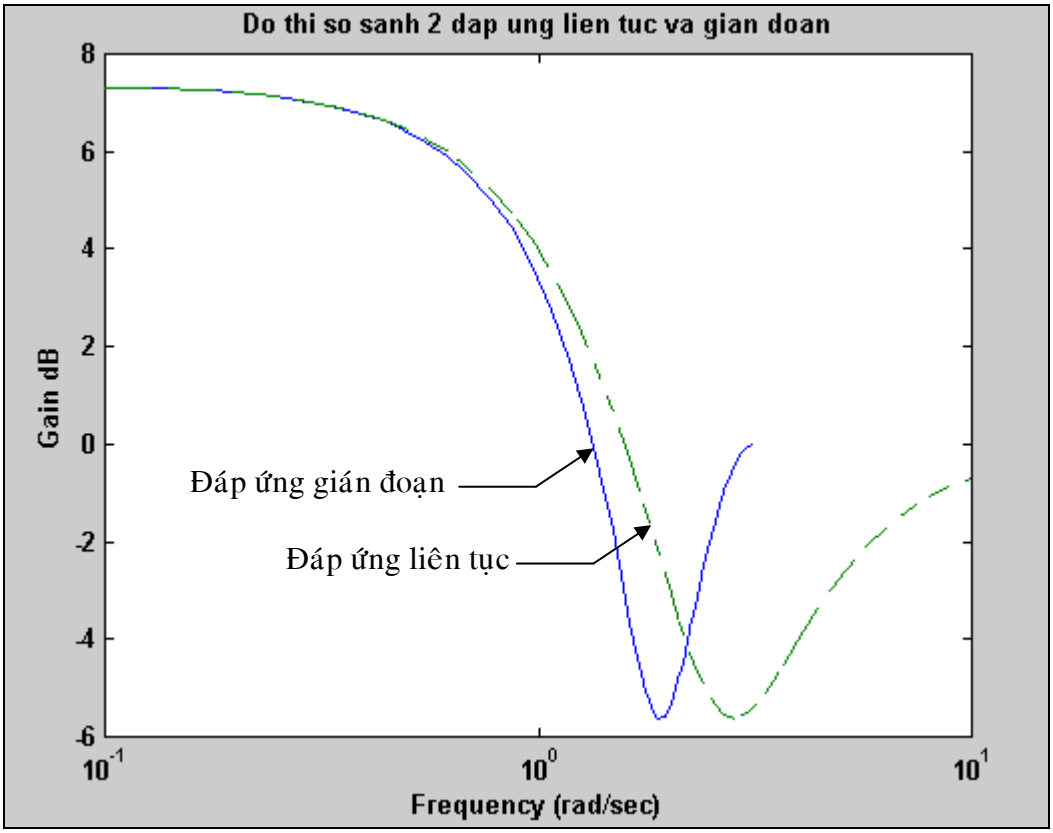
```
A = [11  4; 8  3];
B = [6; 4];
C = [28  12];
D = 15;
Ts = 1;
[ac,bc,cc,dc] = d2cm(a,b,c,d,Ts,'tustin')
d2cm(a,b,c,d,Ts,' tustin') % vẽ đồ thị so sánh
title ('Do thi so sanh 2 dap ung lien tuc va gian doan')
```

ta được đồ thị và các tham số như sau:

```
ac =
     1     1
     2    -1
bc =
     1
     0
cc =
```

2 4

dc = 1



5. Lệnh SS2TF

a) Công dụng:

Chuyển hệ thống từ dạng không gian trạng thái thành dạng hàm truyền.

b) Cú pháp:

[num,den] = ss2tf(a,b,c,d,iu).

c) Giải thích:

[num,den] = ss2tf(a,b,c,d,iu) chuyển hệ thống không gian trạng thái:

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx + Du \end{cases}$$

thành dạng hàm truyền:

$$H(s) = \frac{NUM(s)}{den(s)} = C(sI - A)^{-1} B + D$$

từ ngõ vào thứ iu. Vector den chứa các hệ số của mẫu số theo chiều giảm dần số mũ của s. Ma trận NUM chứa các hệ số tử số với số hàng là số ngõ ra.

d) Ví dụ:

Hàm truyền của hệ thống được xác định bằng lệnh:

[num,den] = ss2tf (a,b,c,d,1)

ta được:

num =

```
     0     0     1.0000
  den =
     1.0000   0.4000   1.0000
```

6. Lệnh TF2SS

a) Công dụng:

Chuyển hệ thống từ dạng không gian hàm truyền thành dạng trạng thái.

b) Cú pháp:

[a,b,c,d] = tf2ss(num,den)

c) Giải thích:

[a,b,c,d] = tf2ss(num,den) tìm hệ phương trình trạng thái của hệ SISO:

$$\dot{x} = Ax + Bu$$
$$y = Cx + Du$$

được cho bởi hàm truyền:

$$H(s) = \frac{NUM(s)}{den(s)} = C(sI - A)^{-1}B + D$$

từ ngõ vào duy nhất. Vector den chứa các hệ số mẫu số hàm truyền theo chiều giảm dần số mũ sủa s. Ma trận NUM chứa các hệ số của tử số với số hàng là số ngõ ra y. Các ma trận a, b, c, c trở thành dạng chính tắt.

* Ví dụ 1:

Xét hệ thống có hàm truyền:

$$H(s) = \frac{\left[\frac{2s+3}{s^2+2s+1}\right]}{s^2+0.4s+1}$$

Để chuyển hệ thống thành dạng không gian trạng thái ta thực hiện các lệnh:

```
Num = [0   2   3
       1   2   3];
den = [1    0.4   1];
[a,b,c,d] = tf2ss (num,den);
```

ta được kết quả:

```
a =
  -0.4000   -1.0000
         1.0000      0
b =
   1
   0
c =
   2.0000    3.0000
   1.0000    2.0000
d =
   0
   1
```

Ví dụ 2: Trích từ sách '**Ứng dụng MATLAB trong điều khiển tự động'** tác giả Nguyễn Văn Giáp.

Cho hàm truyền: $(s^2+7s+2) / (s^3+9s^2+26s+24)$

```
» num=[1 7 2];
» den=[1 9 26 24];
» [A,B,C,D]=tf2ss(num,den)
```

Kết quả:

```
A =

    -9   -26   -24
     1     0     0
     0     1     0


B =

     1
     0
     0


C =

     1     7     2


D =

     0
```

7. Lệnh SS2ZP

a) Công dụng:

Chuyển hệ thống không gian sang trạng thái độ lợi cực-zero (zero pole-gain)

b) Cú pháp:

[z,p,k] = ss2zp(a,b,c,d,iu)

c) Giải thích:

ss2zp tìm các zero, cực và độ lợi không gian trạng thái.

[z,p,k] = ss2zp(a,b,c,d,iu) tìm hàm truyền dưới dạng thừa số.

$$H(s) = \frac{Z(s)}{p(s)} = k\frac{(s - Z(1)(s - Z(2)).....(s - Z(m))}{(s - p(1)(s - p(2)).....(s - p(n))}$$

của hệ thống:

$$\dot{x} = Ax + Bu$$

$$y = Cx + Du$$

từ ngõ vào thứ iu. Vector cột p chứa các cực mẫu số hàm truyền. Các zero của tử số nằm trong các cột của ma trận z với số cột là số ngõ ra y. Độ lợi của tử số hàm truyền nằm trong các cột vector k.

d) Ví dụ:

Xét hệ thống có hàm truyền:

$$H(s) = \frac{2s + 3}{s^2 + 0.4s + 1}$$

num = [2 3];

den = [1 0.4 1];

Có 2 cách để tìm các zero, cực và độ lợi của hệ thống này:

+ Cách 1:

[z,p,k] = tf2zp(num, den)

+ Cách 2:

[a,b,c,d] = tf2ss(num, den);

[z,p,k] = ss2zp(a,b,c,d,1)

và ta được cùng một kết quả như sau:

z = -1.5000

p = -0.2000 + 0.9798i

-0.2000 – 0.9798I

k = 2.0000

8. Lệnh ZP2SS:

a) Công dụng:

Chuyển từ độ cực lợi zero sang hệ không gian trạng thái.

b) Cú pháp:

[a,b,c,d] = zp2ss(z,p,k)

c) Giải thích:

zp2ss hình thành mô hình không gian trạng thái từ các zero, cực và độ lợi của hệ thống dưới dạng hàm truyền.

[a,b,c,d] = zp2ss(z,k,p) tìm hệ không gian trạnng thái:

$$\dot{x} = Ax + Bu$$

$$y = Cx + Du$$

của hệ SIMO được cho bởi hàm truyền:

$$H(s) = \frac{Z(s)}{p(s)} = k\frac{(s - Z(1)(s - Z(2)).....(s - Z(m))}{(s - p(1)(s - p(2)).....(s - p(n))}$$

Vector cột p chứa các cực và ma trận z chứa các zero với số cột là số ngõ ra. Vector k chứa các hệ số độ lợi.Các ma trận a,b,c,d trở về dạng chính tắc.

9. Lệnh TF2ZP

a) Công dụng:

Chuyển hệ thống từ dạng hàm truyền sang dạng độ lợi cực-zero.

b) Cú pháp:

[z,p,k] = tf2zp (NUM,den)

c) Giải thích:

tf2ss tìm các zero, cực và độ lợi của hệ thống được biểu diễn dưới dạng hàm truyền.

[z,p,k]= tf2zp (NUM,den) tìm hàm truyền của hệ SIMO dạng:

$$H(s) = \frac{Z(s)}{p(s)} = k\frac{(s - Z(1)(s - Z(2)).....(s - Z(m))}{(s - p(1)(s - p(2)).....(s - p(n))}$$

được cho bởi hàm truyền:

$$\frac{NUM(s)}{den(s)} = \frac{NUM(1)s^{nn-1} + ... + NUM(nn-1)s + NUM(nn)}{den(1)s^{nd-1} + ... + den(nd-1)s + den(nd)}$$

Vector den chứa các hệ số của mẫu số theo chiều giảm dần số mũ của s. Ma trận NUM chứa các hệ số tử số với số hàng là số ngõ ra. Ma trận z chứa các zero, vector cột p chứa các cực và vector k chứa các hệ số độ lợi của hàm truyền.

b) Ví dụ:

Tìm các zero và cực của hệ thống có hàm truyền:

$$H(s) = \frac{2s + 3}{s^2 + 0.4s + 1}$$

num = [2 3];

den = [1 0.4 1];

[z,p,k] = tft2zp (num,den)

ta được:

z = -1.5000

p = -0.2000 + 0.9798i

-0.2000 – 0.9798i

k = 2

10. Lệnh ZP2TF

a) Công dụng:

Chuyển đổi hệ thống từ dạng độ lợi cực zero sang dạng hàm truyền

b) Cú pháp:

[num,den] = zp2tf (z,p,k)

c) Giải thích:

zp2tf tạo ra hàm truyền đa thức từ các zero, cực và độ lợi của hệ thống.

[num,den] = zp2tf (z,p,k) tìm hàm truyền hữu tỉ:

$$\frac{NUM(s)}{den(s)} = \frac{NUM(1)s^{nn-1} + ...+ NUM(nn-1)s + NUM(nn)}{den(1)s^{nd-1} + ...+ den(nd-1)s + den(nd)}$$

được cho bởi hàm truyền dạng:

$$H(s) = \frac{Z(s)}{p(s)} = k\frac{(s-Z(1)(s-Z(2)).....(s-Z(m))}{(s-p(1)(s-p(2)).....(s-p(n))}$$

Vector cột p chứa các cực, ma trận z chứa các zero với số cột là số ngõ ra, độ lợi của tử số hàm truyền nằm trong vector k. Các hệ mẫu số đa thức nằm trong vector hàng den, các hệ số tử số nằm trong ma trận num số hàng bằng với số cột của z.

11. Lệnh POLY

a) Công dụng:

Tạo ra đa thức từ các nghiệm được chỉ định.

b) Cú pháp:

p = poly(A)

p = poly(r)

c) Giải thích:

p = poly(A), trong đó A là ma trận nxn với các phần tử là các hệ số của đa thức đặc trưng det (sI-A), tạo ra vector hàng có n+1 phần tử xếp theo thứ tự giảm dần số mũ của s.

p = poly(r), tạo ra vector hàngvới các phần tử là các hệ số của đa thức có nghiệm là các phần tử của vector ngõ ra.

d) Ví dụ 1:

Cho ma trận

A =

1 2 3

4 5 6

7 8 0

p = poly (A)

p =

1 -6 -72 -27

Ví dụ 2: Trích từ **Ví dụ 2.5** sách của tác giả Nguyễn Văn Giáp

```
%Vidu2.m
%tim nghiem cua da thuc:
%  s^6+9s^5+31.25s^4+61.25s^3+67.75s^2+14.75s+15
P=[1 9 31.25 61.25 67.75 14.75 15]
R=roots(P)
```

Kết quả:

»

P =

1.0000 9.0000 31.2500 61.2500 67.7500 14.7500 15.0000

R =

-4.0000

-3.0000

-1.0000 + 2.0000i

-1.0000 - 2.0000i

0.0000 + 0.5000i

0.0000 - 0.5000i

12. Lệnh RESIDUE

a) Công dụng:

Chuyển đổi giữa dạng khai triển phân số từng phần và dạng đa thức.

b) Cú pháp:

[r,p,k]= residue(b,a)

[b,a]= residue(r,p,k)

c) Giải thích:

[r,p,k]= residue(b,a) tìm giá trị thặng dư, các cực, và các số hạng khai triển phân số từng phần của 2 đa thức b(s) và a(s) dạng:

$$\frac{b(s)}{a(s)} = \frac{b_1 + b_2 s^{-1} + b_3 s^{-2} + ... + b_{m+1} s^{-m}}{a_1 + a_2 s^{-1} + a_3 s^{-2} + ... + a_{n+1} s^{-n}}$$

[b,a]= residue(r,p,k) chuyển dạng khai triển phân số từng phần:

$$\frac{b(s)}{a(s)} = \frac{r_1}{s - p_1} + \frac{r_2}{s - p_1} + ... + \frac{r_n}{s - p_n} + k(s)$$

về dạng đa thức với các hệ số trong vector a và b.

d) Ví dụ: Trích từ **Ví dụ 2.9** sách của tác giả Nguyễn Văn Giáp

Xác định thành phần tối giản của hàm truyền: F(s)= $(2s^3+9s+1)/(s^3+s^2+4s+4)$

```
%vidu.m
%xac dinh cac thanh phan toi gian cua ham truyen:
%          (2s^3+9s+1)
% H(s)=--------------------
%        (s^3+s^2+4s+4)
b=[2 0 9 1]
a=[1 1 4 4]
[r,p,k]=residue(b,a)
```

Kết quả:

»

b =

2 0 9 1

```
a =

    1     1     4     4


r =

   0.0000 - 0.2500i
   0.0000 + 0.2500i
  -2.0000


p =

  -0.0000 + 2.0000i
  -0.0000 - 2.0000i
  -1.0000


k =

     2
```

Từ đó hàm truyền tối giản là:

$$2 + (-2/(s+1)) + (0,25i/(s - j2)) + (-0,25i/(s - j2)) = 2 + (-2/(s+1))+ 1/(s^2+4)$$

13. <u>Lệnh SS2SS</u>

a) <u>Công dụng:</u>

Biến đổi tương đương hệ không gian trạng thái.

b) <u>Cú pháp</u>:

[at,bt,ct,dt]= ss2ss (a,b,c,d,T)

c) <u>Giải thích</u>:

[at,bt,ct,dt]= ss2ss (a,b,c,d,T) thực hiện biến đổi tương đương: z= Tx

Cuối cùng ta được hệ không gian trạng thái như sau

$$\dot{z} = TAT^{-1} z + TBu$$

$$y = CT^{-1}z+Du$$

d) Ví dụ:

Cho hệ không gian trạng thái:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 2 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u$$

$$y = [2 \quad 4] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + [1]u$$

Thực hiện biến đổi tương đươngđể cải tiến điều kiện của ma trận A.

```
a =  [1   1;2   -1];
b =  [1;0];
c =  [2   4];
d = [1];
T= balance(a);
[at,bt,ct,dt] = ss2ss(a,b,c,d,inv(T))
```

14. Lệnh CANON

a) Công dụng:

Chuyển hệ không gian trạng thái về dạng chính tắc.

b) Cú pháp:

```
[ab,bb,cb,db] = canon(a,b,c,d,'type')
```

c) Giải thích:

Lệnh canon chuyển hệ không gian trạng thái liên tục:

$$\dot{x} = Ax + Bu$$

$$y = Cx + Du$$

Thành dạng chính tắc.

+ 'type' là 'moddal': chuyển thành dạng chính tắc 'hình thái' (modal).

+ 'type' là 'companion': chuyển thành dạng chínnh tắc 'kèm theo' (companion)

Nếu 'type' không được chỉ định thì giá trị mặc nhiên là 'modal'.

Hệ thống đã chuyển đổi có cùng quan hệ vào ra (cùng hàm truyền) nhưng các trạng thái thì khác nhau.

[ab,bb,cb,db]= canon (a,b,c,d,'type') chuyển hệ không gian trạng thái thành dạng 'hình thái' trong đó có giá trị riêng thực nằm trên đường chéo của ma trận Avà các giá trị riêng phức nằm ở khối 2x2 trên đường chéo của ma trận A. Giả sử hệ thống có các giá trị riêng ( ), ma trận A sẽ là:

$$A = \begin{bmatrix} \lambda_1 & 0 & 0 & 0 \\ 0 & \sigma & \omega & 0 \\ 0 & -\omega & \sigma & 0 \\ 0 & 0 & 0 & \lambda_2 \end{bmatrix}$$

[ab,bb,cb,db]= canon (a,b,c,d,'companion') chuyển hệ không gian trạng thái thành dạng chính tắc 'kèm theo' trong đó đa thhức đặc trưng của hệ thống nằm ở cột bên phải ma trận A. Nếu một hệ thống có đa thức đặc trưng:

$$s^n + a_1 s^{n-1} + ... + a_{n-1} s + a_n$$

thì ma trận A tương ứng là:

$$A = \begin{bmatrix} 0 & 0 & 0 & ..... & -a_n \\ 1 & 0 & 0 & ..... & \vdots \\ 0 & 0 & 0 & ..... & -a_3 \\ \vdots & \vdots & \vdots & \ddots & -a_2 \\ 0 & \cdots & \cdots & 1 & -a_1 \end{bmatrix}$$

Nếu thêm vào một đối số ở ngõ ra thì:

[ab,bb,cb,db,T]= canon(a,b,c,d,'type') tạo ra vector chuyển đổi T với z= Tx

CÁC BÀI TẬP

Bài 1: Được viết dưới dạng m_file

```
%Bai tap tinh toan tong quat cua ham truyen
tu1=input('nhap (vi du: tu1=[3]), tu1= ');
mau1=input('nhap (vi du mau1=[1 4]),  mau1= ');
tu2=input('nhap  (tu2=[2 4]), tu2= ');
mau2=input('nhap (mau2=[1 2 3]),  mau2= ');
%ket qua tu3=[0 0 2 12]; mau2=[1 6 11 12]
disp('Ket noi 2 he thong noi tiep la:');
[tu3,mau3]=series(tu1,mau1,tu2,mau2)
pause

chon=input('Ban muon khao sat ham nao 1,2,3: ');
if (chon==1)
    num=tu1;
    den=mau1;
end
if (chon==2)
    num=tu2;
    den=mau2;
end
if (chon==3)
    num=tu3;
    den=mau3;
 end
 if (chon~=1)&(chon~=2)&(chon~=3)
    break
 end
 num
 den
 pause

disp('Nghiem va zero cua ham truyen la:');
[z,p,k] = tf2zp(num,den)
pause
disp('Thanh phan toi gian cua ham truyen la:');
[r,p,k] = residue(num,den)
pause
disp('In ra ham truyen o dang ty so cua hai da thuc:');
printsys(num,den,'s')
pause
disp('Tinh va hien thi tan so tu nhien va he so suy giam cua HT lien tuc
la:');
damp(den)
pause
disp('He so khuyech dai cua he thong:');
k=dcgain(num,den)
pause
disp('He so khuyech dai cua he thong kin voi he so suy giam:');
k=rlocfind(num,den)
pause
disp('Bien doi HAM TRUYEN thanh MO HINH BIEN TRANG THAI');
[A,B,C,D]=tf2ss(num,den)
A
B
C
B
disp('Bien doi ham truyen lien tuc sang roi rac la;');
```

```
Ts=input('nhap thoi gian lay mau(vi du: Ts=0.1), Ts= ');
[numd,dend]=c2dm(num,den,Ts,'zoh')
pause
disp('Gia tri rieng,bien do,tan so');
disp('va he so suy giam tuong duong cua ham truyen cua he thong roi rac');
disp('thoi gian lay mau Ts la:');
ddamp(den,Ts)
```

Sau khi chạy chương trình:

» Bài1.m

nhap (vi du: tu1=[3]), tu1= 3

nhap (vi du mau1=[1 4]), mau1= [1 4]

nhap (tu2=[2 4]), tu2= [2 4]

nhap (mau2=[1 2 3]), mau2= [1 2 3]

Ket noi 2 he thong noi tiep la:

tu3 =

    0   0   6   12

mau3 =

    1   6   11   12

Ban muon khao sat ham nao 1,2,3: 3

num =

    0   0   6   12

den =

    1   6   11   12

Nghiem va zero cua ham truyen la:

z =

    -2

p =

  -4.0000

```
  -1.0000 + 1.4142i
  -1.0000 - 1.4142i


k =

     6

Thanh phan toi gian cua ham truyen la:
r =

  -1.0909
   0.5455 - 0.9642i
   0.5455 + 0.9642i


p =

  -4.0000
  -1.0000 + 1.4142i
  -1.0000 - 1.4142i


k =

     []

In ra ham truyen o dang ty so cua hai da thuc:
num/den =

          6 s + 12
   ----------------------
   s^3 + 6 s^2 + 11 s + 12
Tinh va hien thi tan so tu nhien va he so suy giam cua HT lien tuc la:
       Eigenvalue           Damping     Freq. (rad/s)

 -1.00e+000 + 1.41e+000i     5.77e-001      1.73e+000
 -1.00e+000 - 1.41e+000i     5.77e-001      1.73e+000
 -4.00e+000                  1.00e+000      4.00e+000

He so khuyech dai cua he thong:
k =
```

```
    1

He so khuyech dai cua he thong kin voi he so suy giam:Select a point in the graphics window

selected_point =

  0.1267 + 0.1842i


k =

    1.0521

Bien doi HAM TRUYEN thanh MO HINH BIEN TRANG THAI
A =

    -6   -11   -12
     1     0     0
     0     1     0


B =

     1
     0
     0


C =

     0     6    12


D =

     0


A =

    -6   -11   -12
     1     0     0
     0     1     0
```

```
B =

     1
     0
     0


C =

     0     6    12


B =

     1
     0
     0

Bien doi ham truyen lien tuc sang roi rac la;
nhap thoi gian lay mau(vi du: Ts=0.1), Ts= 0.1

numd =

         0    0.0263    0.0015   -0.0189


dend =

    1.0000   -2.4619    2.0197   -0.5488

Gia tri rieng,bien do,tan so
va he so suy giam tuong duong cua ham truyen cua he thong roi rac
thoi gian lay mau Ts la:

       Eigenvalue          Magnitude    Equiv. Damping    Equiv. Freq. (rad/s)

 -4.00e+000                4.00e+000      -4.04e-001          3.43e+001
 -1.00e+000 + 1.41e+000i   1.73e+000      -2.44e-001          2.25e+001
 -1.00e+000 - 1.41e+000i   1.73e+000      -2.44e-001          2.25e+001
```

NHÓM LỆNH VỀ ĐÁP ỨNG TẦN SỐ
(Frequency Response)

1. Lệnh BODE

a) Công dụng:

Tìm và vẽ đáp ứng tần số giản đồ Bode.

b) Cú pháp:

[mag,phase,w] = bode(a,b,c,d)
[mag,phase,w] = bode(a,b,c,d,iu)
[mag,phase,w] = bode(a,b,c,d,iu,w)
[mag,phase,w] = bode(num,den)
[mag,phase,w] = bode(num,den,w)

c) Giải thích:

Lệnh bode tìm đáp ứng tần số biên độ và pha của hệ liên tục LTI. Giản đồ Bode dùng để phân tích đặc điểm của hệ thống bao gồm: biên dự trữ, pha dự trữ, độ lợi DC, băng thông, khả năng miễn nhiễu và tính ổn định.

Nếu bỏ qua các đối số ở vế trái của dòng lệnh thì lệnh bode sẽ vẽ ra giản đồ Bode trên màn hình.

bode(a,b,c,d) vẽ ra chuỗi giản đồ Bode, mỗi giản đồ tương ứng với một ngõ vào của hệ không gian trạng thái liên tục:

$$\dot{x} = Ax + Bu$$
$$y = Cx + Du$$

với trục tần số được xác định tự động. Nếu đáp ứng thay đổi nhanh thì cần phải xác định nhiều điểm hơn.

bode(a,b,c,d,iu) vẽ ra giản đồ Bode từ ngõ vào duy nhất iu tới tất cả các ngõ ra của hệ thống với trục tần số được xác định tự động. Đại lượng vô hướng iu là chỉ số ngõ vào của hệ thống và chỉ ra ngõ vào nào được sử dụng cho đáp ứng giản đồ Bode.

bode(num,den) vẽ ra giản đồ Bode của hàm truyền đa thức hệ liên tục

$$G(s) = num(s)/den(s)$$

trong đó num và den chứa các hệ số đa thức theo chiều giảm dần số mũ của s.

bode(a,b,c,d,iu,w) hay bode(num,den,w) vẽ ra giản đồ Bode với vector tần số w do người sử dụng xác định. Vector w chỉ ra các điểm tần số (tính bằng rad/s) mà tại đó đáp ứng tần số giản đồ Bode được tính.

Nếu vẫn giữ lại các đối số ở vế trái của dòng lệnh thì:

[mag,phase,w] = bode(a,b,c,d)
[mag,phase,w] = bode(a,b,c,d,iu)
[mag,phase,w] = bode(a,b,c,d,iu,w)
[mag,phase,w] = bode(num,den)
[mag,phase,w] = bode(num,den,w)

Sẽ không vẽ ra giản đồ Bode mà tạo ra các ma trận đáp ứng tần số mag, phase và w của hệ thống. Ma trận mag và phase có số cột bằng số ngõ ra và mỗi hàng ứng với một thành phần trong vector w.

$$G(s) = C(sI - A)^{-1}B + D$$

$$mag(\omega) = |G(j\omega)|$$

$$phase(\omega) = \angle G(j\omega)$$

Góc pha được tính bằng độ. Giá trị biên độ có thể chuyển thành decibel theo biểu thức:

$$magdB = 20*log10(mag)$$

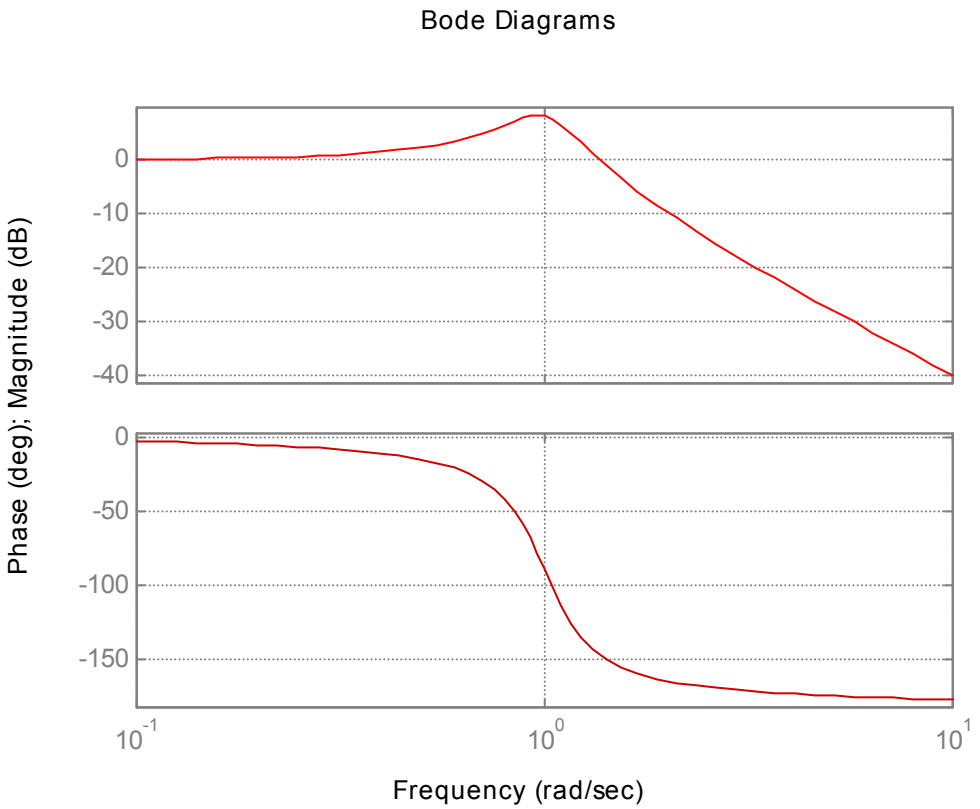
Chúng ta có thể dùng lệnh fbode thay cho lệnh bode đối với các hệ thống có thể chéo nhau. Nó sử dụng các thuật giải nhanh hơn dựa trên sự chéo hóa của ma trận hệ thống A.

d) Ví dụ:

Vẽ đáp ứng biên độ và pha của hệ bậc 2 với tần số tự nhiên $\omega_n = 1$ và hệ số tắt dần $\zeta = 0.2$

```
[a,b,c,d] = ord2(1,0.2);
bode(a,b,c,d)
grid on
```

và ta được giản đồ Bode đáp ứng tần số của hệ thống như sau:



2. Lệnh FBODE

a) Công dụng:

Vẽ đáp ứng tần số giản đồ Bode cho hệ tuyến tính liên tục.

b) Cú pháp:

[mag,phase,w] = fbode(a,b,c,d)

[mag,phase,w] = fbode(a,b,c,d,iu)

[mag,phase,w] = fbode(a,b,c,d,iu,w)

[mag,phase,w] = fbode(num,den)

[mag,phase,w] = fbode(num,den,w)

c) Giải thích:

Lệnh fbode tìm nhanh đáp ứng tần số biên độ và pha của hệ liên tục LTI.

Nếu bỏ qua các đối số ở vế trái của dòng lệnh thì lệnh fbode sẽ vẽ ra giản đồ Bode trên màn hình.

fbode(a,b,c,d) vẽ ra chuỗi giản đồ Bode, mỗi giản đồ tương ứng với một ngõ vào của hệ không gian trạng thái liên tục:

$$\dot{x} = Ax + Bu$$

$$y = Cx + Du$$

với trục tần số được xác định tự động. Nếu đáp ứng thay đổi nhanh thì cần phải xác định nhiều điểm hơn.

fbode(a,b,c,d,iu) vẽ ra giản đồ Bode từ ngõ vào duy nhất iu tới tất cả các ngõ ra của hệ thống với trục tần số được xác định tự động. iu là chỉ số ngõ vào của hệ thống và chỉ ra ngõ vào nào được sử dụng cho đáp ứng giản đồ Bode. fbode nhanh hơn nhưng kém chính xác hơn bode.

fbode(num,den) vẽ ra giản đồ Bode của hàm truyền đa thức hệ liên tục

$$G(s) = num(s)/den(s)$$

trong đó num và den chứa các hệ số đa thức theo chiều giảm dần số mũ của s.

fbode(a,b,c,d,iu,w) hay fbode(num,den,w) vẽ ra giản đồ Bode với vector tần số w do người sử dụng xác định. Vector w chỉ ra các điểm tần số (tính bằng rad/s) mà tại đó đáp ứng tần số giản đồ Bode được tính.

Nếu vẫn giữ lại các đối số ở vế trái của dòng lệnh thì:

[mag,phase,w] = fbode(a,b,c,d)

[mag,phase,w] = fbode(a,b,c,d,iu)

[mag,phase,w] = fbode(a,b,c,d,iu,w)

[mag,phase,w] = fbode(num,den)

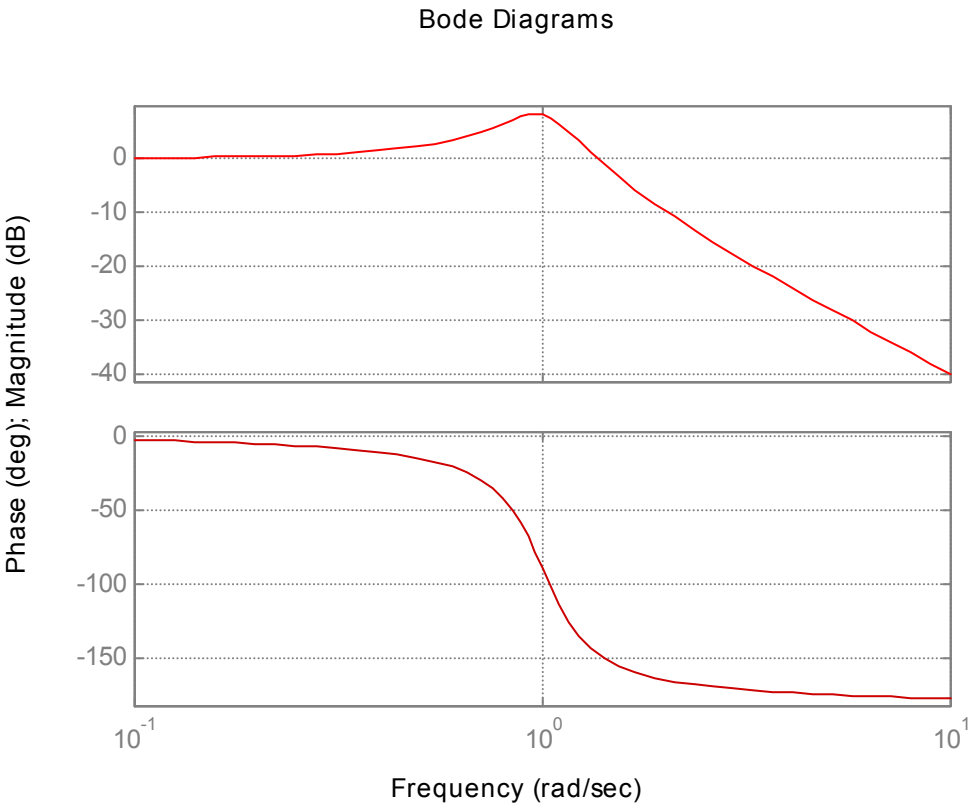
[mag,phase,w] = fbode(num,den,w)

sẽ không vẽ ra giản đồ Bode mà tạo ra các ma trận đáp ứng tần số mag, phase và w của hệ thống. Ma trận mag và phase có số cột bằng số ngõ ra và có số hàng là length(w).

d) Ví dụ:

Vẽ đáp ứng biên độ và pha của hệ bậc 2 với tần số tự nhiên $\omega_n = 1$ và hệ số tắt dần $\zeta = 0.2$

[a,b,c,d] = ord2(1,0.2);
fbode(a,b,c,d); grid on
và ta được đáp ứng như sau:



3. Lệnh DBODE

a) Công dụng:

Tìm và vẽ đáp ứng tần số giản đồ Bode của hệ gián đoạn.

b) Cú pháp:

[mag,phase,w] = dbode(a,b,c,d,Ts)
[mag,phase,w] = bode(a,b,c,d,Ts,iu)
[mag,phase,w] = bode(a,b,c,d,Ts,iu,w)
[mag,phase,w] = bode(num,den,Ts)
[mag,phase,w] = bode(num,den,Ts,w)

c) Giải thích:

Lệnh dbode tìm đáp ứng tần số biên độ và pha của hệ liên tục LTI. Lệnh dbode khác với lệnh freqz mà trong đó đáp ứng tần số đạt được với tần số chưa chuẩn hóa. Đáp ứng có được từ dbode có thể được so sánh trực tiếp với đáp ứng lệnh bode của hệ thống liên tục tương ứng. Nếu bỏ qua các đối số ở vế trái của dòng lệnh thì lệnh dbode sẽ vẽ ra giản đồ Bode trên màn hình.

dbode(a,b,c,d,Ts) vẽ ra chuỗi giản đồ Bode, mỗi giản đồ tương ứng với một ngõ vào của hệ không gian trạng thái liên tục:

$$x[n+] = Ax[n] + Bu\{n]$$
$$y[n] = Cx[n] + Du[n]$$

với trục tần số được xác định tự động. Các điểm tần số được chọn trong khoảng từ π/Ts (rad/sec), trong đó π/Ts (rad/sec) tương ứng với nửa tần số lấy mẫu (tần số Nyquist). Nếu đáp ứng thay đổi nhanh thì cần phải xác định nhiều điểm hơn. Ts là thời gian lấy mẫu.

dbode(a,b,c,d,Ts,iu) vẽ ra giản đồ Bode từ ngõ vào duy nhất iu tới tất cả các ngõ ra của hệ thống với trục tần số được xác định tự động. Đại lượng vô hướng iu là chỉ số ngõ vào của hệ thống và chỉ ra ngõ vào nào được sử dụng cho đáp ứng giản đồ Bode.

dbode(num,den,Ts) vẽ ra giản đồ Bode của hàm truyền đa thức hệ liên tục gián đoạn.

$$G(z) = num(z)/den(z)$$

trong đó num và den chứa các hệ số đa thức theo chiều giảm dần số mũ của s.

dbode(a,b,c,d,Ts,iu,w) hay dbode(num,den,Ts,w) vẽ ra giản đồ Bode với vector tần số w do người sử dụng xác định. Vector w chỉ ra các điểm tần số (tính bằng rad/s) mà tại đó đáp ứng tần số giản đồ Bode được tính. Hiện tượng trùng phổ xảy ra tại tần số lớn hơn tần số Nyquist.

Nếu vẫn giữ lại các đối số ở vế trái của dòng lệnh thì:

[mag,phase,w] = dbode(a,b,c,d,Ts)

[mag,phase,w] = dbode(a,b,c,d,Ts,iu)

[mag,phase,w] = bode(a,b,c,d,Ts,iu,w)

[mag,phase,w] = bode(num,den,Ts)

[mag,phase,w] = bode(num,den,Ts,w)

sẽ không vẽ ra giản đồ Bode mà tạo ra các ma trận đáp ứng tần số mag, phase và w của hệ thống được tính tại các giá trị tần số w. Ma trận mag và phase có số cột bằng số ngõ ra và mỗi hàng ứng với một thành phần trong vector w.

$$G(z) = C(zI - A)^{-1}B + D$$
$$mag(\omega) = \left| G(e^{j\omega T}) \right|$$
$$phase(\omega) = \angle G(e^{j\omega T})$$

trong đó T là thời gian lấy mẫu. Góc pha được tính bằng độ. Giá trị biên độ có thể chuyển thành decibel theo biểu thức:

$$magdB = 20*log10(mag)$$

d) Ví dụ:

Vẽ đáp ứng giản đồ Bode của hệ thống có hàm truyền như sau:

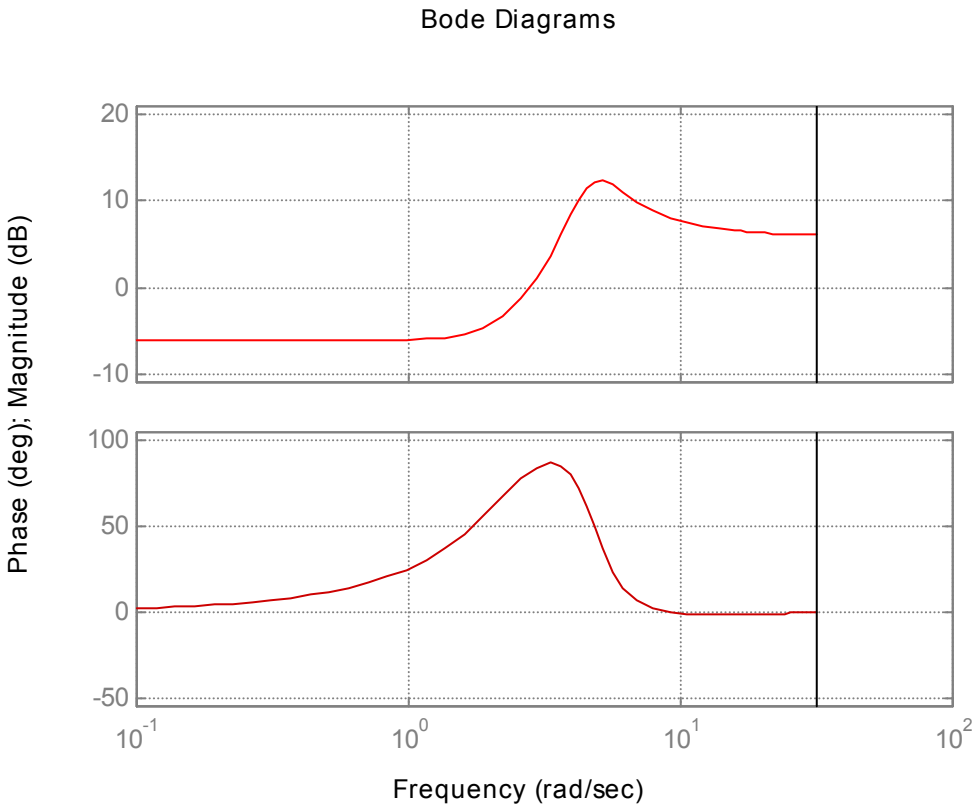
$$H(z) = \frac{2z^2 - 3.4z + 1.5}{z^2 - 1.6s + 0.8}$$

với thời gian lấy mẫu Ts = 0.1

```
num = [2        -3.4        1.5];
den = [1            -1.6            0.8];
```

dbode(num,den,0.1); grid on

và ta được đáp ứng tần số giản đồ Bode của hệ gián đoạn như sau:



4. <u>Lệnh FREQS</u>

a) <u>Công dụng</u>:

Tìm đáp ứng tần số của phép biến đổi Laplace.

b) <u>Cú pháp</u>:

h = freqs(b,a,w)

[h,w] = freqs(b,a)

[h,w] = freqs(b,a,n)

freqs(b,a)

c) <u>Giải thích</u>:

Lệnh freqs trở thành đáp ứng tần số H(jω) của bộ lọc analog.

$$H(s) = \frac{B(s)}{A(s)} = \frac{b(1)s^{nb} + b(2)s^{nb-1} + ... + b(nb+1)}{a(1)s^{na} + a(2)s^{na-1} + ... + a(na+1)}$$

trong đó vector b và a chứa các hệ số của tử số và mẫu số.

h = freqs(b,a,w) tạo ra vector đáp ứng tần số phức của bộ lọc analog được chỉ định bởi các hệ số trong vector b và a. Lệnh freqs tìm đáp ứng tần số trong mặt phẳng phức tại các thời điểm tần số được hcỉ định trong vector w.

[h,w] = freqs(b,a) tự động chọn 200 điểm tần số trong vector w để tính vector đáp ứng tần số h.

[h,w] = freqs(b,a,n) chọn ra n điểm tần số để tìm vector đáp ứng tần số h.

Nếu bỏ qua các đối số ngõ ra ở vế trái thì lệnh freqs sẽ vẽ ra đáp ứng biên độ và pha trên màn hình.

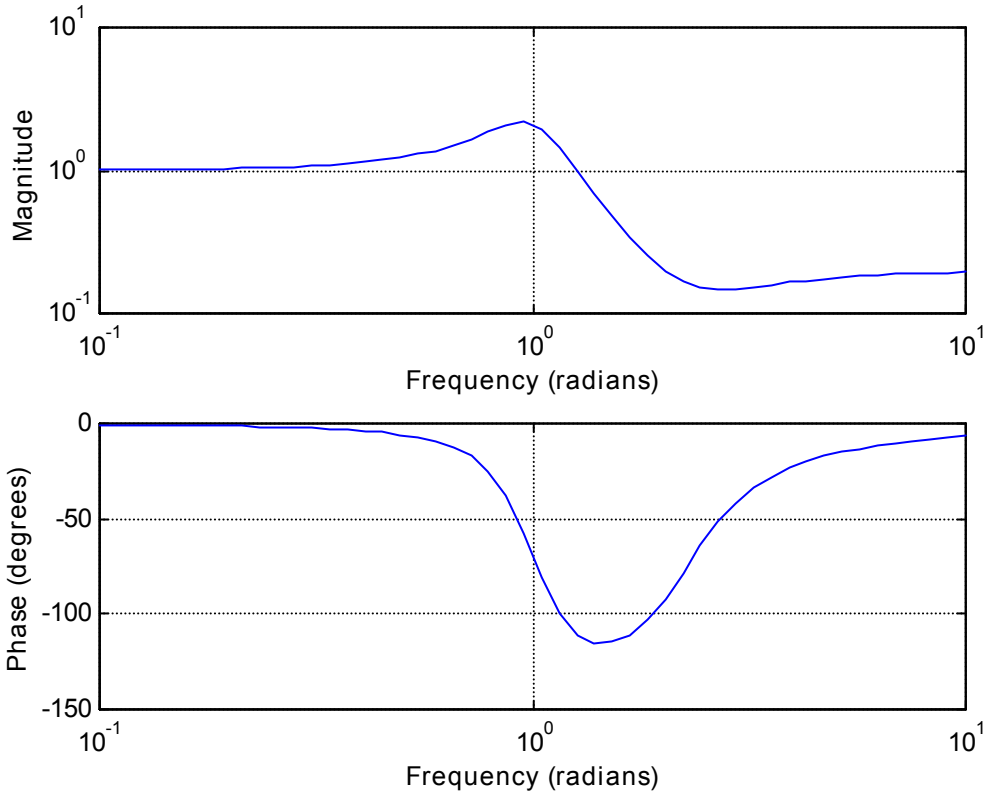
freqs chỉ dùng cho các hệ thống có ngõ vào thực và tần số dương.

d) Ví dụ:

Tìm và vẽ đáp ứng tần số của hệ thống có hàm truyền:

$$H(s) = \frac{0.2s^2 + 0.3s + 1}{s^2 + 0.4s + 1}$$

```
% Khai báo hàm truyền:
    a = [1   0.4      1];
    b = [0.2          0.3   1];
% Xác định trục tần số:
    w = logspace(-1,1);
% Thực hiện vẽ đồ thị:
    freqs(b,a,w)
```



5. Lệnh FREQZ

a) Công dụng:

Tìm đáp ứng tần số của bộ lọc số.

b) Cú pháp:

[h,w] = freqz(b,a,n)

[h,f] = freqz(b,a,n,Fs)
[h,w] = freqz(b,a,n,'whole')
[h,f] = freqz(b,a,n,'whole',Fs)
h = freqz(b,a,w)
h = freqz(b,a,f,Fs)
freqz(b,a)

c) Giải thích:

Lệnh freqz tìm đáp ứng tần số $H(e^{j\omega T})$ của bộ lọc số từ các hệ số tử số và mẫu số trong vector b và a.

[h,w] = freqz(b,a,n) tìm đáp ứng tần số của bộ lọc số với n điểm

$$H(z) = \frac{B(z)}{A(z)} = \frac{b(1)+b(2)z^{-1}+.....+b(nb+1)z^{-nb}}{a(1)+a(2)z^{-1}+.....+a(na+1)z^{-na}}$$

từ các hệ số trong vector b và a. freqz tạo ra vector đáp ứng tần số hồi tiếp và vector w chứa n điểm tần số. freqz xác định đáp ứng tần số tại n điểm nằm đều nhau quanh nửa vòng tròn đơn vị, vì vậy w chứa n điểm giữa 0 và π.

[h,f] = freqz(b,a,n,Fs) chỉ ra tần số lấy mẫu dương Fs (tính bằng Hz). Nó tạo ra vector f chứa các điểm tần số thực giữa 0 và Fs/2 mà tại đó lệng sẽ tính đáp ứng tần số.

[h,w] = freqz(b,a,n,'whole') và [h,f] = freqz(b,a,n,'whole',Fs) sử dụng nđiểm quanh vòng tròn đơn vị (từ 0 tới 2π hoặc từ 0 tới Fs)

h = freqz(b,a,w) tạo ra đáp ứng tần số tại các điểm tần số được chỉ trong vector w. Các điểm tần số này phải nằm trong khoảng (0 ÷2π).

h = freqz(b,a,f,Fs) tạo ra đáp ứng tần số tại các điểm tần số được chỉ trong vector f. Các điểm tần số này phải nằm trong khoảng (0 ÷ Fs).

Nếu bỏ qua các đối số ngõ ra thì lệnh freqz vẽ ra các đáp ứng biên độ và pha trên màn hình.

