선형 제어 시스템 설계(I) : 궤환 시스템의 성능과 안정도 판별

최 한 호
동국대학교 전기공학과

1. 서론

Control system(제어 시스템)은 웹 기반의 무료 백과사전인 Wikipedia(홈페이지 http://en.wikipedia.com)에 따르면 다른 장치들의 동작을 조절하는 장치들을 의미한다. 이에 부합하는 것의 발명은 인간의 탄생에서부터 시작되었다 할 수 있으며 제임스 와트가 증기엔진과 flyball governor(속도 조정기)를 개발한 1769년을 산업혁명의 시작으로 간주하는 것에서 볼 수 있듯이 고도화된 현대 문명은 제어 시스템에 기반하였다고 하여도 과언이 아니다. 제어 시스템은 선형, 비선형, 시불변, 시변, 연속시간, 이산시간, 확정, 확률 시스템 등 여러 가지 방법으로 분류될 수 있는데 선형 시스템에 대한 이론은 다양하고 매우 조직적이어서 복잡한 시스템의 해석과 설계에 기본이 된다. 그러므로 [1]-[9]의 참고문헌을 참조하여 선형 제어 시스템 설계 이론만을 3회에 걸쳐 연재할 예정이다. 그 첫 번째로 본고에서는 궤환 시스템의 성능과 안정도 판별에 관련된 고전 선형 제어이론과 이와 관련된 Matlab 명령어에 대하여 다룬다.

2. 시간 영역 성능

2.1. 기본적인 성능의 정의

- 지연시간(t_d) : 계단 입력에 대한 정상상태 응답 y_{ss}의 50%에 도달하는 데 걸리는 시간(그림 2참조)
- 상승시간(t_r) : 진동이 있는 경우, y_{ss}의 10%에서 90%, 진동이 없는 경우는 0%에서 100%에 이르는데 걸리는 시간
- 정착시간(t_s) : 계단 입력 응답이 $y_{ss}\pm\delta$로 안정하는 데 걸리는 시간(δ는 보통 2% 혹은 5%)
- 최대값시간(t_p) : 최대출력값에 도달했을 때의 시간
- 오버슈트(M_p) : 최대출력과 y_{ss}의 차이(그림 2참조)

2.2. 1차 시스템

- 정규화 1차 시스템 모델 : 입력에 대한 출력의 라플라스 변환 비율인 전달함수가 $H(s)=\frac{a}{s+a}$, $a>0$이다.
- 단위 계단 응답 : 입력이 1일 때의 응답으로 $LAPLACE^{-1}\left[H(s)\frac{1}{s}\right]=LAPLACE^{-1}\left[\frac{a}{s(s+a)}\right]=1-e^{-at}$처럼 주어진다. 그림 1은 $a=1$일 때 응답을 그린 것이다.
- 시상수(T) : $1/a$로 계단입력 응답 최종값의 63.2%에 걸리는 시간이다(그림 1 참조).
- 대역폭(Bandwidth : BW) : a로 주어지며 대역폭이 크면 응답이 빠르고 적으면 응답이 늦다(그림 1 참조).
- 정규화된 1차 시스템의 2% 정착시간 :

$$t_s\approx 4T=4/a$$

- 정규화된 1차 시스템의 오버슈트 : $M_p=0$
- 정규화된 1차 시스템의 상승시간 :

$$t_r\approx 2.2T=2.2/a$$

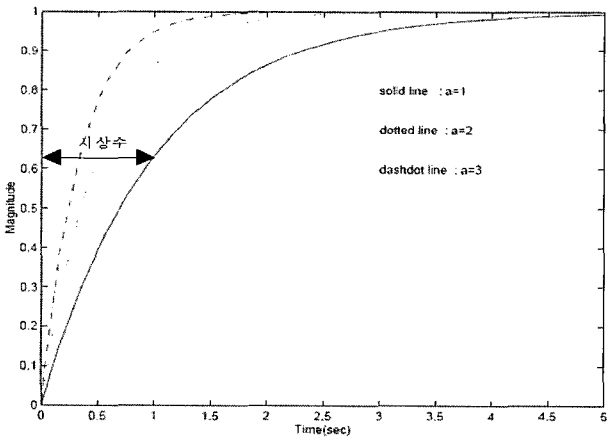

그림 1. $a/(s+a)$의 계단응답곡선과 $a=1$일 때 시상수.