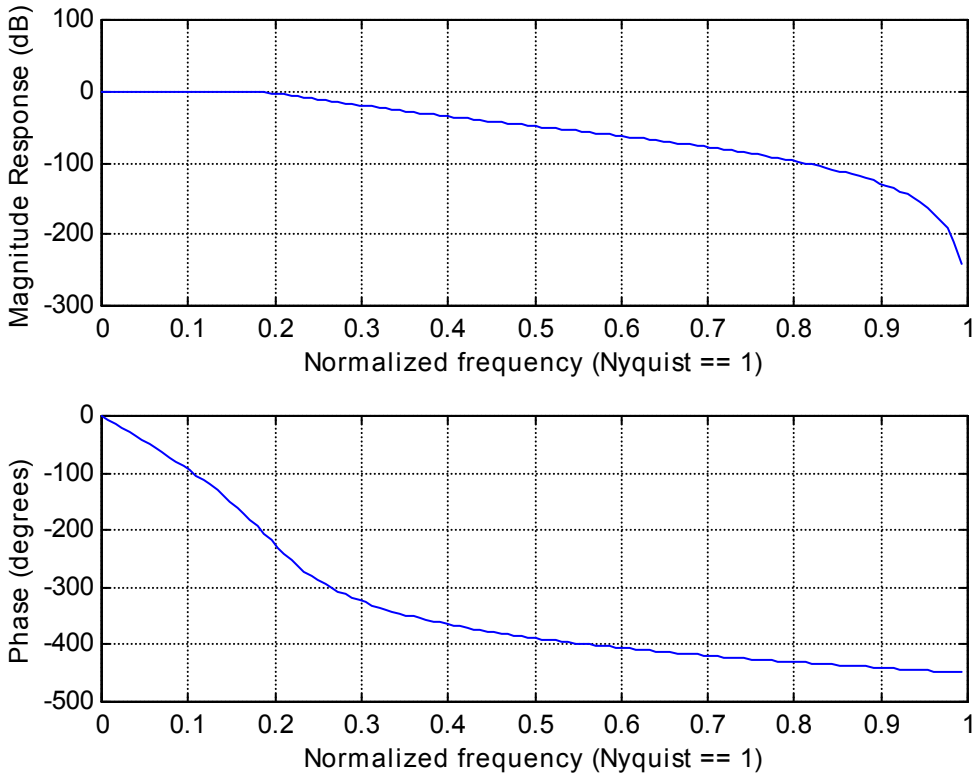
Lệnh freqz dùng cho các hệ thống có ngõ vào thực hoặc phức.

d) Ví dụ:

Vẽ đáp ứng biên độ và pha của bộ lọc Butter.

```
[b,a] = butter(5,0.2);
freqz(b,a,128)
```

và ta được đồ thị đáp ứng:



6. Lệnh NYQUIST

a) Công dụng:

Vẽ biểu đồ đáp ứng tần số Nyquist.

b) Cú pháp:

[re,im,w] = nyquist(a,b,c,d)

[re,im,w] = nyquist(a,b,c,d,iu)

[re,im,w] = nyquist(a,b,c,d,iu,w)

[re,im,w] = nyquist(num,den)

[re,im,w] = nyquist(num,den,w)

c) Giải thích:

Lệnh nyquist tìm đáp ừng tần số Nyquist của hệ liên tục LTI. Biểu đồ Nyquist dùng để phân tích đặc điểm của hệ thống bao gồm: biên dự trữ, pha dự trữ và tính ổn định.

Nếu bỏ qua các đối số ở vế trái của dòng lệnh thì nyquist sẽ vẽ ra biểu đồ Nyquist trên màn hình.

Lệnh nyquist có thể xác định tính ổn định của hệ thống hồi tiếp đơn vị. Cho biểu đồ Nyquist của hàm truyền vòng hở G(s), hàm truyền vòng kín:

$$G_{cl}(s) = \frac{G(s)}{1+G(s)}$$

là ổn định khi biểu đồ Nyquist bao quanh điểm –1+j0 P lần theo chiều kim đồng hồ, trong đó P là số cực vòng hở không ổn định.

nyquist(a,b,c,d) vẽ ra chuỗi biểu đồ Nyquist, mỗi đồ thị ứng vời mối quan hệ giữa một ngõ vào và một ngõ ra của hệ không gian trạng thái liên tục:

$$\dot{x} = Ax + Bu$$

$$y = Cx + Du$$

với trục tần số được xác định tự động. Nếu đáp ứng thay đổi càng nhanh thì cần phải xác định càng nhiều điểm trên trục tần số.

nyquist(a,b,c,d,iu) vẽ ra biểu đồ Nyquist từ ngõ vào duy nhất iu tới tất cả các ngõ ra của hệ thống với trục tần số được xác định tự động. Đại lượng vô hướng iu là chỉ số ngõ vào của hệ thống và chỉ ra ngõ vào nào được sử dụng cho đáp ứng Nyquist.

nyquist(num,den) vẽ ra biểu đồ Nyquist của hàm truyền đa thức hệ liên tục

$$G(s) = num(s)/den(s)$$

trong đó num và den chứa các hệ số đa thức theo chiều giảm dần số mũ của s.

nyquist(a,b,c,d,iu,w) hoặc nyquist(num,den,w) vẽ ra biểu đồ Nyquist với vector tần số w do người sử dụng xác định. Vector w chỉ ra các điểm tần số (tính bằng rad/s) mà tại đó đáp ứng Nyquist được tính.

Nếu vẫn giữ lại các đối số ở vế trái của dòng lệnh thì:

[re,im,w] = nyquist(a,b,c,d)

[re,im,w] = nyquist(a,b,c,d,iu)

[re,im,w] = nyquist(a,b,c,d,iu,w)

[re,im,w] = nyquist(num,den)

[re,im,w] = nyquist(num,den,w)

không vẽ ra biểu đồ Nyquist mà tạo ra đáp ứng tần số của hệ thống dưới dạng các ma trận re, im và w. Các ma trận re và im có số cột bằng số ngõ ra và mỗi hàng ứng với một thành phần trong vector w.

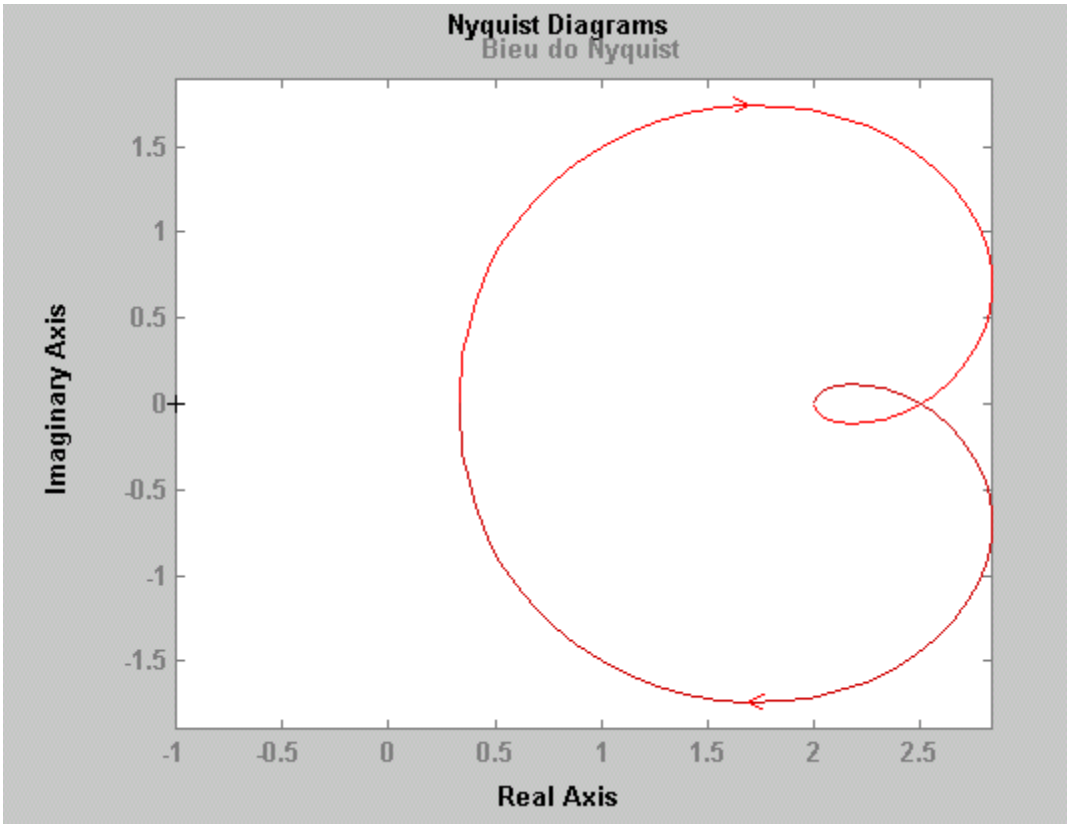
d) Ví dụ:

Vẽ biểu đồ Nyquist của hệ thống có hàm truyền:

$$H(s) = \frac{2s^2 + 5s + 1}{s^2 + 2s + 3}$$

```
num = [2    5      1];
den = [1            2    3];
nyquist(num,den); title('Bieu do Nyquist')
```

và ta được biểu đồ Nyquist như hình vẽ:



7. Lệnh DNYQUIST

a) Công dụng:

Vẽ biểu đồ đáp ứng tần số Nyquist của hệ gián đoạn.

b) Cú pháp:

[re,im,w] = dnyquist(a,b,c,d,Ts)

[re,im,w] = dnyquist(a,b,c,d,Ts,iu)

[re,im,w] = dnyquist(a,b,c,d,Ts,iu,w)

[re,im,w] = dnyquist(num,den,Ts)

[re,im,w] = dnyquist(num,den,Ts,w)

c) Giải thích:

Lệnh dnyquist tìm đáp ừng tần số Nyquist của hệ gián đoạn LTI. Biểu đồ Nyquist dùng để phân tích đặc điểm của hệ thống bao gồm: biên dự trữ, pha dự trữ và tính ổn định. Đáp ứng tần số dùng lệnh dnyquist có thể so sánh trực tiếp với đáp ứng nyquist của hệ liên tục tương ứng.

Nếu bỏ qua các đối số ở vế trái của dòng lệnh thì dnyquist sẽ vẽ ra biểu đồ Nyquist trên màn hình.

Lệnh dnyquist có thể xác định tính ổn định của hệ thống hồi tiếp đơn vị. Cho biểu đồ Nyquist của hàm truyền vòng hở G(s), hàm truyền vòng kín:

$$G_{cl}(z) = \frac{G(z)}{1+G(z)}$$

là ổn định khi biểu đồ Nyquist bao quanh điểm –1+j0 P lần theo chiều kim đồng hồ, trong đó P là số cực vòng hở không ổn định.

dnyquist(a,b,c,d,Ts) vẽ ra chuỗi biểu đồ Nyquist, mỗi đồ thị ứng vời mối quan hệ giữa một ngõ vào và một ngõ ra của hệ không gian trạng thái gián đoạn:

$$x[n+] = Ax[n] + Bu\{n]$$
$$y[n] = Cx[n] + Du[n]$$

với trục tần số được xác định tự động. Các điểm tần số được chọn trong khoảng từ 0 đến π/Ts radians tương ứng với nửa tần số lấy mẫu (tần số Nyquist). Nếu đáp ứng thay đổi càng nhanh thì cần phải xác định càng nhiều điểm trên trục tần số. Tần số là thời gian lấy mẫu.

dnyquist(a,b,c,d,Ts,iu) vẽ ra biểu đồ Nyquist từ ngõ vào duy nhất iu tới tất cả các ngõ ra của hệ thống với trục tần số được xác định tự động. Đại lượng vô hướng iu là chỉ số ngõ vào của hệ thống và chỉ ra ngõ vào nào được sử dụng cho đáp ứng Nyquist.

dnyquist(num,den,Ts) vẽ ra biểu đồ Nyquist của hàm truyền đa thức hệ gián đoạn:

$$G(s) = num(s)/den(s)$$

trong đó num và den chứa các hệ số đa thức theo chiều giảm dần số mũ của s.

dnyquist(a,b,c,d,Ts,iu,w) hoặc dnyquist(num,den,w) vẽ ra biểu đồ Nyquist với vector tần số w do người sử dụng xác định. Vector w chỉ ra các điểm tần số (tính bằng rad/s) mà tại đó đáp ứng Nyquist được tính. Hiện tượng trùng phổ xảy ra tại tần số lớn hơn tần số Nyquist (π/Ts rad/s).

Để tạo ra trục tần số với các khoảng tần số bằng nhau theo logarit ta dùng lệnh logspace.

Nếu vẫn giữ lại các đối số ở vế trái của dòng lệnh thì:

[re,im,w] = dnyquist(a,b,c,d,Ts)

[re,im,w] = dnyquist(a,b,c,d,Ts,iu)

[re,im,w] = dnyquist(a,b,c,d,Ts,iu,w)

[re,im,w] = dnyquist(num,den,Ts)

[re,im,w] = dnyquist(num,den,Ts,w)

không vẽ ra biểu đồ Nyquist mà tạo ra đáp ứng tần số của hệ thống dưới dạng các ma trận re, im và w. Các ma trận re và im chứa các phần thực và phần ảo của đáp ứng tần số của hệ thống được tính tại các giá trị tần số w, re và im có số cột bằng số ngõ ra và mỗi hàng ứng với một thành phần trong vector w.

d) Ví dụ:

Vẽ biểu đồ Nyquist của hệ gián đoạn có hàm truyền:

$$H(z) = \frac{2z^2 - 3.4z + 1.5}{z^2 - 1.6z + 0.8}$$

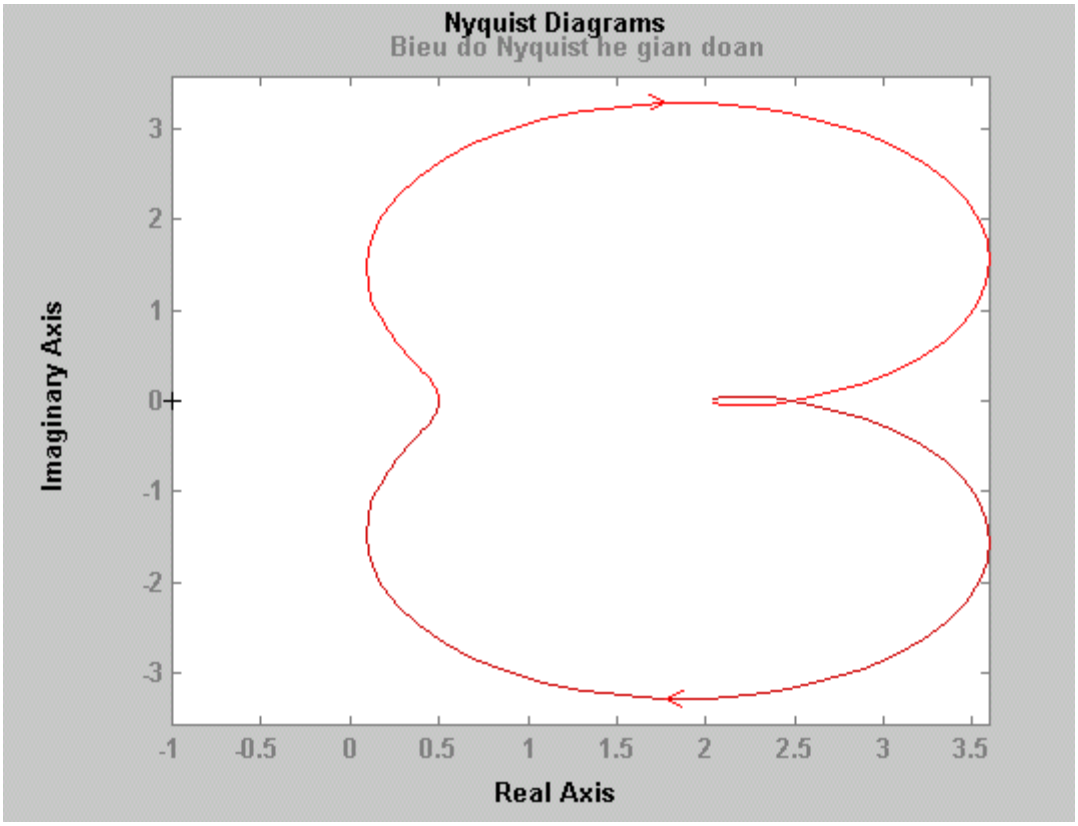
với thời gian lấy mẫu Ts = 0.1

% Xác định hàm truyền:

```
num = [2   -3.4   1.5];
```

```
        den = [1    -1.6     0.8];
% Vẽ biểu đồ Nyquist:
        dnyquist(num,den,0.1)
        title('Bieu do Nyquist he gian doan')
```

và ta được biểu đồ Nyquist hệ gián đoạn như sau:



8. <u>Lệnh NICHOLS</u>

a) <u>Công dụng</u>:

Vẽ biểu đồ đáp ứng tần số Nichols.

b) <u>Cú pháp</u>:

```
[mag,phase,w] = nichols(a,b,c,d)
[mag,phase,w] = nichols(a,b,c,d,iu)
[mag,phase,w] = nichols(a,b,c,d,iu,w)
[mag,phase,w] = nichols(num,den)
[mag,phase,w] = nichols(num,den,w)
```

c) <u>Giải thích</u>:

Lệnh nichols tìm đáp ứng tần số Nichols của hệ liên tục LTI. Biểu đồ Nichols được dùng để phân tích đặc điểm của hệ vòng hở và hệ vòng kín.

Nếu bỏ qua các đối số ở vế trái của dòng lệnh thì lệnh nichols sẽ vẽ ra biểu đồ Nichols trên màn hình.

nichols(a,b,c,d) vẽ ra chuỗi biểu đồ Nichols, mỗi đồ thị tương ứng với mối quan hệ giữa một ngõ vào và một ngõ ra của hệ không gian trạng thái liên tục:

$$\dot{x} = Ax + Bu$$

$$y = Cx + Du$$

với trục tần số được xác định tự động. Nếu đáp ứng thay đổi nhanh thì cần phải xác định càng nhiều điểm trên trục tần số.

nichols(a,b,c,d,iu) vẽ ra biểu đồ Nichols từ ngõ vào duy nhất iu tới tất cả các ngõ ra của hệ thống với trục tần số được xác định tự động. Đại lượng vô hướng iu là chỉ số ngõ vào của hệ thống và chỉ ra ngõ vào nào được sử dụng cho đáp ứng Nichols.

nichols(num,den) vẽ ra biểu đồ Nichols của hàm truyền đa thức hệ liên tục

$$G(s) = num(s)/den(s)$$

trong đó num và den chứa các hệ số đa thức theo chiều giảm dần số mũ của s.

nichols(a,b,c,d,iu,w) hay nichols(num,den,w) vẽ ra biểu đồ Nichols với vector tần số w do người sử dụng xác định. Vector w chỉ định những điểm tần số (tính bằng rad/s) mà tại đó đáp ứng Nichols được tính.

Để tạo ra trục tần số với các khoảng tần số bằng nhau theo logarit ta dùng lệnh logspace.

Nếu giữ lại các đối số ở vế trái của dòng lệnh thì:

[mag,phase,w] = nichols(a,b,c,d)

[mag,phase,w] = nichols(a,b,c,d,iu)

[mag,phase,w] = nichols(a,b,c,d,iu,w)

[mag,phase,w] = nichols(num,den)

[mag,phase,w] = nichols(num,den,w)

sẽ không vẽ ra biểu đồ Nichols mà tạo ra đáp ứng tần số của hệ thống dưới dạng các ma trận mag, phase và w. Các ma trận mag và phase chứa đáp ứng biên độ và pha của hệ thống được xác định tại những điểm tần số w. Ma trận mag và phase có số cột bằng số ngõ ra và mỗi hàng ứng với một thành phần trong vector w.

$$G(s) = C(sI - A)^{-1}B + D$$

$$mag(\omega) = |G(j\omega)|$$

$$phase(\omega) = \angle G(j\omega)$$

Góc pha được tính bằng độ và nằm trong khoảng -360^0 tới 0^0.

Giá trị biên độ có thể chuyển về đơn vị decibel theo công thức:

$$magdB = 20*log10(mag)$$

Để vẽ lưới biểu đồ Nichols ta dùng lệnh ngrid.

d) Ví dụ: Trích trang 11-150 sách **'Control System Toolbox'**

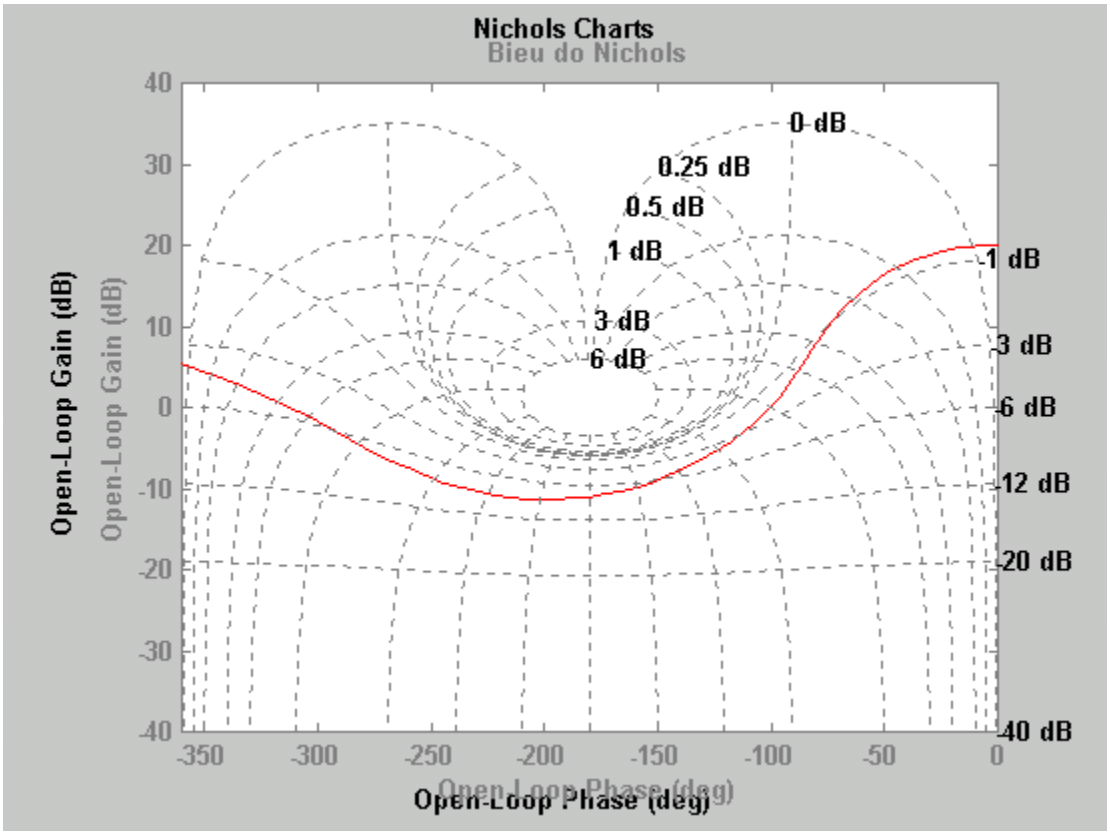
Vẽ đáp ứng Nichols của hệ thống có hàm truyền:

$$H(s) = \frac{-4s^4 + 48s^3 - 18s^2 + 250s + 600}{s^4 + 30s^3 + 282s^2 + 525s + 60}$$

```
num = [-4   48   -18   250   600];
den = [1          30   282   525   60];
```

```
nichols(num,den)
title('Bieu do Nichols')
ngrid('new')
```

và ta được biểu đồ Nichols như hình vẽ:



9. Lệnh DNICHOLS

a) Công dụng:

Vẽ biểu đồ đáp ứng tần số Nichols của hệ gián đoạn.

b) Cú pháp:

```
[mag,phase,w] = dnichols(a,b,c,d,Ts)
[mag,phase,w] = dnichols(a,b,c,d,Ts,iu)
[mag,phase,w] = dnichols(a,b,c,d,Ts,iu,w)
[mag,phase,w] = dnichols(num,den,Ts)
[mag,phase,w] = dnichols(num,den,Ts,w)
```

c) Giải thích:

Lệnh dnichols tìm đáp ứng tần số Nichols của hệ gián đoạn LTI. Biểu đồ Nichols được dùng để phân tích đặc điểm của hệ vòng hở và hệ vòng kín. Đáp ứng từ lệnh dnichols có thể so sánh trực tiếp với đáp ứng từ lệnh nichols của hệ liên tục tương ứng.

Nếu bỏ qua các đối số ở vế trái của dòng lệnh thì lệnh dnichols sẽ vẽ ra biểu đồ Nichols trên màn hình.

dnichols(a,b,c,d,Ts) vẽ ra chuỗi biểu đồ Nichols, mỗi đồ thị tương ứng với mối quan hệ giữa một ngõ vào và một ngõ ra của hệ không gian trạng thái gián đoạn:

$$x[n+] = Ax[n] + Bu\{n]$$
$$y[n] = Cx[n] + Du[n]$$

với trục tần số được xác định tự động. Các điểm tần số được chọn trong khoảng từ 0 tới π/Ts radians. Nếu đáp ứng thay đổi nhanh thì cần phải xác định càng nhiều điểm trên trục tần số.

dnichols(a,b,c,d,Ts,iu) vẽ ra biểu đồ Nichols trên màn hình từ ngõ vào duy nhất iu tới tất cả các ngõ ra của hệ thống với trục tần số được xác định tự động. Đại lượng vô hướng iu là chỉ số ngõ vào của hệ thống và chỉ ra ngõ vào nào được sử dụng cho đáp ứng Nichols.

dnichols(num,den,Ts) vẽ ra biểu đồ Nichols của hàm truyền đa thức hệ gián đoạn

$$G(z) = num(z)/den(z)$$

trong đó num và den chứa các hệ số đa thức theo chiều giảm dần số mũ của s.

dnichols(a,b,c,d,Ts,iu,w) hay dnichols(num,den,Ts,w) vẽ ra biểu đồ Nichols với vector tần số w do người sử dụng xác định. Vector w chỉ định những điểm tần số (tính bằng rad/s) mà tại đó đáp ứng Nichols được tính. Hiện tượng trùng phổ xảy ra tại tần số lớn hơn tần số Nyquist (π/Ts rad/s).

Để tạo ra trục tần số với các khoảng tần số bằng nhau theo logarit ta dùng lệnh logspace.

Nếu giữ lại các đối số ở vế trái của dòng lệnh thì:

[mag,phase,w] = dnichols(a,b,c,d,Ts)

[mag,phase,w] = dnichols(a,b,c,d,Ts,iu)

[mag,phase,w] = dnichols(a,b,c,d,Ts,iu,w)

[mag,phase,w] = dnichols(num,den,Ts)

[mag,phase,w] = dnichols(num,den,Ts,w)

không vẽ ra biểu đồ Nichols mà tạo ra đáp ứng tần số của hệ thống dưới dạng các ma trận mag, phase và w. Các ma trận mag và phase chứa đáp ứng biên độ và pha của hệ thống được xác định tại những điểm tần số w. Ma trận mag và phase có số cột bằng số ngõ ra và mỗi hàng ứng với một thành phần trong vector w.

$$G(z) = C(zI - A)^{-1}B + D$$
$$mag(\omega) = \left| G(e^{j\omega T}) \right|$$
$$phase(\omega) = \angle G(e^{j\omega T})$$

trong đó T là thời gian lấy mẫu. Góc pha được tính bằng độ và nằm trong khoảng -360^0 tới 0^0.

Giá trị biên độ có thể chuyển về đơn vị decibel theo công thức:

$$magdB = 20*log10(mag)$$

Để vẽ lưới biểu đồ Nichols ta dùng lệnh ngrid.

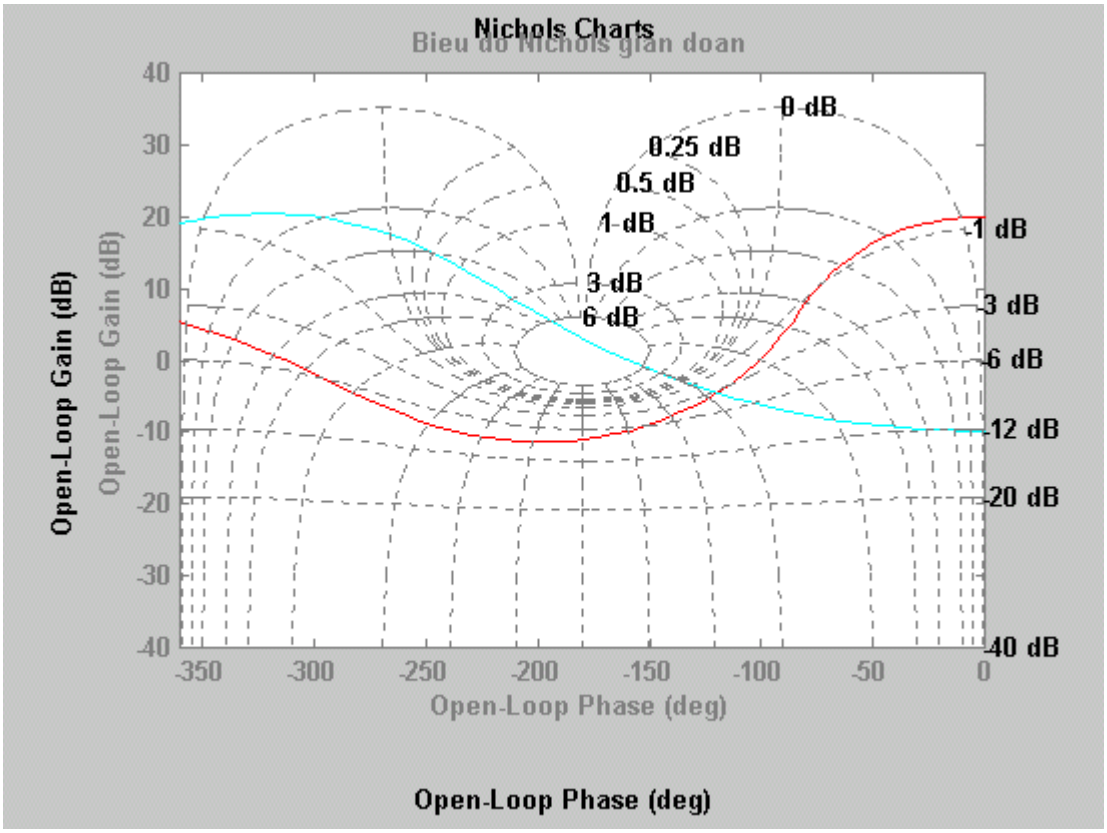
d) Ví dụ:

Vẽ đáp ứng Nichols của hệ thống có hàm truyền:

$$H(z) = \frac{1.5}{z^4 + 1.1z^3 + 1.36z^2 + 0.88z + 0.31}$$

```
num = 1.5;
den = [1        1.1   1.36   0.88   0.31];
ngrid('new')
dnichols(num,den,0.05)
title('Bieu do Nichols gian doan')
```

và ta được biểu đồ Nichols của hệ gián đoạn:



10. <u>Lệnh NGRID</u>

a) <u>Công dụng</u>:

Tạo lưới cho đồ thị Nichols.

b) <u>Cú pháp</u>:

ngrid

ngrid('new')

c) <u>Giải thích</u>:

Lệnh grid tạo lưới cho đồ thị Nichols. Đồ thị này có liên hệ với số phức H/(1+H), trong đó H là một số phức bất kỳ. Nếu H là một điểm trên đáp ứng tần số vòng hở của hệ SISO thì H/(1+H) là giá trị tương ứng trên đáp ứng tần số vòng kín của hệ thống.

ngrid tạo ra lưới trong vùng có biên độ từ –40 dB tới 40 dB và góc pha từ -360^0 tới 0^0 với các đường hằng số mag(H/(1+H)) và angle(H/(1+H)) được vẽ.

ngrid vẽ lưới đồ thị Nichols ngoài biểu đồ Nichols đã có như biểu đồ được tạo ra bởi lệnh nichols hoặc dnichols.

ngrid('new') xóa màn hình đồ họa trước khi vẽ lưới và thiết lập trạng thái giữ để đáp ứng Nichols có thể được vẽ bằng cách dùng lệnh:

ngrid('new')

nichols(num,den) hay nichols(a,b,c,d,iu)

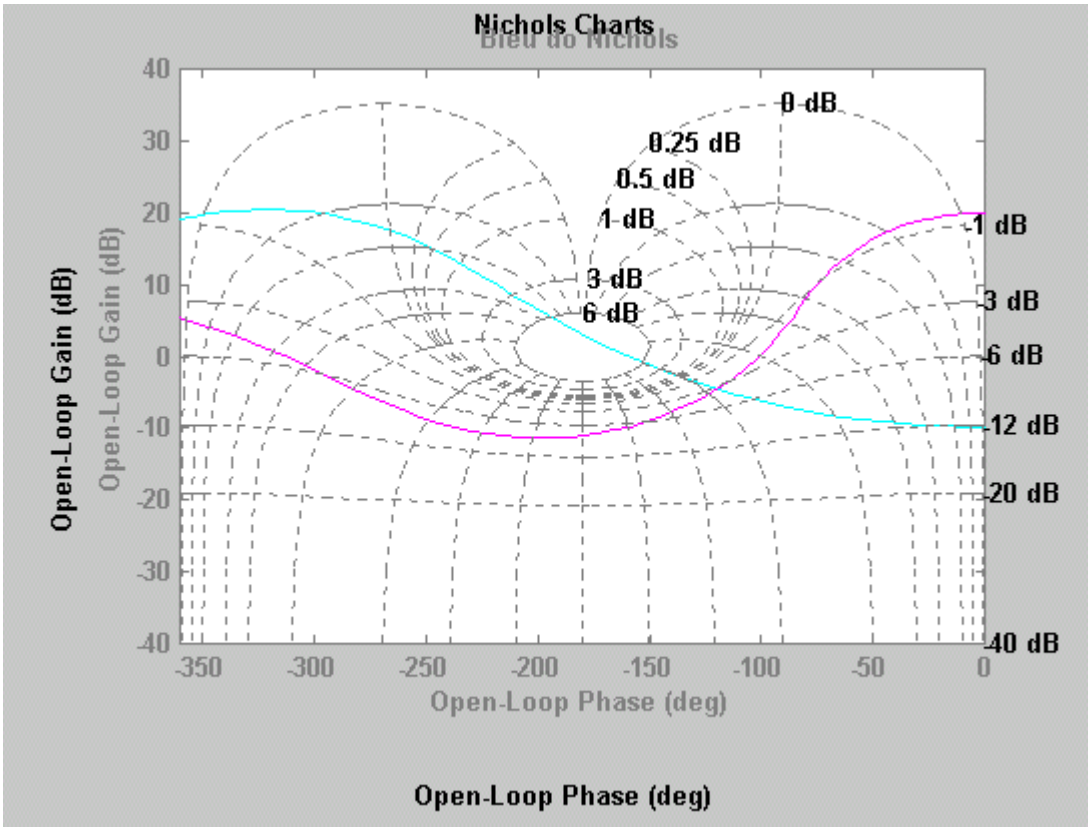
d) Ví dụ:

Vẽ lưới trên biểu đồ Nichols của hệ thống:

$$H(s) = \frac{-4s^4 + 48s^3 - 18s^2 + 250s + 600}{s^4 + 30s^3 + 282s^2 + 525s + 60}$$

```
num = [-4   48   -18   250   600];
den = [1          30   282   525   60];
nichols(num,den)
title('Bieu do Nichols')
ngrid('new')
```

và ta được đồ thị đáp ứng như sau:



11. Lệnh MARGIN

a) Công dụng:

Tính biên dự trữ và pha dự trữ.

b) Cú pháp:

[Gm,Pm,Wcp,Wcg] = margin(mag,phase,w)

[Gm,Pm,Wcp,Wcg] = margin(num,den)

[Gm,Pm,Wcp,Wcg] = margin(a,b,c,d)

c) Giải thích:

Lệnh margin tính biên dự trữ (gain margin), pha dự trữ (phase margin) và tần số cắt (crossover frequency) từ dữ liệu đáp ứng tần số. Biên dự trữ và pha dự trữ dựa trên hệ thống vòng hở SISO và cho biết tính ổn định tương đối của hệ thống khi hệ thống là hệ thống vòng kín.

Nếu bỏ qua các đối số ở vế trái dòng lệnh thì giản đồ Bode với biên dự trữ và pha dự trữ sẽ được vẽ trên màn hình.

Biên dự trữ là độ lợi cần tăng thêm để tạo ra độ lợi vòng đơn vị tại tần số mà góc pha bằng -180^0. Nói cách khác, biên dự trữ là 1/g nếu g là độ lợi tại tần số góc pha -180^0. Tương tự, pha dự trữ là sự khác biệt giữa góc pha đáp ứng và -180^0 khi độ lợi là 1. Tần số mà tại đó biên độ là 1 được gọi là tần số độ lợi đơn vị (unity-gain frequency) hoặc tần số cắt.

margin(num,den) tính biên dự trữ và pha dự trữ của hàm truyền liên tục:

$$G(s) = num/den$$

Tương tự, margin(a,b,c,d) tính độ dự trữ của hệ không gian trạng thái (a,b,c,d). Với cách này, lệnh margin chỉ sử dụng cho hệ liên tục. Đối với hệ gián đoạn, ta sử dụng lệnh dbode để tìm đáp ứng tần số rồi gọi margin.

[mag,phase,w] = dbode(a,b,c,d,Ts)

margin(mag,phase,w)

[Gm,Pm,Wcp,Wcg] = margin(mag,phase,w) sẽ không vẽ ra các đồ thị đáp ứng mà tạo ra các ma trận biên dự trữ Gm, pha dự trữ Pm, tần số kết hợp Wcp, Wcg được cho bởi các vector biên độ mag, phase và tần số w của hệ thống. Các giá trị chính xác được tìm ra bằng cách dùng phép nội suy giữa các điểm tần số. Góc pha được tính bằng độ.

d) Ví dụ:

Tìm biên dự trữ, pha dự trữ và vẽ giản đồ Bode của hệ bậc 2 có $\omega_n = 1$ và $\zeta = 0.2$

```
[a,b,c,d] = ord(1,0.2);
bode(a,b,c,d)
margin(a,b,c,d)
[Gm,Pm,Wcp,Wcg] = margin(a,b,c,d)
```

và ta được kết quả:

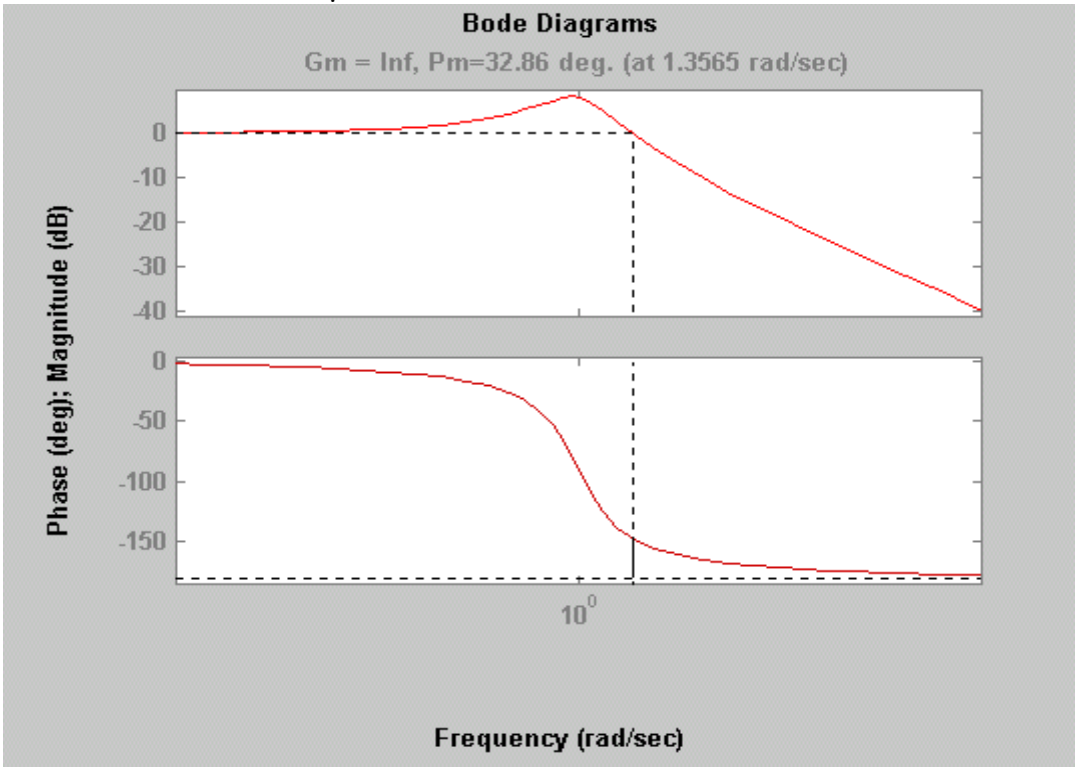
Gm = lnf(∞)

Pm = 32.8599 độ

Wcg = NaN (không xác định)

Wcp = 1.3565

Giản đồ Bode của hệ:



12. Lệnh SIGMA

a) Công dụng:

Tìm giản đồ Bode giá trị suy biến của hệ không gian trạng thái.

b) Cú pháp:

[sv,w] = sigma(a,b,c,d)

[sv,w] = sigma(a,b,c,d,'inv')

[sv,w] = sigma(a,b,c,d,w)

[sv,w] = sigma(a,b,c,d,w,'inv')

c) Giải thích:

Lệnh sigma tính các giá trị suy biến của ma trận phức $C(j\omega I-A)^{-1}B+D$ theo hàm của tần số ω. Các giá trị suy biến là mở rộng của đáp ứng biên độ giản đồ Bode của hệ MIMO.

Nếu bỏ qua các đối số ở vế trái của dòng lệnh thì sigma sẽ vẽ ra giản đồ Bode của giá trị suy biến trên màn hình.

[sv,w] = sigma(a,b,c,d) vẽ ra giản đồ suy biến của ma trận phức:

$$G(w) = C(j\omega I-A)^{-1}B+D$$

theo hàm của tần số. Trục tần số được chọn tự động và phối hợp nhiều điểm nếu đồ thị thay điểm nhanh.

Đối với các ma trận vuông, sigma(a,b,c,d,'inv') vẽ đồ thị các giá trị suy biến của ma trận phức đảo:

$$G^{-1}(w) = [C(j\omega I-A)^{-1}B+D]^{-1}$$

sigma(a,b,c,d,w) hoặc sigma(a,b,c,d,w,'inv') vẽ đồ thị các giá trị suy biến với vector tần số do người sử dụng xác định. Vector w chỉ ra những tần số (tính bằng rad/s) mà tại đó đáp ứng các giá trị suy biến được tính.

Nếu giữ lại các đối số ở vế trái dòng lệnh thì:

[sv,w] = sigma(a,b,c,d)

[sv,w] = sigma(a,b,c,d,'inv')

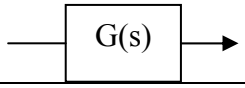
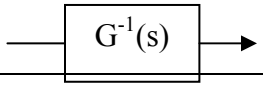
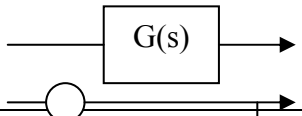
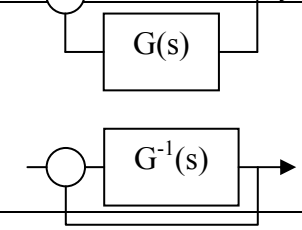
[sv,w] = sigma(a,b,c,d,w)

[sv,w] = sigma(a,b,c,d,w,'inv')

không vẽ ra các đồ thị đáp ứng mà tạo ra các ma trận suy biến theo chiều giảm dần của bậc tương ứng với các điểm tần số trong vector w.

Đối với phép phân tích rắn chắc, các giá trị suy biến của ma trận hàm truyền đặc biệt được phân tích.

Về thực hiện các lệnh để đạt được ma trận hàm truyền mong muốn của một số khối được trình bày trong bảng sau:

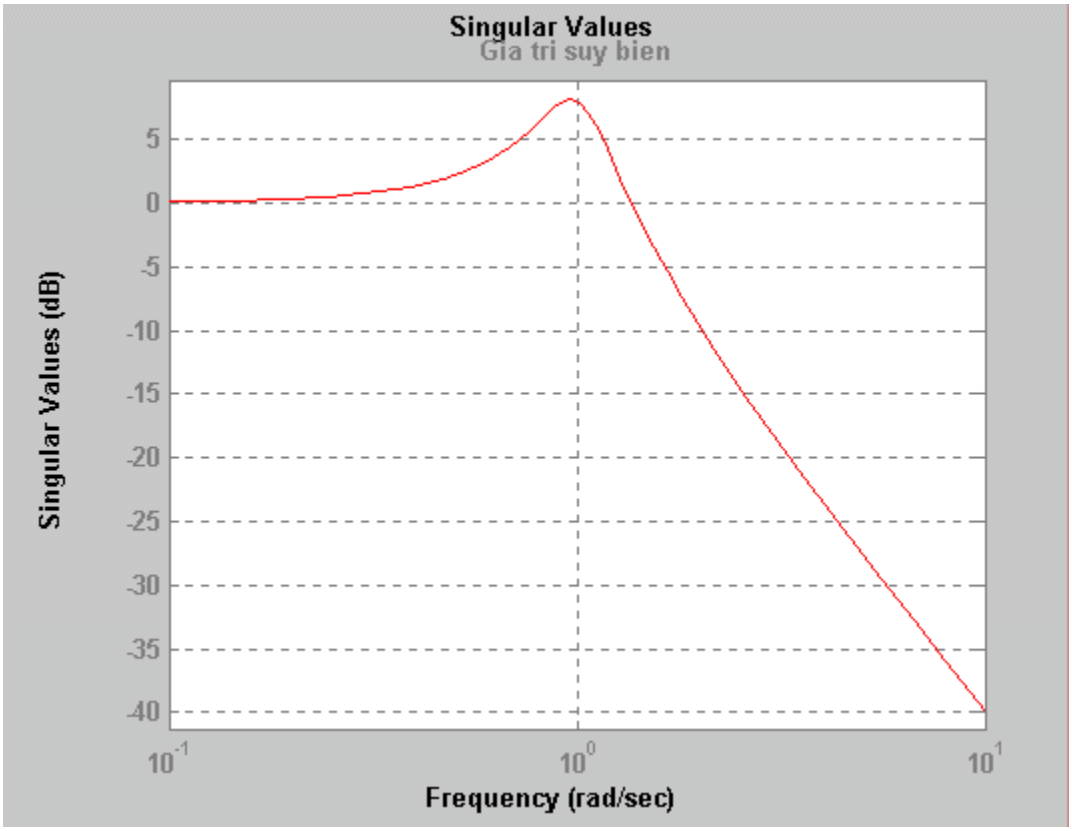
Ma trận hàm truyền	Sơ đồ khối	Lệnh
		
$G(j\omega)$		sigma(a,b,c,d)
$G^{-1}(j\omega)$		sigma(a,b,c,d,'inv')
$1+G(j\omega)$		[a,b,c,d] = parallel(a,b,c,d,[ ],[ ],[ ],eye(d)) sigma(a,b,c,d) [a,b,c,d] = feedback([ ],[ ],[ ],eye(d),a,b,c,d) sigma(a,b,c,d,'inv')
$1+G^{-1}(j\omega)$		[a,b,c,d] = feedback(a,b,c,d,[ ],[ ],[ ],eye(d)) sigma(a,b,c,d)

Đáp ứng giá trị suy biến của hệ SISO tương đương với đáp ứng biên độ giản đồ Bode của hệ đó.

d) Ví dụ:

Xét hệ bậc 2 có $\omega_n = 1$ và $\zeta = 0.2$. Vẽ đồ thị giá trị suy biến của hệ thống.

[a,b,c,d] = ord(1,0.2);
margin(a,b,c,d)
title('Gia tri suy bien')
và ta được đáp ứng như hình vẽ:



13. Lệnh DSIGMA

a) Công dụng:

Tìm giản đồ Bode giá trị suy biến của hệ không gian trạng thái.

b) Cú pháp:

[sv,w]= dsigma(a,b,c,d,Ts)
[sv,w]= dsigma(a,b,c,d,Ts,'inv')
[sv,w]= dsigma(a,b,c,d,Ts,w)
[sv,w]= dsigma(a,b,c,d,Ts,w,'inv')

c) Giải thích:

Lệnh dsigma tính các giá trị suy biến của ma trận phức $C(e^{j\omega T}I-A)^{-1}+B+D$ theo hàm của tần số ω. Các gia trị suy biến là mở rộng của đáp ứng biên độ giản đồ Bode của hệ MIMO và có thể được dùng để xác định độ rắn chắc của hệ thống.

Nếu bỏ qua các đối số ở vế trái dòng lệnh thì dsigma sẽ vẽ ra giản đồ Bode của giá trị suy biến trên màn hình.

dsigma(a,b,c,d,Ts) vẽ giản đồ suy biến của ma trận phức :

$$G(w) = C(e^{j\omega T}I-A)^{-1}+B+D$$

theo hàm của tần số. Các điểm tần số được chọn tự động trong khoảng từ 0 tới π/Ts rad/sec trong đó π/Ts rad/sec tương ứng với nửa tần số lấy mẫu (tần số Nyquist). Nếu đồ thị thay đổi nhanh thì cần chọn nhiều điểm tần số hơn.

Đối với các hệ thống có ma trận vuông, dsigma(a,b,c,d,Ts,'inv') vẽ đồ thị các giá trị suy biến của ma trận phức đảo :

$$G^{-1}(w) = [C(e^{j\omega T}I-A)^{-1}B+D]^{-1}$$

dsigma(a,b,c,d,Ts,w) hoặc dsigma(a,b,c,d,Ts,'inv') vẽ đồ thị các giá trị suy biến với vector tần số do người sử dụng xác định. Vector w chỉ ra những tần số (tính bằng rad/sec) mà tại đó đáp ứng các giá trị suy biến được tính. Hiện tượng trùng phổ xảy ra tại tần số lớn hơn tần số Nyquist (π/Ts rad/sec).

Để tạo ra vector tần số được chia đều theo logarit tần số ta dùng lệnh logspace.

Nếu giữ lại các đối số ở vế trái dòng lệnh thì :

[sv,w]= dsigma(a,b,c,d,Ts)

[sv,w]= dsigma(a,b,c,d,Ts,'inv')

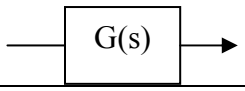
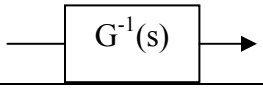
[sv,w]= dsigma(a,b,c,d,Ts,w)

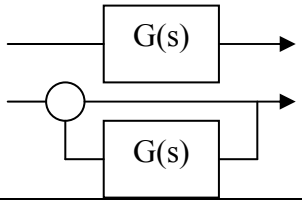
[sv,w]= dsigma(a,b,c,d,Ts,w,'inv')

không vẽ ra các đồ thị đáp ứng mà tạo ra các giá trị suy biến trong sv và các điểm tần số w. Mỗi hàng của ma trận sv chứa các giá trị suy biến theo chiều giảm dần của bậc tương ứng với các điểm tần số trong vector w.

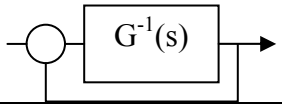
Đối với phép phân tích rắn chắc, các giá trị suy biến của ma trận hàm truyền đặc biệt được phân tích.

Việc thực hiện các lệnh để đạt được ma trận hàm truyền mong muốn của một số khối được trình bày trong bảng sau :

Ma trận hàm truyền	Sơ đồ khối	Lệnh
		
G(jω)		dsigma(a,b,c,d)
		
$G^{-1}(j\omega)$		dsigma(a,b,c,d, 'inv')



1+ G(jω)		[a,b,c,d]= parallel(a,b,c,d,[ ],[ ],[ ],eye(d)) dsigma(a,b,c,d) [a,b,c,d]=feedback([ ],[ ],[ ],eye(d),a,b,c,d) dsigma(a,b,c,d,'inv')



1+G^{-1}(jω)		[a,b,c,d]= feedback(a,b,c,d,[ ],[ ],[ ],eye(d)) dsigma(a,b,c,d)

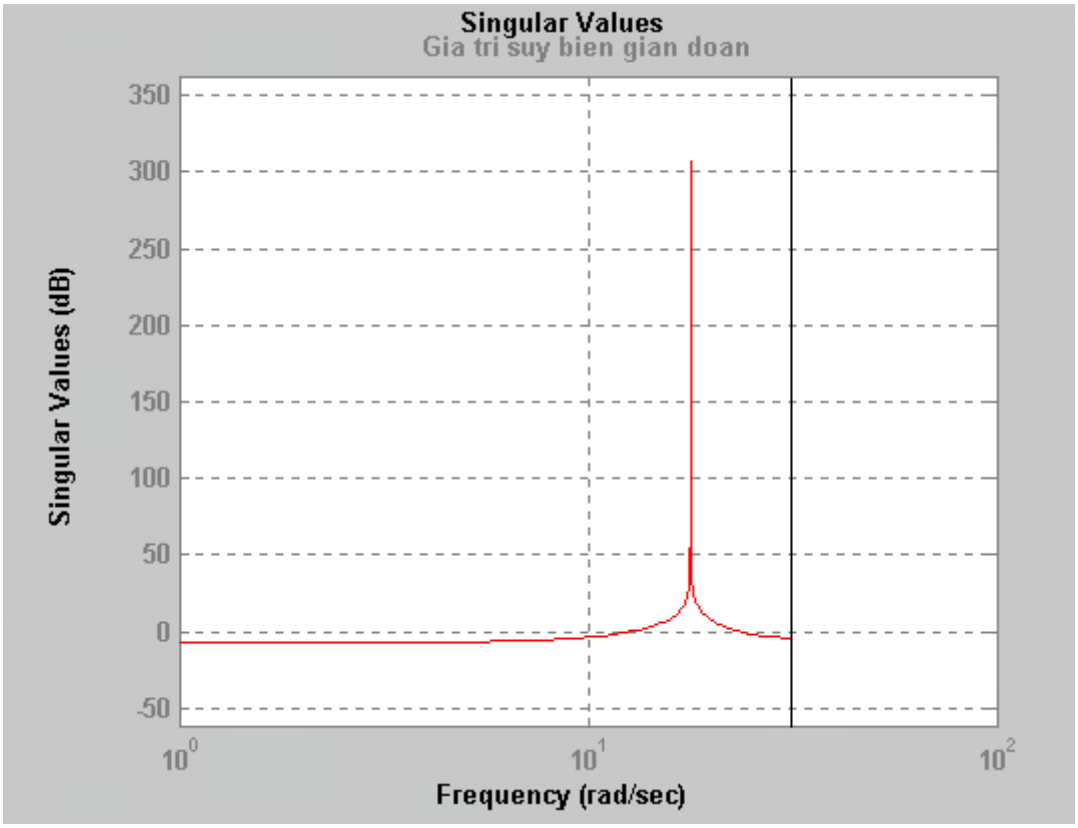
Đáp ứng giá trị suy biến của hệ SISO tương đương với đáp ứng biên độ giản đồ Bode của hệ đó.

d) Ví dụ:

Xét hệ bậc 2 có $\omega_n = 1$ và $\zeta = 0.2$. Vẽ đồ thị giá trị suy biến của hệ thống với thời gian lấy mẫu Ts = 0.1

```
[a,b,c,d]= ord2(1,0.2);
bode(a,b,c,d)
dsigma(a,b,c,d,0.1)
title('Gia tri suy bien gian doan')
```

và ta có giản đồ Bode giá trị suy biến :



14. Lệnh LTIFR

a) Công dụng:

Đáp ứng tần số của hệ tuyến tính bất biến.

b) Cú pháp:

ltifr(a,b,s)

c) Giải thích:

Lệnh ltifr dùng để mở rộng đáp ứng tần số của hệ không gian trạng thái tuyến tính bất biến.

G = Ltifr(a,b,s) tìm đáp ứng tần số của hệ thống với một ngõ vào duy nhất :

$$G(s) = (sI - A)^{-1}B$$

Vector s chỉ ra số phức mà tại đó đáp ứng tần số được xác định. Đối với đáp ứng giản đồ Bode hệ liên tục, s nằm trên trục ảo. Đối với đáp ứng giản đồ Bode hệ gián đoạn, s nhận các giá trị quanh vòng tròn đơn vị.

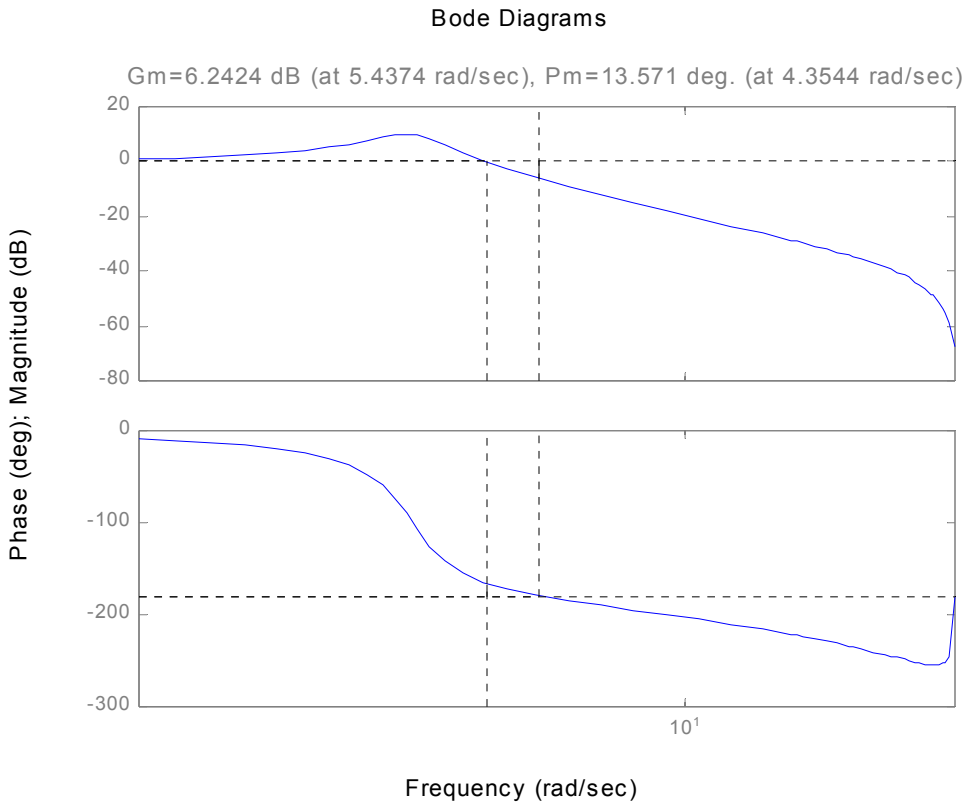
ltifr tạo ra đáp ứng tần số dưới dạng ma trận phức G với số cột bằng số trạng thái hay số hàng của ma trận A và có số hàng là length(s).

CÁC BÀI TẬP VỀ ĐÁP ỨNG TẦN SỐ

Bài 1: hàm **margin** (bài tập này trích từ trang 11-138 sách **'Control System Toollbox'**

```
» hd=tf([0.04798 0.0464],[1 -1.81 0.9048],0.1)

Transfer function:
 0.04798 z + 0.0464
--------------------
z^2 - 1.81 z + 0.9048

Sampling time: 0.1  ; Thời gian lấy mẫu: 0,1
» [Gm,Pm,Wcg,Wcp]=margin(hd);
» [Gm,Pm,Wcg,Wcp]

ans =

   2.0517   13.5712    5.4374    4.3544

» margin(hd)
```

Kết quả:



Bài 2: lệnh **modred** (bài tập này trích từ trang 11-142 sách **'Control System Toollbox'**

$$h(s) = \frac{s3 + 11s2 + 36s + 26}{s4 + 14{,}6s3 + 74{,}96s2 + 153{,}7s + 99{,}65}$$

» h=tf([1 11 36 26],[1 14.6 74.96 153.7 99.65])

Transfer function:

```
        s^3 + 11 s^2 + 36 s + 26
-------------------------------------------
s^4 + 14.6 s^3 + 74.96 s^2 + 153.7 s + 99.65
```

» [hb,g]=balreal(h)

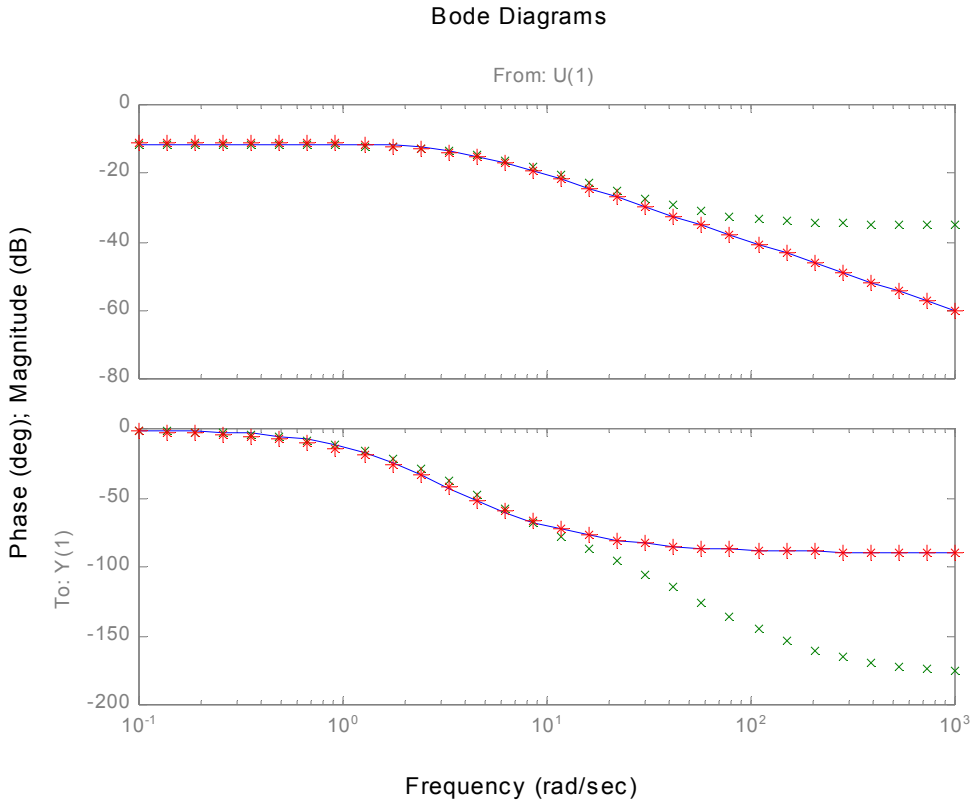
a =

	x1	x2	x3	x4
x1	-3.6014	-0.82121	-0.61634	-0.058315

```
        x2     0.82121    -0.59297     -1.0273    -0.090334
        x3    -0.61634      1.0273     -5.9138     -1.1272
        x4    0.058315   -0.090334      1.1272     -4.4918


b =
                u1
        x1      1.002
        x2   -0.10641
        x3   0.086124
        x4  -0.0081117


c =
                x1          x2          x3          x4
        y1      1.002     0.10641    0.086124   0.0081117


d =
                u1
        y1       0

Continuous-time model.

g =

   0.1394
   0.0095
   0.0006
   0.0000

» g'

ans =

   0.1394    0.0095    0.0006    0.0000
» hmdc=modred(hb,2:4,'mdc')

a =
                x1
        x1    -4.6552
```

```
b =
             u1
    x1      1.1392

c =
             x1
    y1      1.1392

d =
             u1
    y1   -0.017857

Continuous-time model.
» hdel=modred(hb,2:4,'del')

a =
             x1
    x1     -3.6014

b =
             u1
    x1      1.002

c =
             x1
    y1      1.002

d =
             u1
    y1         0

                    Continuous-time model.

» bode(h,'-',hmdc,'x',hdel,'*')
```

Kết quả:

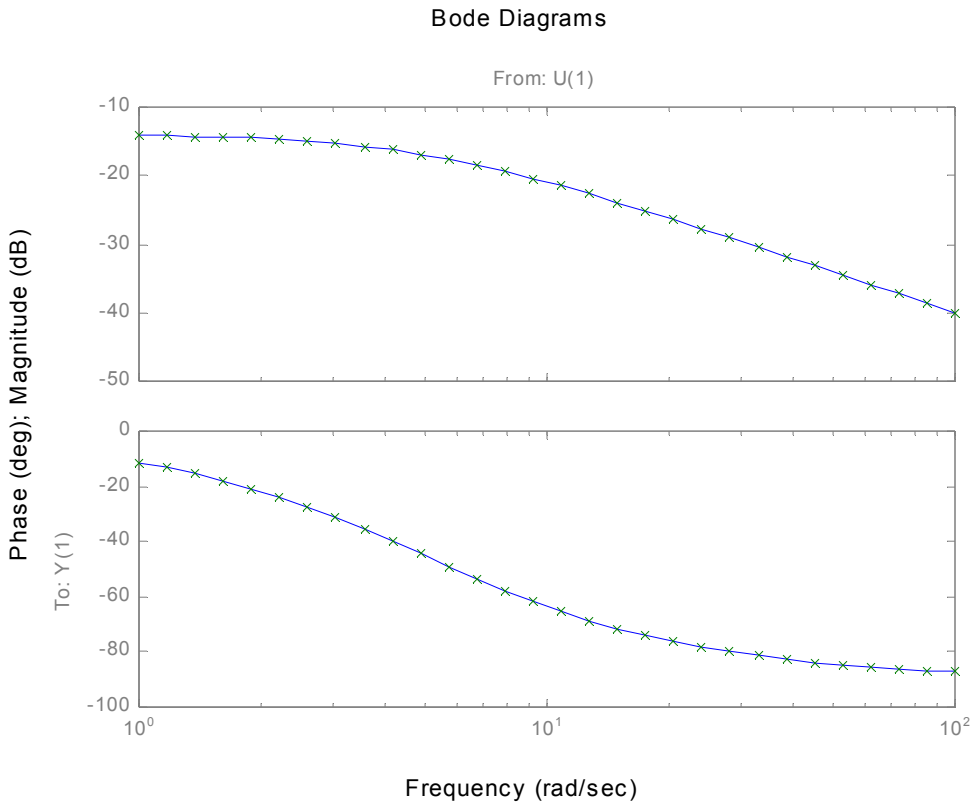


Bài 3: (Trang 11-16 sách **'Control System Toollbox')**
Xem zero-pole-gain (zero-cực-độ lợi) của hệ thống sau:

```
» sys=zpk([-10 -20.01],[-5 -9.9 -20.1],1)

Zero/pole/gain:
  (s+10) (s+20.01)
----------------------
(s+5) (s+9.9) (s+20.1)

»
» [sys,g]=balreal(sys)

a =
             x1         x2         x3
   x1    -4.9697     0.2399   -0.22617
   x2    -0.2399    -4.2756     9.4671
   x3   -0.22617    -9.4671    -25.755


b =
```

```
              u1
        x1     1
        x2     0.024121
        x3     0.022758


c =
              x1          x2          x3
        y1     1    -0.024121     0.022758


d =
              u1
        y1     0

Continuous-time model.

g =

   0.1006
   0.0001
   0.0000

» g'

ans =

   0.1006    0.0001    0.0000
» sysr=modred(sys,[2 3],'del')

a =
              x1
        x1     -4.9697


b =
              u1
        x1     1


c =
```

```
              x1
y1             1


d =
              u1
y1             0

Continuous-time model.
» zpk(sysr)

Zero/pole/gain:
 1.0001
--------
(s+4.97)
» bode(sys,'-',sysr,'x')
```

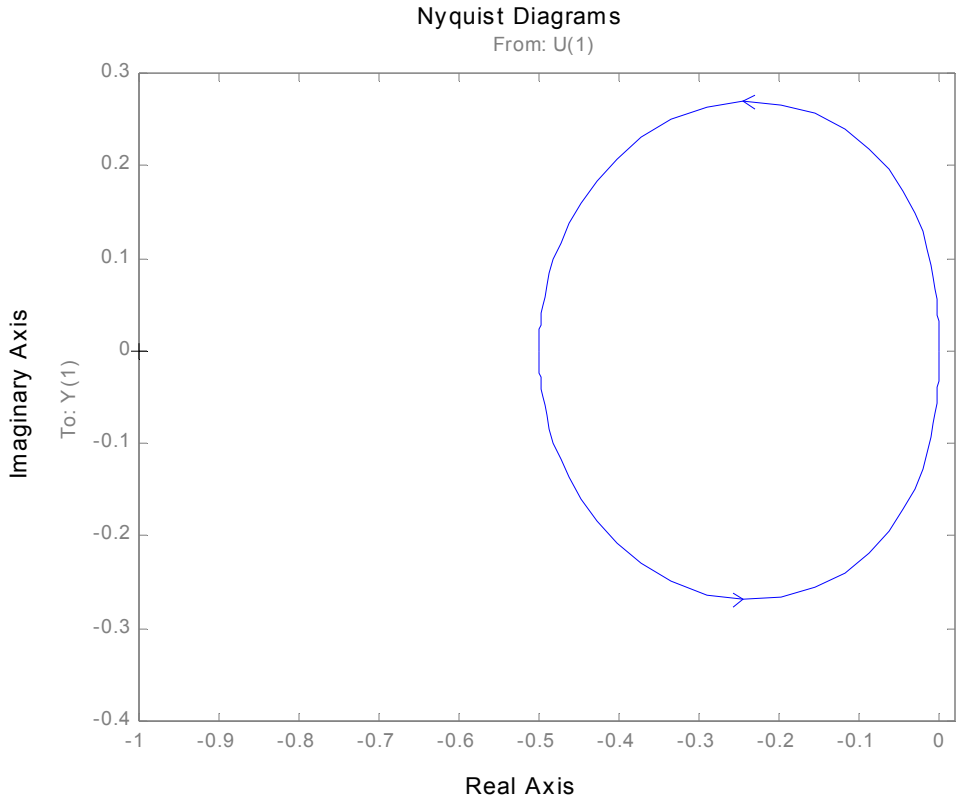


Bài 4: Trích từ trang 55 sách **'Hướng dẫn sử dụng MATLAB'** tác giả Nguyễn Văn Giáp.

Vẽ biểu đồ nyquist của hệ thống:

H(s) = (s+4)/(s^2 + 3s – 8)

```
» num=[1 4];
» den=[1 3 -8];
» nyquist(num,den);
```

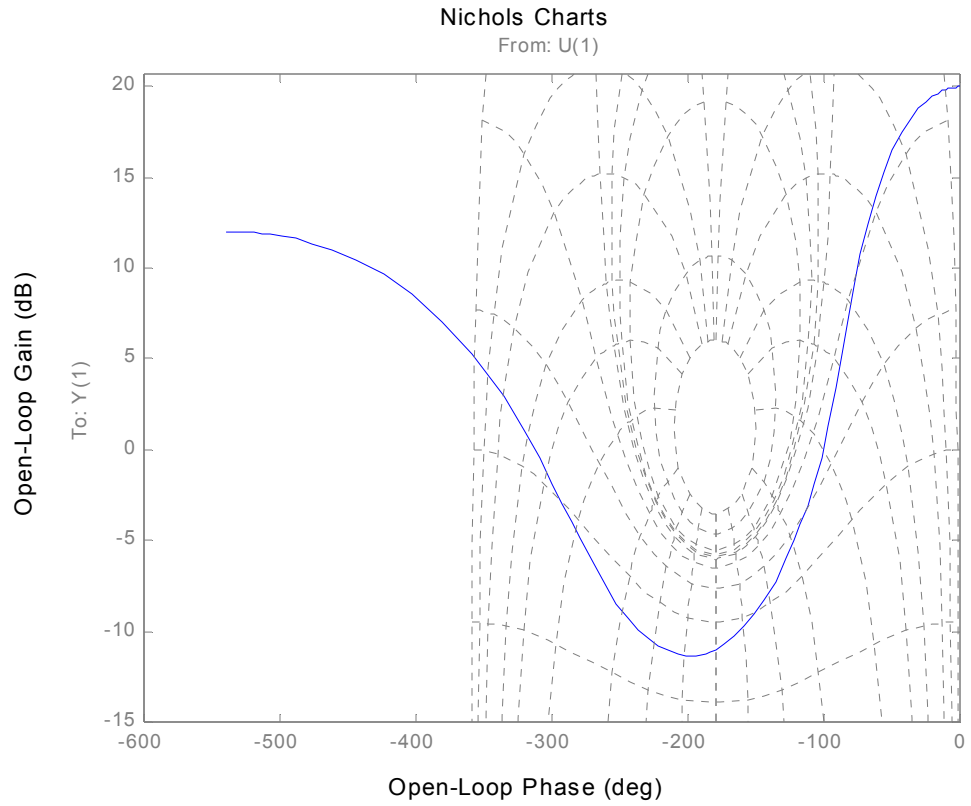


Bài 5: Trích trang 11-147 sách **'Control System Toolbox'**

Vẽ đáp ứng Nichols của hệ thống có hàm truyền:

$$H(s) = \frac{-4s^4 + 48s^3 - 18s^2 + 250s + 600}{s^4 + 30s^3 + 282s^2 + 525s + 60}$$

```
» H=tf([-4 48 -18 250 600],[1 30 282 525 60])

Transfer function:
-4 s^4 + 48 s^3 - 18 s^2 + 250 s + 600
-------------------------------------
 s^4 + 30 s^3 + 282 s^2 + 525 s + 60
Nichols(H)
ngrid
```



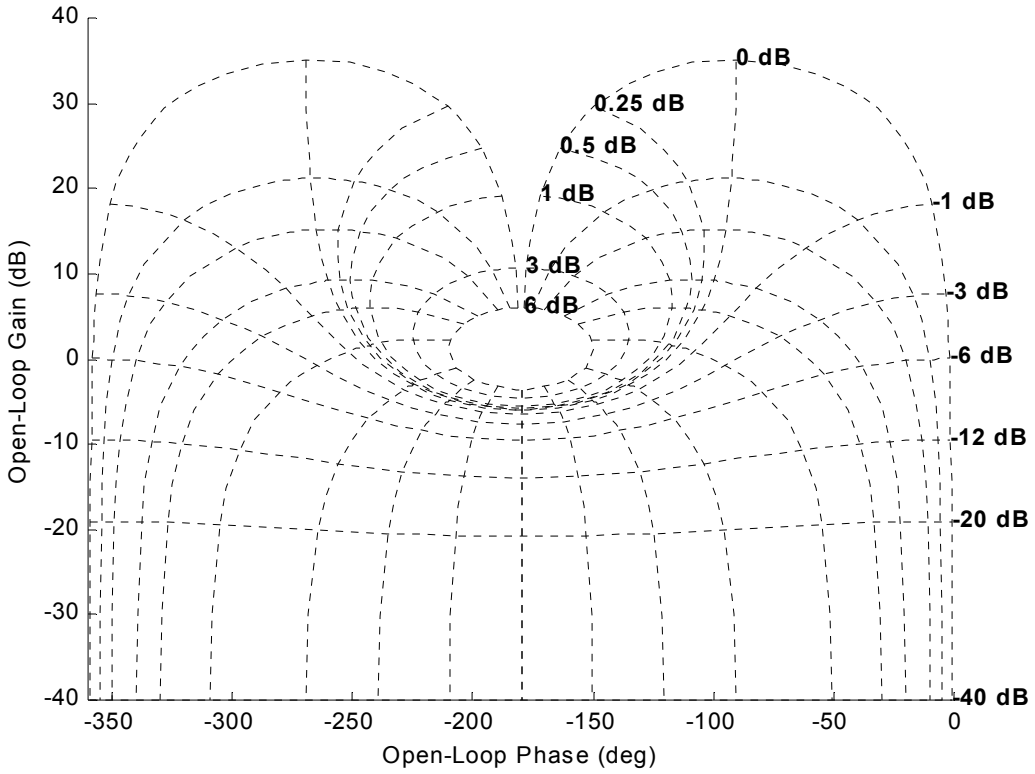
Bài 6: Trang 131 sách **'Ứng dụng MATLAB trong điều khiển tự động'** tác giả Nguyễn Văn Giáp.

Trên giản đồ Nichols vẽ đường cong logarit biên độ – pha của hàm truyền hệ thống

$$H(s) = \frac{k}{S^3+52s^2+100s}$$

```
» k=438;
» num=k;
» den=[1 52 100 0];
» w=.1:.1:10;
» [mag,phase]=bode(num,den,w);
» ngrid,
```

Kết quả:


0 dB
0.25 dB
0.5 dB
1 dB
3 dB
6 dB
-1 dB
-3 dB
-6 dB
-12 dB
-20 dB
-40 dB
Open-Loop Gain (dB)
Open-Loop Phase (deg)
40
30
20
10
0
-10
-20
-30
-40
-350
-300
-250
-200
-150
-100
-50
0

NHÓM LỆNH VỀ ĐÁP ỨNG THỜI GIAN (Time Response)

1. Lệnh IMPULSE

a) Công dụng:

Tìm đáp ứng xung đơn vị.

b) Cú pháp:

[y,x,t] = impulse(a,b,c,d)
[y,x,t] = impulse(a,b,c,d,iu)
[y,x,t] = impulse(a,b,c,d,iu,t)
[y,x,t] = impulse(num,den)
[y,x,t] = impulse(num,den,t)

c) Giải thích:

Lệnh impulse tìm đáp ứng xung đơn vị của hệ tuyến tính. Nếu bỏ qua các đối số bên trái thì lệnh impulse sẽ vẽ ra đáp ứng xung trên màn hình.

impulse(a,b,c,d) tạo ra chuỗi đồ thị đáp ứng xung, mỗi đồ thị ứng với một mối quan hệ vào ra của hệ liên tục LTI:

$$\dot{x} = Ax + Bu$$
$$y = Cx + Du$$

với vector thời gian được xác định tự động.

impulse(a,b,c,d,iu) tạo ra đáp ứng xung từ ngõ vào duy nhất iu tới toàn bộ các ngõ ra của hệ thống với vector thời gian được xác định tự động. iu là chỉ số ngõ vào của hệ thống và chỉ ra ngõ vào nào được dùng cho đáp ứng xung.

impulse(num,den) tạo ra đồ thị đáp ứng xung của đa thức hàm truyền:

$$G(s) = num(s)/den(s)$$

trong đó num và den chứa các hệ số đa thức theo chiều giảm dần số mũ của s.

impulse(a,b,c,d,iu,t) hay impulse(num,den,t) dùng vector thời gian t do người sử dụng quy định. Vector t chỉ định những thời điểm mà đáp ứng xung được tính và vector t phải được chỉ chia thành các khoảng đều nhau.

Nếu giữ các đối số bên trái:

[y,x,t] = impulse(a,b,c,d)
[y,x,t] = impulse(a,b,c,d,iu)
[y,x,t] = impulse(a,b,c,d,iu,t)
[y,x,t] = impulse(num,den)
[y,x,t] = impulse(num,den,t)

không vẽ ra các đồ thị mà tạo ra các ma trận đáp ứng trạng thái và đáp ứng ngõ ra của hệ thống và vector thời gian t. Ma trận y và x chứa các đáp ứng trạng thái và đáp ứng ngõ ra của hệ thống được xác định tại những thời điểm t. Ma trận y có số

cột là số ngõ ra và mỗi hàng ứng với một thành phần trong vector t. Ma trận x có số cột là số trạng thái và mỗi hàng ứng với một thành phần trong vector t.

d) Ví dụ: (Trích từ trang 11-95 sách **'control System Toolbox'**)

Vẽ đáp ứng xung của hệ không gian trạng thái bậc 2 sau:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -0.5 & -0.8 \\ 0.8 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u$$

$$y = \begin{bmatrix} 1.9 & 6.5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + [0]\, u$$

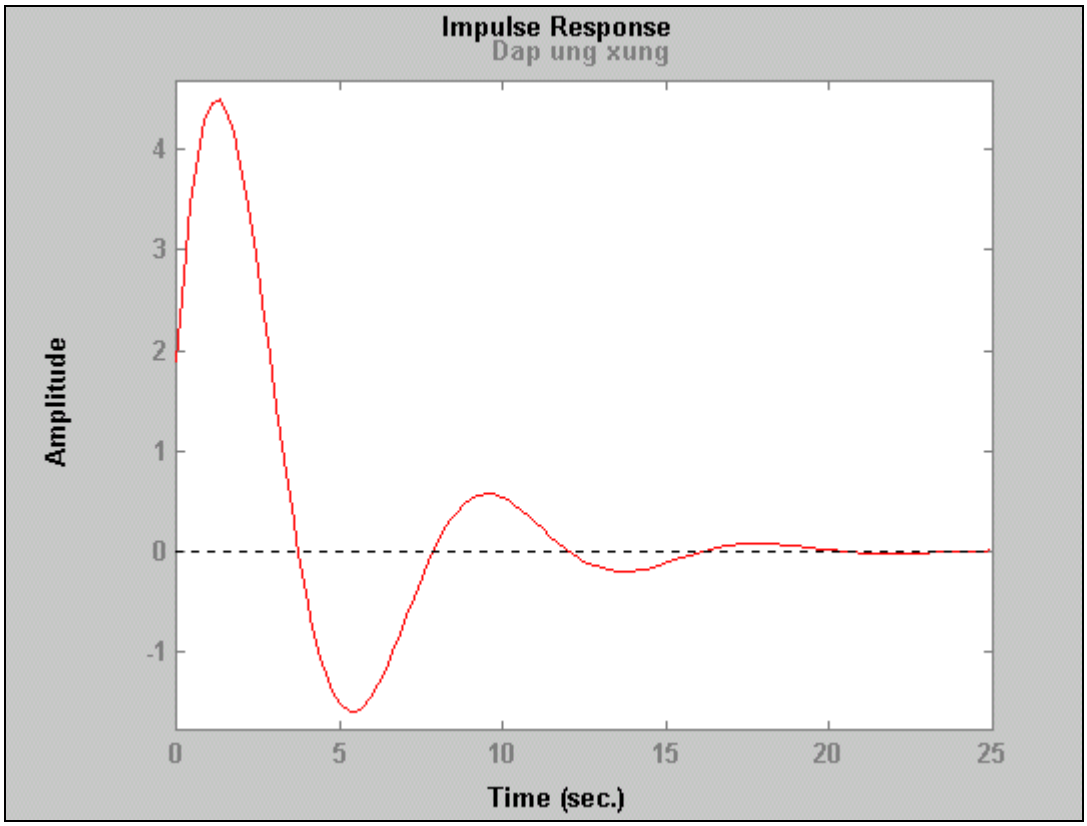
% Khai báo hệ thống:

```
a = [-0.5   -0.8 ; 0.8    0];
b = [1 ; 0];
c = [1.9          6.5];
d = [0];
```

% Vẽ đáp ứng xung:

impulse(a,b,c,d); title('Dap ung xung') (đặt tiêu đề cho đồ thị)

và cuối cùng ta nhận được đồ thị đáp ứng xung như sau:



2. Lệnh DIMPULSE

a) Công dụng:

Tìm đáp ứng xung đơn vị của hệ gián đoạn.

b) Cú pháp:

[y,x] = dimpulse(a,b,c,d)

[y,x] = dimpulse(a,b,c,d,iu)

[y,x] = dimpulse(a,b,c,d,iu,n)

[y,x] = dimpulse(num,den)

[y,x] = dimpulse(num,den,n)

c) Giải thích:

Lệnh dimpulse tìm đáp ứng xung đơn vị của hệ tuyến tính gián đoạn. Nếu bỏ qua các đối số bên trái thì thì lệnh dimpulse sẽ vẽ ra đáp ứng xung trên màn hình.

dimpulse(a,b,c,d) tạo ra chuỗi đồ thị đáp ứng xung, mỗi đồ thị đáp ứng với một mối quan hệ vào ra của hệ gián đoạn LTI:

$$x[n+1] = Ax[n] + Bu[n]$$
$$y[n] = Cx[n] + Du[n]$$

với số điểm lấy mẫu được xác định tự động.

dimpulse(a,b,c,d,iu) tạo ra đồ thị đáp ứng xung từ ngõ vào duy nhất iu tới toàn bộ các ngõ ra của hệ thống với số điểm lấy mẫu được xác định tự động. iu là chỉ số ngõ vào của hệ thống và chỉ ra ngõ vào nào được dùng cho đáp ứng xung.

dimpulse(num,den) tạo ra đồ thị đáp ứng xung của đa thức hàm truyền:

$$G(z) = num(z)/den(z)$$

trong đó num và den chứa các hệ số đa thức theo chiều giảm dần số mũ của z.

dimpulse(num,den,n) hay dimpulse(a,b,c,d,iu,n) dùng số điểm lấy mẫu n do người sử dụng chỉ định.

Nếu giữ các đối số bên trái:

[y,x] = dimpulse(a,b,c,d)

[y,x] = dimpulse(a,b,c,d,iu)

[y,x] = dimpulse(a,b,c,d,iu,n)

[y,x] = dimpulse(num,den)

[y,x] = dimpulse(num,den,n)

không vẽ ra các đồ thị mà tạo ra các ma trận đáp ứng ngõ ra và đáp ứng trạng thái của hệ thống. Ma trận y và x chứa các đáp ứng trạng thái và ngõ ra của hệ thống được xác định tại những điểm lấy mẫu. Ma trận y có số cột là số ngõ ra. Ma trận x có số cột là số trạng thái.

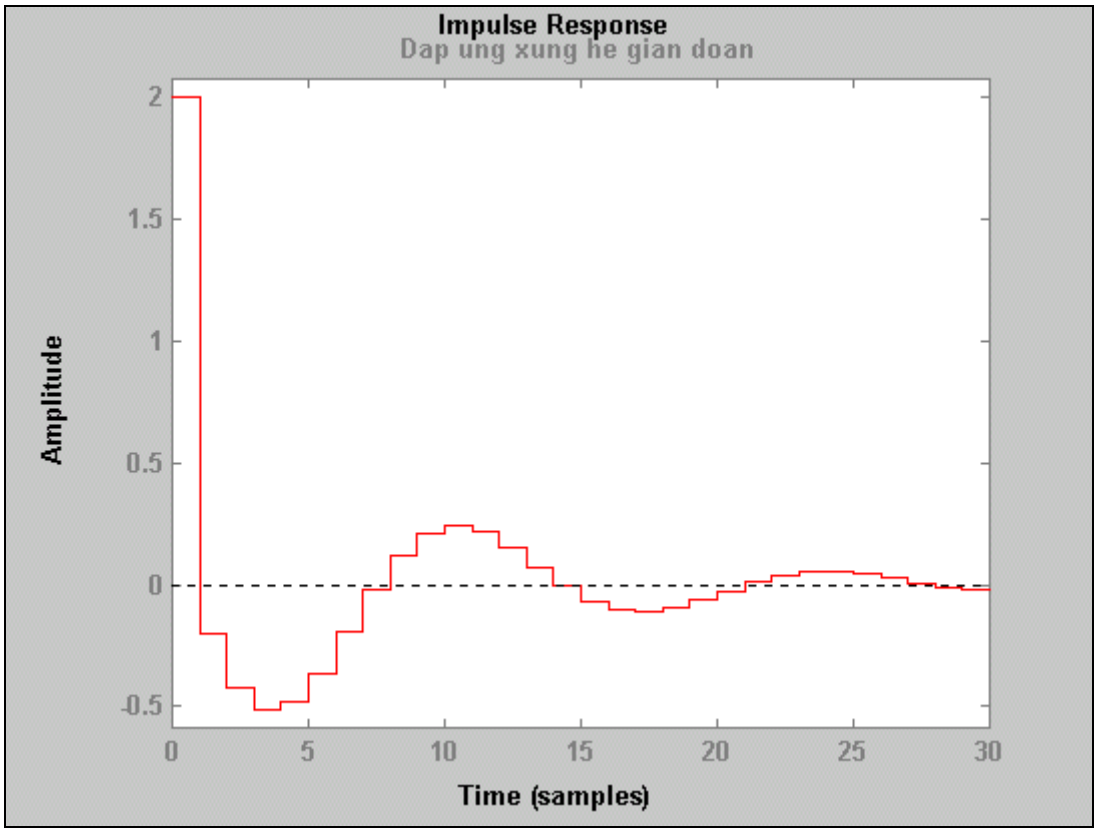
d) Ví dụ:

Vẽ đáp ứng xung của hệ gián đoạn có hàm truyền sau:

$$H(z) = \frac{2z^2 + 3.4z + 1.5}{z^2 - 1.6 + 0.8}$$

```
num = [2   -3.4   1.5];
den = [1   -1.6   0.8];
dimpulse(num,den); title('Dap ung xung he gian doan')
```

và cuối cùng ta được đồ thị đáp ứng xung hệ gián đoạn như sau:



3. Lệnh INITIAL

a) Công dụng:

Tìm đáp ứng điều kiện ban đầu.

b) Cú pháp:

[y,x,t] = initial(a,b,c,d,x0)

[y,x,t] = initial(a,b,c,d,x0,t)

c) Giải thích:

Lệnh initial dùng để tìm đáp ứng của hệ tuyến tính liên tục ứng với điều kiện ban đầu của các trạng thái. Nếu bỏ qua các đối số ở bên trái thì lệnh initial sẽ vẽ ra đáp ứng điều kiện ban đầu trên màn hình.

initial(a,b,c,d,x0) vẽ ra đồ thị đáp ứng điều kiện ban đầu của tất cả các ngõ ra của hệ liên tục LTI:

$$\dot{x} = Ax + Bu$$
$$y = Cx + Du$$

với vector thời gian được xác định tự động. x0 là vector trạng thái ban đầu.

initial(a,b,c,d,x0,t) vẽ ra đồ thị đáp ứng ban đầu với vector thời gian t do người sử dụng xác định. Vector t sẽ chỉ ra những thời điểm mà tại đó đáp ứng điều kiện ban đầu được tính.

Nếu sử dụng các đối số ở vế trái của dòng lệnh thì:

[y,x,t] = initial(a,b,c,d,x0)

[y,x,t] = initial(a,b,c,d,x0,t)

sẽ không vẽ ra các đồ thị đáp ứng mà tạo ra các ma trận đáp ứng trạng thái x, đáp ứng ngõ ra y và vector thời gian t của hệ thống đối với điều kiện ban đầu x0. Ma trận y và x chứa các đáp ứng ngõ ra và đáp ứng trạng thái của hệ thống được tính tại thời điểm t. Ma trận y có số cột bằng số ngõ ra và mỗi hàng ứng với một thành phần trong vector t. Ma trận x có số cột bằng số trạng thái và mỗi cột ứng với một thành phần trong vector t.

d) Ví dụ:

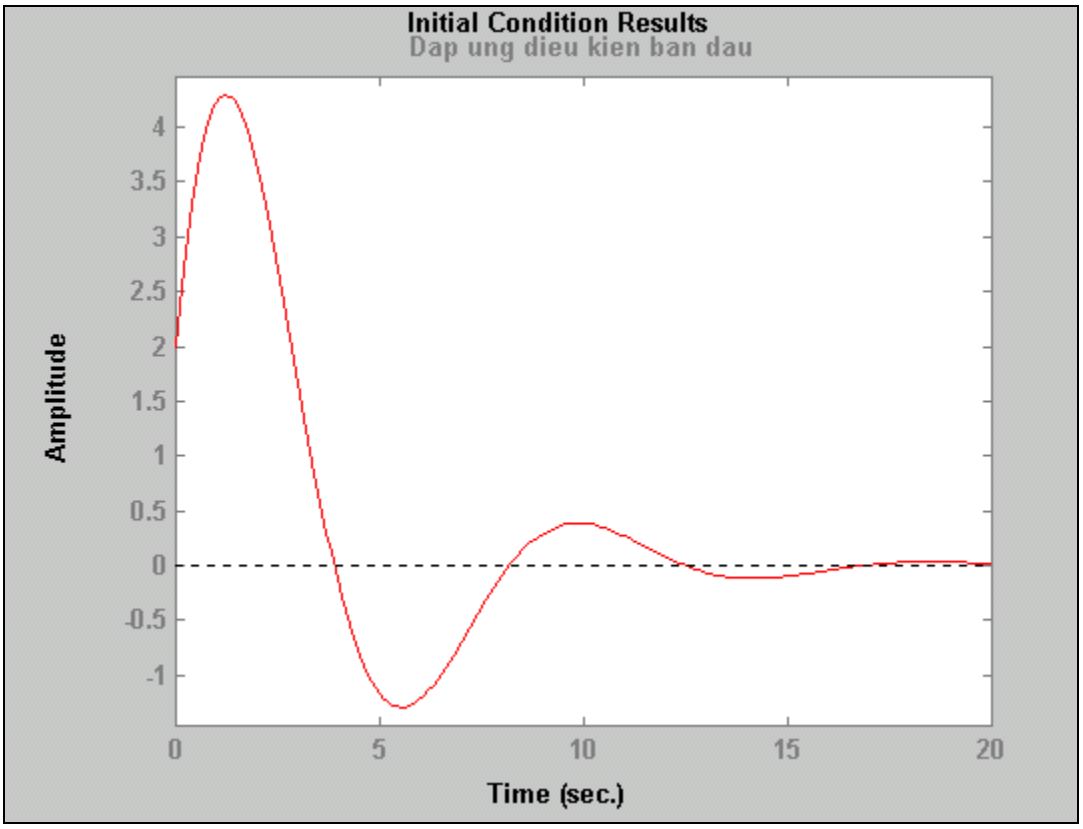
Vẽ đáp ứng điều kiện ban đầu của hệ không gian trạng thái bậc 2 sau:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -0.5572 & -0.7814 \\ 0.7814 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u$$

$$y = \begin{bmatrix} 1.9691 & 6.4493 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \end{bmatrix} u$$

với điều kiện ban đầu x0 = [1 0]

```
% Khai báo hệ thống, điều kiện ban đầu và trục thời gian:
    a = [-0.5572   -0.7814 ; 0.7814   0];
    b = [1 ; 0];
    c = [1.9691   6.4493];
    d = [0];
    x0 = [1 0];
    t = 0:0.1:20;
% Vẽ đáp ứng:
    initial(a,b,c,d,x0,t)
    title('Dap ung dieu kien ban dau')
```



4. Lệnh DINITIAL

a) Công dụng:

Tìm đáp ứng điều kiện ban đầu của hệ gián đoạn.

b) Cú pháp:

[y,x] = dinitial(a,b,c,d,x0)

[y,x] = dinitial(a,b,c,d,x0,n)

c) Giải thích:

Lệnh dinitial dùng để tìm đáp ứng của hệ tuyến tính gián đoạn ứng với điều kiện ban đầu của các trạng thái. Nếu bỏ qua các đối số ở bên trái thì lệnh dinitial sẽ vẽ ra đáp ứng điều kiện ban đầu trên màn hình.

dinitial(a,b,c,d,x0) vẽ ra đồ thị đáp ứng điều kiện ban đầu của tất cả các ngõ ra của hệ gián đoạn LTI:

$$x[n+1] = Ax[n] + Bu[n]$$
$$y[n] = Cx[n] + Du[n]$$

với số điểm lấy mẫu được xác định tự động. x0 là vector trạng thái ban đầu.

dinitial(a,b,c,d,x0,n) vẽ ra đồ thị đáp ứng ban đầu với số điểm lầy mẫu n do người sử dụng xác định.

Nếu sử dụng các đối số ở vế trái của dòng lệnh thì:

[y,x] = dinitial(a,b,c,d,x0)

[y,x] = dinitial(a,b,c,d,x0,n)

sẽ không vẽ ra các đồ thị đáp ứng mà tạo ra các ma trận đáp ứng trạng thái x, đáp ứng ngõ ra y của hệ thống đối với điều kiện ban đầu x0. Ma trận y có số cột bằng số ngõ ra và ma trận x có số cột bằng số trạng thái.

d) Ví dụ:

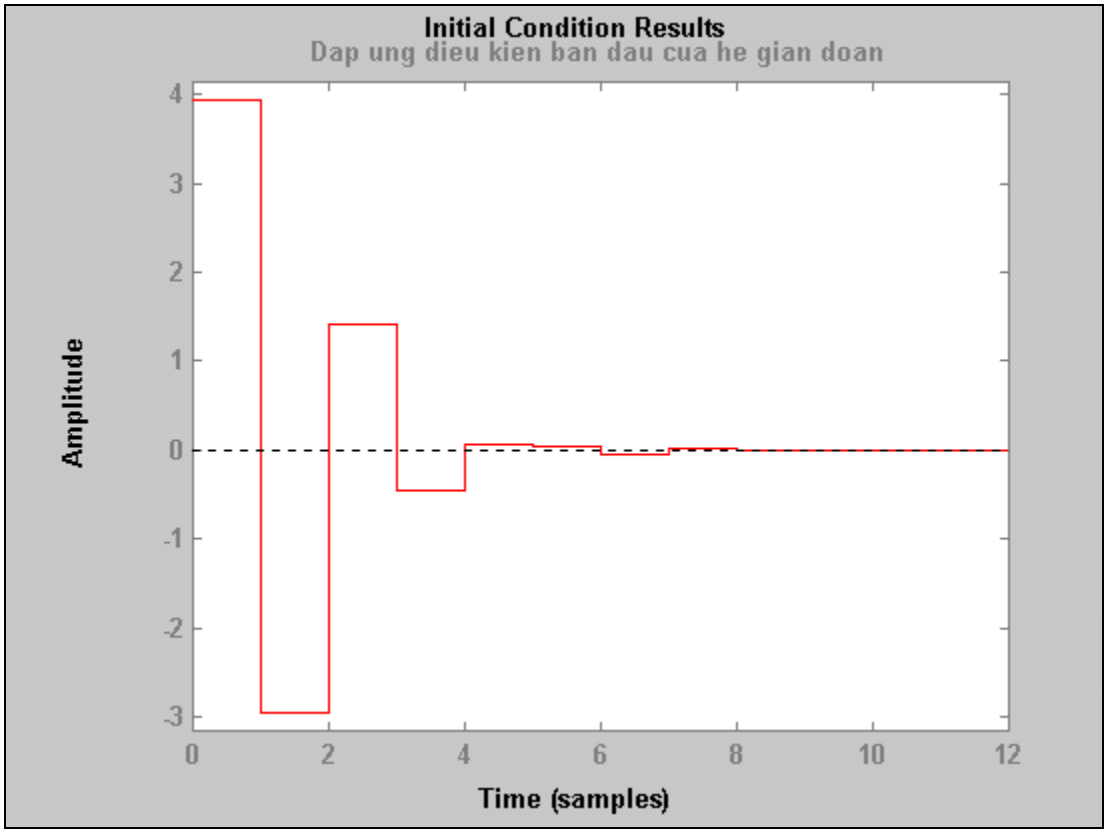
Vẽ đáp ứng điều kiện ban đầu của hệ không gian trạng thái bậc 2:

$$\begin{bmatrix} x_1[n+1] \\ x_2[n+1] \end{bmatrix} = \begin{bmatrix} -0.7497 & -0.2027 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x_1[n] \\ x_2[n] \end{bmatrix} + \begin{bmatrix} -4.1841 \\ -6.5049 \end{bmatrix} u$$

$$y = \begin{bmatrix} 3.9321 & 0 \end{bmatrix} \begin{bmatrix} x_1[n] \\ x_2[n] \end{bmatrix}$$

với điều kiện ban đầu x0 = [1 0]

```
a = [-0.7497   -0.2027 ; 1   0];
b = [-4.1841 ; -6.5049];
c = [3.9321   0];
d = [0];
dinitial(a,b,c,d,[1   0]);
title('Dap ung dieu kien ban dau cua he gian doan')
```



5. Lệnh LSIM

a) Công dụng:

Mô phỏng hệ thống liên tục với các ngõ vào tùy ý.

b) Cú pháp:

[y,c] = lsim(a,b,c,d,u,t)

[y,c] = lsim(a,b,c,d,u,t,x0)

[y,c] = lsim(num,den,u,t)

c) Giải thích:

Lệnh lsim dùng để mô phỏng hệ tuyến tính liên tục với các ngõ vào tùy ý.