2.3. 2차 시스템

- 정규화 2차시스템 모델 : $H(s)=\frac{w_n^2}{s^2+2\zeta w_n s+w_n^2}$ 로 여기에서 ζ는 감쇠비, w_n은 고유진동수이다.
- 계단 응답 : 아래처럼 주어진다.

$$1-\frac{1}{\sqrt{1-\zeta^2}}e^{-\zeta w_n t}\sin(w_n\sqrt{1-\zeta^2}t+\cos^{-1}\zeta),$$

$0<\zeta<1$

- 정규화된 2차 시스템의 BW :

$$BW=w_n\sqrt{1-2\zeta^2+\sqrt{2-4\zeta^2+4\zeta^4}}$$

• 정규화된 2차 시스템의 최대값시간 :

$$t_p = \pi / w_n \sqrt{1-\zeta^2}$$

• 정규화된 2차 시스템의 2% 정착시간 :

$$t_s \approx 4BW \approx 4/\zeta w_n$$

• 정규화된 2차 시스템의 오버슈트 :

$$M_p = e^{-\zeta\pi\sqrt{1-\zeta^2}}$$

• 정규화된 2차 시스템의 상승시간 :

$$t_r \approx (2.16\zeta + 0.6)/w_n,$$

$$t_r \approx 1.8/w_n (\text{if} \quad \zeta = 0.5)$$

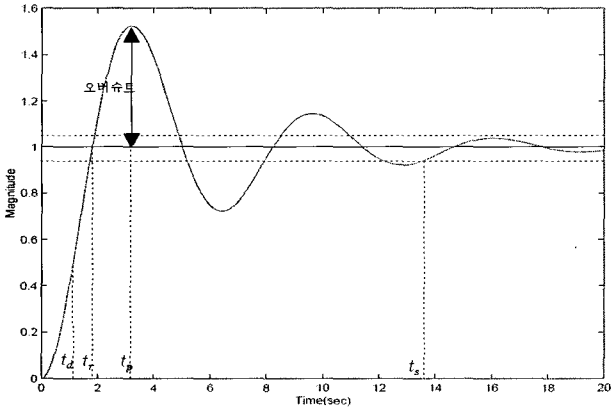

그림 2. 전형적인 2차 시스템의 응답곡선.

• 2차 시스템에서 감쇠비 ζ의 영향 : 그림 3처럼 고유진동수를 일정하게 놓고 감쇠비를 증가시키면 오버슈트와 정착시간이 감소하나 상승시간이 증가하여 오버슈트와 상승시간은 상충되는 관계를 갖는다. $\zeta > 1$이면 과제동, $\zeta = 1$이면 임계제동, $0 < \zeta < 1$이면 부족제동이라 부른다.

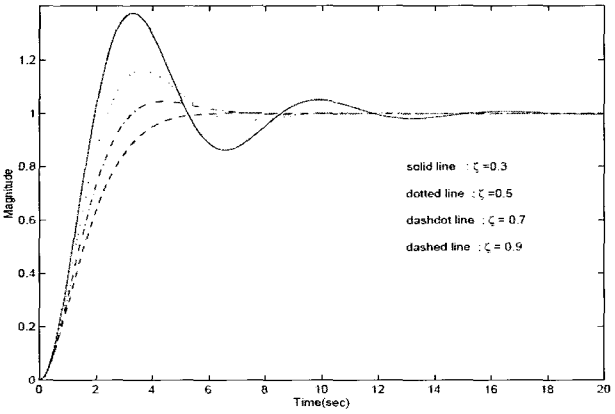

그림 3. ζ값에 따른 시스템 $1/(s^2+2\zeta s+1)$에 대한 단위계단응답의 변화.