Nếu bỏ qua các đối số ở vế trái của dòng lệnh thì lệnh lsim vẽ ra ra đồ thị trên màn hình.

Cho hệ không gian trạng thái LTI:

$$\dot{x} = Ax + Bu$$
$$y = Cx + Du$$

lsim(a,b,c,d,u,t) vẽ ra đồ thị đáp ứng thời gian của hệ thống với ngõ vào thời gian ban đầu nằm trong ma trận u. Ma trận u phải có số cột bằng số ngõ vào u. Mỗi hàng của ma trận u tương ứng với một thời gian mới và ma trận u phải có số hàng là length(t). Vector t chỉ ra trục thời gian cho quá trình mô phỏng và phải chia thành các đoạn bằng nhau. Nếu dùng thêm đối số x0 ở vế phải thì lệnh lsim(a,b,c,d,u,t,x0) sẽ chỉ ra điều kiện ban đầu của các trạng thái.

lsim(num,den,u,t) vẽ ra đáp ứng thời gian của hàm truyền đa thức:

$$G(s) = num(s)/den(s)$$

trong đó num và den chứa các hệ số đa thức theo chiều giảm dần số mũ của s.

Nếu giữ lại các đối số ở vế trái thì:

[y,c] = lsim(a,b,c,d,u,t)

[y,c] = lsim(a,b,c,d,u,t,x0)

[y,c] = lsim(num,den,u,t)

sẽ không vẽ ra các đồ thị đáp ứng mà tạo ra các ma trận y và x, trong đó ma trận y là đáp ứng ngõ ra và ma trận x là đáp ứng trạng thái của hệ thống. Ma trận y có số cột bằng số ngõ ra y và mỗi hàng ứng với một hàng của ma trận u. Ma trận x có số cột bằng số trạng thái x và mỗi hàng ứng với một hàng của ma trận u.

d) Ví dụ: (Trích từ trang 11-127 sách **'Control System Toolbox'**)

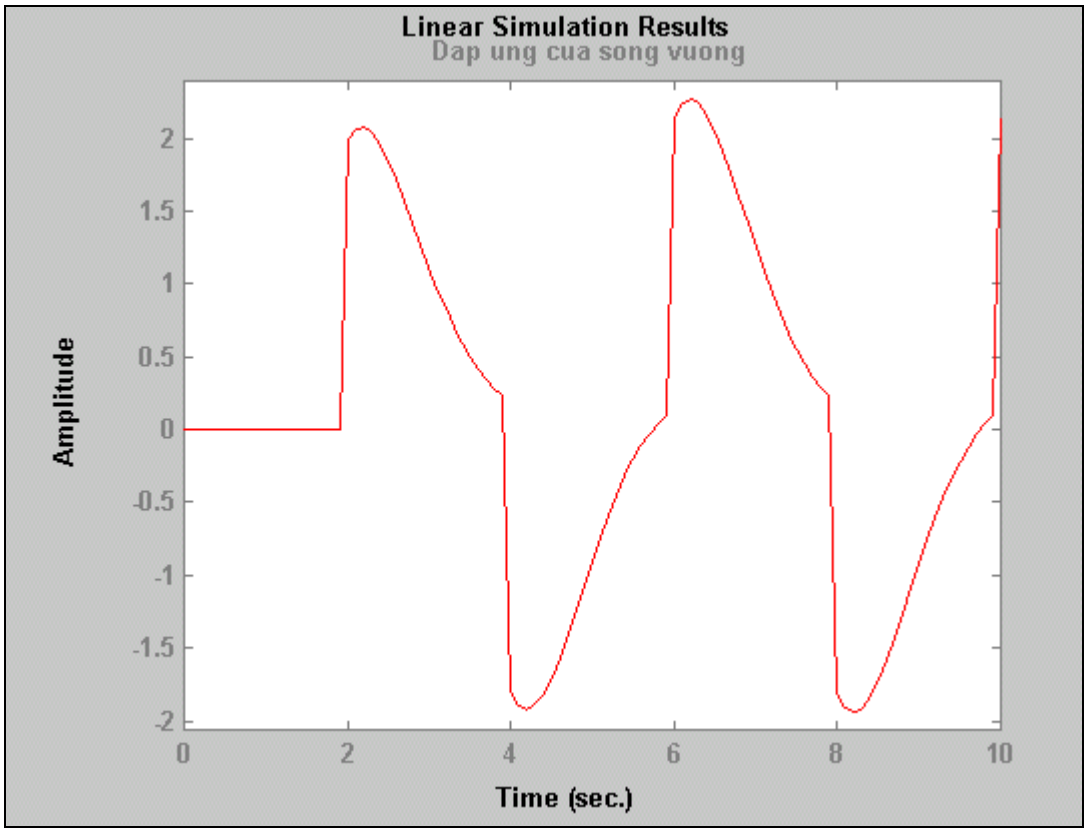
Mô phỏng và vẽ đồ thị đáp ứng của hệ thống có hàm truyền:

$$H(s) = \frac{2s^2 + 5s + 1}{s^2 + 2s + 3}$$

với chu kỳ sóng vuông là 10s.

```
num = [2    5    1];
den = [1    2    3];
t = 0:.1:10;
period = 4;
```

u = (rem(t,period)) >= period./2);
lsim(num,den,u,t); title('Dap ung cua song vuong')
và ta được đồ thị đáp ứng của hệ như sau:



6. Lệnh DLSIM

a) Công dụng:

Mô phỏng hệ thống gián đoạn với các ngõ vào tùy ý.

b) Cú pháp:

[y,c] = dlsim(a,b,c,d,u,t)
[y,c] = dlsim(a,b,c,d,u,x0)
[y,c] = dlsim(num,den,u)

c) Giải thích:

Lệnh lsim dùng để mô phỏng hệ tuyến tính gián đoạn với các ngõ vào tùy ý.

Nếu bỏ qua các đối số ở vế trái của dòng lệnh thì lệnh dlsim vẽ ra ra đồ thị trên màn hình.

Cho hệ không gian trạng thái LTI:

$$x[n+1] = Ax[n] + Bu[n]$$
$$y[n] = Cx[n] + Du[n]$$

dlsim(a,b,c,d,u) vẽ ra đồ thị đáp ứng thời gian của hệ thống với ngõ vào thời gian ban đầu nằm trong ma trận u. Ma trận u phải có số cột bằng số ngõ vào u. Mỗi

hàng của ma trận u tương ứng với một thời điểm mới. Nếu dùng thêm đối số x0 ở vế phải thì lệnh lsim(a,b,c,d,u,x0) sẽ chỉ ra điều kiện ban đầu của các trạng thái.

lsim(num,den,u) vẽ ra đáp ứng thời gian của hàm truyền đa thức:

$$G(z) = num(z)/den(z)$$

trong đó num và den chứa các hệ số đa thức theo chiều giảm dần số mũ của s.

Nếu giữ lại các đối số ở vế trái thì:

[y,c] = dlsim(a,b,c,d,u)

[y,c] = dlsim(a,b,c,d,u,x0)

[y,c] = dlsim(num,den,u)

sẽ không vẽ ra các đồ thị đáp ứng mà tạo ra các ma trận y và x, trong đó ma trận y là đáp ứng ngõ ra và ma trận x là đáp ứng trạng thái của hệ thống. Ma trận y có số cột bằng số ngõ ra y và mỗi hàng ứng với một hàng của ma trận u. Ma trận x có số cột bằng số trạng thái x và mỗi hàng ứng với một hàng của ma trận u.

d) Ví dụ:

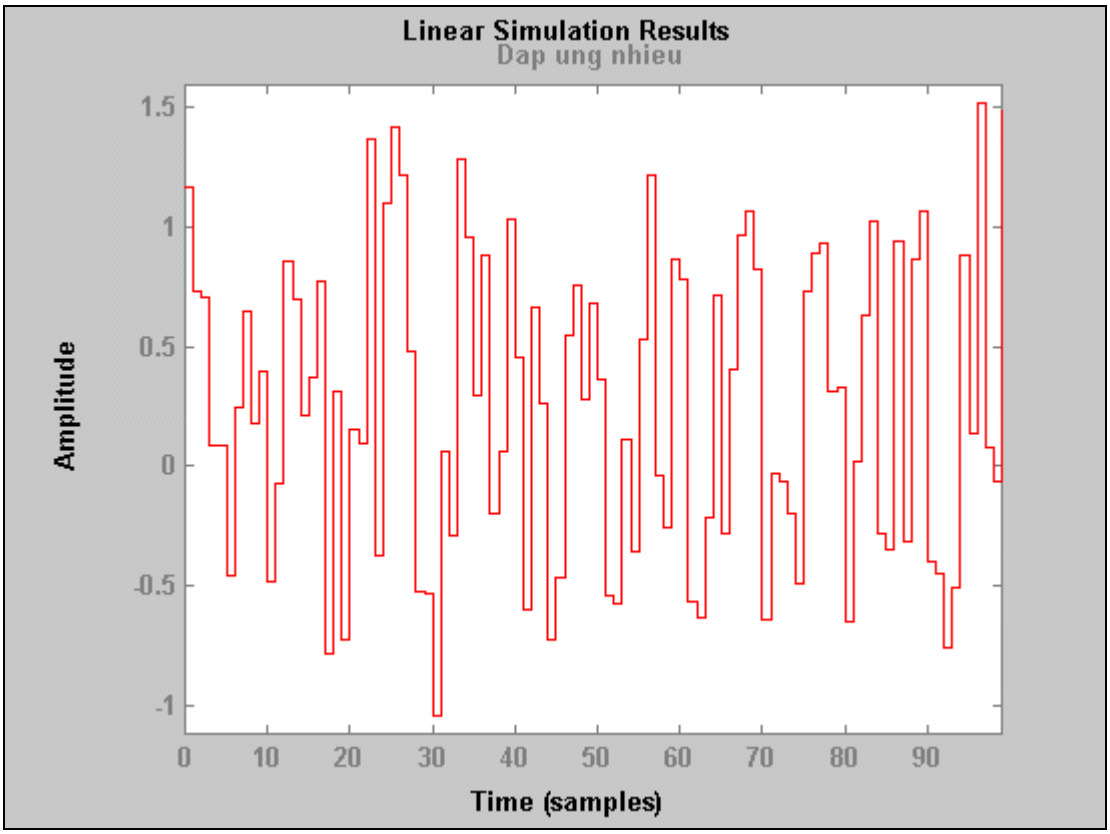
Mô phỏng đáp ứng của hệ thống gián đoạn có hàm truyền:

$$H(z) = \frac{2z^2 - 3.4z + 1.5}{z^2 - 1.6z + 0.8}$$

với 100 mẫu của nhiễu ngẫu nhiên.

```
num = [2    -3.4    1.5];
den = [1    -1.6    0.8];
rand('nomal')
u = rand(100,1);
dlsim(num,den,u)
title('Dap ung nhieu')
```

và ta được đồ thị đáp ứng của hệ như sau:



7. Lệnh STEP

a) Công dụng:

Tìm đáp ứng nấc đơn vị.

b) Cú pháp:

[y,x,t] = step(a,b,c,d)

[y,x,t] = step(a,b,c,d,iu)

[y,x,t] = step(a,b,c,d,iu,t)

[y,x,t] = step(num,den)

[y,x,t] = step(num,den,t)

c) Giải thích:

Lệnh step tìm đáp ứng nấc đơn vị của hệ tuyến tính liên tục.

Nếu bỏ qua các đối số ở vế trái của dòng lệnh thì lệnh step vẽ ra đáp ứng nấc trên màn hình.

step(a,b,c,d) vẽ ra chuỗi đồ thị đáp ứng nấc, mỗi đồ thị tương ứng với mối quan hệ giữa một ngõ vào và một ngõ ra của hệ liên tục LTI:

$$\dot{x} = Ax + Bu$$

$$y = Cx + Du$$

với vector thời gian được xác định tự động.

step(a,b,c,d,iu) vẽ ra đồ thị đáp ứng nấc từ một ngõ vào duy nhất tới tất cả các ngõ ra của hệ thống với vector thời gian được xác định tự động. Đại lượng vô hướng iu

là chỉ số ngõ vào của hệ thống và nó chỉ ra ngõ vào nào được sử dụng cho đáp ứng xung.

step(num,den) vẽ ra đồ thị đáp ứng nấc của hàm truyền đa thức:

G(s) =num(s)/den(s)

trong đó num và den chứa các hệ số đa thức theo chiều giảm dần số mũ của s.

step(a,b,c,d,iu,t) hay step(num,den,t) cũng vẽ ra đáp ứng nấc của hệ không gian trạng thái hay hàm truyền với vector thời gian t do người sử dụng xác định. Vector t chỉ ra những thời điểm mà tại đó đáp ứng nấc được tính và vector t phải được chia thành những đoạn đều nhau.

Nếu giữ lại các đối số ở vế trái của dòng lệnh thì:

[y,x,t] = step(a,b,c,d)

[y,x,t] = step(a,b,c,d,iu)

[y,x,t] = step(a,b,c,d,iu,t)

[y,x,t] = step(num,den)

[y,x,t] = step(num,den,t)

không vẽ ra các đồ thị đáp ứng mà tạo ra các ma trận đáp ứng ngõ ra y và ma trận đáp ứng trạng thái x củahệ thống được xác định tại những thời điểm t. Ma trận y có số cột bằng số ngõ ra và mỗi hàng ứng với một thành phần trong vector t. Ma trận x có số cột bằng số trạng thái và mỗi hàng ứng với một thành phần trong vector t.

d) Ví dụ:

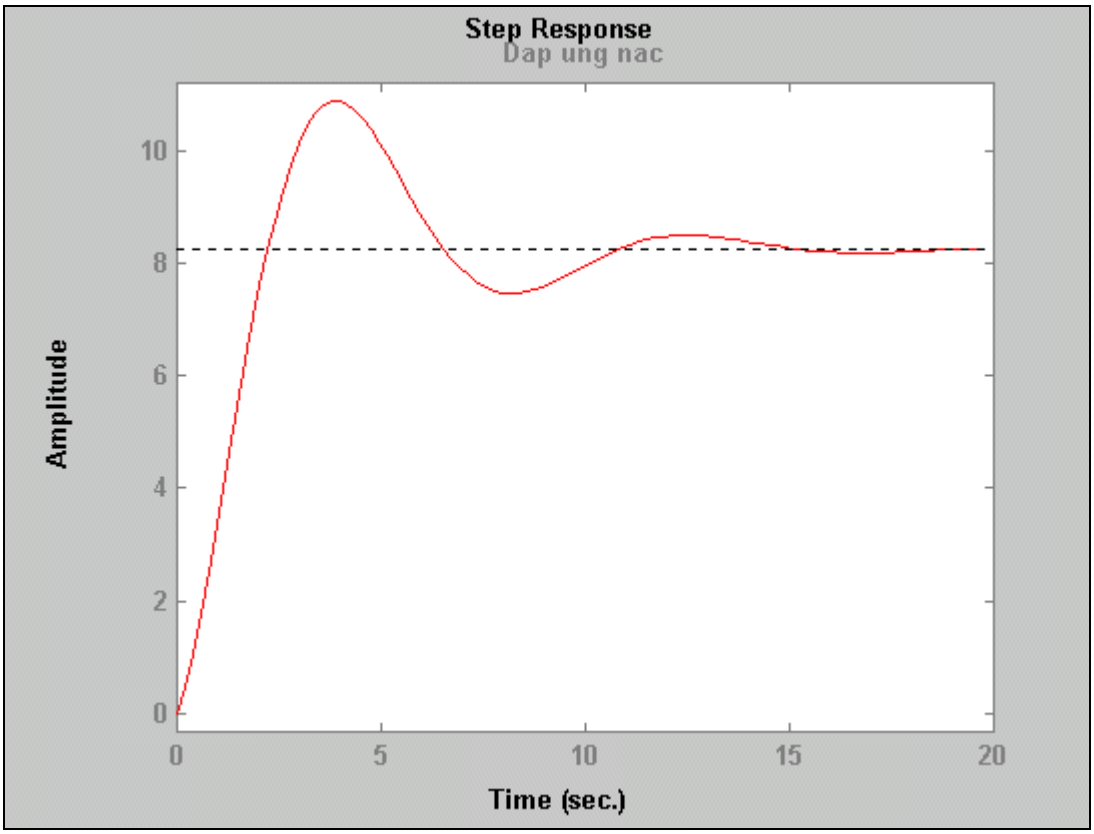
Vẽ đồ thị đáp ứng nấc của hệ không gian trạng thái bậc 2 sau:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -0.5572 & -0.7814 \\ 0.7814 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u$$

$$y = \begin{bmatrix} 1.9691 & 6.4493 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + [0]\, u$$

```
a = [-0.5572     -0.7814 ; 0.7814          0];
b = [1 ; 0];
c = [1.9691      6.4493];
d = [0];
step(a,b,c,d); title('Dap ung nac')
```

và ta được đồ thị đáp ứng nấc của hệ thống như sau:



8. Lệnh DSTEP

a) Công dụng:

Tìm đáp ứng nấc đơn vị của hệ gián đoạn.

b) Cú pháp:

[y,x] = dstep(a,b,c,d)
[y,x] = dstep(a,b,c,d,iu)
[y,x] = dstep(a,b,c,d,iu,n)
[y,x] = dstep(num,den)
[y,x] = dstep(num,den,n)

c) Giải thích:

Lệnh dstep tìm đáp ứng nấc đơn vị của hệ tuyến tính gián đoạn.

Nếu bỏ qua các đối số ở vế trái của dòng lệnh thì lệnh dstep vẽ ra đáp ứng nấc trên màn hình.

dstep(a,b,c,d) vẽ ra chuỗi đồ thị đáp ứng nấc, mỗi đồ thị tương ứng với mối quan hệ giữa một ngõ vào và một ngõ ra của hệ gián đoạn LTI:

$$x[n + 1] = Ax[n] + Bu[n]$$
$$y[n] = Cx[n] + Du[n]$$

với số điểm lấy mẫu được xác định tự động.

dstep(a,b,c,d,iu) vẽ ra đồ thị đáp ứng nấc từ một ngõ vào duy nhất tới tất cả các ngõ ra của hệ thống với số điểm lấy mẫu được xác định tự động. Đại lượng vô

hướng iu là chỉ số ngõ vào của hệ thống và nó chỉ ra ngõ vào nào được sử dụng cho đáp ứng xung.

dstep(num,den) vẽ ra đồ thị đáp ứng nấc của hàm truyền đa thức:

G(z) =num(z)/den(z)

trong đó num và den chứa các hệ số đa thức theo chiều giảm dần số mũ của s.

dstep(a,b,c,d,iu,n) hay dstep(num,den,n) cũng vẽ ra đáp ứng nấc của hệ không gian trạng thái hay hàm truyền với số điểm lấy mẫu do người sử dụng xác định.

Nếu giữ lại các đối số ở vế trái của dòng lệnh thì:

[y,x] = dstep(a,b,c,d)

[y,x] = dstep(a,b,c,d,iu)

[y,x] = dstep(a,b,c,d,iu)

[y,x] = dstep(num,den)

[y,x] = dstep(num,den,n)

không vẽ ra các đồ thị đáp ứng mà tạo ra các ma trận đáp ứng ngõ ra y và ma trận đáp ứng trạng thái x củahệ thống. Ma trận y có số cột bằng số ngõ ra. Ma trận x có số cột bằng số trạng thái.

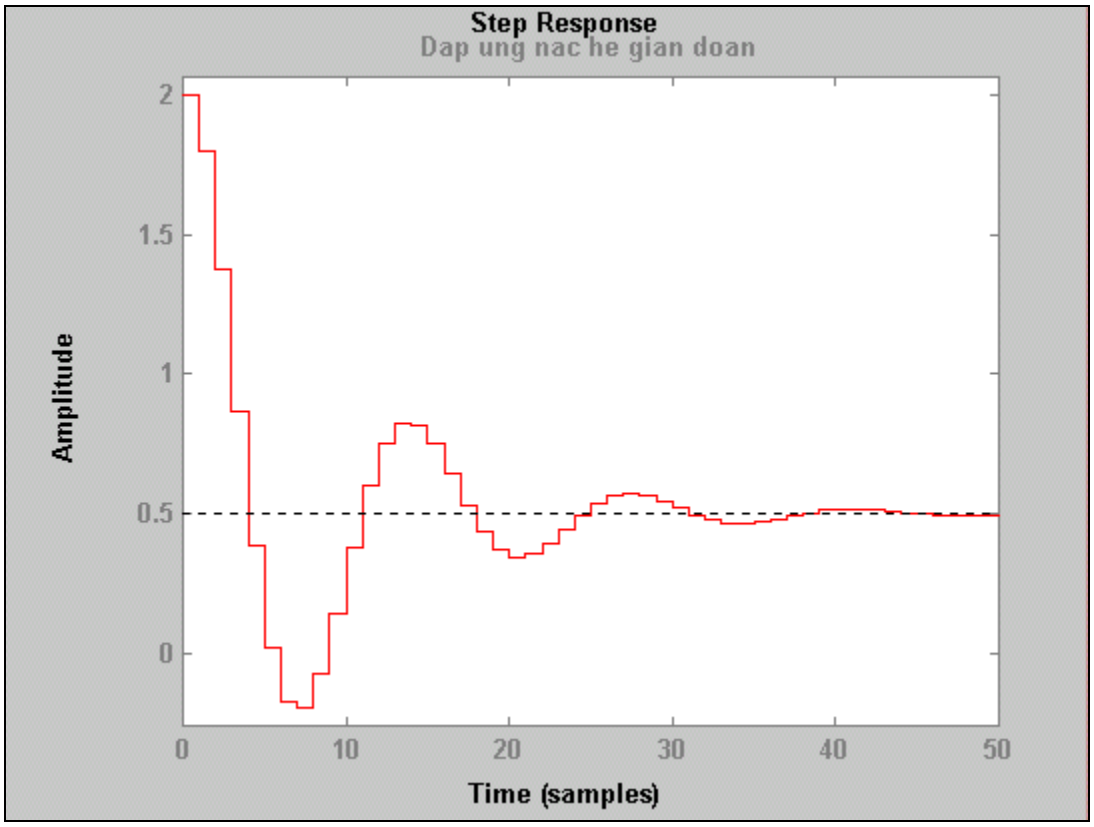
d) Ví dụ:

Vẽ đáp ứng nấc của hệ gián đoạn của hệ có hàm truyền như sau:

$$H(z) = \frac{2z^2 - 3.4z + 1.5}{z^2 - 1.6z + 0.8}$$

```
num = [2    -3.4    1.5];
den = [1    -1.6    0.8];
dstep(num,den)
title('Dap ung nac he gian doan')
```

và ta được đồ thị đáp ứng nấc của hệ như hình bên:



9. Lệnh LTITR

a) Công dụng:

Tìm đáp ứng thời gian của hệ tuyến tính bất biến.

b) Cú pháp:

ltitr(a,b,u)

ltitr(a,b,u,x0)

c) Giải thích:

Lệnh ltitr dùng để mở rộng đáp ứng thời gian của hệ tuyến tính bất biến. Nó mô phỏng cho hệ không gian trạng thái gián đoạn:

x = ltitr(a,b,u) mở rộng đáp ứng của hệ gián đoạn:

$$x[n + 1] = Ax[n] + Bu[n]$$

đối với ngõ vào u. Ma trận u phải có số cột bằng số ngõ vào u. Mỗi hàng của ma trận u tương ứng với một điểm thời gian mới.

ltitr tạo ra ma trận x với số cột bằng số trạng thái x và có số hàng là length(u).

Nếu thêm vào vế phải dòng lệnh tham số x0 thì điều kiện ban đầu sẽ được thiết lập với lệnh x = ltitr(a,b,u,x0)

10. Lệnh FILTER

a) Công dụng:

Lọc dữ liệu với đáp ứng xung không xác định hay đáp ứng xung xác định.

b) Cú pháp:

```
y = filter(b,a,X)
[y,zf] = filter(b,a,X)
[y,zf] = filter(b,a,X,zi)
y = filter(b,a,X,zi,dim)
[...] = filter(b,a,X,[ ],dim)
```

c) Giải thích:

Lệnh fiter lọc dữ liệu tuần tự sử dụng bộ lọc số cho các ngõ vào thực và phức.

y = filter(b,a,X) lọc dữ liệu trong vector X với bộ lọc được mô tả bởi vector hệ số tử số b và vector hệ số mẫu số a. Nếu a(1) không bằng 1, bộ lọc sẽ chuẩn hóa hệ số lọc bởi a(1). Nếu a(1) bằng 0 thì sẽ báo lỗi.

Nếu X là một ma trận, bộ lọc sẽ thực hiện trên các cột của X. Nếu X là một mảng đa chiều, bộ lọc sẽ thực hiện theo chiều duy nhất.

[y,zf] = filter(b,a,X) tạo ma trận điều kiện cuối cùng zf của bộ trễ. Ngõ ra zf là một vector của max(size(a),size(b)) hoặc một tập hợp các vector với mỗi vector là một cột của X.

[y,zf] = filter(b,a,X,zi) chấp nhận điều kiện ban đầu zi và tạo ra điều kiện cuối cùng cuối cùng zf của bộ lọc trễ. Ngõ vào zi là một vector có kích thước length(a),length(b)) – 1.

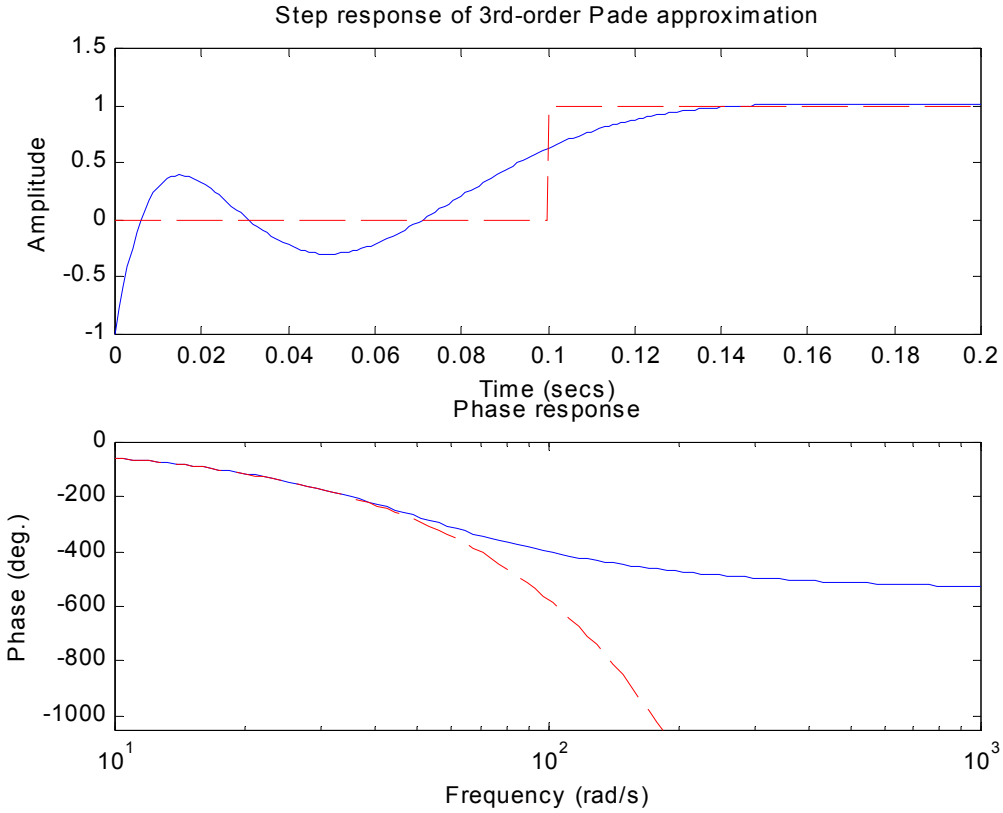
y = filter(b,a,X,zi,dim) và [...] = filter(b,a,X,[ ],dim) thực hiện lọc theo chiều dim.

CÁC BÀI TẬP VỀ ĐÁP ỨNG THỜI GIAN

Bài1: Lệnh **pade:** Tính toán sắp xỉ

Bài này trích từ trang 11-66 sách **'Control System Toollbox'**

» pade(0.1,3)



Bài 2: Trích từ trang 11-24 sách **'Control System Toollbox'**

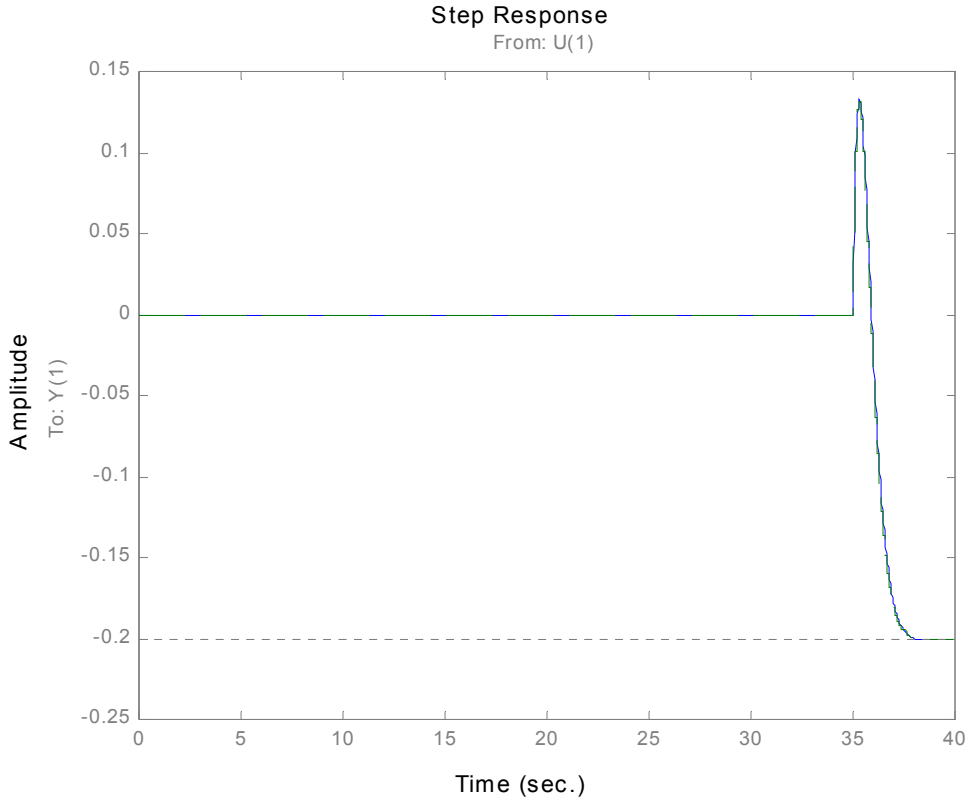
$$H(s) = \frac{s - 1}{s^2 + 4s + 5}$$

```
» H=tf([1 -1],[1 4 5],'inputdelay',035)

Transfer function:
          s - 1
exp(-35*s) * -------------
        s^2 + 4 s + 5

» Hd=c2d(H,0.1,'foh')

Transfer function:
```

$$z^{-350} * \frac{0.04226\,z^2 - 0.01093\,z - 0.03954}{z^2 - 1.629\,z + 0.6703}$$

Sampling time: 0.1

```
» step(H,'-',Hd,'--')
```

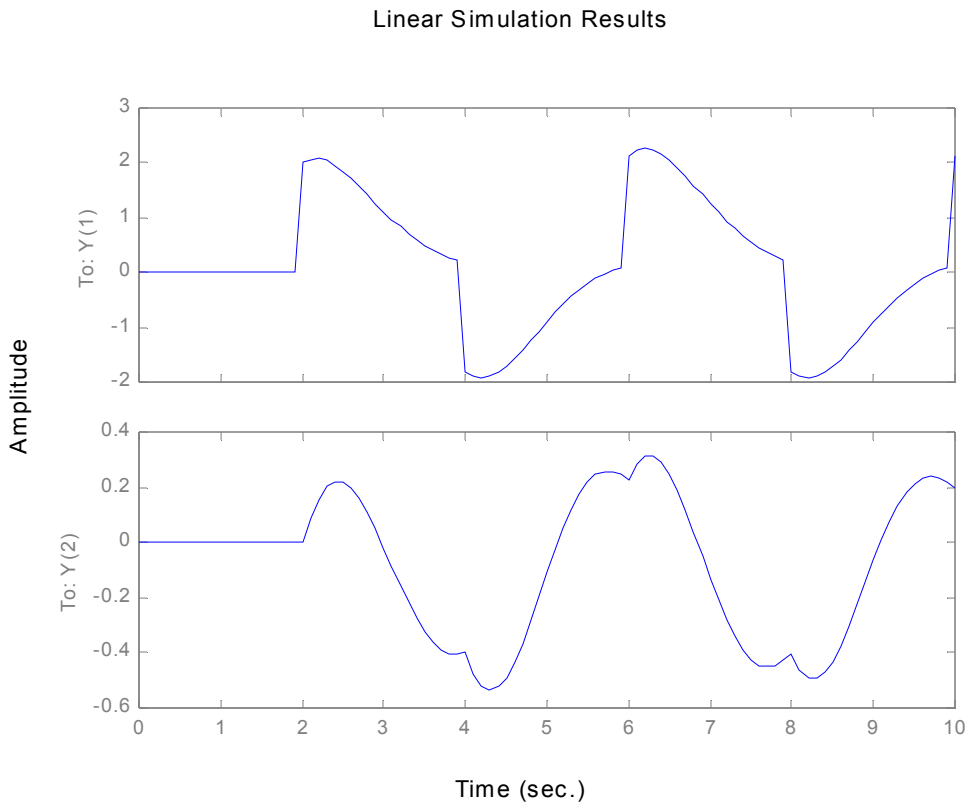


Bài 3: Trang 11-127, $H(s) = \begin{pmatrix} \frac{2s^2 + 5s + 1}{s^2 + 2s + 3} \\ \frac{s - 1}{s^2+s+5} \end{pmatrix}$

```
» [u,t]=gensig('square',4,10,0.1);
» H=[tf([2 5 1],[1 2 3]);tf([1 -1],[1 1 5])];
» lsim(H,u,t)
```

Kết quả:

Bài tập này được trích từ trang 11-127 sách **'Control System Toolbox'**



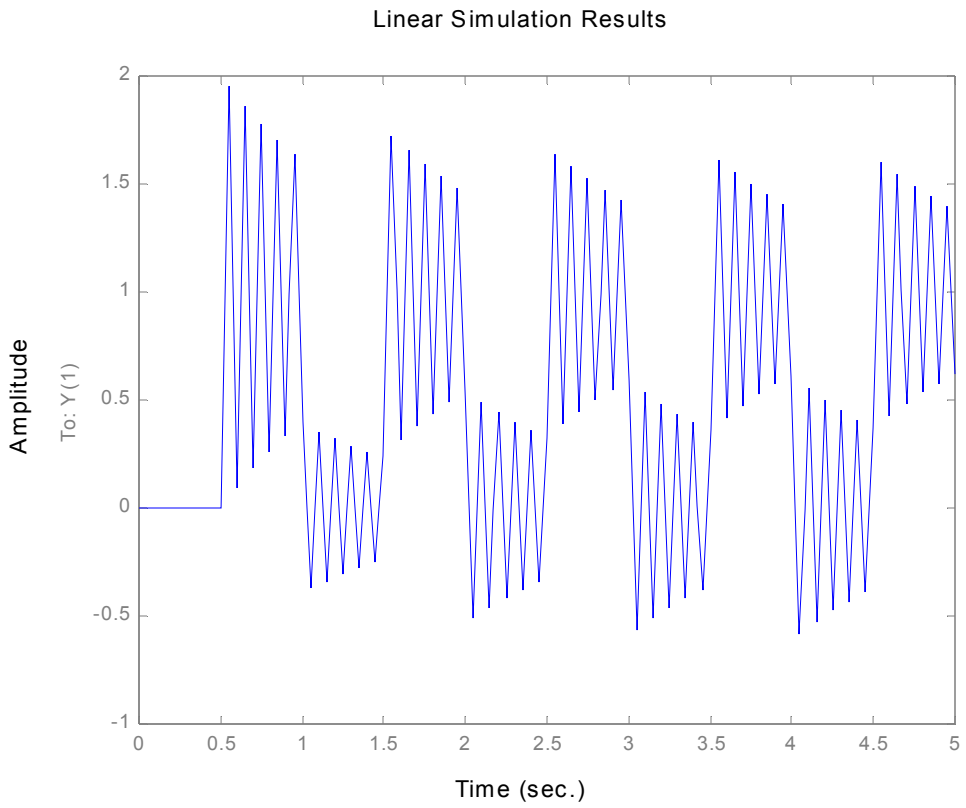
Bài 4: Dùng lệnh **lsim**, trích từ trang 11-130 sách **'Control Systen Toollbox'**
Dịch đề: Vẽ đáp ứng khâu bậc 2 của hàm truyền sau:

$$h(s) = \frac{\omega^2}{s^2 + 2s + \omega^2}$$

$\omega = 62,83$

```
» w2=62.83^2

w2 =

  3.9476e+003

» h=tf(w2,[1 2 w2]);
» t=0:0.1:5;          %vector of time sample:
» u=(rem(t,1)>=0.5);  %square ware value    :
» lsim(h,u,t)
```

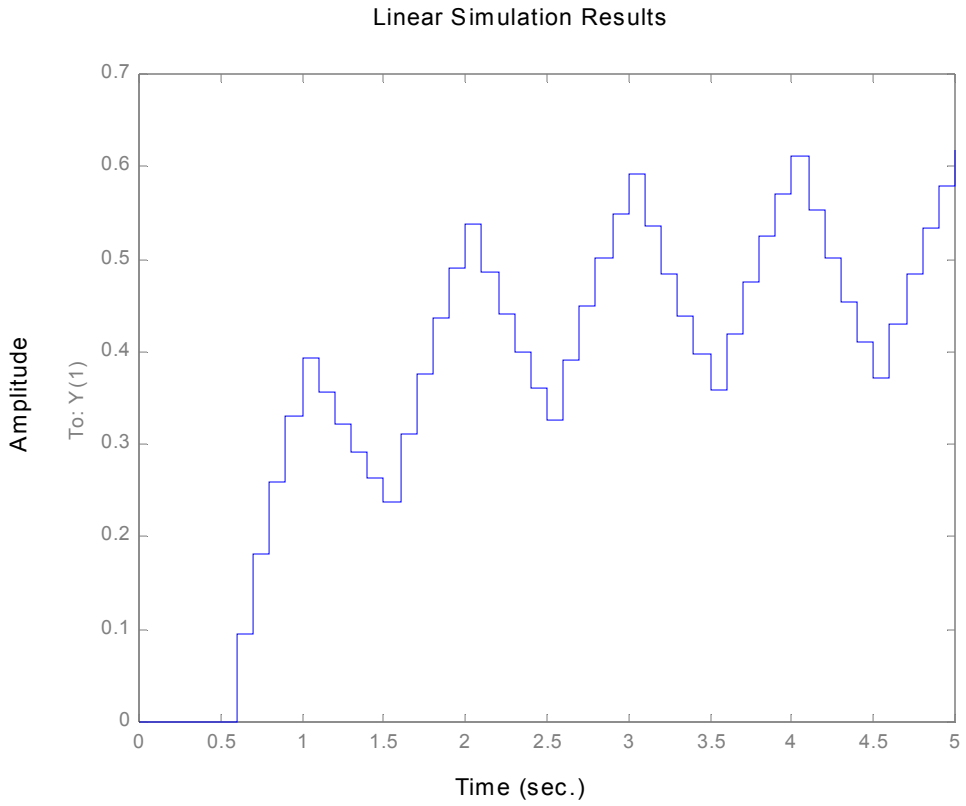
Kết quả:



Bài 5: Trang 11-131 sách **'Control Systen Toollbox'**
Ta lấy số liệu bài 24 nhưng thời gian mẫu là 0,1.

Chương trình:

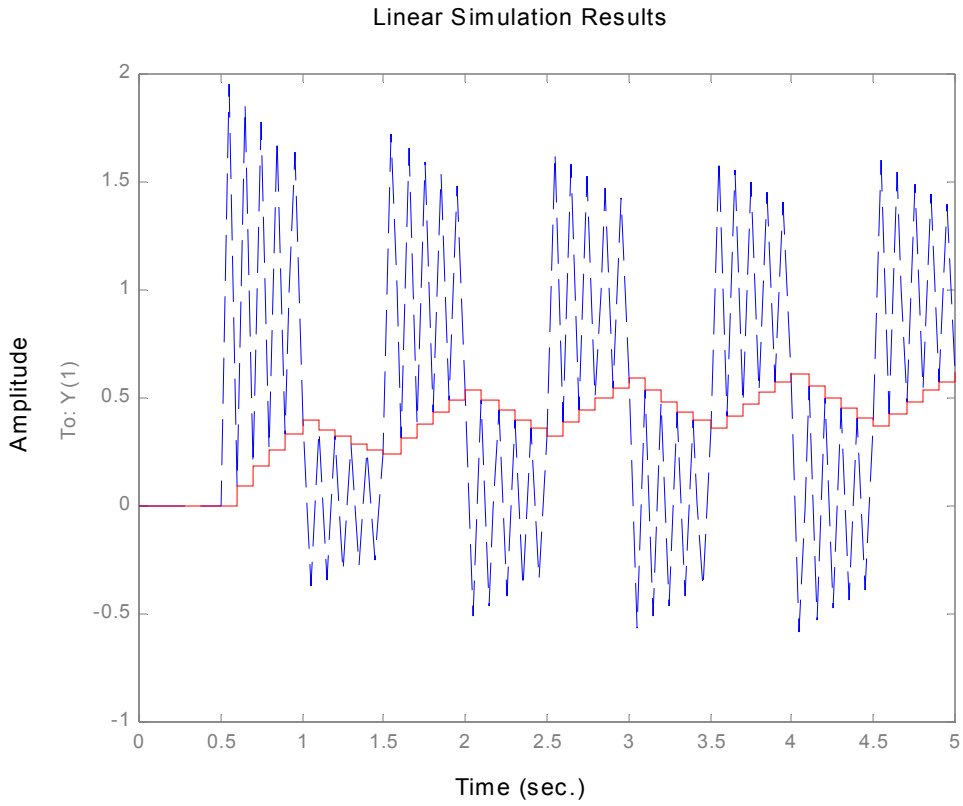
```
» w2=62.83^2;
» hd=c2d(h,0.1);
» t=0:0.1:5;          %vector of time sample:
» u=(rem(t,1)>=0.5);  %square ware value    :
» lsim(hd,u,t)
```



Bài 6: Trang 11-132 sách **'Control Systen Toollbox'**

Cũng lấy số liệu 2 bài trên.

```
» w2=62.83^2;
» h=tf(w2,[1 2 w2]);
» t=0:0.1:5;          %vector of time sample:
» u=(rem(t,1)>=0.5);  %square ware value    :
» hd=c2d(h,0.1);
» lsim(h,'b--',hd,'r-',u,t)   %
```



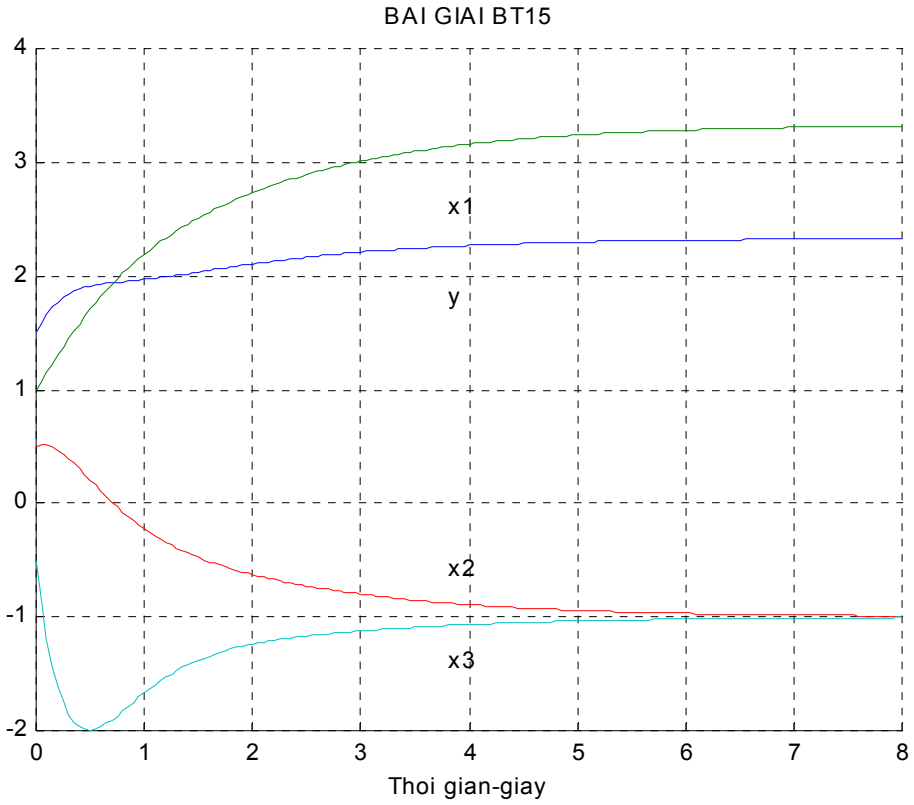
Bài 7: Trích từ trang 46 sách **'ứng dụng matlab trong điều khiển tự động'**

Phương trình biến trang thái của hệ thống tuyến tính bất biến theo thời gian là:

Chương trình được viết trong file.m:

```
%function  [yout,x] = lsim(A, B, C, D, U, t, x0)
%Phuong trinh bien trang thai cua mot he thong tuyen tinh
% bat bien theo thoi gian la:
%   .
%    x1
%       .       0   1   0       x1          1
%  {x2} = { 0   0   1 }  { x2 }  + {1} r(t)
%       .      -6 -11 -6        x3          1
%    x3
%                    1
% y=[1 1 0]x,  x(0)=  0.5
%                  -0.5
% Xac dinh x(t),y(t) khi r(t) la ham bac don vi
hold on
grid on
A=[0 1 0;0 0 1;-6 -13 -6];
B=[1;1;1];%xac dinh vi ban dau va hinh dang cua do thi x1,y,x2,x3
C=[1 1 0];
```

```
D=0;
x0=[1 .5 -.5]; %vecto hang dieu kien ban dau
t=0:.05:8;    %buoc nhay
U=ones(1,length(t));%tao vecto hang u(t)
[x,y]=lsim(A,B,C,D,U,t,x0);
plot(t,x,t,y)
title('BAI GIAI BT15')
xlabel('Thoi gian-giay')
text(3.8,1.8,'y'),text(3.8,2.6,'x1');%Canh vi tri cua y va x1 tren do thi
text(3.8,-0.6,'x2'),text(3.8,-1.4,'x3')%Canh vi tri cua x2 va x3 tren do thi
```



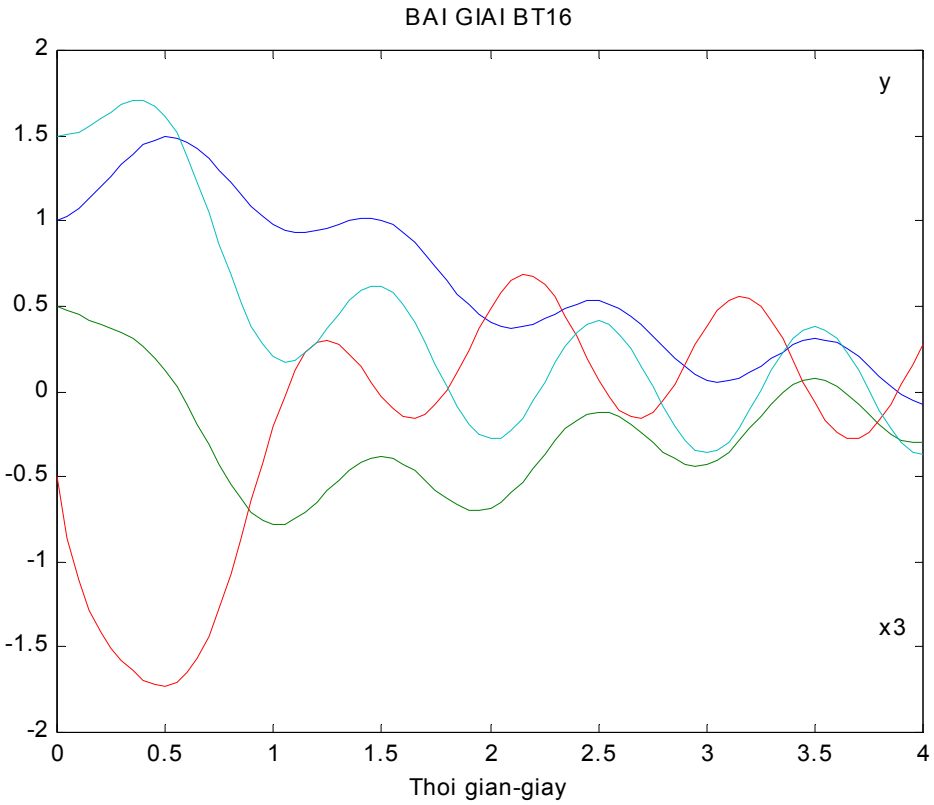
Bài 9: trích từ trang 48 sách tác giả Nguyễn Văn Giáp.

Cũng với yêu cầu như bài 28, nhưng r(t)=sin(2Πt).

Chương trình soạn trong file.m:

```
%function  [yout,x] = lsim(A, B, C, D, U, t, x0)
%BT16:Ve do thi y(t),x(t) cua bai BT15 neu r(t)=sin(2pit)
A=[0 1 0;0 0 1;-6 -11 -6];
B=[1;1;1];C=[1 1 0];D=0;
```

```
x0=[1 .5 -.5];          %vecto hang dieu kien ban dau
t=0:.05:4;              %buoc nhay
r=sin(2*pi*t);
[y,x]=lsim(A,B,C,D,r,t,x0);
plot(t,x,t,y)
title('BAI GIAI BT16')
xlabel('Thoi gian-giay')
text(3.8, 1.8,'y'),text(3.8, 2.6,'x1')
text(3.8, -8,'x2'),text(3.8, -1.4,'x3')
```



ài

Bài 10: Xét hàm truyền sau:

$$G(s) = \frac{s+10}{s^2+8s+25}$$

Để tính đáp ứng bước của hệ thống này ta dùng cấu trúc như sau:

[out,state,tt]=step([1 10],[1 8 25])

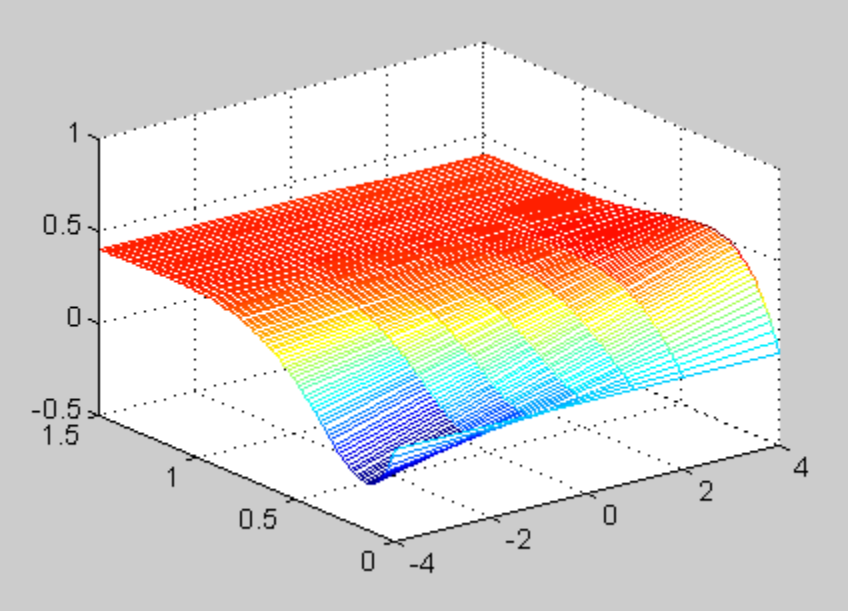
Giả sử ta muốn phân tích một đáp ứng bước của hệ thống thay đổi, với zero của hàm truyền thay đổi nhưng độ lợi dc (dc gain) của hệ thống không đổi, để giữ lại cho hệ thống cùng mẫu và thay đổi hệ số của số hạng đầu trong đa thức của tử,tức là hệ số của s, vì vậy mà dc gain là hằng số và zero thay đổi.

Ví dụ : hệ thống như ví dụ trên nhưng số hạng ban đầu của đa thức ở tử số thay đổi thành (-4,-2,-1,0,1,2,4)

Ta thực hiện trong cửa sổ lệnh của matlab như sau:

```
» coef=[-4 -2 -1 0 1 2 4];
» den=[1 8 25];
» [y,x,t]=step([coef'  10*ones(length(coef),1)],den);
» mesh(coef,t,y)
```

Kết quả như hình:



Hình 3.7: So sánh giữa các đáp ứng step

Bài 11: đáp ứng xung (impulse)

Ví dụ hệ thống có hàm truyền sau:

$$G(s) = \frac{s+10}{s^2+2s+25}$$

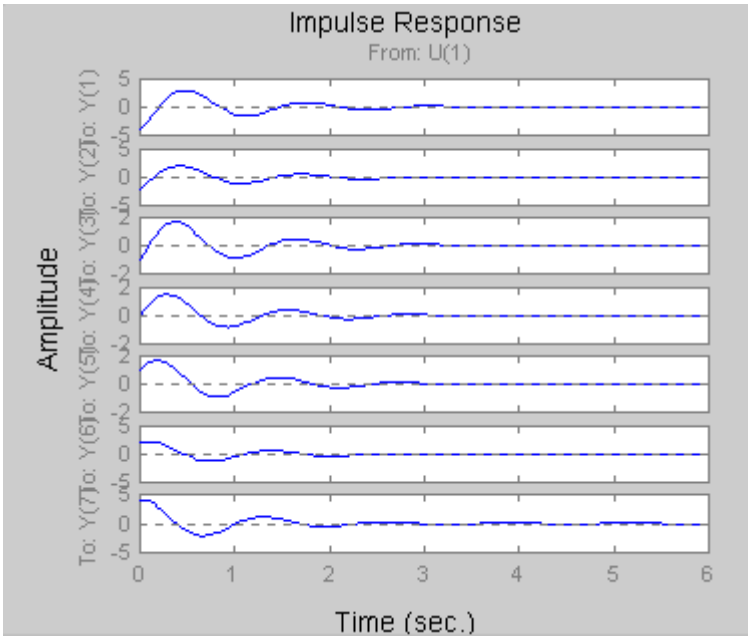
Vẽ đáp ứng xung của hệ thống:

```
impulse([1  10],[1  2  25])
```

Giả sử ta muốn phân tích đáp ứng xung thay đổi như thế nào khi zero của hàm truyền thay đổi, không thay đổi dc gain của hệ thống. giống như ví dụ ở phần trước ta có :

```
» coef=[-4 -2 -1 0 1 2 4];
» den=[1 2 25];
» impulse([coef' 10*ones(length(coef),1)],den);
```

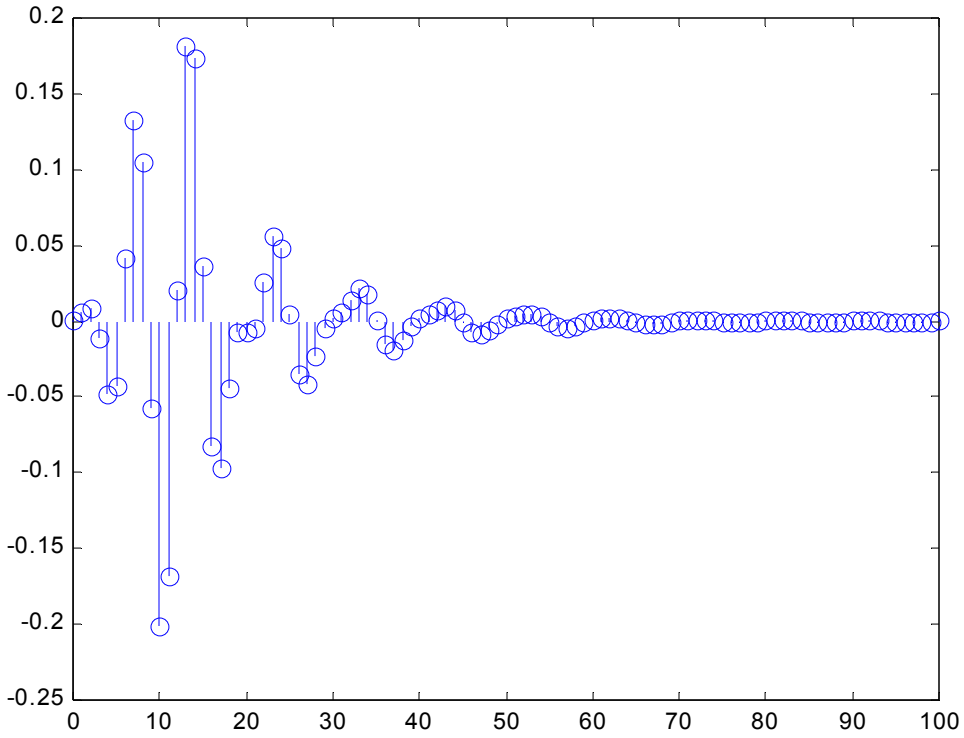
Kết quả như hình sau:



Bài 12: Trích từ trang 716 sách **'The Student Edition of MATLAB'**

Dịch đề: Thiết kế 1 khâu gồm 10 bộ lọc của dãi băng truyền ngang có tần số từ 100 đến 200 Hz và vẽ đáp ứng xung của nó:

```
» n=5;wn=[100 200]/500;
» [b,a]=butter(n,wn);
» [y,t]=impz(b,a,101);
» stem(t,y)
```



Bài 13: Đáp ứng từng ngõ vào

Một vấn đề tổng quát hơn là ta có thể tính được tín hiệu ngõ ra của hệ thống LTI với một tính hiệu ngõ vào không đồng nhất.

Ví dụ như hệ thống bậc nhất sau:

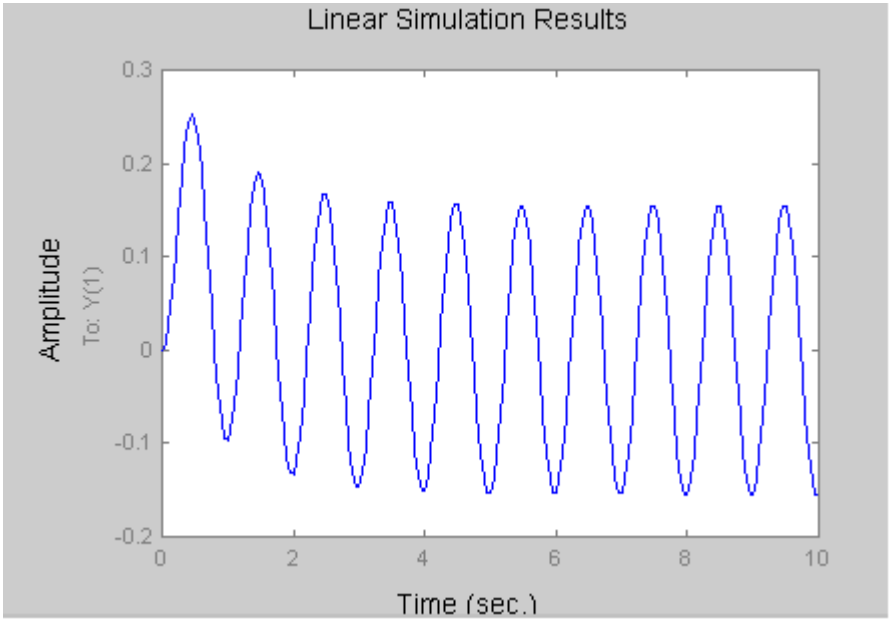
$$\dot{x} = -x + u$$

$$y = x$$

Hệ thống này bị tác động với một tín hiệu ngõ vào hình sin có tần số là 1Hz, tín hiệu ngõ ra thu được bởi cấu trúc:

```
>> freq=1; t=0:0.05:10;
>> u=sin(2*pi*freq*t); lsim(-1,1,1,0,u,t)
```

Kết quả là hình sau:



Hình : Đáp ứng từng ngõ vào

VẼ GIẢN ĐỒ BODE, NyQuist, Nichols

LÝ THUYẾT:

Giản đồ Bode gồm hai đồ thị: Đồ thị logarith biên độ của hàm truyền và góc pha theo logarith tần số. (một đơn vị ở trục hoành gọi là một decade).

Biên độ : $|G(j\omega)|_{dB} = 20 \log_{10} |G(j\omega)|$ (2.22)

Pha : $\varphi = \underline{/G(j\omega)}$ (hay arg G(jω)) (2.23)

Giản đồ Bode của các khâu cơ bản:

* Khâu khuếch đại:

Hàm truyền đạt G(s) = K

Giản đồ Bode L(ω) = 20 lgM(ω) = 20 lgK là 1 đường thẳng song song với trục hoành.

* Khâu quán tính bậc 1:

Hàm truyền đạt $G(s) = \frac{K}{Ts+1}$

Biểu đồ Bode L(ω) = 20 lgM(ω) = 20 lgK – $20\lg\sqrt{T^2\omega^2+1}$ có độ dốc giảm

–20dB/decade

* Khâu vi phân bậc 1:

Hàm truyền đạt G(s) = K(Ts + 1)

Giản đồ Bode L(ω) = 20 lgM(ω) = 20 lgK + $20\lg\sqrt{T^2\omega^2+1}$ có độ dốc tăng 20dB/decade

* Khâu tích phân:

Hàm truyền đạt $G(s) = \frac{K}{s}$

Giản đồ Bode L(ω) = 20 lgM(ω) = 20 lgK – 20lgω

* Khâu bậc 2:

Hàm truyền đạt $G(s) = \frac{\omega_n^2}{s^2 + 2\varepsilon\omega_n s + \omega_n^2}$

Giản đồ Bode L(ω) = -20lg $\sqrt{\left(1-\omega^2t^2\right)^2 + 4\varepsilon^2\omega^2t^2}$

BÀI TẬP

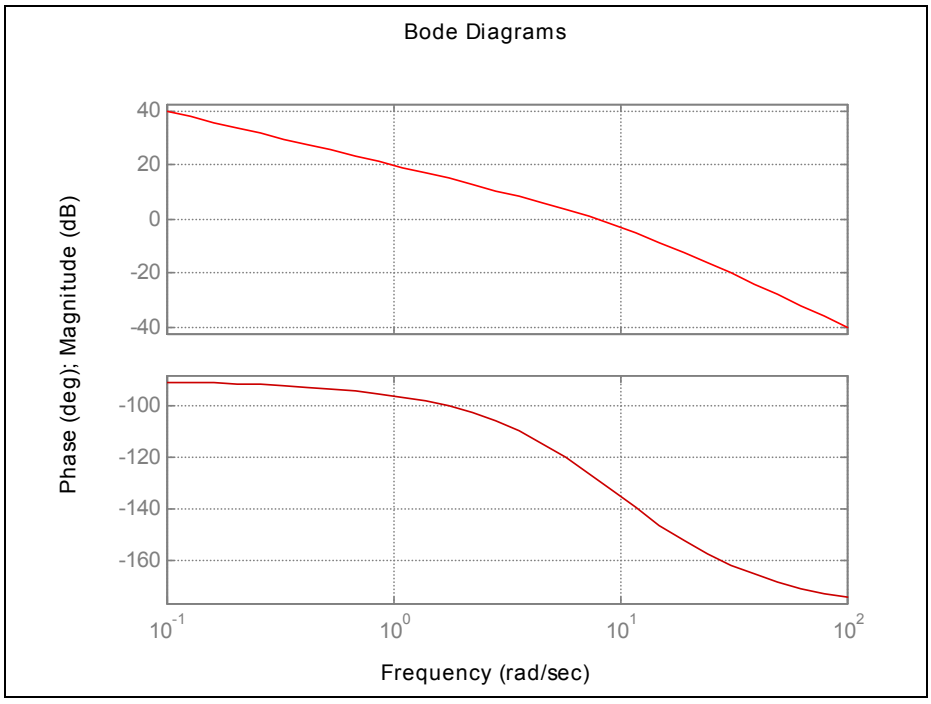
Bài 1:

Vẽ giản đồ Bode hệ thống hồi tiếp đơn vị của hàm truyền vòng hở sau:

$$G(s) = \frac{10}{s(1+0.1s)}$$

```
» num = 10;
» den =  [0.1 1 0];
» bode(num,den)
```

Kết quả:



Hệ thống gồm 1 khâu khuếch đại bằng 10, một khâu tích phân và một khâu quán tính bậc 1

Tần số gãy: 10.

$|G(jw)|_{dB} = 20dB - 20\log\omega$

Tại tần số $\omega = 1rad/sec$ $|G(jw)|_{dB} = 20dB$ và độ dốc $-20dB/decade$ (do khâu tích phân).

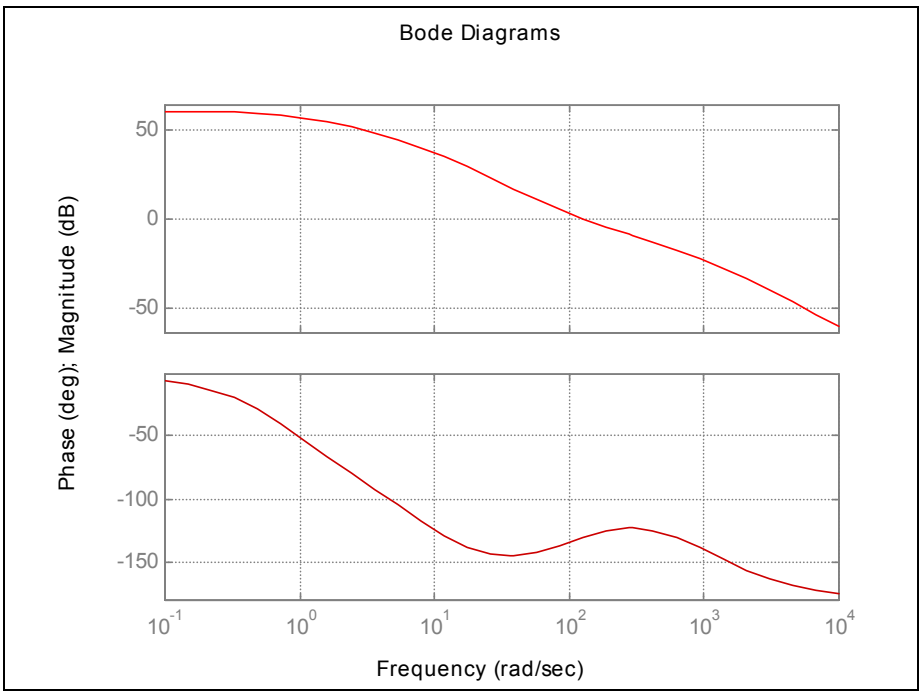
Độ dốc $-20dB/decade$ tiếp tục cho đến khi gặp tần số cắt $\omega = 10rad/sec$, tại tần số này ta cộng thêm $-20dB/decade$ (do khâu quán tính bậc nhất) và tạo ra độ dốc -40dB/dec.

Bài 2:

$$G(s) = \frac{10^5 (s+100)}{(s+1)(s+10)(s+1000)}$$

```
» num = 100000*[1 100];
» den = [1 1011 11010 10000];
» bode(num,den)
```

Kết quả:



Hệ thống gồm một khâu khuếch đại 10^5, một khâu vi phân bậc nhất và 3 khâu quán tính bậc 1.

Tần số gãy: 1,10,100,1000.

$|G(jw)|_{dB|w=0} = 60dB$

Tại tần số gãy ω = 1rad/sec có độ lợi 60dB và độ dốc –20dB/decade (vì khâu quán tính bậc 1). Độ dốc –20dB/decade được tiếp tục đến khi gặp tần số gãy ω = 10rad/sec tại đây ta cộng thêm -20dB/decade(vì khâu quán tính bậc 1), tạo ra độ dốc –40dB/dec. Độ dốc - 20dB ở tần số ω = 100rad/dec (do khâu vi phân bậc 1). Tại tần số gãy ω = 100rad/sec tăng 20dB (vì khâu vi phân bậc 1). Tạo ra độ dốc có độ dốc -20dB.

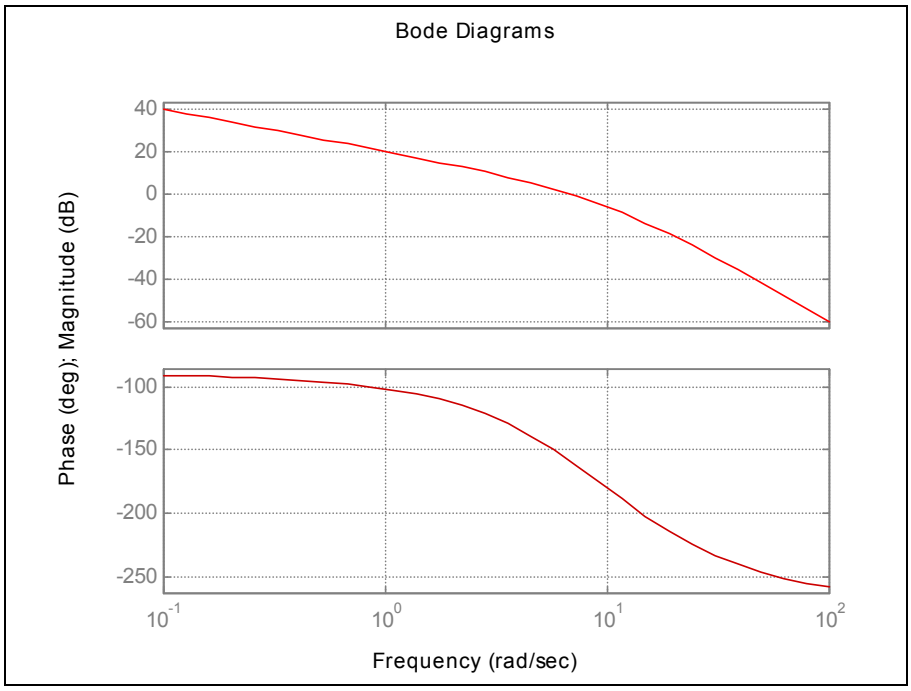
Tại tần số gãy ω = 1000rad/sec giảm 20dB (vì khâu quán tính bậc 1). Tạo ra độ dốc -40dB.

Bài 3:

$$G(s) = \frac{10}{s(1+0.1s)^2}$$

» num = 10;
» den = [0.01 0.2 1 0];
» bode(num,den)

Kết quả:



Hệ thống gồm một khâu khuếch đại 10, một khâu tích phân và 1 thành phần cực kép.

Tần số gãy: 10.

$|G(jw)|_{dB} = 20dB - 20\log\omega$

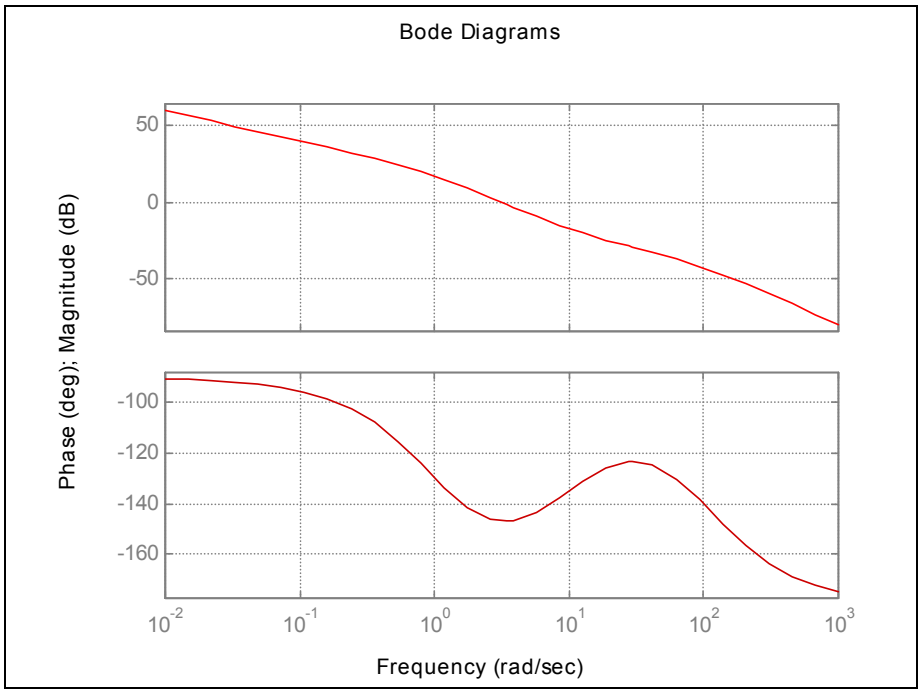
Tần số gãy nhỏ nhất ω = 0.1 rad/sec tại tần số này có độ lợi 40dB và độ dốc –20dB (do khâu tích phân). Độ dốc này tiếp tục cho tới tần số gãy kép ω = 10. Ở tần số này sẽ giảm 40dB/decade, tạo ra độ dốc –60dB/dec.

Bài 4:

$$G(s) = \frac{10^2(s+10)}{s(s+1)(s+100)}$$

```
» num = 100*[1 10];
» den = [1 101 100 0];
» bode(num,den)
```

Kết quả:



Hệ thống gồm một khâu khuếch đại 100, một khâu tích phân và 2 khâu quán tính bậc 1, 1 khâu vi phân.

Tần số gãy: 1,10,100

$|G(jw)|_{dB|w=0} = 20\log 10 - 20\log\omega$

Ta chỉ xét trước tần số gãy nhỏ nhất 1decade. Tại tần số gãy ω = 0.1rad/sec có độ lợi 40dB và độ dốc –20dB/dec, độ dốc –20dB/dec tiếp tục cho đến khi gặp tần số gãy ω = 1rad/sec, ta cộng thêm –20dB/dec (vì khâu quán tính bậc 1) và tạo ra độ dốc –40dB/dec. Tại tần số ω =10 sẽ tăng 20dB/dec (vì khâu vi phân) tạo ra độ dốc –20dB/dec, độ dốc –20db/dec được tiếp tục cho đến khi gặp tần số gãyω = 100rad/sec sẽ giảm 20dB/dec (vì khâu quán tính bậc 1) sẽ tạo độ dốc –40dB/decade.

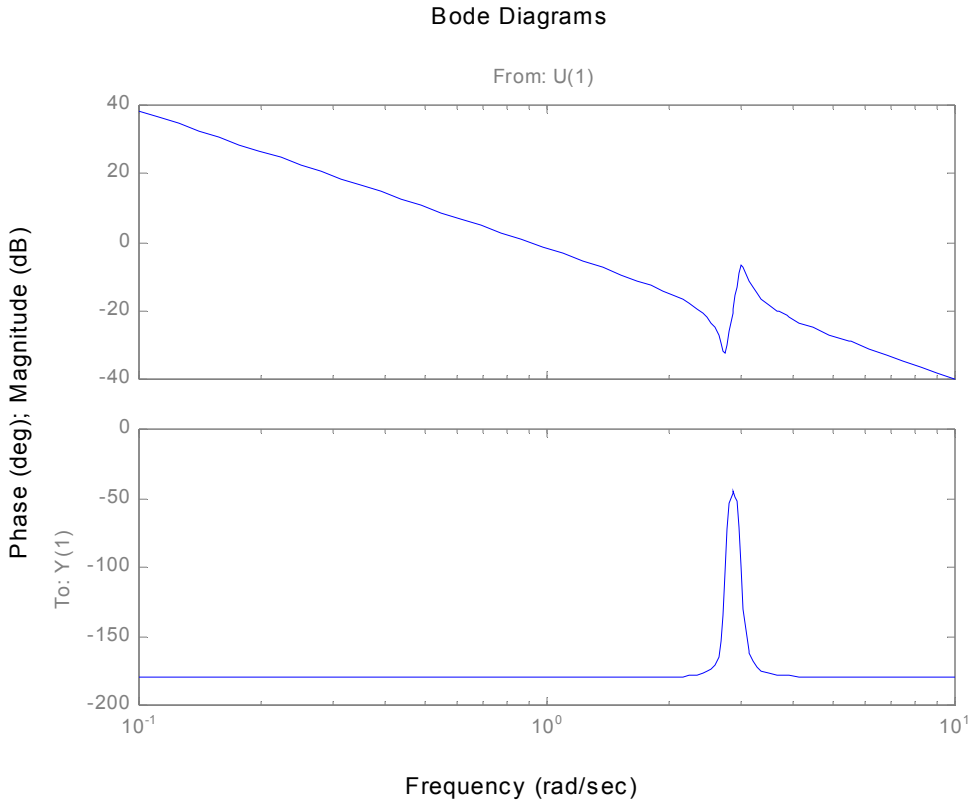
Bài 5: Bài này trích từ trang 11-21 sách **'Control System Toollbox'**

Vẽ giản đồ bode của hệ thống hồi tiếp SISO có hàm sau:

$$S^2+01.s+7.5$$

$$H(s) = \frac{}{S^2+0.12s^3+9s^2}$$

```
» g=tf([1 0.1 7.5],[1 0.12 9 0 0]);
» bode(g)
```



Bài 6: Trang 11-153 sách **'Control System Toolbox'**
Vẽ giản đồ bode của hàm rời rạc sau, với thời gian lấy mẫu là: 0,1.

$$H(z) = \frac{z^3-2.841z^2+2.875z-1.004}{z^3+2.417z^2+2.003z-0.5488}$$

```
» H=tf([1 -2.841 2.875 -1.004],[1 -2.417 2.003 -0.5488],0.1);
» norm(H)

ans =

   1.2438

» [ninf,fpeak]=norm(H,inf)
```

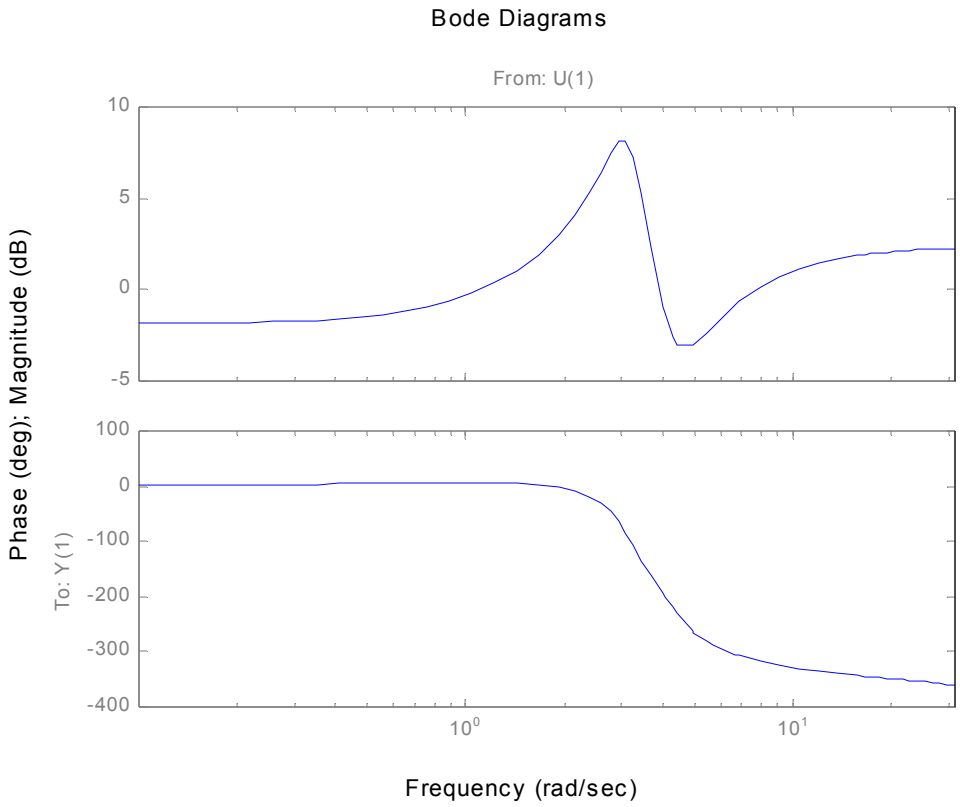
ninf =

2.5488

fpeak =

3.0844

» bode(H)



» 20*log(ninf)

ans =

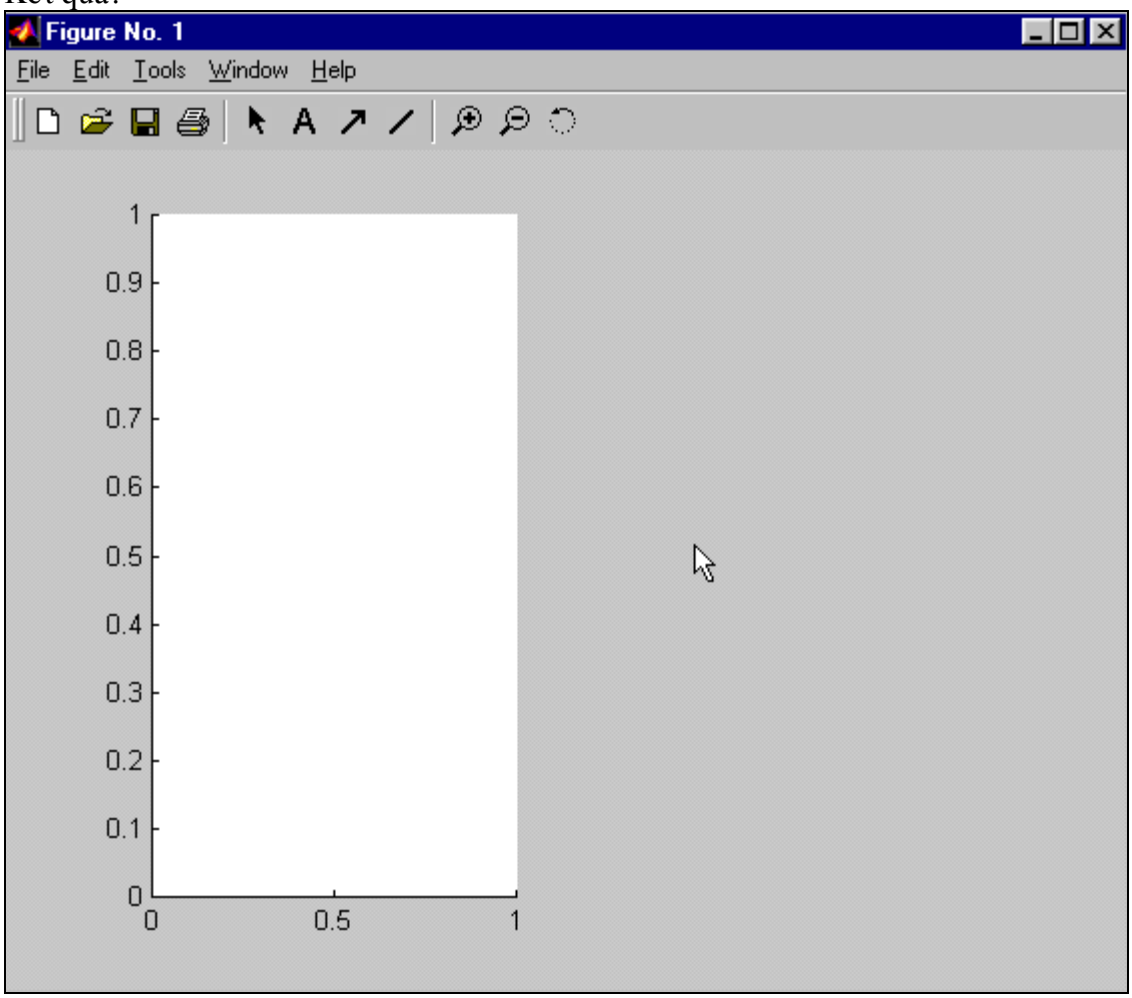
18.7127

Bài 7: Trích từ trang 5-18 sách **'Control System Toolbox'**

Bài này cho ta xem công dụng của lệnh chia trục **subplot**

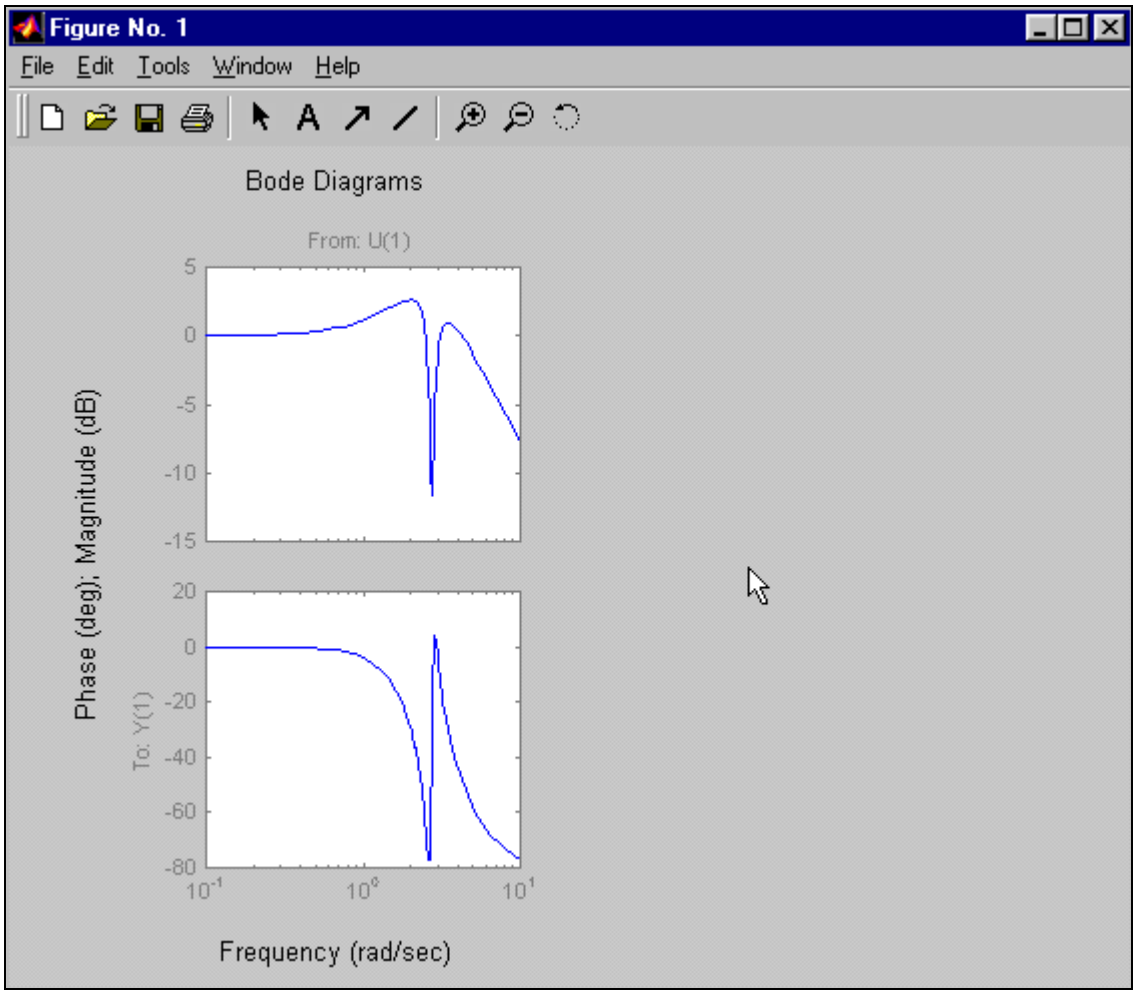
```
» h=tf([4 8.4 30.8 60],[1 4.12 17.4 30.8 60]);
» subplot(121)
```

Kết quả:

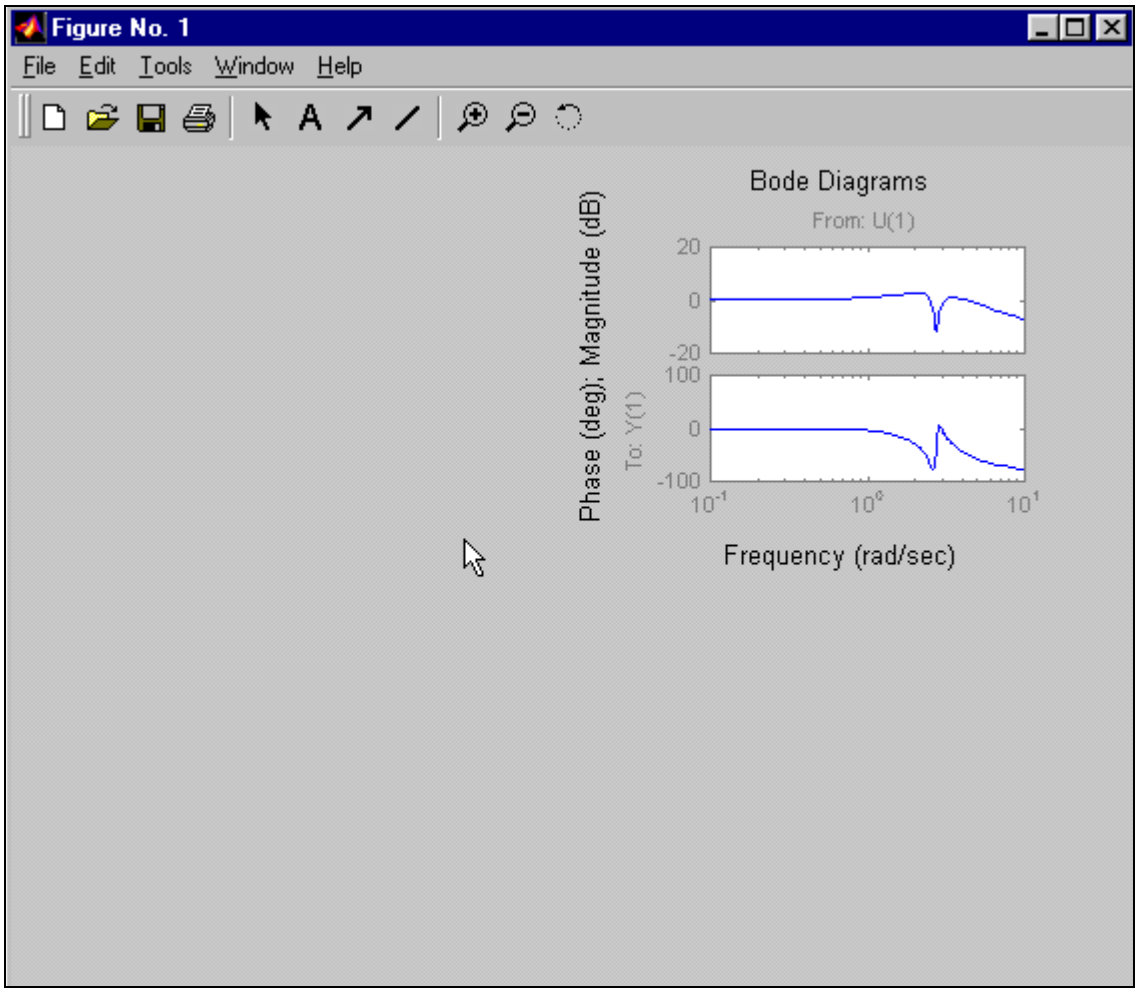



```
» h=tf([4 8.4 30.8 60],[1 4.12 17.4 30.8 60]);
» subplot(121)
» bode(h)
```

Kết quả:



» h=tf([4 8.4 30.8 60],[1 4.12 17.4 30.8 60]);
» subplot(222)
» bode(h)
Kết quả:




```
» h=tf([4 8.4 30.8 60],[1 4.12 17.4 30.8 60]);
» subplot(121)
» bode(h)
» subplot(222)
» bode(h)
» subplot(224)
» bode(h)
```

Kết quả:

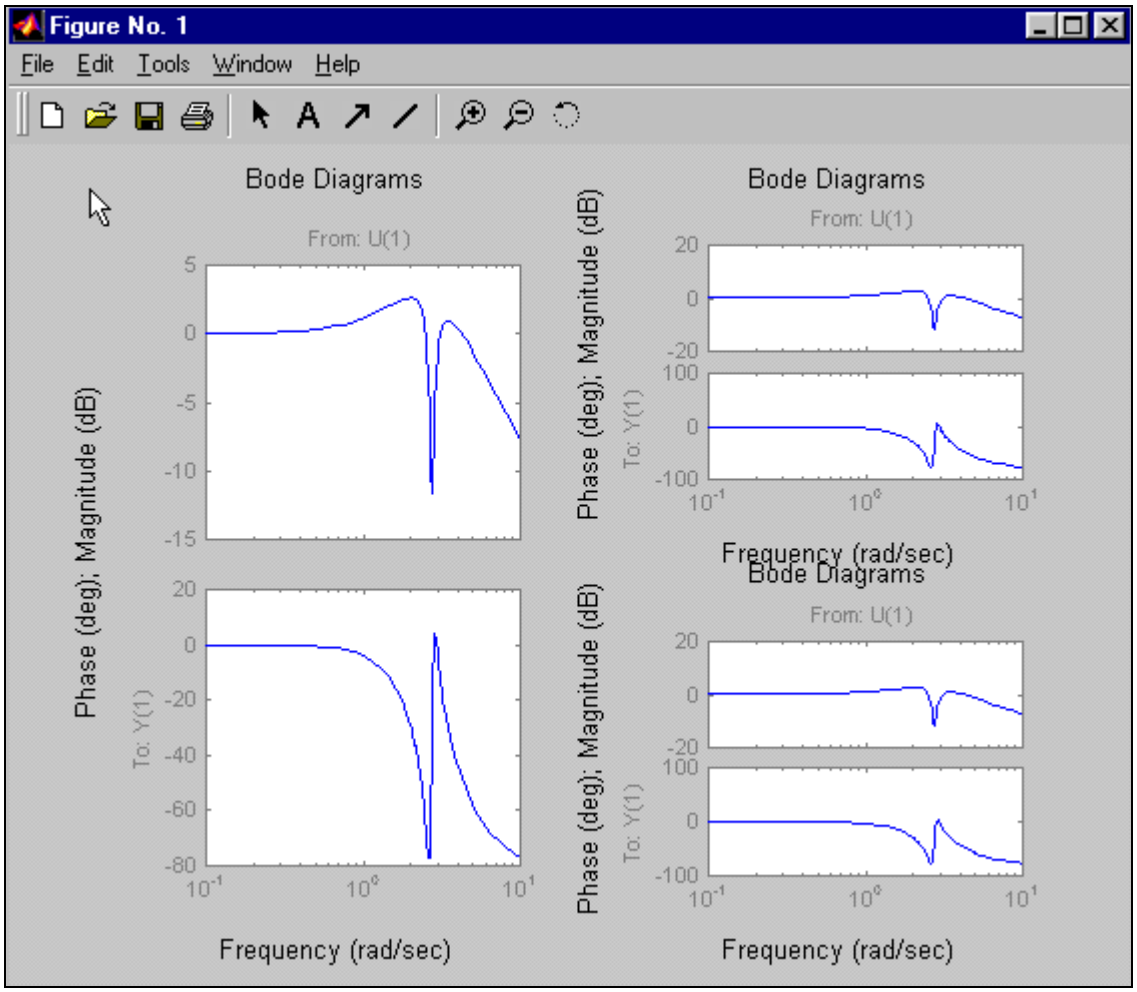


Figure No. 1
File Edit Tools Window Help
Bode Diagrams
From: U(1)
Phase (deg); Magnitude (dB)
To: Y(1)
Frequency (rad/sec)
Bode Diagrams
From: U(1)
Phase (deg); Magnitude (dB)
To: Y(1)
Frequency (rad/sec)
Bode Diagrams
From: U(1)
Phase (deg); Magnitude (dB)
To: Y(1)
Frequency (rad/sec)

Biểu đồ Nichols

Lý thuyết:

Công dụng: Để xác định độ ổn định và đáp ứng tần số vòng kín của hệ thống hồi tiếp ta sử dụng biểu đồ Nichols. Sự ổn định được đánh giá từ đường cong vẽ mối quan hệ của độ lợi theo đặc tính pha của hàm truyền vòng hở. Đồng thời đáp ứng tần số vòng kín của hệ thống cũng được xác định bằng cách sử dụng đường cong biên độ và độ di pha vòng kín không đổi phủ lên đường cong biên độ – pha vòng hở.