• 2차 시스템에서 고유진동수 w_n의 영향 : 그림 4처럼 감쇠비를 일정하게 놓고 고유진동수를 증가시키면 응답의 진동주기, 정착시간, 상승시간이 감소한다.

• 0점의 영향 : $(z-s)/z(s^2+s+1)$를 고려하자. 그림 5의 위쪽 그림처럼 좌반 평면 0점을 첨가하였을 때 허수축에서 멀면($z \ll 0$) 별 영향이 없으나 허수축에 가까우면 ($-\zeta w_n \ll z < 0$) 오버슈트를 증가시키는 경향이 있다. 그림 5의 아래 그림처럼 우반 평면 0점을 첨가하면 언더슈트(undershoot)가 생기는데 허수축에서 멀면($z \gg 0$) 별 영향이 없으나 허수축에서 가까우면($0 < z \ll \zeta w_n$) 언더슈트가 매우 커져 성능이 나빠진다. 일반적으로 허수축에 영점이 가까우면 오버슈트가 굉장히 증가하고 상승시간과 최대값 시간이 감소하고 BW가 증가한다.

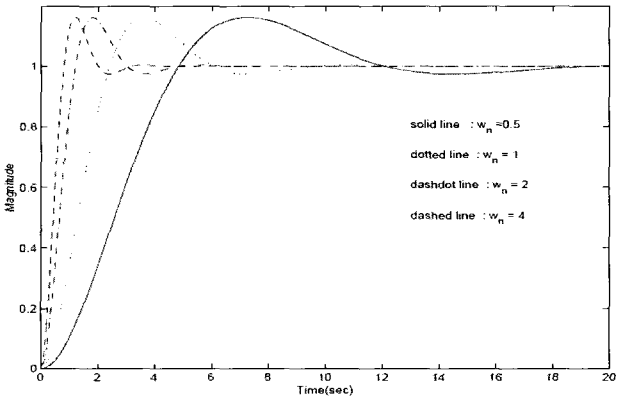

그림 4. w_n값에 따른 시스템 $w_n^2/(s^2+w_n s+w_n^2)$에 대한 단위계단응답의 변화.

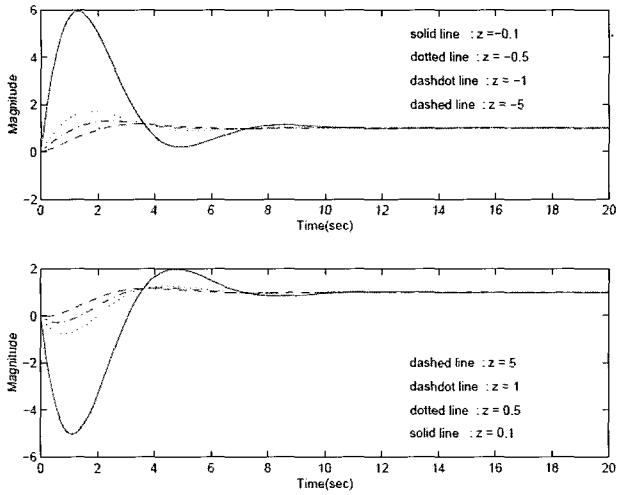

그림 5. z값에 따른 시스템 $\dfrac{(z-s)}{z(s^2+s+1)}$에 대한 단위계단 응답 (위: $z<0$, 아래 $z<0$).

• 우세근(dominant pole) : 허수축에 제일 가까운 극점을 우세근이라 하는데 그림 6 처럼 응답은 우세근에 좌우된다. 우세근보다 허수축에서 4배이상 멀리 떨어진 좌반면의 극점은 무시할 수 있고 보통 대역폭(BW)은 우세근으로 시상수 T는 우세근의 역수로 근사화 할 수 있다.