Cú pháp:

```
[mod,phase,puls]= nichols(A,B,C,D);
[mod,phase,puls]= nichols(A,B,C,D,ui);
[mod,phase]= nichols(A,B,C,D,ui,w);
[mod,phase,puls]= nichols(num,den);
[mod,phase]= nichols(num,den,w);
```

Những cấu trúc trên cho độ lớn là những giá trị tự nhiên, pha là độ và vectơ của diểm tần số là rad/s. Sự tồn tại của điểm tần số mà đáp ứng tần số được định giá bằng vectơ w, và ui là biến khai báo với hệ thống nhiều ngõ vào.

Chú ý:

+ khi sử dụng lệnh nichols với cấu trúc không có biến ngỏ ra thì ta được biểu đồ nichols.

+ lệnh nichols luôn luôn cho pha trong khoảng $[-360^0,0^0]$

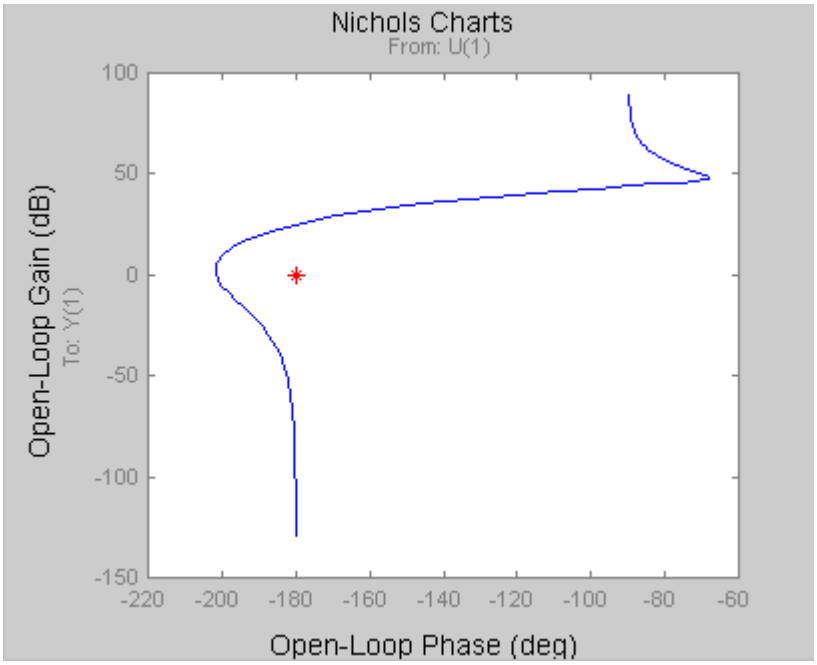
Bài 8: cho hệ thống có hàm truyền sau:

$$G(s) = 30\frac{s^2 + 7s + 1}{s(s+1)^3}$$

Các bước thực hiện:

```
» num=30*[1 7 1];
» den=[poly([-1 -1 -1]) 0];
» hold on, plot(-180,0,'*r'), hold on;
» nichols(num,den)
```

Trả về biểu đồ nichols với điểm tới hạn **"critical point"** $(-180^0,0)$ được biểu diễn như hình sau:



Hình: Biểu đồ Nichols

DẠNG BÀI TẬP VẼ BIỂU ĐỒ NYQUYST VÀ KHẢO SÁT ỔN ĐỊNH DÙNG GIẢN ĐỒ BODE

LÝ THUYẾT:

- **Hệ thống ổn định ở trạng thái hở, sẽ ổn định ở trạng thái kín nếu biểu đồ Nyquist không bao điểm (-1+i0) trên mặt phẳng phức.**
- **Hệ thống không ổn định ở trạng thái hở, sẽ ổn định ở trạng thái kín nếu biểu đồ Nyquist bao điểm (-1+i0)p lần ngược chiều kim đồng hồ (p là số cực GH nằm ở phải mặt phẳng phức).**

BÀI TẬP:

Từ dấu nhắc của cửa sổ MATLAB, ta nhập:

» num = [*nhập các hệ số của tử số theo chiều giảm dần của số mũ*].

» den = [*nhập các hệ số của mẩu số theo chiều giảm dần của số mũ*].

» nyquist(num,den)

Bài 9:

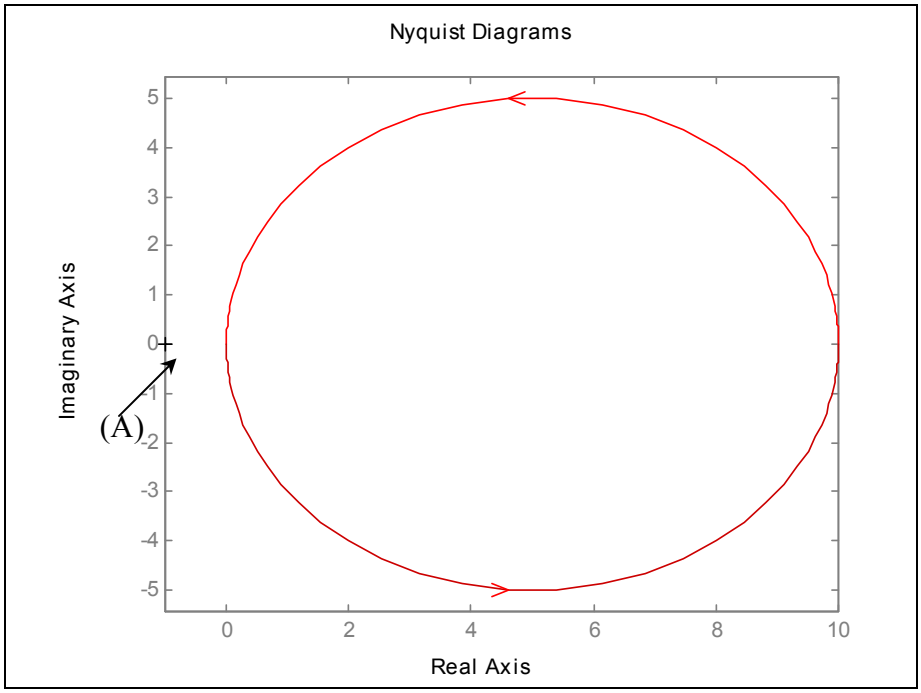
$$GH(s) = \frac{k}{1-st} \quad (\text{với } k = 10,\ t = 1)$$

» num = 10;

» den = [-1 1];

» nyquist(num,den)

Kết quả:



Nhận xét: hàm truyền vòng hở có 1 cực nằm bên phải mặt phẳng phức. Biểu đồ Nyquist không bao điểm A (-1+j0).

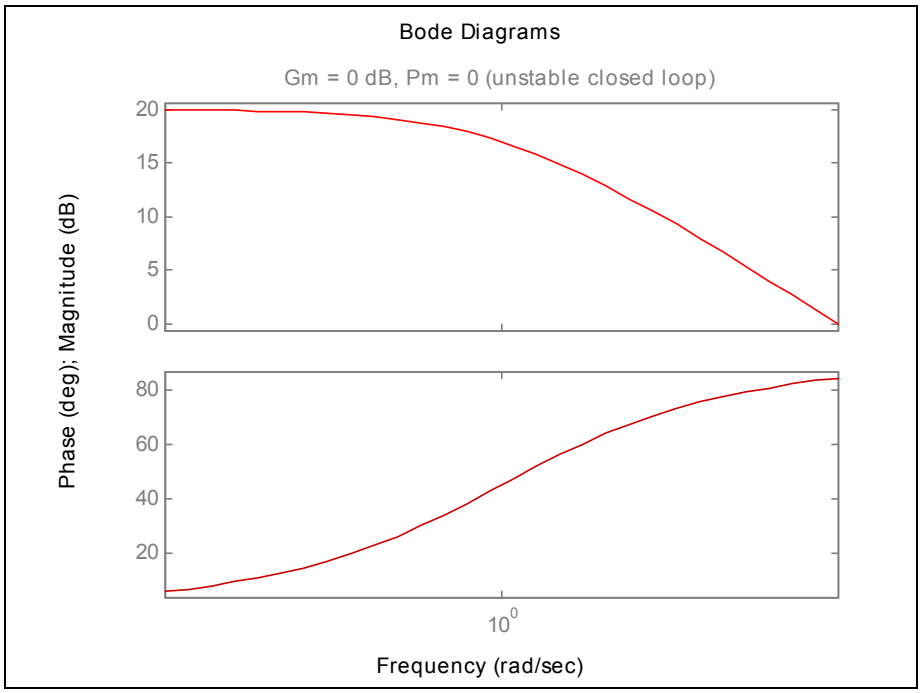
Điểm –1 ký hiệu (+) nằm trên trục thực âm (Real Axis), điểm 0 nằm trên trục ảo (Imaginary Axis).

Kết luận: hệ không ổn định.

*** Dùng lệnh margin để tìm biên dự trữ và pha dự trữ.**

Từ dấu nhắc của cửa sổ lệnh MATLAB ta dùng lệnh 'margin':

```
» num = 10;
» den = [-1 1];
» margin(num,den);
```



Kết luận:

Độ dự trữ biên (Gm = 0 dB).

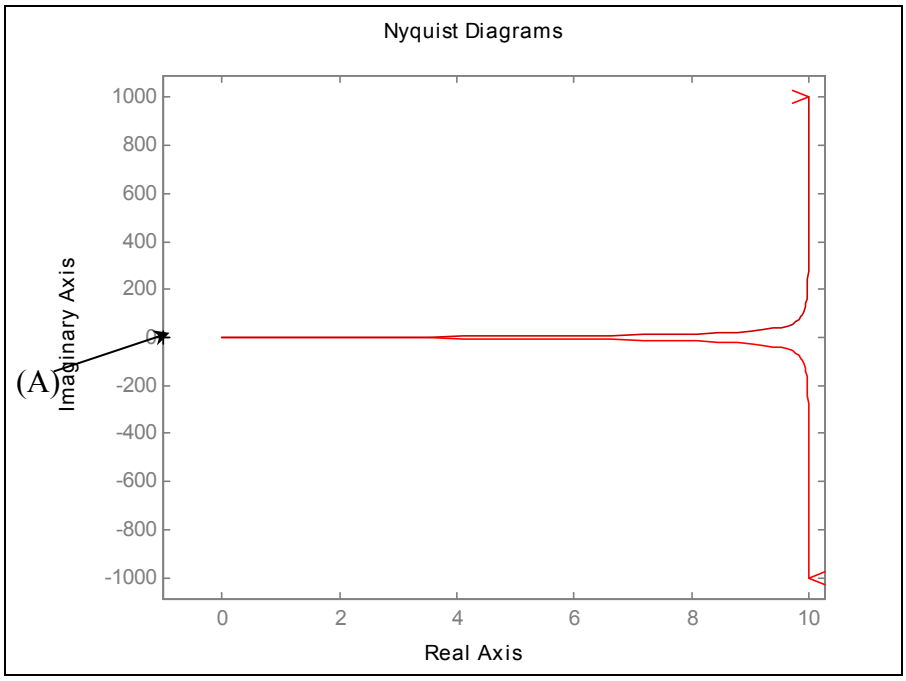
Độ dự trữ pha (Pm = 0°).

Warning: Closed loop is unstable (hệ vòng kín không ổn định).

Bài 10: Cho hàm ttuyền:

$$GH(s) = \frac{k}{s(1-st)} \quad (k = 10, t = 1)$$

```
» num = 10;
» den = [-1 1 0];
» nyquist(num,den)
```



Nhận xét: hàm truyền vòng hở có 1 cực nằm bên phải mặt phẳng phức và 1 cực nằm tại gốc tọa độ. Biểu đồ Nyquist không bao điểm A (-1+j0).

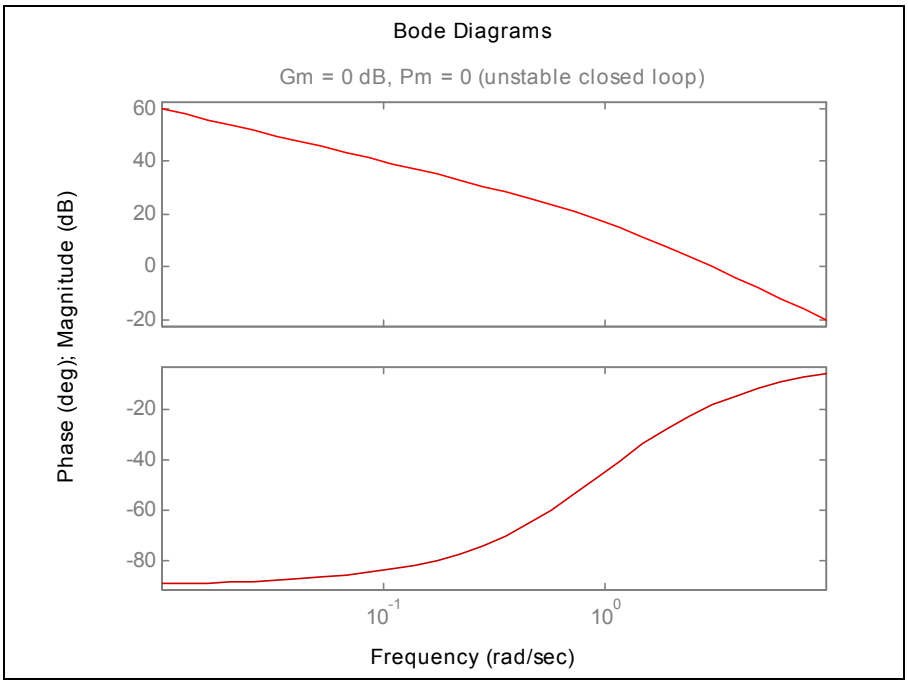
Điểm –1 ký hiệu (+) nằm trên trục thực âm (Real Axis) , điểm 0 nằm trên trục ảo (Imaginary Axis).

Kết luận: hệ không ổn định.

*** Dùng lệnh margin để tìm biên dự trữ và pha dự trữ.**

Từ dấu nhắc của cửa sổ lệnh MATLAB ta dùng lệnh 'margin':

```
» num = 10;
» den = [-1 1 0];
»margin(num,den)
```

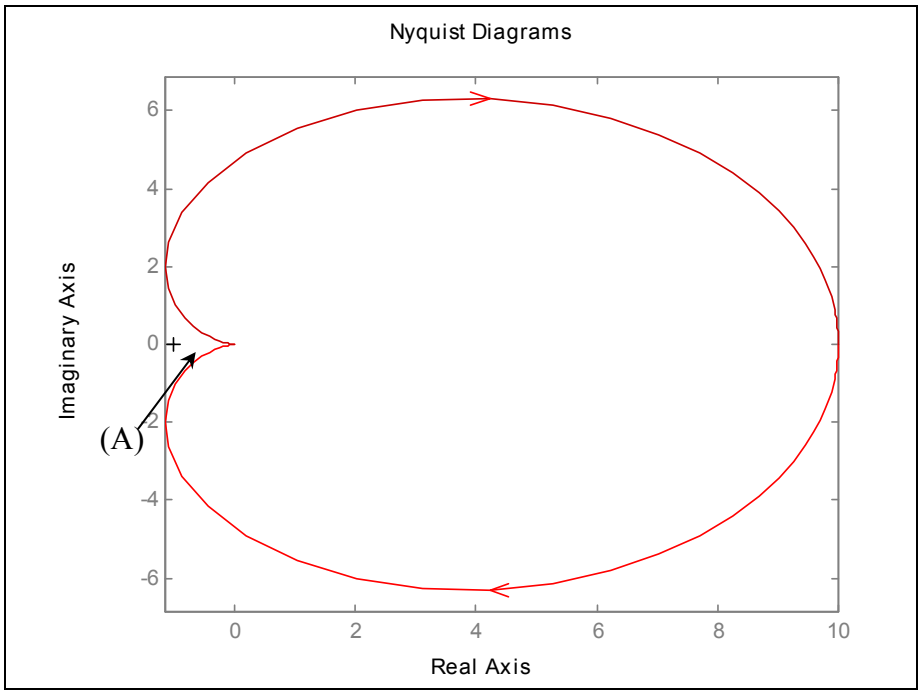


Kết luận:
Độ dự trữ biên (Gm = 0 dB).
Độ dự trữ pha (Pm = 0°).
Warning: Closed loop is unstable (hệ vòng kín không ổn định).

Bài 11: Cho hệ thống sau

$$GH(s) = \frac{k}{(t_1 s + 1)(t_2 s + 1)} \quad (k = 10,\ t_1 = 1,\ t_2 = 2)$$

» num = 10;
» den = [2 3 1];
» nyquist(num,den)



Nhận xét: hàm truyền vòng hở có 2 cực nằm bên trái mặt phẳng phức. Biểu đồ Nyquist không bao điểm A (-1+j0).

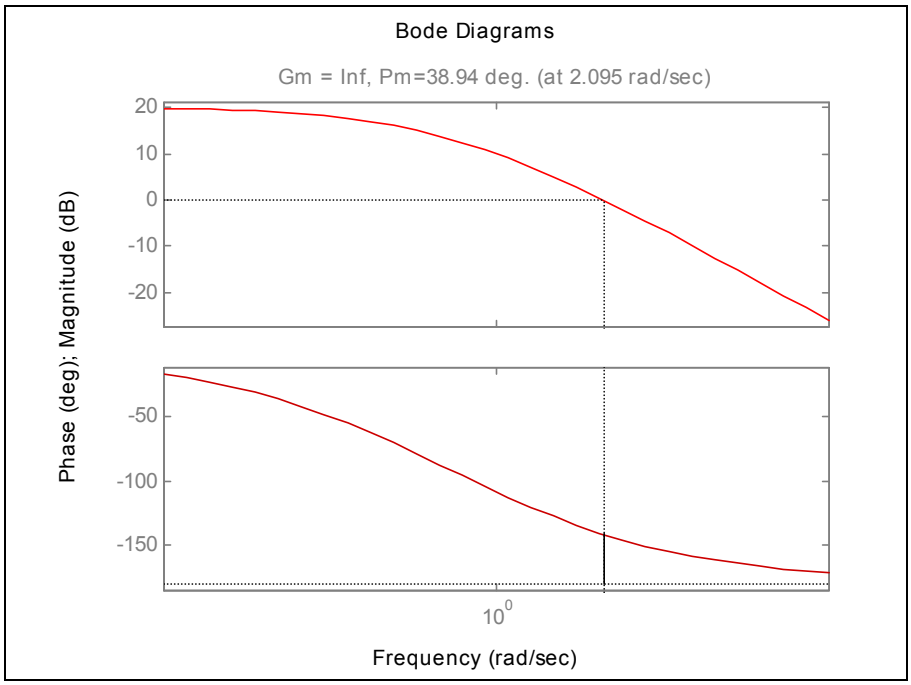
Điểm –1 ký hiệu (+) nằm trên trục thực âm (Real Axis) , điểm 0 nằm trên trục ảo (Imaginary Axis).

Kết luận: hệ thống ổn định.

*** Dùng lệnh margin để tìm biên dự trữ và pha dự trữ.**

Từ dấu nhắc của cửa sổ MATLAB dùng lệnh 'margin'.

```
» num = 10;
» den = [2 3 1];
» margin(num,den)
```

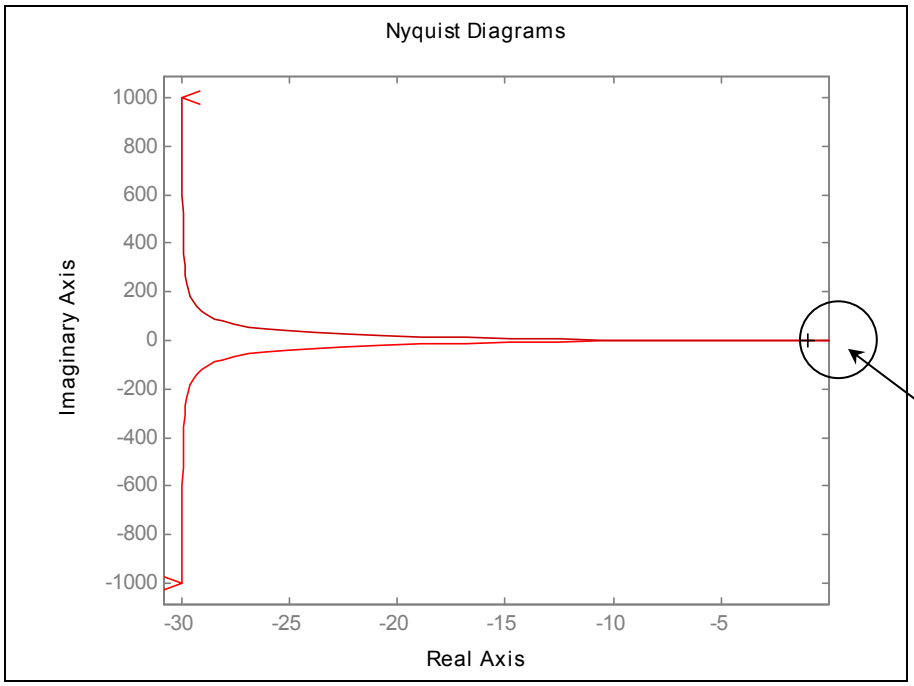


Kết luận: hệ thống ổn định.
Độ dự trữ biên (Gm = ∞).
Độ dự trữ pha (Pm = 38.94°), tại tần số cắt biên 2.095 rad/sec.

Bài 12: Cho hệ thống có hàm truyền sau:

$$GH(s) = \frac{k}{s(t_1 s+1)(t_2 s+1)} \qquad (k = 10 \quad t_1=1,\ t_2 =2)$$

```
» num = 10;
» den = [2 3 1 0];
» nyquist(num,den)
```



Nhận xét: hàm truyền vòng hở có 2 cực nằm bên trái mặt phẳng phức và 1 cực ở zero. Biểu đồ Nyquist bao điểm A(-1+j0).

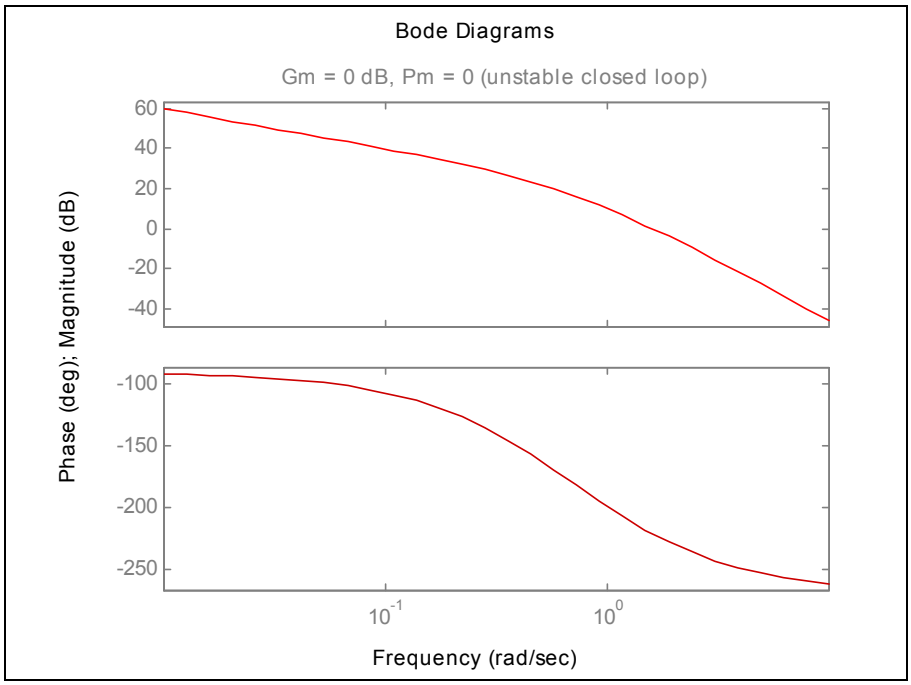
Điểm –1 ký hiệu (+) nằm trên trục thực âm (Real Axis) , điểm 0 nằm trên trục ảo (Imaginary Axis).

Kết luận: hệ không ổn định.

*** Dùng lệnh margin để tìm biên dự trữ và pha dự trữ.**

Từ dấu nhắc của cửa sổ MATLAB ta dùng lệnh 'margin' để kiểm chứng lại hệ:

```
» num = 10;
» den = [2 3 1 0];
»margin(num,den)
```

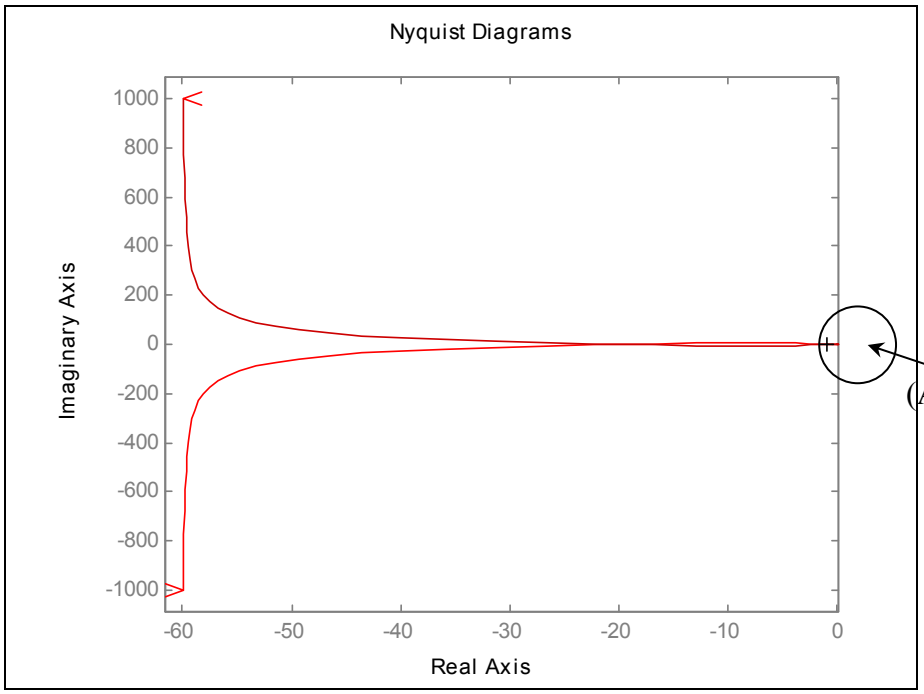


Kết luận: hệ thống không ổn định.
Độ dự trữ biên (Gm = 0 dB).
Độ dự trữ pha (Pm = 0°)

Bài 12:

$$GH(s) = \frac{k}{s(t_1 s+1)(t_2 s+1)(t_3 s+1)} \qquad (t_1 = 1,\ t_2 = 2,\ t_3 = 3,\ k = 10)$$

```
» num = 10;
» den = [6 11 6 1 0];
» nyquist(num,den)
```



Nhận xét: hàm truyền vòng hở có 3 cực nằm bên trái mặt phẳng phức và 1 cực ở zero. Biểu đồ Nyquist bao điểm A (-1+i0).

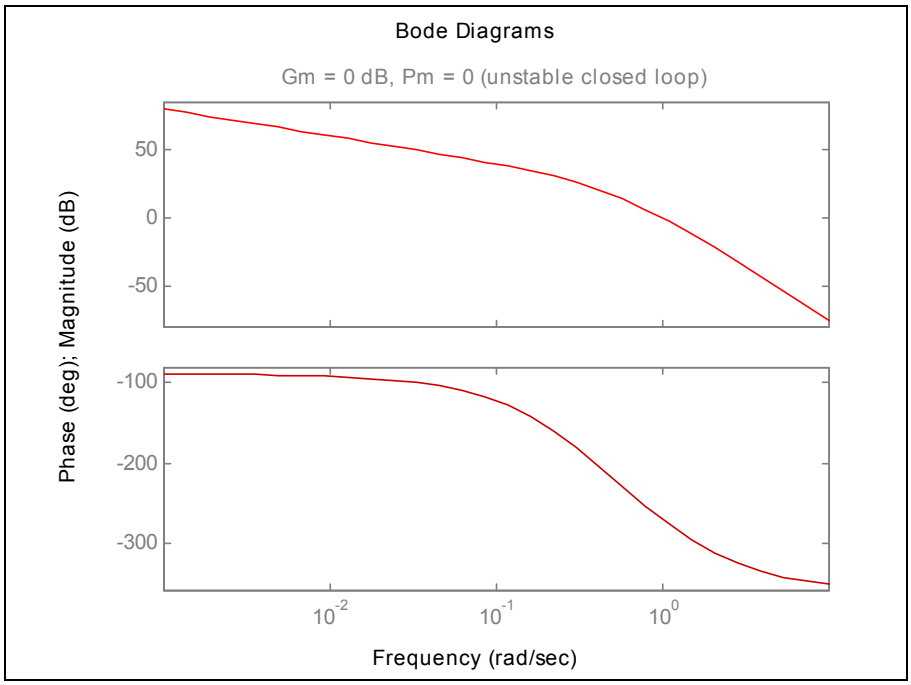
Điểm –1 ký hiệu (+) nằm trên trục thực âm (Real Axis) , điểm 0 nằm trên trục ảo (Imaginary Axis).

Kết luận: hệ không ổn định.

*** Dùng lệnh margin để tìm biên dự trữ và pha dự trữ.**

Từ dấu nhắc của cửa sổ MATLAB, dùng lệnh 'margin' để kiểm chứng lại hệ:

```
» num = 10;
» den = [6 11 6 1 0];
» margin(num,den)
```



Kết luận: hệ thống không ổn định.
Độ dự trữ biên (Gm = 0 dB).
Độ dự trữ pha (Pm = 0°).

NHÓM LỆNH VỀ QUỸ ĐẠO NGHIỆM
(Roots Locus)

1. Lệnh PZMAP

a) Công dụng:

Vẽ biểu đồ cực-zero của hệ thống.

b) Cú pháp:

[p,z]= pzmap(num,den)

[p,z]= pzmap(a,b,c,d)

[p,z]= pzmap(a,b,c,d)

c) Giải thích:

Lệnh pzmap vẽ biểu đồ cực-zero của hệ LTI. Đối với hệ SISO thì các cực và zero của hàmtruyền được vẽ.

Nếu bỏ qua các đối số ngõ ra thì lệnh pzmap sẽ vẽ ra biểu đồ cực-zero trên màn hình.

pzmap là phương tiện tìm ra các cực và zero tuyền đạt của hệ MIMO.

pzmap(a,b,c,d) vẽ các cực và zero của hệ không gian trạng thái trong mặt phẳng phức. Đối với các hệ thống MIMO, lệnh sẽ vẽ tất cả các zero truyền đạt từ tất cả các ngõ vào tới tất cả các ngõ ra. Trong mặt phẳng phức, các cực được biểu diễn bằng dấu × còn các zero được biểu diễn bằng dấu o.

pzmap(num,den) vẽ các cực và zero của hàm truyền trong mặt phẳng phức. Vector num và den chứa các hệ số tử số và mẫu số theo chiều giảm dần số mũ của s.

pzmap(p,z) vẽ các cực và zero trong mặt phẳng phức. Vector cột p chứa tọa độ các cực và vector cột z chứa tọa độ các zero trong mặt phẳng phức. Lệnh này vẽ các cực và zero đã được tính sẵn trong mặt phẳng phức.

Nếu giữ lại các đối số ngõ ra thì :

[p,z]= pzmap(num,den)

[p,z]= pzmap(a,b,c,d)

[p,z]= pzmap(a,b,c,d)

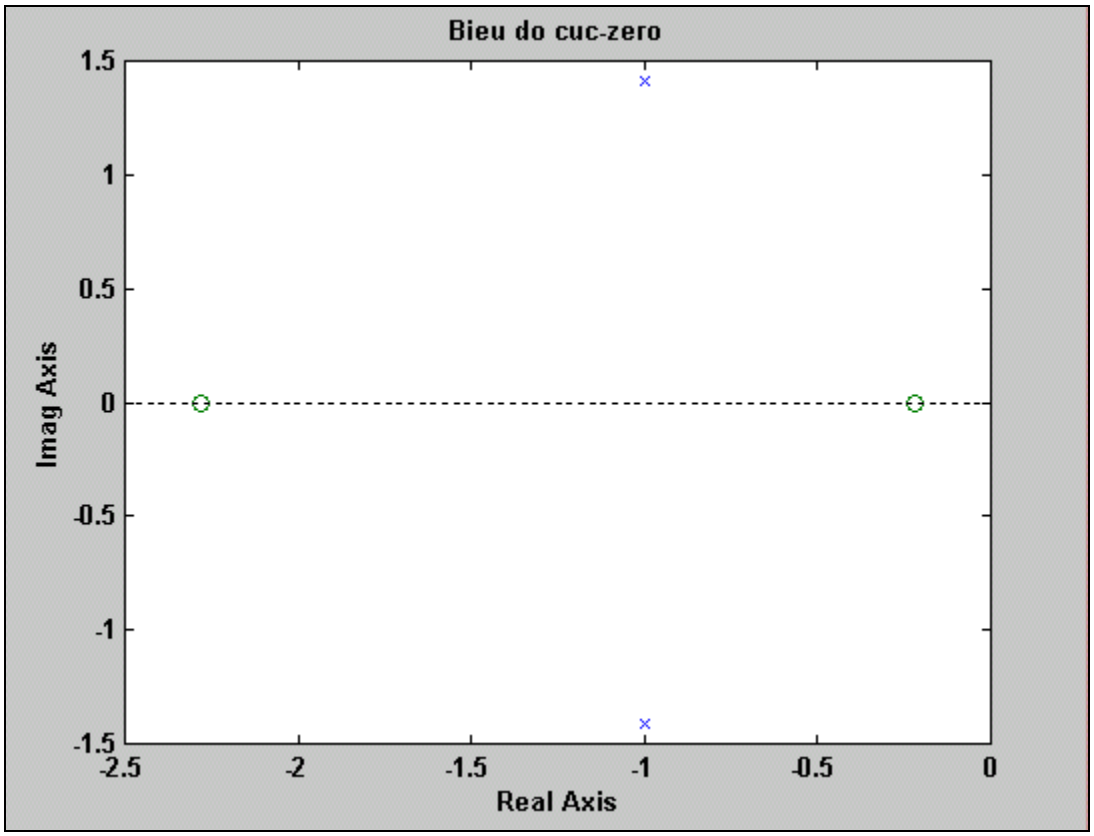
tạo ra các ma trận p và z trong đó p chứa các cực còn z chứa các zero.

d) Ví dụ: (Trích trang 11-174 sách '**Control system Toolbox'**)

Vẽ các cực và zero của hệ liên tục có hàm truyền :

$$H(s) = \frac{2s^2 + 5s + 1}{s^2 + 2s + 3}$$

```
num = [2   5   1];
den  = [1   2   3];
pzmap(num,den)
title('Bieu do cuc-zero')
```



2. Lệnh RLOCFIND

a) Công dụng:

Tìm độ lợi quỹ đạo nghiệm với tập hợp nghiệm cho trước.

b) Cú pháp:

[k,poles]= rlocfind(a,b,c,d)

[k,poles]= rlocfind(num,den)

[k,poles]= rlocfind(a,b,c,d,p)

[k,poles]= rlocfind(num,den,p)

c) Giải thích:

Lệnh rlocfind tạo ra độ lợi quỹ đạo nghiệm kết hợp với các cực trên quỹđạo nghiệm. Lệnh rlocfind được dùng cho hệ SISO liên tục và gián đoạn.

[k,poles]= rlocfind(a,b,c,d) tạo ra dấu x trong cửa sổ đồ họa mà ta dùng để chọn một điểm trên quỹ đạo nghiệm có sẵn. Độ lợi của điểm này được tạo ra trong k và các cực ứng với độ lợi này nằm trong poles. Để sử dụng lệnh này thì quỹ đạo nghiệm phải có sẵn trong cửa sổ đồ họa.

[k,poles]= rlocfind(num,den) tạo ra dấu x trong cửa sổ đồ họa mà ta dùng để chọn một điểm trên quỹ đạo nghiệm của hệ thống có hàm truyền G = num/den trong đó có num và den chứa các hệ số đa thức theo chiều giảm dần số mũ của s hoặc z.

[k,poles]= rlocfind(a,b,c,d,p) hoặc [k,poles]= rlocfind(num,den,p) tạo ra vector độ lợi k và vector các cực kết hợp pole với mỗi thành phần trong mỗi vector ứng với mỗi nghiệm trong p.

d) Ví dụ: (Trích từ trang 11-180 sách '**Control System Toolbox**')

Xác định độ lợi hồi tiếp để các cực vòng kín của hệ thống có hệ số tắt dần $\zeta = 0.707$ và có hàm truyền :

$$H(s) = \frac{2s^2 + 5s + 1}{s^2 + 2s + 3}$$

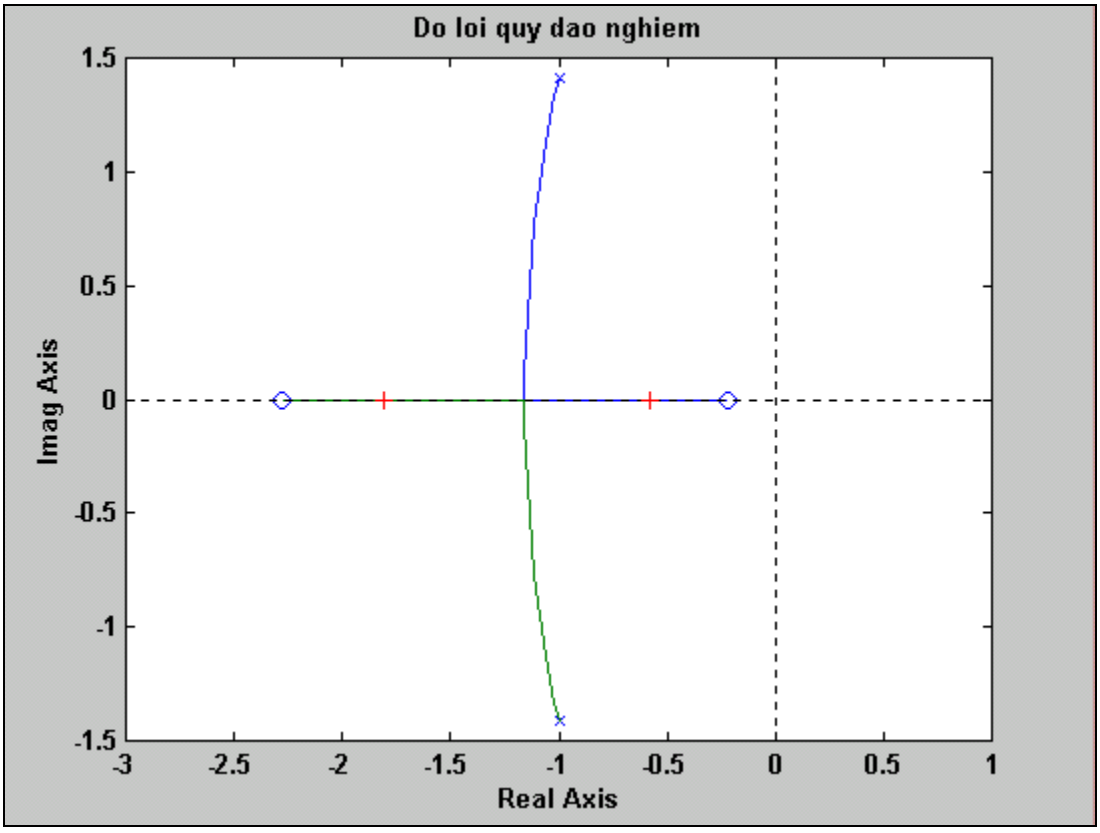
```
num = [2   5   1];
den  = [1   2   3];
% Vẽ quỹ đạo nghiệm:
rlocus(num,den);title('Do loi quy dao nghiem');
% Tìm độ lợi tại điểm được chọn:
rlocfind(num,den);
```

Sau khi nhập xong lệnh, trên màn hình của Matlab sẽ xuất hiện dòng chữ:

Select a point in the graphics window

và trên hình vẽ có thước để ta kéo chuột và chọn điểm

ta có quỹ đạo nghiệm:



3. Lệnh RLOCUS

a) Công dụng:

Tìm quỹ đạo nghiệm Evans.

b) Cú pháp:

r = rlocus(num,den)
r = rlocus(num,den,k)
r = rlocus(a,b,c,d)
r = rlocus(a,b,c,d,k)

c) Giải thích:

Lệnh rlocus tìm quỹ đạo nghiệm Evans của hệ SISO. Quỹ đạo nghiệm được dùng để nghiên cứu ảnh hưởng của việc thay đổi độ lợi hồi tiếp lên vị trí cực của hệ thống, cung cấp các thông tin về đáp ứng thời gian và đáp ứng tần số. Đối với đối tượng điều khiển có hàm truyền G(s) và khâu bổ chính hồi tiếp k*f(s), hàm truyền vòng kín là :

$$h(s) = \frac{g(s)}{1 + kg(s)f(s)} = \frac{g(s)}{q(s)}$$

Nếu bỏ qua các đối số ngõ ra thì lệnh rlocus sẽ vẽ ra quỹ đạo trên màn hình. Lệnh rlocus dùng cho cả hệ liên tục và gián đoạn.

r = rlocus(num,den) vẽ quỹ đạo nghiệm của hàm truyền :

$$\text{q(s)} = 1 + \text{k}\frac{num(s)}{den(s)} = 0$$

với vector độ lợi k được xác định tự động. Vector num và den chỉ ra hệ tử số và mẫu số theo chiều giảm dần số của s hoặc z.

$$\frac{num(s)}{den(s)} = \frac{num(1)s^{nn-1} + num(2)s^{nn-2} + ... + num(nn)}{den(1)s^{nd-1} + den(2)s^{nd-2} + ... + den(nd)}$$

r = rlocus(a,b,c,d) vẽ ra quỹ đạo nghiệm của hệ không gian trạng tái SISO liên tục và gián đoạn với vector độ lợi được xác định tự động

r = rlocus(num,den,k) hoặc r = rlocus(a,b,c,d,k) vẽ ra quỹ đạo nghiệm với vector độ lợi k do người sử dụng xác định. Vector k chứa các giá trị và độ lợi mà nghiệm hệ vòng kín được tính.

Nếu sử dụng các đối số ngõ ra thì :

[r,k] = rlocus(num,den)
[r,k] = rlocus(num,den,k)
[r,k] = rlocus(a,b,c,d)
[r,k] = rlocus(a,b,c,d,k)

tạo ra ma trận ngõ ra chứa các nghiệm và vector độ lợi k. Ma trận r có length(k) hàng và (length(den) –1) cột, ngõ ra chứa vị trí các nghiệm phức. Mỗi hàng trong ma trận tương ứng với một độ lợi trong vector k. Quỹ đạo nghiệm có thể được vẽ bằng lệnh plot(r,'x').

d) Ví dụ: (Trích từ trang 11-183 sách **'Control System Toolbox'**)

Tìm và vẽ quỹ đạo nghiệm của hệ thống có hàm truyền :

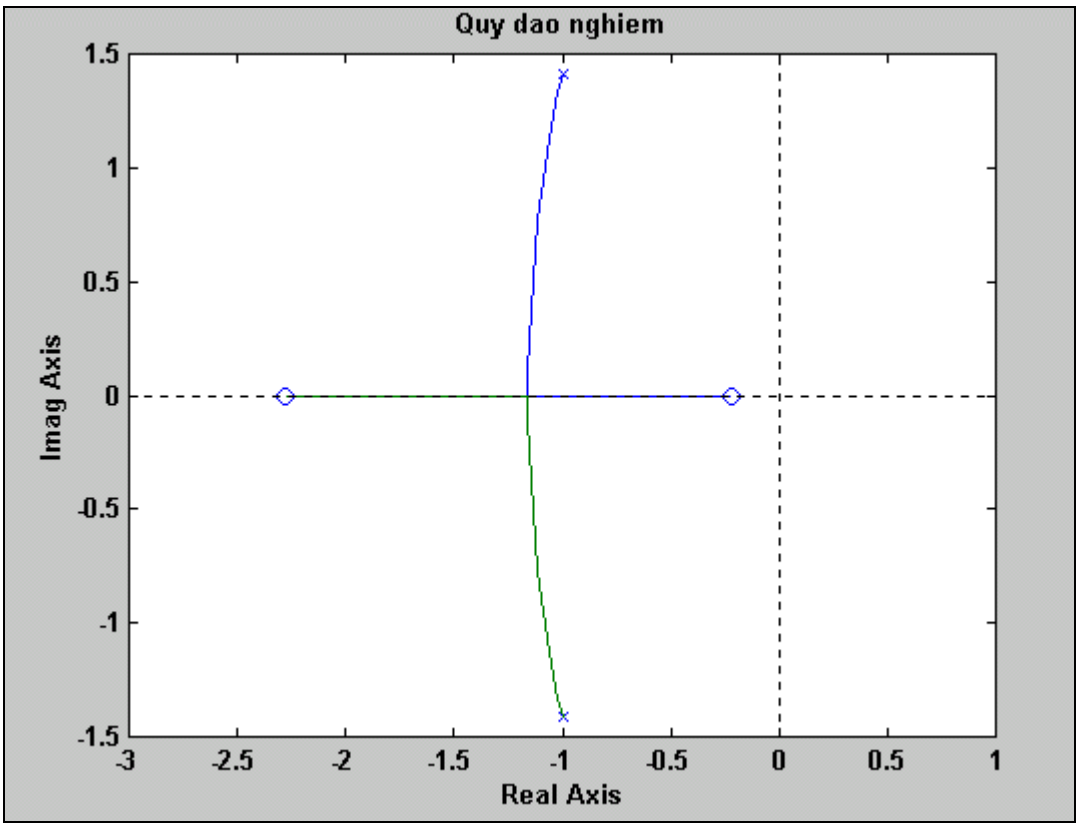
$$H(s) = \frac{2s^2 + 5s + 1}{s^2 + 2s + 3}$$

% Xác định hàm truyền :

```
num = [2   5   1];
den  = [1   2   3];
```

% Vẽ quỹ đạo nghiệm :

```
rlocus(num,den)
title('Quy dao nghiem')
```



4. <u>Lệnh SGRID</u>

a) <u>Công dụng</u>:

Tạo lưới cho quỹ đạo nghiệm và biểu đồ cực-zero liên tục.

b) <u>Cú pháp</u>:

```
sgrid
sgrid('new')
sgrid(z,wn)
sgrid(z,wn,'new')
```

c) Giải thích:

Lệnh sgrid tạo lưới cho quỹ đạo nghiệm và biểu đồ cực-zero liên tục trong mặt phẳng s. Đường lưới vẽ là các đường hằng số tỉ số tắt dần (ζ) và tần số tự nhiên (ω_n). Đường tỉ số tắt dần được vẽ từ 0 tới 1 theo từng nấc là 0.1.

sgrid('new') xóa màn hình đồ họa trước khi vẽ và thiết lập trạng thái hold on để quỹ đạo nghiệm hay biểu đồ cực-zero được vẽ lên lưới bằng các lệnh :

sgrid('new')

rlocus(num,den) hoặc pzmap(num,den)

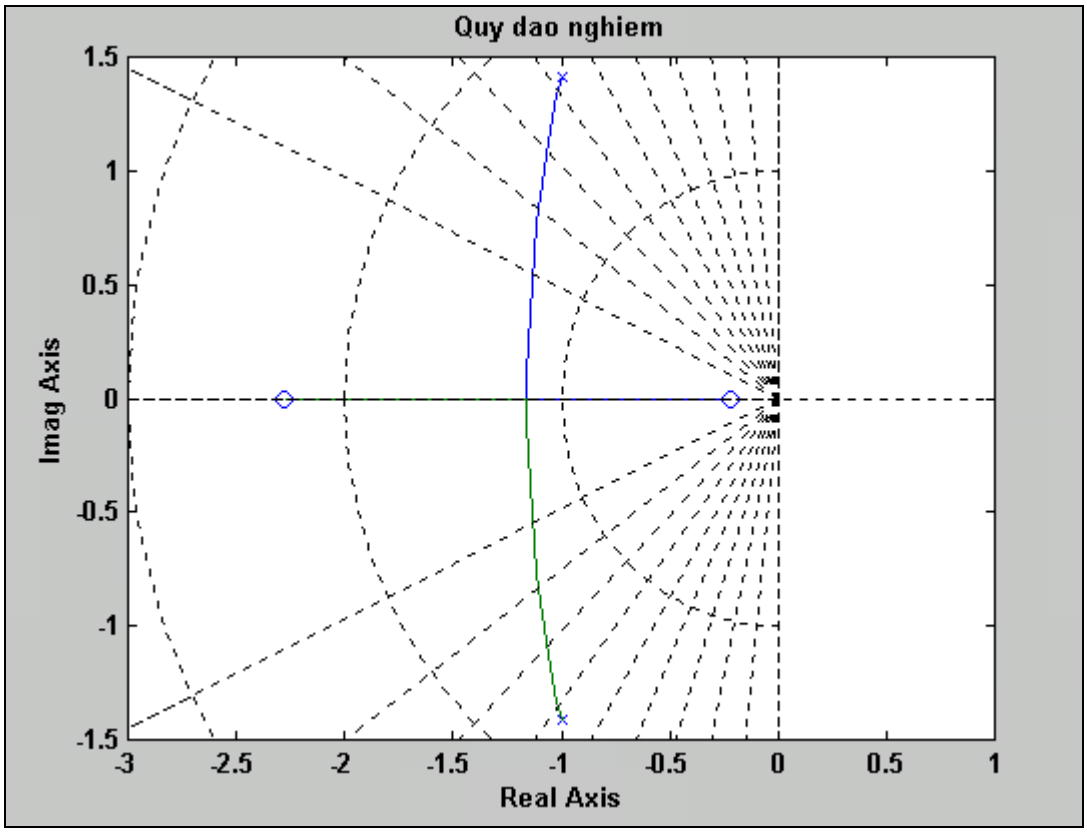
sgrid(z,wn) vẽ các đường hằng số tỉ lệ tắt dần được chỉ định trong vector z và vẽ đường tần số tự nhiên được chỉ định trong vector wn.

sgrid(z,wn,'new') xóa màn hình đồ họa trước khi vẽ các đường tỉ số tắt dần và tần số tự nhiên được chỉ định trong vector z và wn. Trạng thái hold on được thiết lập.

d) Ví dụ: Trích từ trang 11-200 sách **'Control System Toolbox'**

Vẽ lưới trong mặt phẳng s trên quỹ đạo nghiệm của hệ thống có hàm truyền :

```
num = [2   5   1];  % ta có thể thay đổi 2 dòng num=..., den=... thành dòng lệnh sau:
den  = [1   2   3];  %  H(s)=tf([2 5 1],[1 2 3]);
rlocus(num,den)
title('Quy dao nghiem')
sgrid
```



5. Lệnh ZGRID

a) Công dụng:

Vẽ lưới tỉ lệ tắt dần và tần số tự nhiên cho quỹ đạo nghiệm gián đoạn.

b) Cú pháp:

```
zgrid
zgrid('new')
zgrid(z,wn)
zgrid(z,wn,'new')
```

c) Giải thích:

Lệnh zgrid tạo lưới quỹ đạo cho nghiệm hoặc biểu đồ cực-zero trong mặt phẳng z. Các đường hằng số tỉ lệ tắt dần (ζ) và tần số tự nhiên chuẩn hóa sẽ được vẽ. ζ được thay đổi từ 0 tới 1 theo từng nấc thay đổi là 0.1 và tần số tự nhiên được vẽ từ 0 tới π với từng nấc thay đổi là π/ω.

zgrid('new') xóa màn hình đồ họa trước khi vẽ lưới và thiết lập trạng thái hold on để quỹ đạo nghiệm hoặc biểu đồ cực-zero được vẽ lên lưới sử dụng các lệnh :

```
zgrid('new')
rlocus(num,den) hoặc pzmap(num,den)
```

zgrid(z,wn) vẽ hằng số tắt dần được chỉ định trong vector z và vẽ hằng số tần số tự nhiên cho các tần số chuẩn hóa được chỉ định trong vector wn. Các tần số chuẩn hóa có thể được vẽ bằng lệnh zgrid(z,wn/Ts) với tần số là thời gian lấy mẫu.

zgrid(z,wn,'new') xóa màn hình đồ họa trước khi vẽ tỉ số tắt dần và tần số tự nhiên được chỉ định trong vector z và wn. Trạng thái hold on được thiết lập.

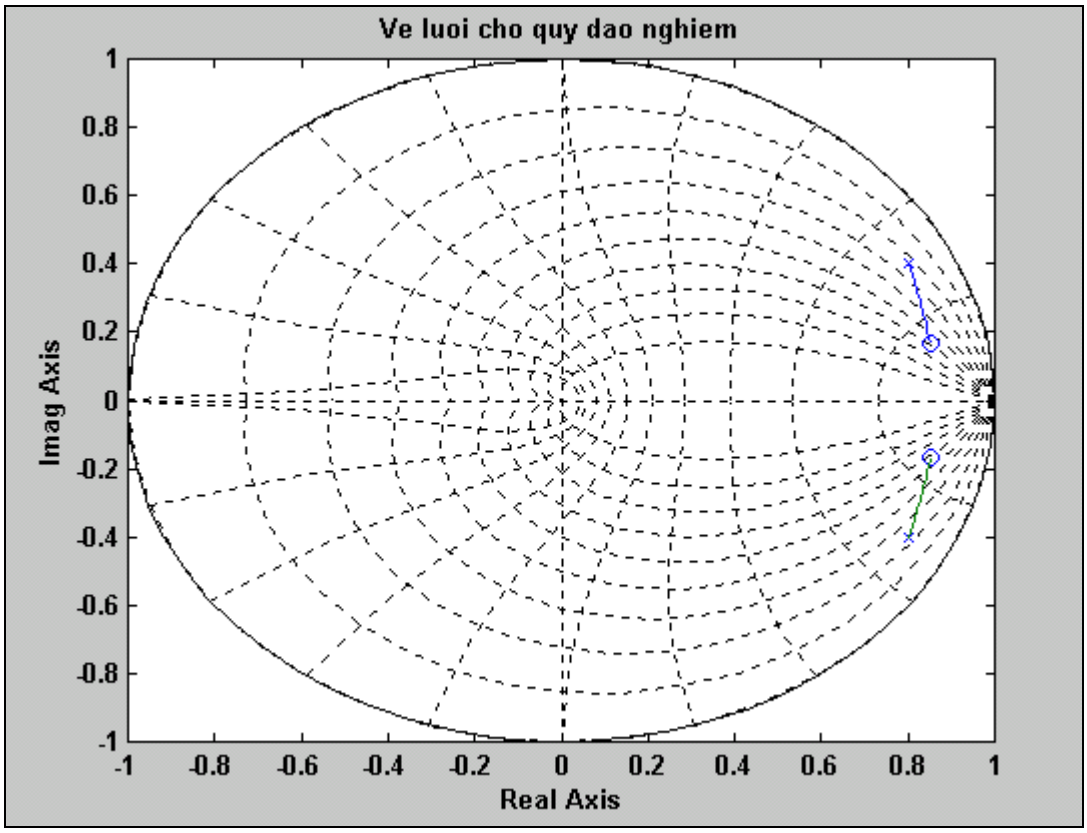
zgrid([ ],[ ]) sẽ vẽ ra vòng tròn đơn vị.

d) Ví dụ: Trích từ 11-236 sách **'Control System Toolbox'**

Vẽ lưới trong mặt phẳng cho quỹ đạo nghiệm của hệ thống có hàm truyền :

$$H(z) = \frac{2z^2 - 3.4z + 1.5}{z^2 - 1.6s + 0.8}$$

```
num = [2    -3.4    1.5];
den  = [1    -1.6    0.8];
axis('square')
zgrid('new')
rlocus(num,den)
title('Ve luoi cho quy dao nghiem')
```

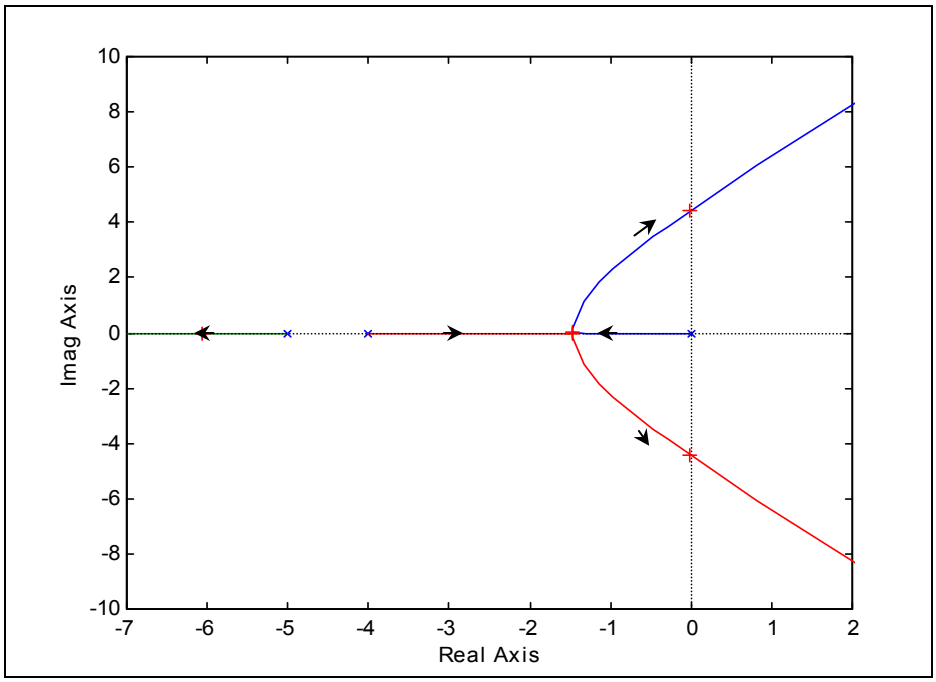

Ve luoi cho quy dao nghiem
Imag Axis
Real Axis
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1

CÁC BÀI TẬPVỀ QUỸ ĐẠO NGHIỆM

Bài 1:

$$KGH = \frac{k}{s(s+4)(s+5)} \quad \text{với } k = 2$$

» num = 2;
» den = [1 9 20 0];
» rlocus(num,den)



Từ đồ thị cho ta:

1. Điểm cực: 0 ,-4,-5.
2. Quỹ đạo nghiệm có 3 nhánh.
3. Điểm zero ở vô cùng (∞).
4. Điểm tách được xác định bằng cách từ cửa sổ MATLAB ta nhập:

» num = 2;
» den = [1 9 20 0];
» rlocus(num,den);
» rlocfind(num,den)

Sau khi nhập lệnh thì trên cửa sổ lệnh sẽ xuất hiện hàng chữ:

Select a point in the graphics window (hãy chọn 1 điểm trên đồ thị minh họa).

Trên đồ thị sẽ có thước cho ta chọn điểm – kéo rê chuột để chọn điểm cần chọn.

selected_point = -1.4516

Điểm tách có giá trị: -1.4516

Giao điểm của quỹ đạo nghiệm với trục ảo (tương tự như tìm điểm tách): +4.472j, -4.472j.

Từ giá trị tại giao điểm của quỹ đạo nghiệm với trục ảo ta thế vào phương trình đặc trưng:

$F(s) = s^3+9s^2+20s+k = 0$

$F(jw) = -jw^3-9w^2+20jw+k = 0$

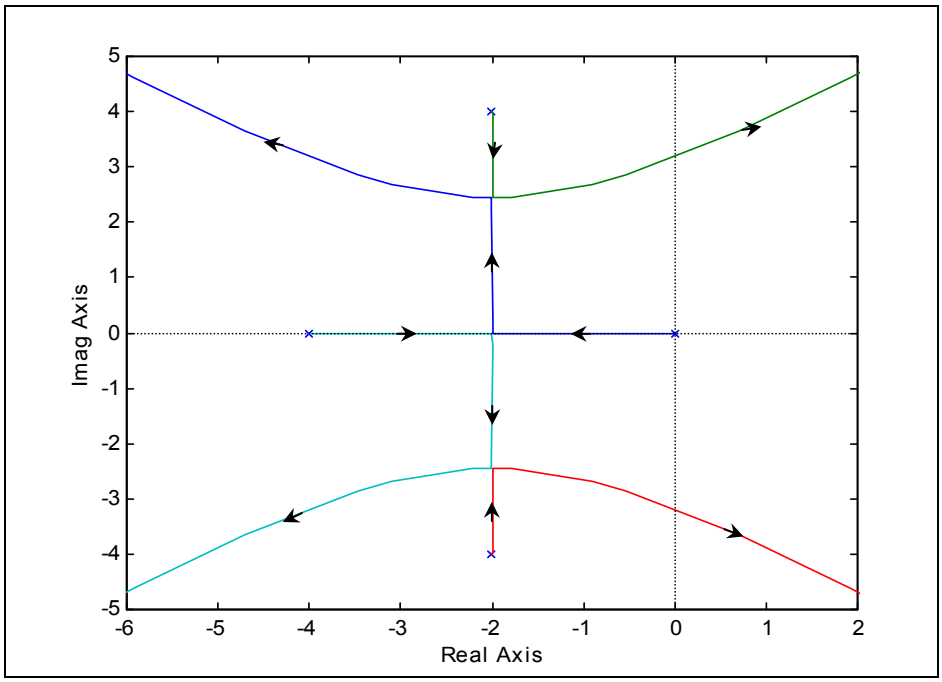
$\Rightarrow$ **$k_{gh} = 180$**

Kết luận: hệ thống sẽ ổn định khi $0 < k < 180$

Bài 2:

$$KGH = \frac{k}{s(s+4)(s^2+4s+20)} \qquad (k = 2)$$

```
» num = 2;
» den = [1 8 36 80 0];
» rlocus(num,den)
```



Từ đồ thị cho ta:

1. Điểm cực: 0,-4,-2+4j,-2-4j;
2. Quỹ đạo nghiệm có 4 nhánh
3. Điểm zero ở vô hạn (∞)

4. Điểm tách được xác định bằng cách từ cửa sổ MATLAB ta nhập:

```
» num = 2;
» den = [1 8 36 80 0];
» rlocus(num,den);
» rlocfind(num,den);
```

Sau khi nhập lệnh thì trên cửa sổ lệnh sẽ xuất hiện hàng chữ:

Select a point in the graphics window (hãy chọn 1 điểm trên đồ thị minh họa).

Trên đồ thị sẽ có thước cho ta chọn điểm – kéo rê chuột để chọn điểm cần chọn.

selected_point = -2, -2.0184 + 2.4561j, -2.0184 - 2.4561j

Điểm tách có giá trị: -2, -2.0184 + 2.4561j, -2.0184 - 2.4561j

Giao điểm của quỹ đạo nghiệm với trục ảo (tương tự như tìm điểm tách): +3.16j, -3.16j

Từ giá trị tại giao điểm của quỹ đạo nghiệm với trục hoành ta thế vào phương trình đặc trưng:

$$F(jw) = w^4 - 8jw^3 - 36w^2 + 80jw + k$$

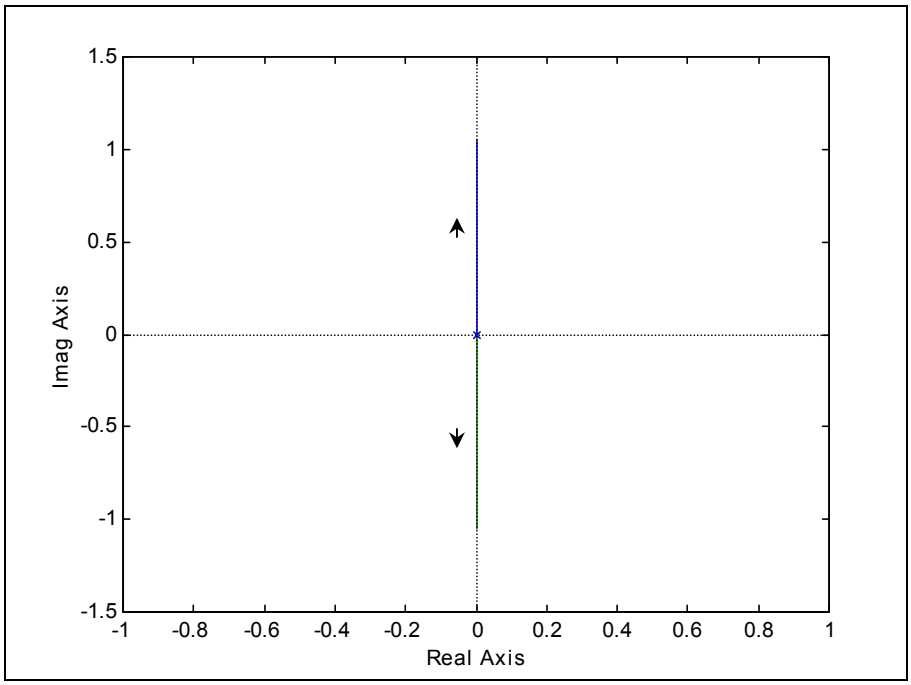
$$\Rightarrow \underline{k_{gh} = 260}$$

Kết luận : hệ thống sẽ ổn định khi $0 < k < 260$

Bài 3:

$$KGH = \frac{k}{s^2} \quad (k = 2)$$

```
» num = 2;
» den =[1 0 0 ];
» rlocus(num,den)
```



Từ đồ thị ta có:

1. Điểm cực : 0
2. Quỹ đạo nghiệm có 2 nhánh
3. Điểm zero ở vô hạn (∞)

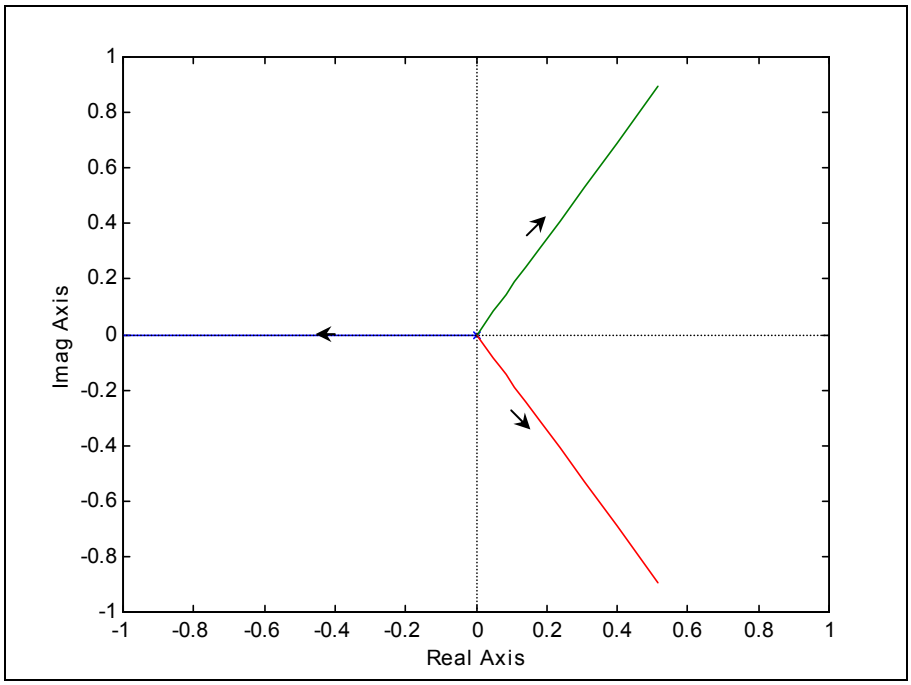
Điểm tách có giá trị: 0

Kết luận: hệ thống không ổn định.

Bài 4:

$$KGH = \frac{k}{s^3}$$

```
» num = 2;
» den =[1 0 0 0];
» rlocus(num,den)
```



Từ đồ thị ta có:

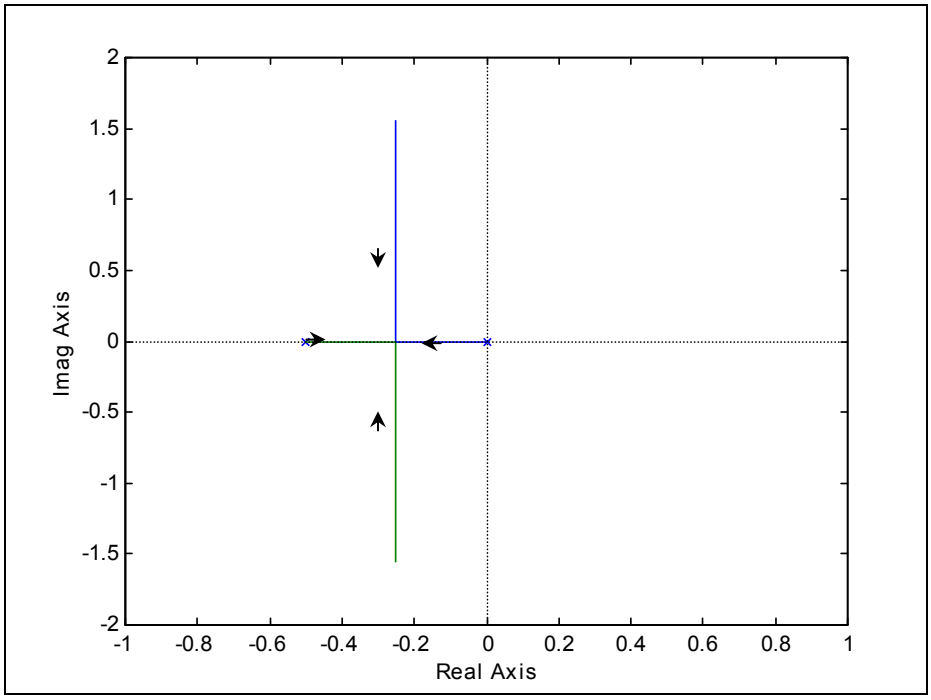
1. Điểm cực: 0.
2. Quỹ đạo nghiệm có 3 nhánh.
3. Điểm zero ở vô hạn (∞).
4. Điểm tách có giá trị: 0

Kết luận: hệ thống không ổn định (vì hai nhánh của quỹ đạo nghiệm số luôn nằm nửa phải mặt phẳng phức).

Bài 5:

$$KGH = \frac{k}{s(ts+1)} \quad (k = 1, t = 2)$$

» num = 1;
» den = [2 1 0];
» rlocus(num,den)



1. Điểm cực : 0,-0.5
2. Quỹ đạo nghiệm có 2 nhánh
3. Điểm zero ở vô hạn (∞)
4. Điểm tách được được xác định bằng cách từ cửa sổ MATLAB ta nhập:

» num = 1;
» den = [2 1 0];
» rlocus(num,den);
» rlocfind(num,den)

Sau khi nhập lệnh thì trên cửa sổ lệnh sẽ xuất hiện hàng chữ:
Select a point in the graphics window (hãy chọn 1 điểm trên đồ thị minh họa).

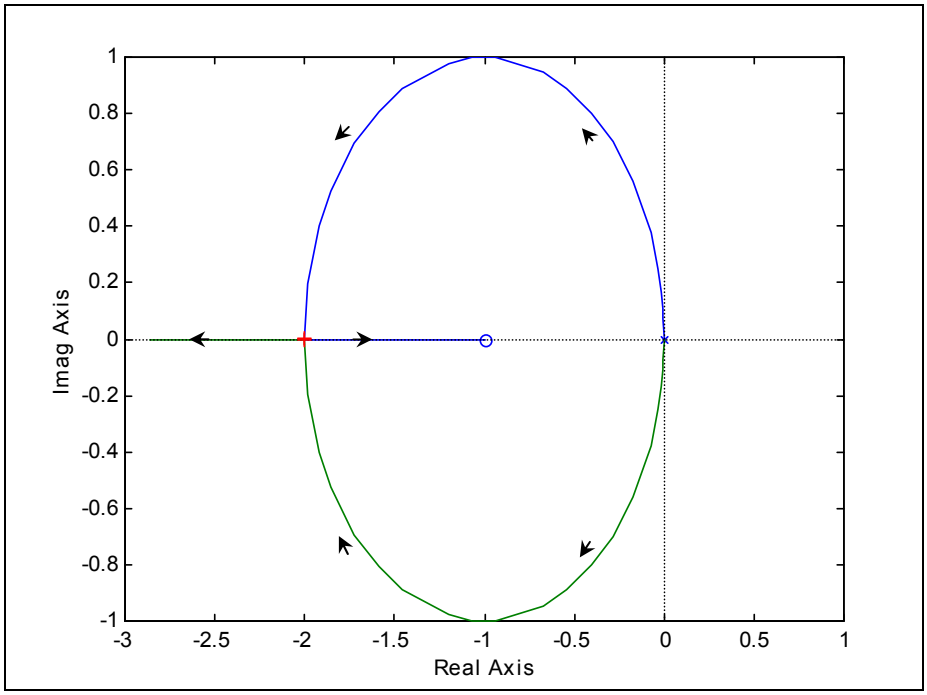
Trên đồ thị sẽ có thước cho ta chọn điểm – kéo rê chuột để chọn điểm cần chọn.
selected_point = -0.253
Điểm tách có giá trị: -0.253-

Kết luận: hệ thống luôn ổn định (vì quỹ đạo nghiệm luôn nằm ở nửa trái mặt phẳng phức).

Bài 6:

$$KGH = \frac{k(ts+1)}{s^2} \quad (k = 1, t = 1)$$

» num = [1 1];
» den = [1 0 0];
» rlocus(num,den)



1. Điểm cực: 0
2. Quỹ đạo nghiệm có 2 nhánh
3. Điểm zero ở ∞, -1
4. Điểm tách được được xác định bằng cách từ cửa sổ MATLAB ta nhập:

» num = [1 1];
» den = [1 0 0];
» rlocus(num,den);
» rlocfind(num,den)

Sau khi nhập lệnh thì trên cửa sổ lệnh sẽ xuất hiện hàng chữ:
Select a point in the graphics window (hãy chọn 1 điểm trên đồ thị minh họa).

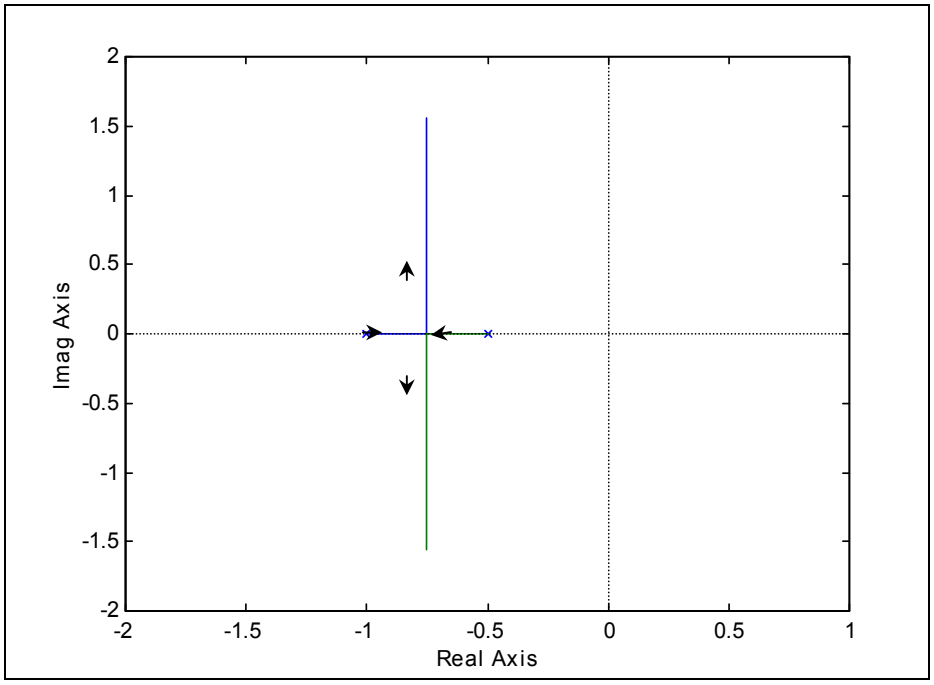
Trên đồ thị sẽ có thước cho ta chọn điểm – kéo rê chuột để chọn điểm cần chọn.
selected_point = -2
Điểm tách có giá trị: -2.

Kết luận: hệ thống ở biên ổn định.

Bài 7:

$$KGH = \frac{k}{(t_1 s + 1)(t_2 s + 1)} \quad (k = 1, t_1 = 2, t_2 = 1)$$

» num = 1;
» den = [2 3 1];
» rlocus(num,den)



1. Điểm cực: -0.5, -1.
2. Quỹ đạo nghiệm có 2 nhánh
3. Điểm zero ở vô hạn (∞)
4. Điểm tách được được xác định bằng cách từ cửa sổ MATLAB ta nhập:

» num = 1;
» den = [2 3 1];
» rlocus(num,den);
» rlocfind(num,den)

Sau khi nhập lệnh thì trên cửa sổ lệnh sẽ xuất hiện hàng chữ:
Select a point in the graphics window (hãy chọn 1 điểm trên đồ thị minh họa).

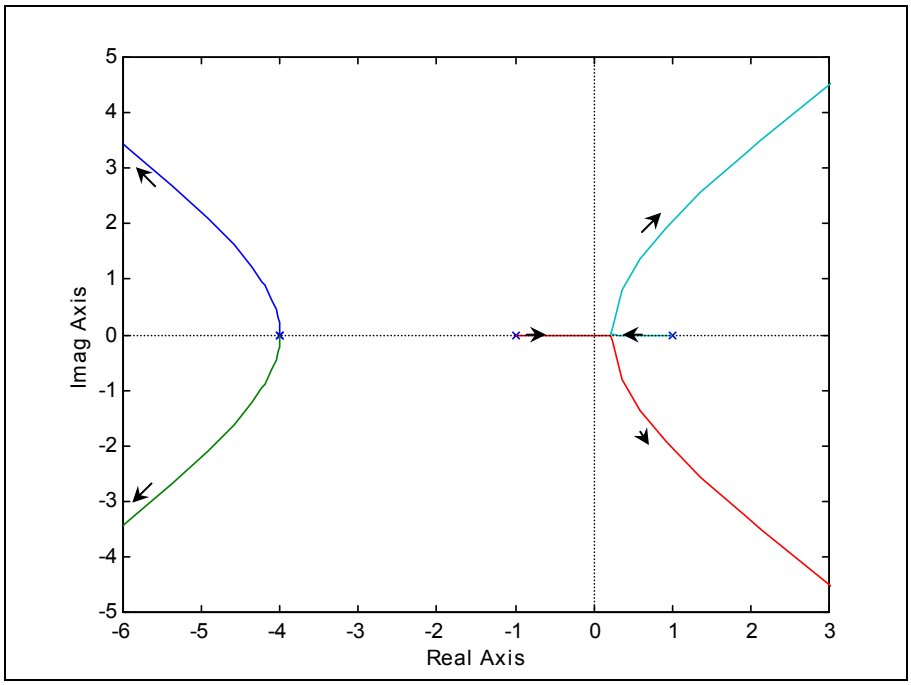
Trên đồ thị sẽ có thước cho ta chọn điểm – kéo rê chuột để chọn điểm cần chọn.
selected_point = -0.75.
Điểm tách có giá trị: -0.75

Kết luận: hệ thống luôn ổn định.

Bài 8:

$$KGH = \frac{k}{(s+1)(s-1)(s+4)^2} \quad (k = 10)$$

» num =10;
» den = [1 8 15 -8 -16];
» rlocus(num,den)



1. Điểm cực : 1, -1 và 1 cực kép -4.
2. Quỹ đạo nghiệm có 4 nhánh.
3. Điểm zero: có 4 zero ở vô cùng (∞).
4. Điểm tách được được xác định bằng cách từ cửa sổ MATLAB ta nhập:

» num =10;
» den = [1 8 15 -8 -16];
» rlocus(num,den);
» rlocfind(num,den)

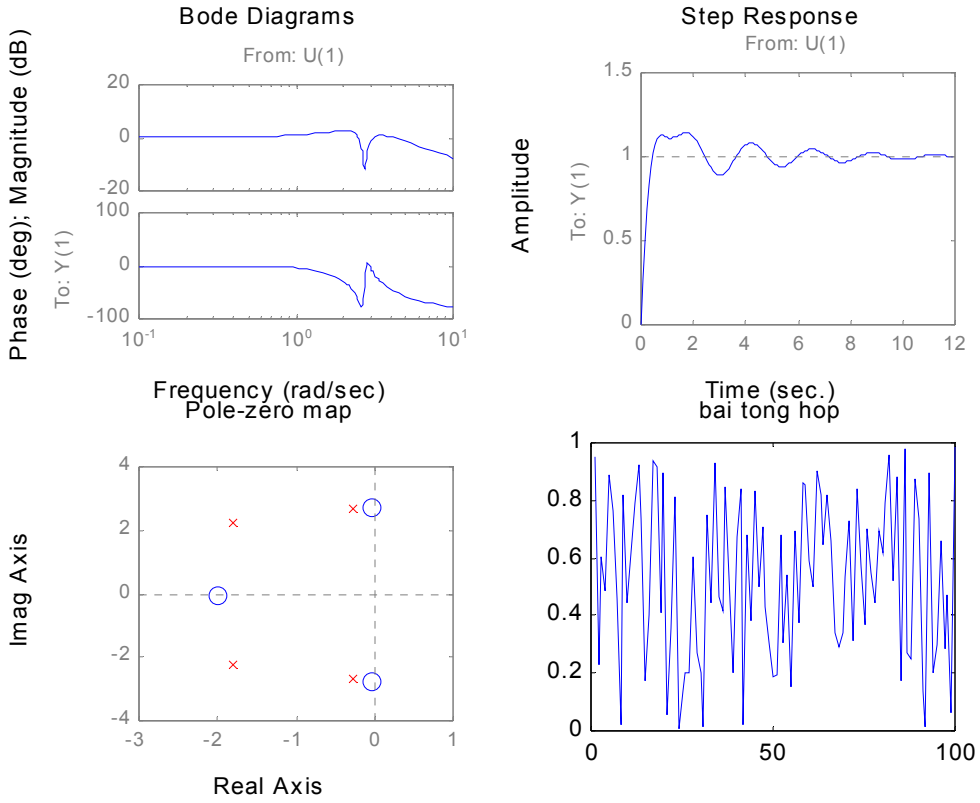
Sau khi nhập lệnh thì trên cửa sổ lệnh sẽ xuất hiện hàng chữ:
Select a point in the graphics window (hãy chọn 1 điểm trên đồ thị minh họa).

Trên đồ thị sẽ có thước cho ta chọn điểm – kéo rê chuột để chọn điểm cần chọn.
selected_point = 0.2308, -4
Điểm tách có giá trị: 0.2308, -4

Kết luận: Hệ thống luôn không ổn định vì tồn tại 1 nghiệm của phương trình đặc trưng nằm bên phải mặt phẳng phức.

Bài 9: Trích từ trang 5-19 sách **'Control System Toolbox'**
Bài này tổng hợp các lệnh:

```
» h=tf([4 8.4 30.8 60],[1 4.12 17.4 30.8 60]);
» subplot(221)
» bode(h)
» subplot(222)
» step(h)
» subplot(223)
» pzmap(h)
» subplot(224)
» plot(rand(1,100)
» plot(rand(1,100))
```



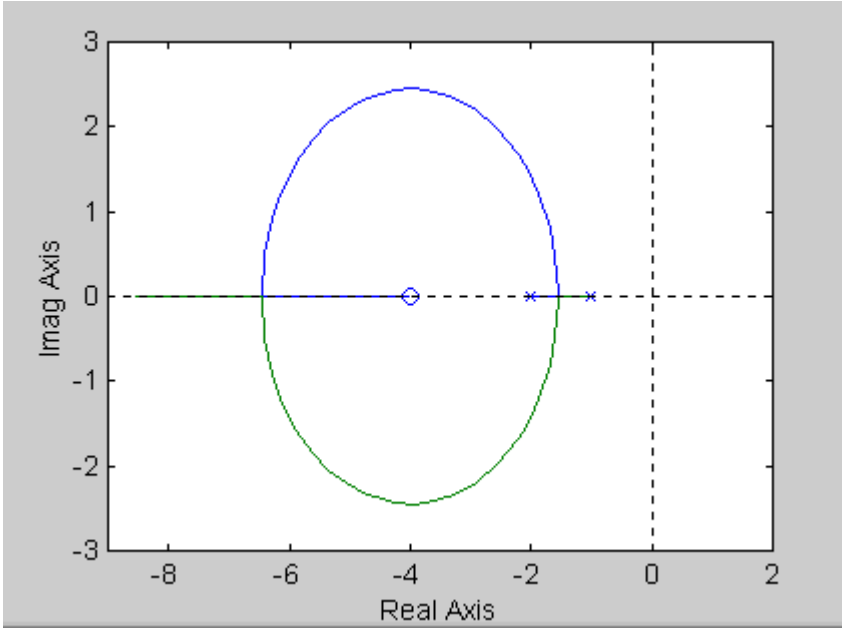
Bài 10: Cho hàm truyền như sau:

$$G(s) = \frac{s+4}{(s+1)(s+2)}$$

Viết theo cấu trúc sau ta có được đồ thị biểu diễn quỹ đạo nghiệm:

```
» num=[1 4];
» den=conv([1 1],[1 2])
» rlocus(num,den)
```

Kết quả như hình sau:



KHẢO SÁT SỰ ỔN ĐỊNH CỦA HỆ THỐNG

LÝ THUYẾT:

- **Hệ thống ổn định ở trạng thái hở, sẽ ổn định ở trạng thái kín nếu biểu đồ Nyquist không bao điểm (-1+i0) trên mặt phẳng phức.**
- **Hệ thống không ổn định ở trạng thái hở, sẽ ổn định ở trạng thái kín nếu biểu đồ Nyquist bao điểm (-1+i0)p lần ngược chiều kim đồng hồ (p là số cực GH nằm ở phải mặt phẳng phức).**

Từ dấu nhắc của cửa sổ MATLAB, ta nhập:

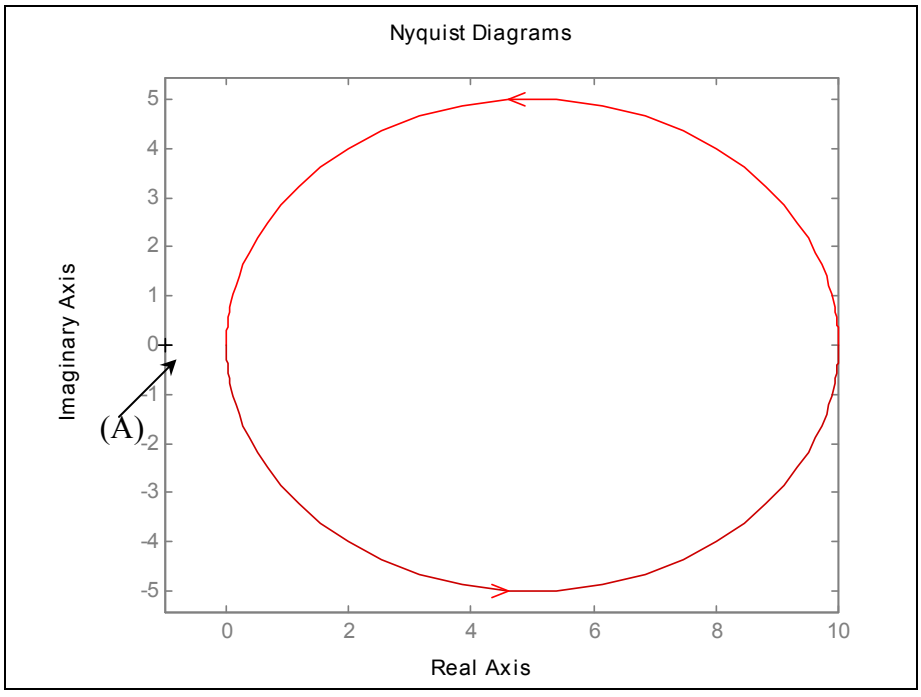
» num = [*nhập các hệ số của tử số theo chiều giảm dần của số mũ*].
» den = [*nhập các hệ số của mẩu số theo chiều giảm dần của số mũ*].
» nyquist(num,den)

Bài tập 1:

$$GH(s) = \frac{k}{1-st} \quad (\text{với } k = 10,\ t = 1)$$

» num = 10;
» den = [-1 1];
» nyquist(num,den)

Kết quả:



Nhận xét: hàm truyền vòng hở có 1 cực nằm bên phải mặt phẳng phức. Biểu đồ Nyquist không bao điểm A (-1+j0).

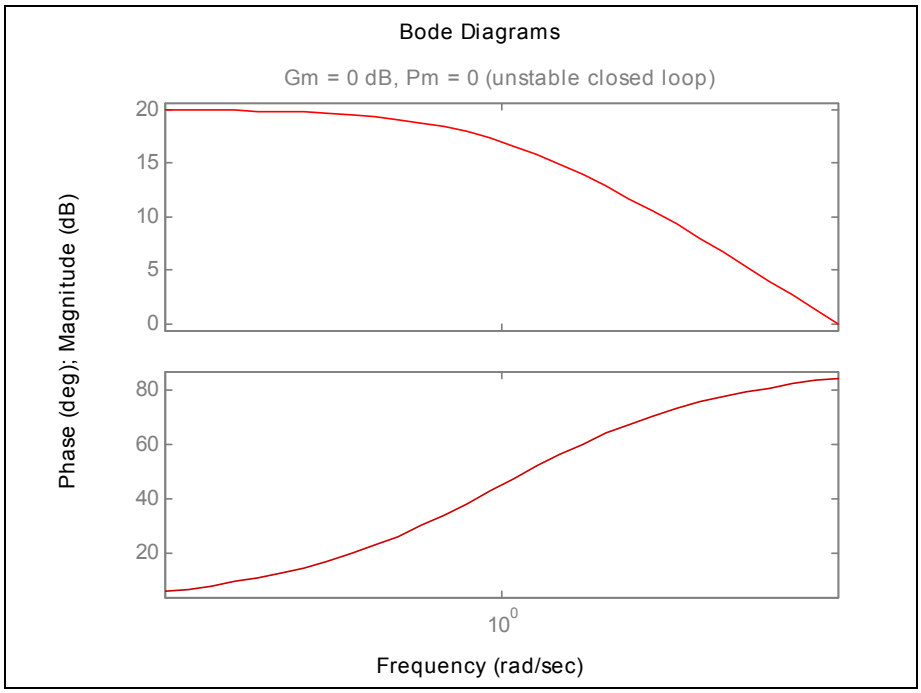
Điểm –1 ký hiệu (+) nằm trên trục thực âm (Real Axis), điểm 0 nằm trên trục ảo (Imaginary Axis).

Kết luận: hệ không ổn định.

*** Dùng lệnh margin để tìm biên dự trữ và pha dự trữ.**

Từ dấu nhắc của cửa sổ lệnh MATLAB ta dùng lệnh 'margin':

```
» num = 10;
» den = [-1 1];
» margin(num,den);
```



Kết luận:

Độ dự trữ biên (Gm = 0 dB).

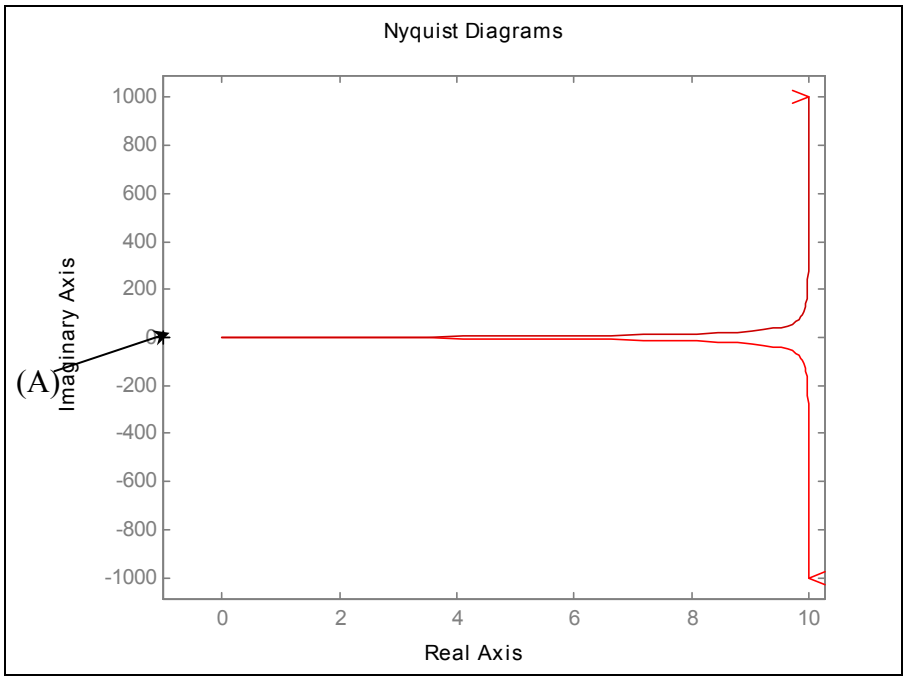
Độ dự trữ pha (Pm = 0°).

Warning: Closed loop is unstable (hệ vòng kín không ổn định).

Bài tập 2:

$$GH(s) = \frac{k}{s(1-st)} \quad (k = 10, t = 1)$$

```
» num = 10;
» den = [-1 1 0];
» nyquist(num,den)
```



Nhận xét: hàm truyền vòng hở có 1 cực nằm bên phải mặt phẳng phức và 1 cực nằm tại gốc tọa độ. Biểu đồ Nyquist không bao điểm A (-1+j0).

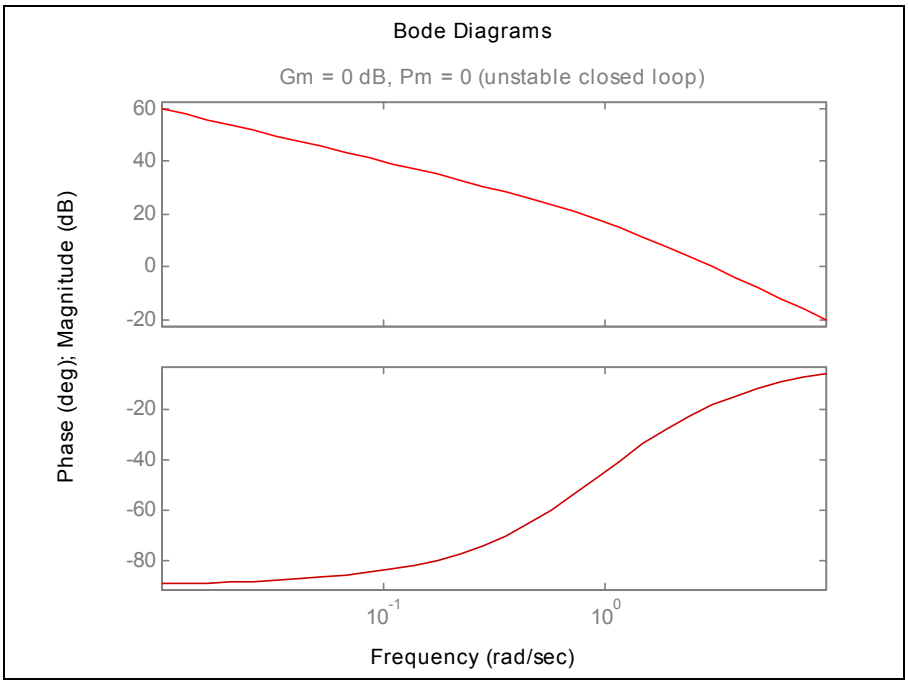
Điểm –1 ký hiệu (+) nằm trên trục thực âm (Real Axis) , điểm 0 nằm trên trục ảo (Imaginary Axis).

Kết luận: hệ không ổn định.