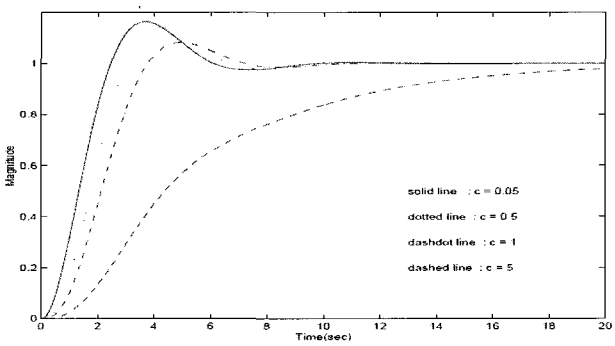

그림 6. c값에 따른 시스템 $1/(cs+1)(s^2+s+1)$에 대한 단위 계단 응답.

• 극점에 따른 응답 특성 : 앞의 해석을 참조해서 시스템의 극점에 따라 아래 그림 7과 같이 극점이 복소평면 좌반면에 위치하면 입력을 임펄스로 했을 때의 응답인 임펄스 응답이 0으로 가되 허수축에서 멀면 급속히 안정상태로 됨을 알 수 있다. 우반면에 위치하면 임펄스 응답은 발산하되 허수축에서 멀면 급속히 발산함을 알 수 있다. 복소수근을 갖으면 진동이 존재하고 실수축에서 멀면 진동의 주기가 커짐을 알 수 있다.

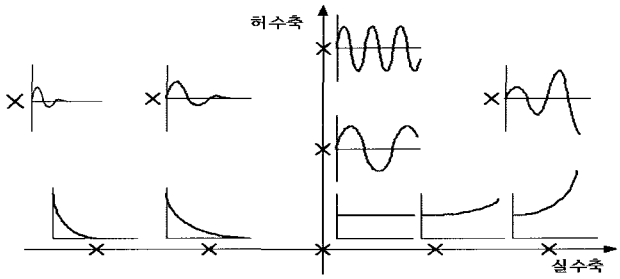

그림 7. 극점에 따른 임펄스 응답 특성.

3. 궤환 시스템의 장단점

• 일반적인 궤환 제어시스템 : 그림8처럼 주어진다. $r(s)$는 시스템의 원하는 기준입력이고, $e(s)$ 오차신호이고, $d(s)$는 임의적이거나 결정론적으로 알려지거나 알려지지 않은 외란 입력을 의미하고 모델 불확실성이나 비선형성 등을 포함한다. $n(s)$는 센서 또는 임의적인 높은 주파수의 신호에 의한 잡음을 의미한다.

• 폐루프의 출력 :

$$y(s)=\frac{GK}{1+GK}r(s)+\frac{1}{1+GK}d(s)+\frac{GK}{1+GK}n(s)$$

• 루프이득(Loop gain) : $L(s)=K(s)G(s)$

• Return difference :

$$J(s)=1+G(s)K(s)=1+L(s)$$

• 감도 : $S(s)=\dfrac{1}{1+G(s)K(s)}=\dfrac{1}{J(s)}$

• 상보 감도 : $T(s)=\dfrac{GK}{1+GK}$

• 오차신호 : $e(s)=S(s)[r-d]+T(s)n$

• 입력신호 : $u(s)=K(s)S(s)[r-d-n]$

• 외란제거 성능 : 외란의 영향을 줄이기 위해서는 감도 $S(s)$를 줄여야 즉 $\|GK\|=\|L\|\gg 1$을 유지해야 한다.

• 추적 성능 : 오차를 줄이기 위해서는 감도 $S(s)$를 줄이면 된다.

• 잡음제거 성능 : 잡음을 줄이기 위해서는 $T(s)$를 줄여야 하나 $T+S=1$로 잡음제거를 위해 $T(s)$를 줄이면 감도 $S(s)$가 증가해 외란제거와 추적 성능이 나빠져 서로 상충되는 성능조건이다.

• 입력의 제한 : 시스템에 손상을 주지 않기 위해서 입력의 크기가 제한되어야 하고 이를 위해 $K(s)S(s)$의 크기는 제한되어야 한다. 루프이득 GK가 크면 $K(s)S(s)\approx G^{-1}(s)$이고 일반적으로 $G(s)$는 strictly proper하므로 $G^{-1}(s)$는 높은 주파수에서 매우 값이 커지므로 고주파수에서 입력이 커져 구동기 포화가 일어나는 것에 대비를 해야 한다.

• 일반적인 loop shaping : 일반적으로 외란 $d(s)$과 기준신호 $r(s)$는 낮은 주파수 성분을 갖고 잡음 $n(s)$는 높은 주파수 성분을 갖는다. 그러므로 저주파수에서는 루프이득이 커야 외란제거와 추적 성능을 좋게 할 수 있고 고주파수에서는 루프이득을 작게 해야 잡음 제거 성능을 좋게 할 수 있다 중간 주파수에서는 시스템의 안정을 위한 여유조건을 갖도록 루프이득을 조절한다.

• 궤환의 장점 : 불확실성에 대한 강인성, 잡음 신호 제거 성능, 추적성능 등을 증가시킬 수 있다.

• 궤환의 단점 : 복잡성이 증가한다. 시스템 이득이 감소한다. 불안정해질 수 있다.

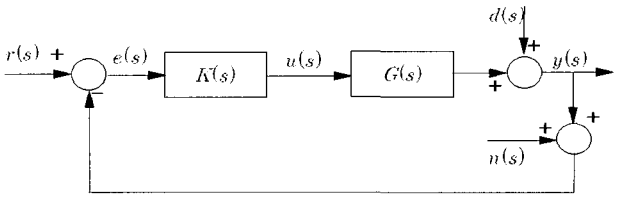

그림 8. 일반적인 궤환 제어시스템의 블록선도.

4. Routh-Hurwitz 안정성 판별법.

- 안정성의 정의 : 임의의 ε에 대하여 시스템의 초기상태가 $\|x(0)\|<\delta$로 주어지기만 하면 그 이후의 상태가 $\|x(t)\|<\varepsilon$를 만족시키도록 하는 δ가 존재하면 그 시스템은 안정하다고 한다.
- 선형시스템의 안정성 : 선형시스템은 특성근이 복소평면의 좌반면에 위치하면 안정하다.
- 판별배열 작성법 : 주어진 특성방정식에 대해 아래와 같은 배열을 만든다.

$$\begin{array}{c|cccc} s^n & a_n & a_{n-2} & a_{n-4} & \cdots \\ s^{n-1} & a_{n-1} & a_{n-3} & a_{n-5} & \cdots \\ s^{n-2} & b_{n-1} & b_{n-3} & b_{n-5} & \cdots \\ s^{n-3} & c_{n-1} & c_{n-3} & c_{n-5} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ s^0 & h_{n-1} & & & \end{array},$$

여기에서 $b_{n-1}=-\dfrac{1}{a_{n-1}}\begin{vmatrix} a_n & a_{n-2} \\ a_{n-1} & a_{n-3}\end{vmatrix}$,

$$b_{n-3}=-\frac{1}{a_{n-1}}\begin{vmatrix} a_n & a_{n-4} \\ a_{n-1} & a_{n-5}\end{vmatrix}$$

$\cdots, c_{n-1}=-\dfrac{1}{b_{n-1}}\begin{vmatrix} a_{n-1} & a_{n-3} \\ b_{n-1} & b_{n-3}\end{vmatrix}$으로 주어지는 데 마찬가지 방식으로 배열을 완성한다.

- 판별법 : 특성방정식에서 양의 실수부분을 가지는 근의 개수는 판별배열의 첫째 열에서 부호변화 횟수와 같으므로 안정하려면 부호변화가 없고 0이 아니어야 한다.
- 상대 안정성 판별 : 만약 특성 근이 $-\alpha(\alpha>0)$보다 작은지 여부를 판별하기 위해서는 특성 방정식의 변수 s를 $s-\alpha$로 대체하여 판별하면 된다.