*** Dùng lệnh margin để tìm biên dự trữ và pha dự trữ.**

Từ dấu nhắc của cửa sổ lệnh MATLAB ta dùng lệnh 'margin':

```
» num = 10;
» den = [-1 1 0];
»margin(num,den)
```

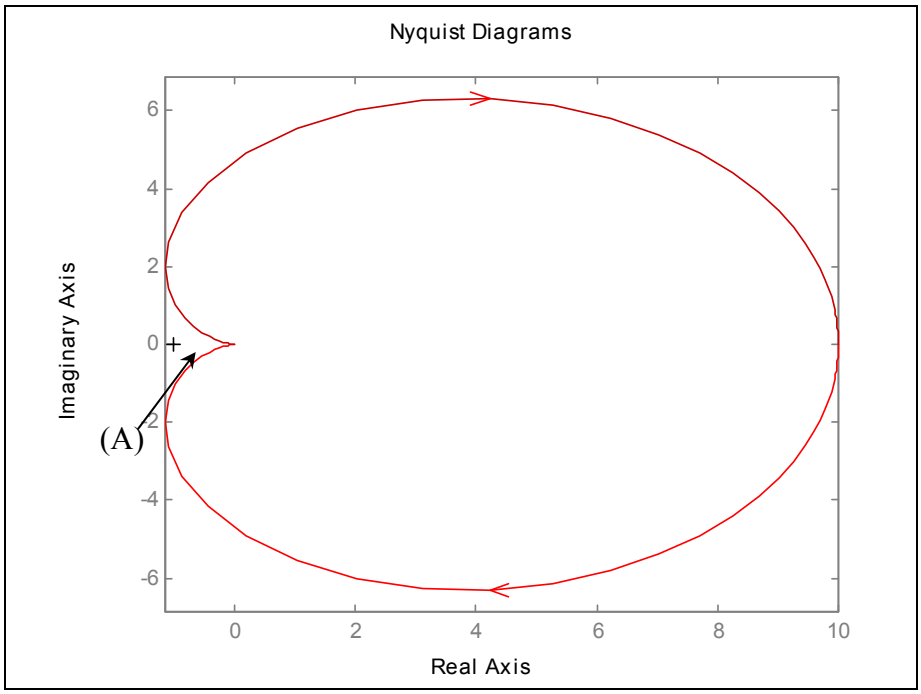


Kết luận:
Độ dự trữ biên (Gm = 0 dB).
Độ dự trữ pha (Pm = 0°).
Warning: Closed loop is unstable (hệ vòng kín không ổn định).

Bài tập 3:

$$GH(s) = \frac{k}{(t_1 s + 1)(t_2 s + 1)} \quad (k = 10,\ t_1 = 1,\ t_2 = 2)$$

» num = 10;
» den = [2 3 1];
» nyquist(num,den)



Nhận xét: hàm truyền vòng hở có 2 cực nằm bên trái mặt phẳng phức. Biểu đồ Nyquist không bao điểm A (-1+j0).

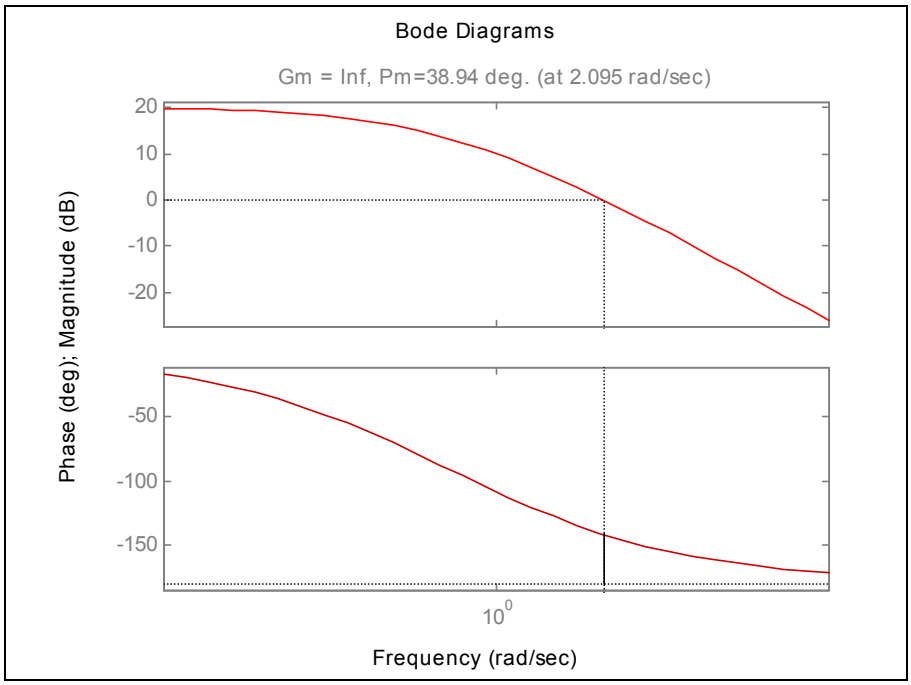
Điểm –1 ký hiệu (+) nằm trên trục thực âm (Real Axis) , điểm 0 nằm trên trục ảo (Imaginary Axis).

Kết luận: hệ thống ổn định.

*** Dùng lệnh margin để tìm biên dự trữ và pha dự trữ.**

Từ dấu nhắc của cửa sổ MATLAB dùng lệnh 'margin'.

```
» num = 10;
» den = [2 3 1];
» margin(num,den)
```

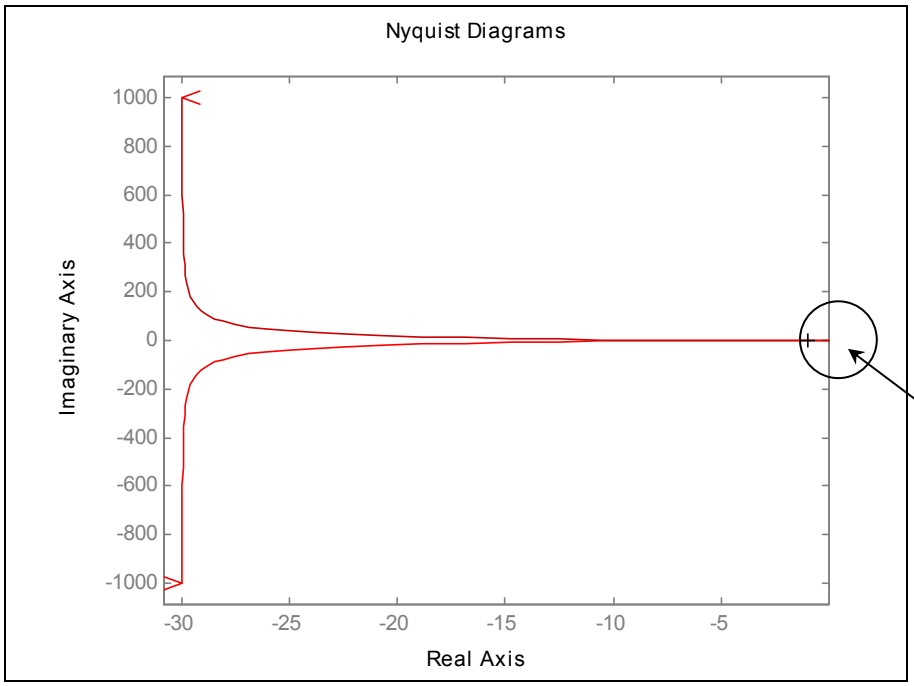


Kết luận: hệ thống ổn định.
Độ dự trữ biên (Gm = ∞).
Độ dự trữ pha (Pm = 38.94°), tại tần số cắt biên 2.095 rad/sec.

Bài tập 4:

$$GH(s) = \frac{k}{s(t_1 s+1)(t_2 s+1)} \qquad (k = 10 \quad t_1=1,\ t_2 =2)$$

```
» num = 10;
» den = [2 3 1 0];
» nyquist(num,den)
```



Nhận xét: hàm truyền vòng hở có 2 cực nằm bên trái mặt phẳng phức và 1 cực ở zero. Biểu đồ Nyquist bao điểm A(-1+j0).

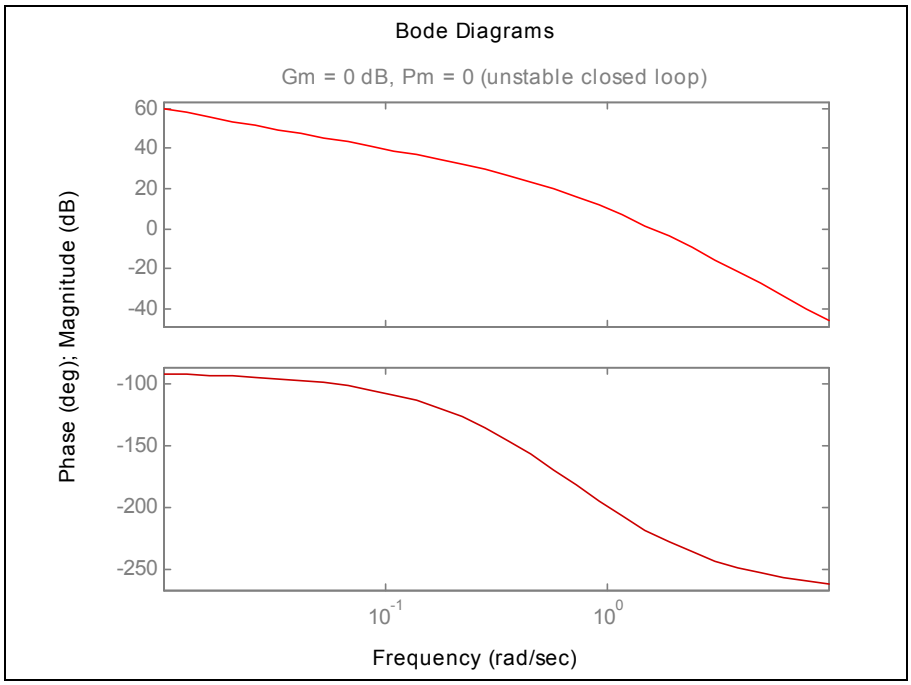
Điểm –1 ký hiệu (+) nằm trên trục thực âm (Real Axis) , điểm 0 nằm trên trục ảo (Imaginary Axis).

Kết luận: hệ không ổn định.

*** Dùng lệnh margin để tìm biên dự trữ và pha dự trữ.**

Từ dấu nhắc của cửa sổ MATLAB ta dùng lệnh 'margin' để kiểm chứng lại hệ:

```
» num = 10;
» den = [2 3 1 0];
»margin(num,den)
```

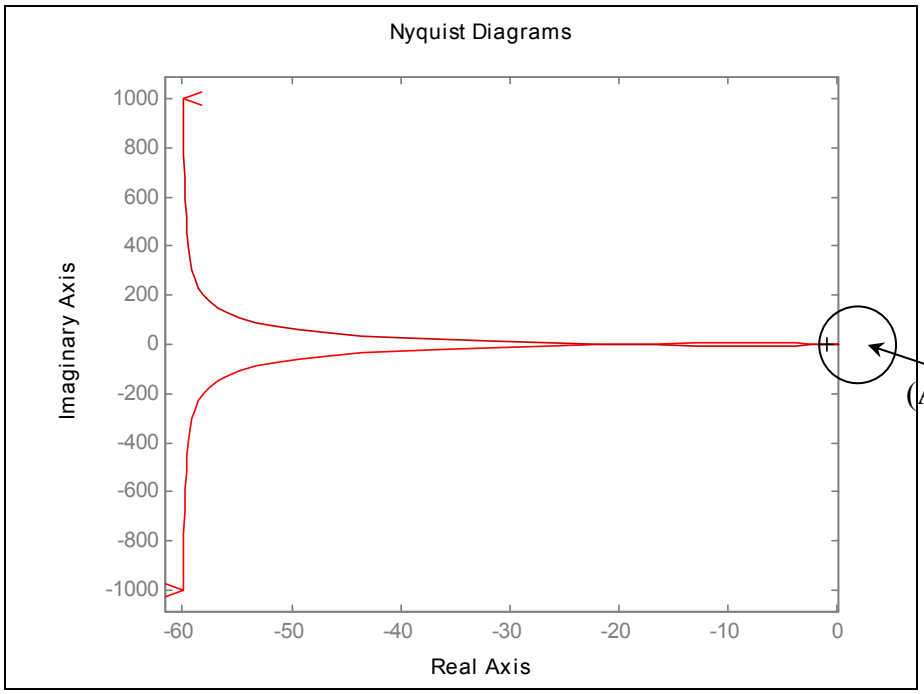


Kết luận: hệ thống không ổn định.
Độ dự trữ biên (Gm = 0 dB).
Độ dự trữ pha (Pm = 0°)

Bài tập 5:

$$GH(s) = \frac{k}{s(t_1 s+1)(t_2 s+1)(t_3 s+1)} \qquad (t_1 = 1,\ t_2 = 2,\ t_3 = 3,\ k = 10)$$

```
» num = 10;
» den = [6 11 6 1 0];
» nyquist(num,den)
```



Nhận xét: hàm truyền vòng hở có 3 cực nằm bên trái mặt phẳng phức và 1 cực ở zero. Biểu đồ Nyquist bao điểm A (-1+i0).

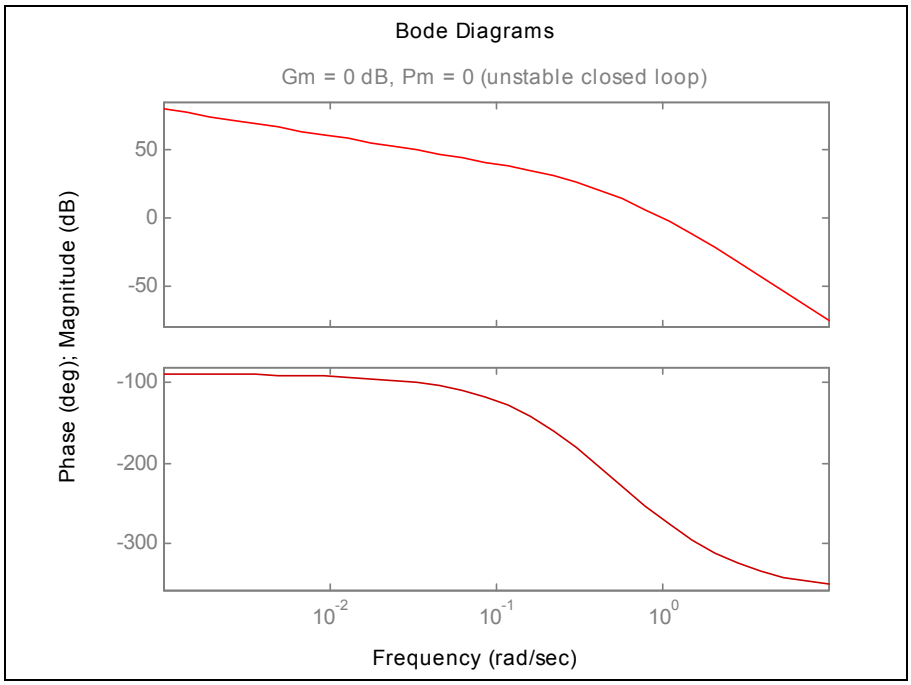
Điểm –1 ký hiệu (+) nằm trên trục thực âm (Real Axis) , điểm 0 nằm trên trục ảo (Imaginary Axis).

Kết luận: hệ không ổn định.

*** Dùng lệnh margin để tìm biên dự trữ và pha dự trữ.**

Từ dấu nhắc của cửa sổ MATLAB, dùng lệnh 'margin' để kiểm chứng lại hệ:

```
» num = 10;
» den = [6 11 6 1 0];
» margin(num,den)
```



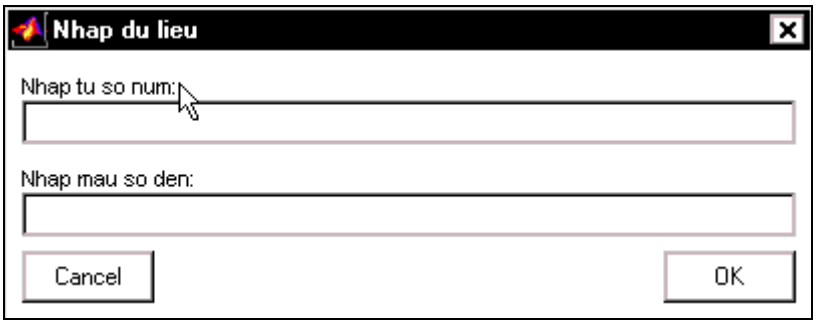
Kết luận: hệ thống không ổn định.
Độ dự trữ biên (Gm = 0 dB).
Độ dự trữ pha (Pm = 0°).

Bài tập 6: Sau đây là dạng bài tập tổng quát với tử và mẫu của một hàm truyền là các số liệu mà ta phải nhập vào.
Chương trình:

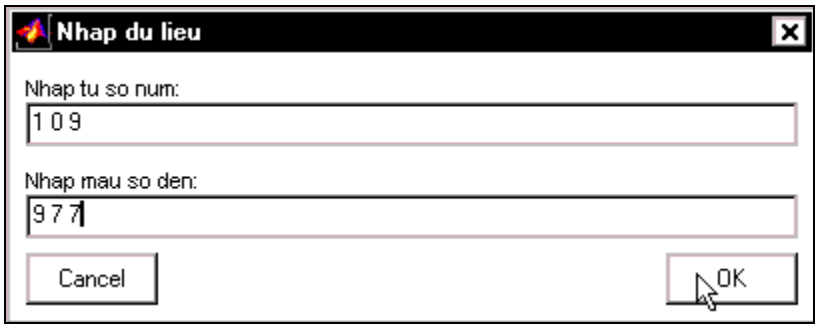
```
%%Tap tin khao sat on dinh he thong
%%PHAM QUOC TRUONG - DT: 9230774
function ondinh()
promptstr={'Nhap tu so num:','Nhap mau so den:'};
inistr={'',''};
dlgTitle='Nhap du lieu';
lineNo=1;
result=inputdlg(promptstr,dlgTitle,lineNo,inistr);
num=str2num(char(result(1)));
den=str2num(char(result(2)));
[z,p,k]=residue(num,den); %Tim cac cuc p
z=roots(num)              %Tim cac zero z
zplane(z,p)               %Ve cuc va zero
```

Sau khi chạy chương trình ta được kết quả:

Bạn hãy nhập số liệu vào:



Gỉa sử ta nhập số liệu sau và chọn OK:



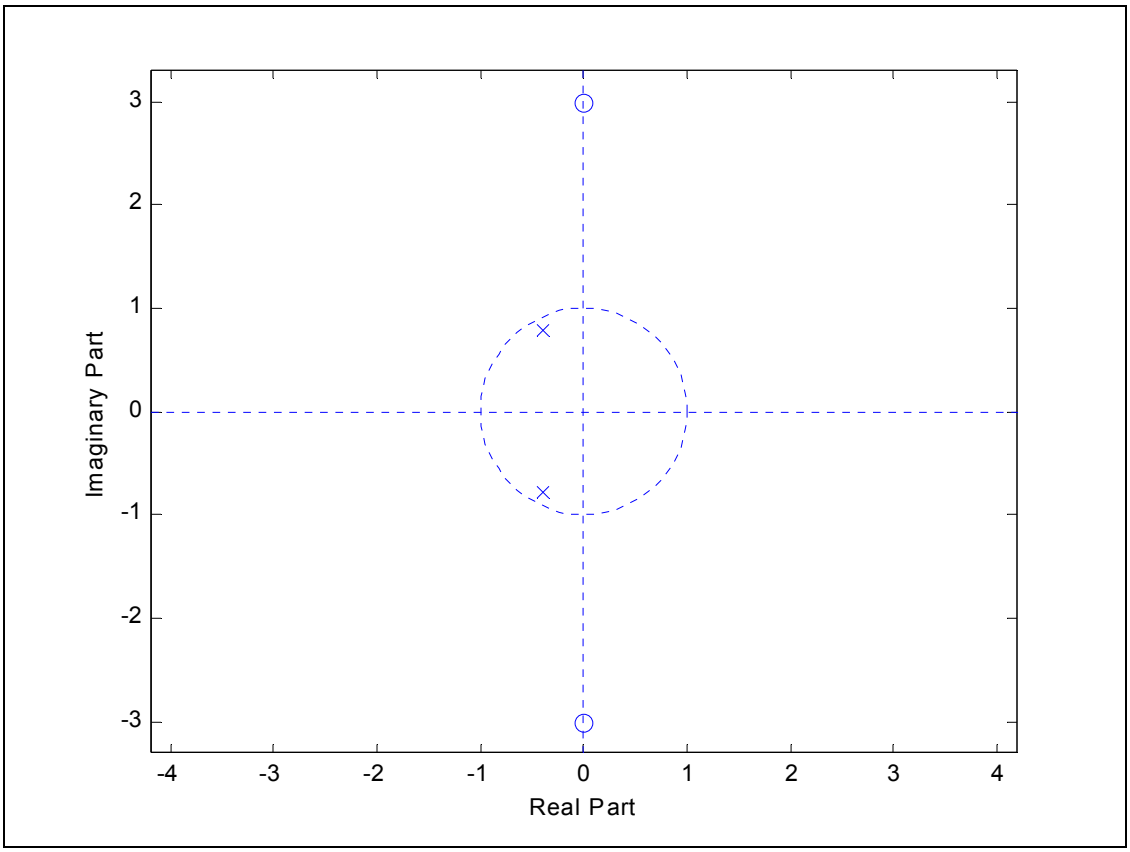
Kết quả ngoài cửa sổ MATLAB Command Windows

z =

0 + 3.0000i

0 - 3.0000i

Hình vẽ cực và zero:



Khảo sát hệ thống theo tiêu chuẩn Hurwitz

ÔN LẠI LÝ THUYẾT:

Xét Phương trình đặc trưng:

$$F(s) = a_n s^n + a_{n-1} + ... + a_0 \quad \text{với } a_n \neq 0$$

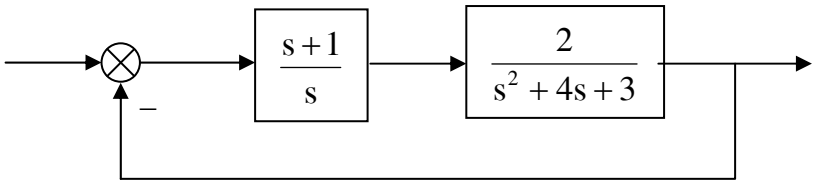
1. **Điều kiện cần để hệ ổn định:**
   - **Các hệ số a_j (j = 0, ... n-1) cùng dấu với a_n.**
   - **$a_j \neq 0$ (j = 0,...,n)**
2. **Tiêu chuẩn Hurwitz:**

Điều kiện cần và đủ để hệ ổn định (các nghiệm của phương trình đặt trưng nằm bên trái mặt phẳng phức) là tất cả các định thức Hurwitz D_k đều cùng dấu (k = 0..n)

3. **Tiêu chuẩn Routh:**

Điều cần và đủ để hệ ổn định là tất cả các phần tử của cột 1 bảng Routh đều cùng dấu, nếu có sự đổi dấu thì số lần đổi dấu thì số lần đổi dấu bằng số nghiệm ở phải mặt phẳng phức.

Bài tập 7: Cho hệ thống điều khiển phản hồi:



Dùng giản đồ Bode để khảo sát ổn định của hệ thống trên.
Khảo sát hệ xem hệ có ổn định hay không.
Trước tiên ta dùng lệnh 'series'kết nối 2 hệ thống:

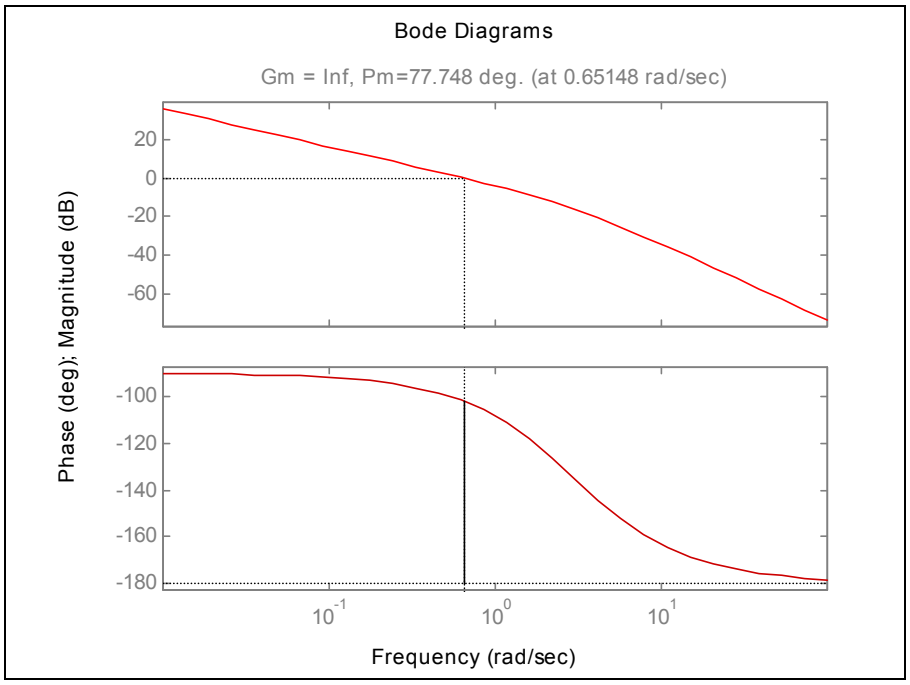
```
» num1 = [1 1];
» den1 = [1 0];
» num2 = 2;
» den2 = [1 4 3];
» [num,den] = series(num1,den1,num2,den2)
num =

   0    0    2    2
den =

   1    4    3    0
```

Hàm truyền nối tiếp là:

$$GH(s) = \frac{2s+2}{s^3+4s^2+3s}$$

Dùng giản đồ Bode để khảo sát ổn định:

```
» num = [2 2];
» den = [1 4 3 0];
» margin(num,den)
```



Kết luận:

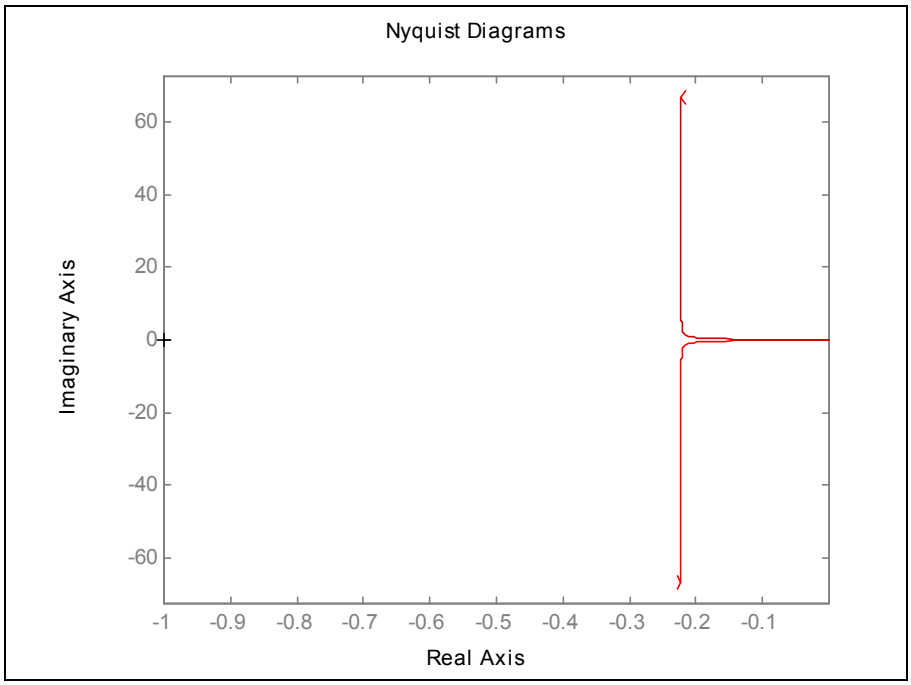
Biên dự trữ: Gm = ∞

Pha dự trữ Pm = 77.74° tại tần số cắt biên $w_b = 0.65$

Vậy hệ thống ổn định.

Vẽ biểu đồ Nyquist:

» nyquist(num,den)



Bên cạnh đó ta có thể khảo sát ổn định bằng tiêu chuẩn đại số:

Phương trình đặc trưng: $s^3 + 4s^2 + 5s + 2 = 0$

Trước tiên ta gọi 'hurwitz' từ cửa sổ lệnh:(liên hệ PQT để có chương trình)

» hurwitz

Cho biet so bac cao nhat cua ham: 3

Cho biet he so a(0): 1

Cho biet he so a(1): 4

Cho biet he so a(2): 5

Cho biet he so a(3): 2

Cac dinh thuc Hurwitz:

D[1] = 1

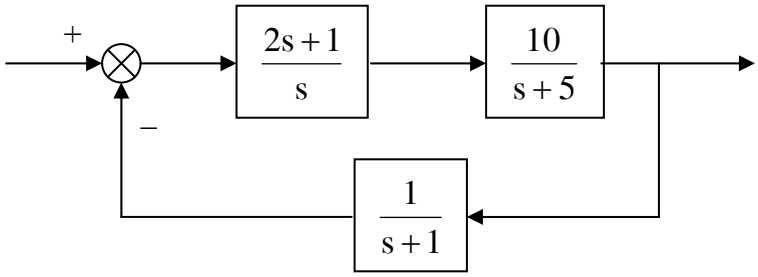
D[2] = 4

D[3] = 18

D[4] = 36

- HE THONG ON DINH. -

Bài tập 8: Khảo sát hệ thống:



Trước tiên, ta kết nối hệ thống:

Từ cửa sổ lệnh của MATLAB, ta nhập lệnh:

```
» num1 = [2 1];
» den1 = [1 0];
» num2 = 10;
» den2 = [1 5];
» [num,den] = series(num1,den1,num2,den2)
```

Và ta sẽ có:

```
num =

    0    20    10


den =

    1     5     0
```

Ta nhập tiếp:

```
» numc = [20 10];
» denc = [1 5 0];
» numd = 1;
» dend = [1 1];
» [num,den] = feedback(numc,denc,numd,dend)
```

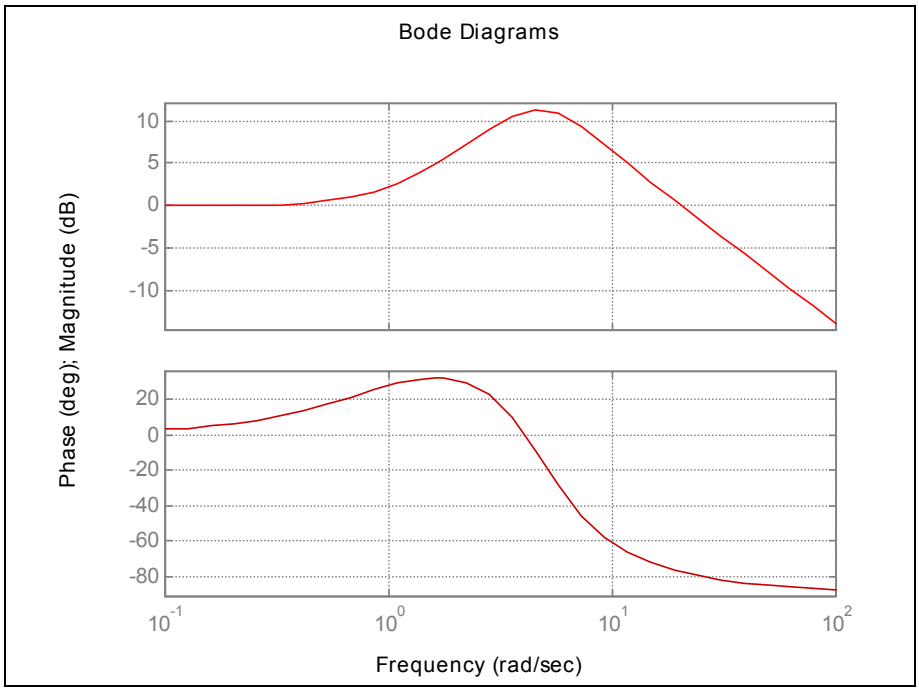
(nếu sau dend, có 1 tức là hồi tiếp dương)

```
num =

    0    20    30    10


den =

    1     6    25    10
```

Hàm truyền của hệ thống là: $G(s)H(s) = \dfrac{20s^2 + 30s + 10}{s^3 + 6s^2 + 25s + 10}$

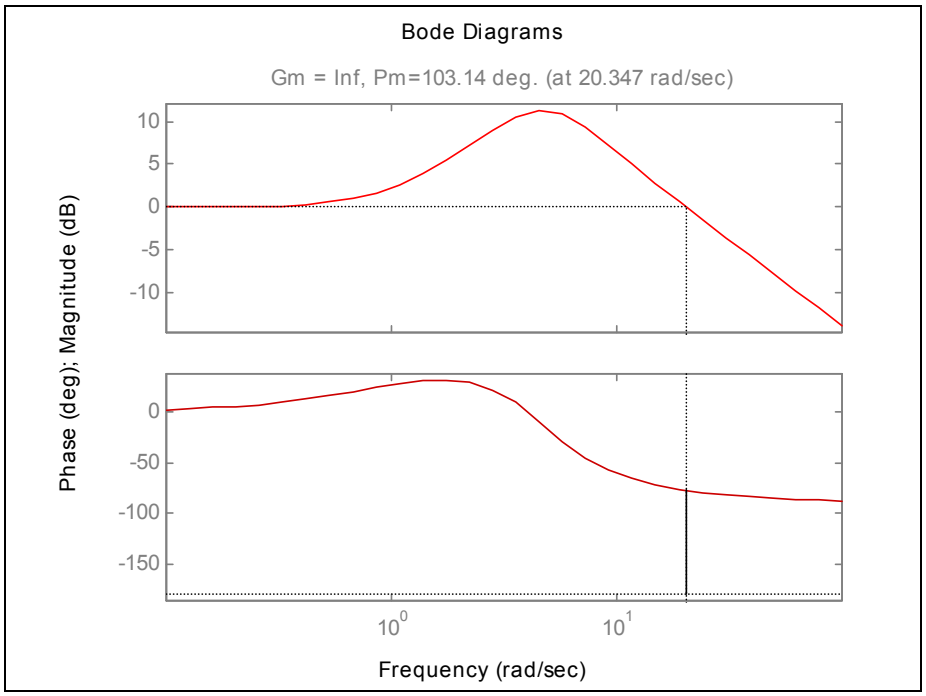
Vẽ giản đồ Bode của hệ:

```
» num = [20 30 10];
» den = [1 6 25 10];
» bode(num,den)
```



Tính biên dự trữ và pha dự trữ của hệ:

```
» margin(num,den)
```



Kết luận:

Hệ ổn định.

Biên dự trữ: Gm = ∞.

Pha dự trữ: Pm = 103.14^o tại tần số cắt biên là 20.347 rad/sec.

Chú ý: Sau khi đã vào cửa sổ lập trình, ta lập chương trình khảo sát hệ có phương trình đặc trưng theo tiêu chuẩn đại số (tiêu chuẩn Hurwitz) xem hệ có ổn định hay không.

Trong cửa sổ lệnh (cửa sổ làm việc), gọi lệnh » hurwitz (chương trình đã được soạn thảo trong phần lập trình mang tên Hurwitz) sẽ có những hàng chữ:

cho biet so bac cao nhat cua ham: (nhập vào hệ số a_n)

cho biet he so a(0):

. . .

cho biet he so a(n):

Dưới dây là phần đánh vào cửa sổ lập trình

```
%%%%%%%%%% PHAM QUOC TRUONG - MSSV: 97102589 %%%%%%%%%%
%%%%%%%%%%            DT: 9230774             %%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function Hurwitz()
   %
   % * Cong dung: Xet tinh on dinh cua he thong theo tieu chuan
Hurwitz.
   %
   % * Cach su dung:
   % Truoc tien, nhap vao da thuc dac trung f theo dang:
```

```
   %     f = [a(n)  a(n-1)  a(n-2) ..... a(1)  a(0)]
   % voi a(n), a(n-1), a(n-2), ....., a(1),a(0) la cac he so cua da
thuc dac trung.
   % Sau do, goi lenh Hurwitz(f)
```

XIN VUI LONG LIÊN HỆ PHẠM QUỐC TRƯỜNG ĐỂ CÓ CHƯƠNG TRÌNH

Chạy chương trình các ví dụ:

Ví dụ 1: Cho phương trình đặc trưng:

$$F(s) = s^4 + 3s^3 + 2s^2 + 2s + 1$$

» Hurwitz
Cho biet so bac cao nhat cua ham: 4 (nhập xong nhấn Enter)
Cho biet he so a(0) = 1
Cho biet he so a(1) = 3
Cho biet he so a(2) = 2
Cho biet he so a(3) = 2
Cho biet he so a(4) = 1

Sau khi đã nhập các hệ số, MATLAB sẽ tự động giải và cho ta kết quả:

Cac dinh thuc Hurwitz:
D[1] = 1

D[2] = 3

D[3] = 4

D[4] = -1

D[5] = -1

- HE THONG KHONG ON DINH. –

Ví dụ 2: Cho phương trình đặc trưng:

$$F(s) = 5s^4 + 8s^3 + 21s^2 + 10s + 3$$

» Hurwitz
Cho biet so bac cao nhat cua ham: 4
Cho biet he so a(0) = 5
Cho biet he so a(1) = 8
Cho biet he so a(2) = 21
Cho biet he so a(3) = 10
Cho biet he so a(4) = 3
Cac dinh thuc Hurwitz:
D[1] = 5

D[2] = 8

D[3] = 118

D[4] = 988

D[5] = 2964

- HE THONG ON DINH. -

Ví dụ 3: Cho phương trình đặc trưng:

$$F(s) = s^5 + 10s^4 + 16s^3 + 160s^2 + s + 10$$

» hurwitz
Cho biet so bac cao nhat cua ham: 5
Cho biet he so a(0) = 1
Cho biet he so a(2) = 10
Cho biet he so a(3) = 16
Cho biet he so a(4) = 160
Cho biet he so a(5) = 1
Cho biet he so a(6) = 10

Sau khi đã nhập các hệ số, MATLAB sẽ tự động giải và cho ta kết quả:

Cac dinh thuc Hurwitz:
D[1] = 1

D[2] = 10

D[3] = 0

D[4] = 0

D[5] = 0

D[6] = 0

- HE THONG O BIEN ON DINH. –

Khảo sát hệ thống theo tiêu chuẩn Routh

Chương trình:(liên hệ PQT)

```
%%%%%%%%%%%%%%%%%%% PHAM QUOC TRUONG MSSV:97102589  %%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%       Dien thoai: 9230774       %%%%%%%%%%%%%%%%%%%%
```

%%%%%%%%%%%%%%%%%%PHẠM QUỐC TRƯỜNG%%%%%%%%%%%%%%%%%%%GVHD%%%%%%%%%%%PHẠM QUANG HUY%%%%%%%%%%%%%%%%

Chạy chương trình các ví dụ:

Ví dụ 1: Cho phương trình đặc trưng

$$F(s) = s^4 + 3s^3 + 2s^2 + 2s + 1$$

» routh
- CHUONG TRINH TAO HAM ROUTH -
Cho biet so bac cao nhat cua he: 4
Cho biet he so a(0) = 1
Cho biet he so a(1) = 3
Cho biet he so a(2) = 2
Cho biet he so a(3) = 2
Cho biet he so a(4) = 1
- HE THONG KHONG ON DINH. -

Ví dụ 2: Cho phương trình đặc trưng

$$F(s) = s^5 + s^4 + 4s^3 + 4s^2 + 2s +1$$

» routh
- CHUONG TRINH TAO HAM ROUTH -
Cho biet so bac cao nhat cua he: 5
Cho biet he so a(0) = 1
Cho biet he so a(1) = 1
Cho biet he so a(2) = 4
Cho biet he so a(3) = 4
Cho biet he so a(4) = 2
Cho biet he so a(5) = 1
- HE THONG KHONG ON DINH. -

Ví dụ 3: Cho phương trình đặc trưng

$$F(s) = s^5 + 10s^4 + 16s^3 + 160s^2 + s + 10$$

» routh
- CHUONG TRINH TAO HAM ROUTH -
Cho biet so bac cao nhat cua he: 5
Cho biet he so a[0] = 1
Cho biet he so a[1] = 10
Cho biet he so a[2] = 16
Cho biet he so a[3] = 160
Cho biet he so a[4] = 1
Cho biet he so a[5] = 10

- HE THONG ON DINH. -

MỘT SỐ CHƯƠNG TRÌNH KHẢO SÁT, THIẾT KẾ HỆ THỐNG ĐIỀU KHIỂN TỰ ĐỘNG

(Nếu bạn nào quan tâm đến các chương trình thì liên hệ với PQT)

1. Chương trình 1:

Viết chương trình xác định hàm truyền vòng kín có khâu hồi tiếp đơn vị.

2. Chương trình 2:

Viết chương trình tìm cực và zero của hàm truyền.

3. Chương trình 3:

Viết chương trình khảo sát tính ổn định của hệ tuyến tính liên tục dùng giản đồ Bode.

4. Chương trình 4:

Tạo ra lệnh hurwitz để xét tính ổn định của hệ thống tuyến tính liên tục theo tiêu chuẩn Hurwitz.

5. Chương trình 5:

Viết chương trình tự động vẽ giản đồ Bode, biểu đồ Nyquist, quỹ đạo nghiệm của hệ tuyến tính liên tục.

6. Chương trình 6:

Viết chương trình để tìm các chỉ tiêu trong miền thời gian của hệ bậc 2.

7. Chương trình 7:

Viết chương trình để thực hiện bổ chính cho một hệ thống tuyến tính liên tục bằng giản đồ Bode.

8. Chương trình 8:

Viết chương trình khảo sát ảnh hưởng của khâu PID vào hệ thống tuyến tính bậc 2.
trong các tập tin này chương trình sẽ không thực hiện được.

9. Chương trình 9:

Viết lệnh dùng để khảo sát tính ổn định của hệ thống tuyến tính gián đoạn theo tiêu chuẩn Jury.

11. Chương trình 11:

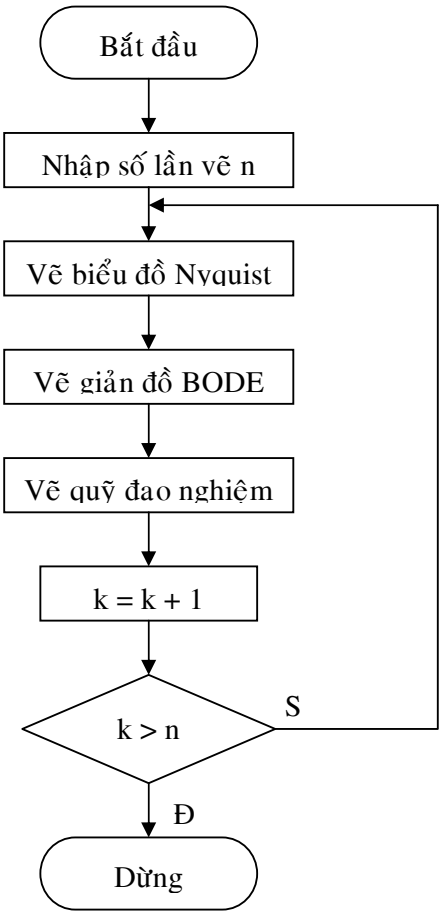
Viết chương trình đồ họa để vẽ các đáp ứng tần số và đáp ứng thời gian bằng cách chọn trong menu.

Chương trình được soạn thảo trong 2 tập tin dohoa.m và action.m và hệ thống trong chương trình này có hàm truyền là:

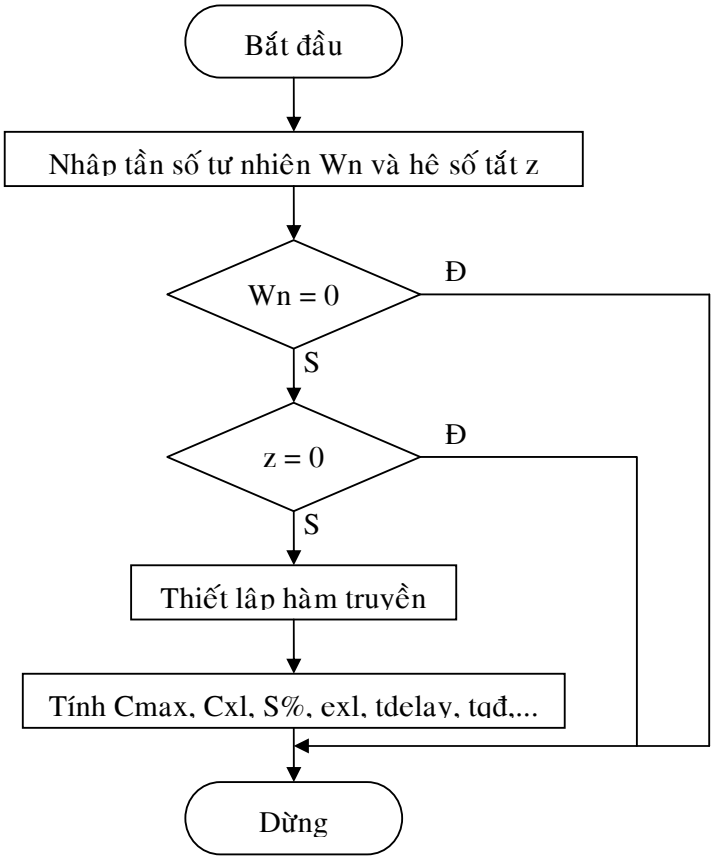
$$G(s) = \frac{1}{s(s+4)(s+5)}$$

PHỤ CHƯƠNG: LƯU ĐỒ CÁC CHƯƠNG TRÌNH

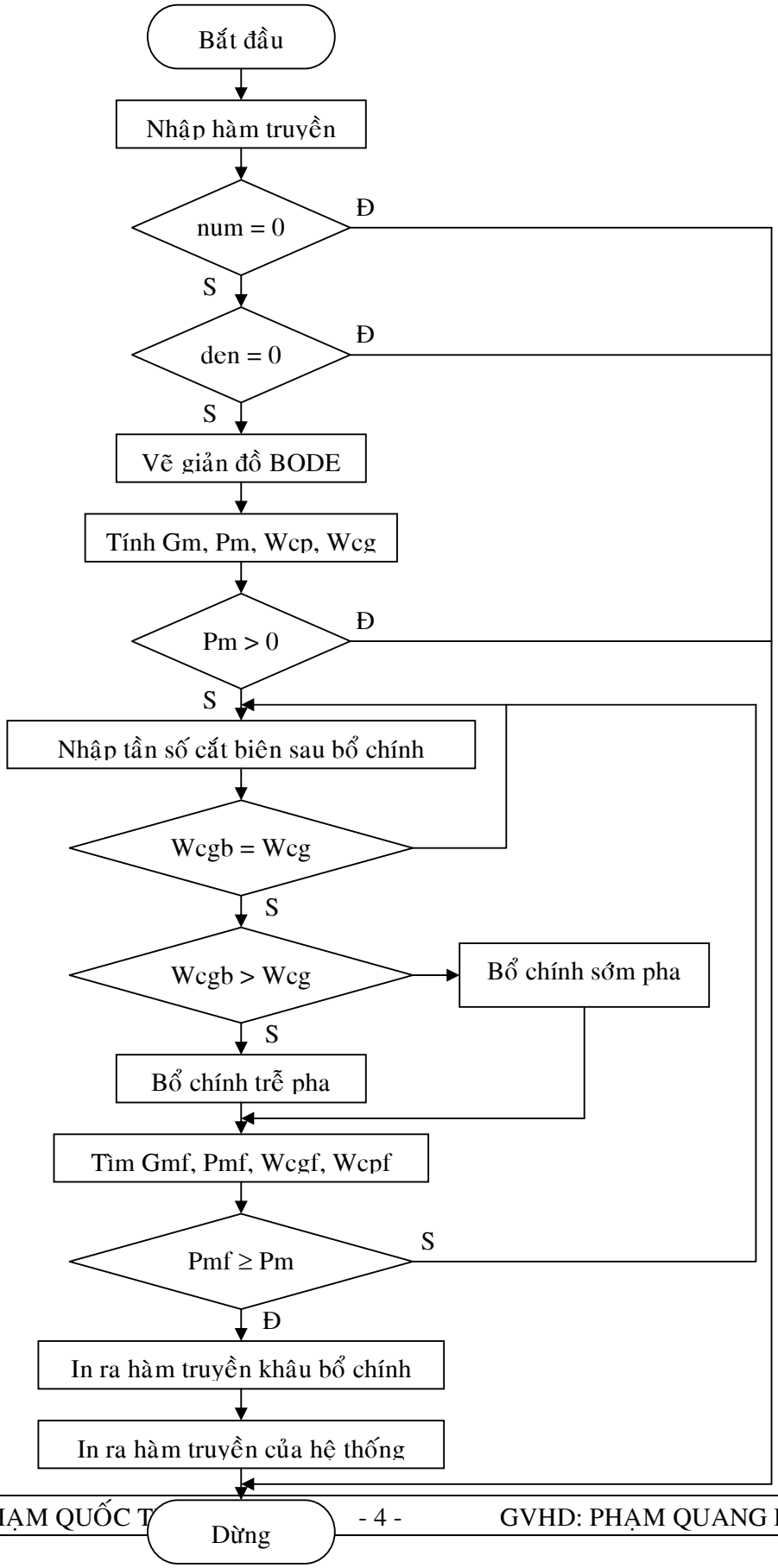
Lưu đồ chương trình tự động vẽ biểu đồ Nyquist, giản đồ Bode và quỹ đạo nghiệm



Chương trình tìm các chỉ tiêu trong miền thời gian của hệ bậc 2



Chương trình bổ chính cho hệ thống tuyến tính liên tục



Chương trình khảo sát ảnh hưởng của khâu PID vào hệ thống