예제 1: $s^2+Ks+2=0$의 근이 -1보다 작게 될 K값의 범위를 구해보자. 특성 방정식의 변수 s를 $s-1$으로 치환해 $(s-1)^2+K(s-1)+2=s^2+(K-2)s+3-K=0$를 얻고 다음의 판별배열을 얻을 수 있다.

$$\begin{array}{c|cc} s^2 & 1 & 3-K \\ s & K-2 & 0 \\ 1 & 3-K & \end{array}$$

결국 부호변화가 없으려면 $3>K>2$를 만족시켜야 됨을 알 수 있다.

5. 근궤적 기법

- 근궤적을 위한 시스템의 개루프 전달함수의 일반 형태 :

$$G(s)=\frac{(s-z_1)\cdots(s-z_m)}{(s-p_1)\cdots(s-p_n)}$$

- 근궤적을 위한 시스템의 루프이득 :

$$L(s)=K\frac{(s-z_1)\cdots(s-z_m)}{(s-p_1)\cdots(s-p_n)}$$

- 근궤적의 정의 : K의 변화에 따른 폐루프 전달함수 $L/(1+L)$의 극점 변화를 그린 것이 근궤적이다. 그러므로 이를 이용하면 특성근이 원하는 복소평면에 위치시키기 위한 이득을 찾아 제어기를 설계할 수 있다.
- 일반적인 극점에 따른 응답의 특성 변화 : 우세근이 $-\alpha\pm j\beta$로 주어질 때 극점의 변화에 따른 응답의 변화는 다음과 같다. 여기에서 t_p는 최대값시간, t_s는 정착시간, t_r은 상승시간, M_p는 오버슈트, BW는 오버슈트이다.
  ① β를 고정시키고 α를 증가시키면 t_p는 고정되고 t_s, t_r, M_p는 감소하고 BW는 증가한다.
  ② α를 고정시키고 β를 증가시키면 t_s는 고정되고 t_p, t_r은 감소하고 M_p, BW는 증가한다.
  ③ α, β를 동시에 같은 비율로 증가시키면 M_p는 고정되고 t_r, t_p, t_s는 감소하고 BW는 증가한다.
- 적절한 극점의 위치 : 원하는 오버슈트를 M_p, 원하는 정착시간을 t_s라 하면 $M_p=e^{\zeta\pi\sqrt{1-\zeta^2}}$, $t_s\approx 4/\zeta w_n$에서 $\theta<\cos^{-1}\sqrt{(\ln M_p)^2/[\pi^2+(\ln M_p)^2]}$, $\zeta w_n>4/t_s$를 얻어 그림 9와 같은 극점 범위를 얻을 수 있다.

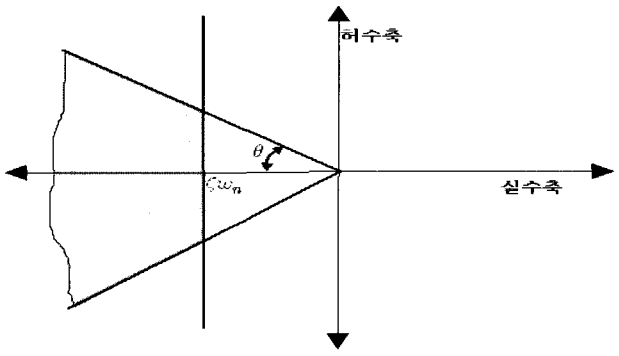

그림 9. 적절한 근의 위치.

- 근궤적의 특성 : 근궤적은 다음과 같은 특성을 갖는다.
  ① 근궤적의 수 : 특성방정식의 근의 개수와 같은 n이다.
  ② 근궤적의 출발점과 종착점 : 근궤적은 $K=0$일 때 개루프 극점 $p_1,\cdots p_n$에서 출발하여 $K=\infty$일 때 개루프 영점 $z_1,\cdots z_m$과 무한대에 위치한 영점에서 끝난다.

③ 실수축상의 근궤적 : 실수축상 임의의 점은 우측에 있는 개루프 전달함수 $G(s)$의 극점과 영점을 합한 개수가 홀수이면 근궤적상에 있고 만약 짝수이면 그 점은 근궤적에 위치하고 있지 않다.

④ 근궤적의 대칭성 : 근궤적은 실수축에 대하여 대칭이다.

⑤ 점근선 : 중심점은 $\dfrac{\sum_{i=1}^{n} p_i - \sum_{i=1}^{m} z_i}{n-m}$에 $\dfrac{2k+1}{n-m}\pi(k=0,\pm1,\cdots)$의 각도로 주어지는 점근선 $n-m$개 존재한다.

⑥ 허수축과의 교점 : 다음의 다항식을 만족시키는 주파수 값 w와 이득 K에서 허수축과 만난다. Routh-Hurwitz 판별배열을 이용해 부호가 변하는 곳을 찾아 교점을 찾을 수도 있다.

$$(jw-p_1)\cdots(jw-p_n)+K(jw-z_1)\cdots(jw-z_m)=0$$

6. 보데 선도 등 여러 가지 선도

- 주파수 응답 : 전달함수 $G(s)$에 $s=jw$를 대입한 $G(jw)$를 말한다.
- 주파수 응답의 특성 : 전달함수 $G(s)$의 선형시스템에 $\cos wt = Re(e^{iwt})$를 입력하였을 때 영상태 응답은

$$y(t)=Re\left(LAPLACE^{-1}\left[G(s)\frac{1}{s-jw}\right]\right)=$$

$$Re\left(\int_0^t g(\tau)e^{jw(t-\tau)}d\tau\right)=Re\left(e^{jwt}\int_0^t g(\tau)e^{-jw\tau}d\tau\right)$$

정상 상태 즉 $t\to\infty$이면 위식은 출력이 다음처럼 주어짐을 의미한다.

$$y(t)=Re(G(jw)e^{jwt})=Re(|G(jw)|e^{jwt+\angle G(jw)})$$

결국 입력 $\cos wt$에 대하여 정상상태의 출력은 $y(t)=|G(jw)|\cos(wt+\angle G(jw))$로 주어짐을 알 수 있다. 주파수 응답의 크기만큼 증폭되고 주파수 응답의 위상만큼 차가 남을 알 수 있다.

- Bode 선도 : 주파수 응답의 크기 $|G(jw)|$와 위상 $\angle G(jw)$를 주파수 w에 대하여 그린 것으로 주파수 w는 대수눈금으로 x축 변수로, 응답의 크기는 db(상용대수 값의 20배)로, 위상은 각도 단위로 y축에 보통 그린다.
- Bode 선도의 장점 : 다음과 같은 장점을 갖는다.

① 주파수 응답이 여러 개 기본 요소들의 곱으로 표시되므로 보데선도에서 응답의 크기는 각 기본 요소 크기의 합으로, 위상도 각 기본 요소의 위상의 합으로 표시되 복잡한 시스템의 경우에도 쉽게 얻을 수 있다.

② 점근선을 가지고 실제의 보데 선도를 근사시켜 쉽게 구할 수 있다.

③ 광범위한 주파수 범위에서 응답을 표시할 수 있다.

6.1. 보데선도의 기본 요소

- 상수의 보데 선도 : 상수 K는 크기가 모든 주파수에 대하여 $20\log K$의 일정한 값을 갖고 위상도 모든 주파수에 대하여 0도이므로 보데선도의 크기 선도는 일정한 $20\log K$를 갖는 주파수축에 평행한 직선으로 위상은 0을 갖는 주파수축에 평행한 직선으로 표시된다.
- 미, 적분 요소의 보데 선도 : 미분 요소 $G(jw)=jw$는 크기가 $20\log w$ 로 크기선도에 기울기 20[dB/dec]의 직선으로 나타나고 위상은 90도로 주어져 위상선도에 크기 90도를 갖는 주파수축에 평행한 직선으로 표시된다. 마찬가지로 적분요소와 이중적분, 이중 미분에 대한 보데 선도를 그림 10처럼 구할 수 있다.

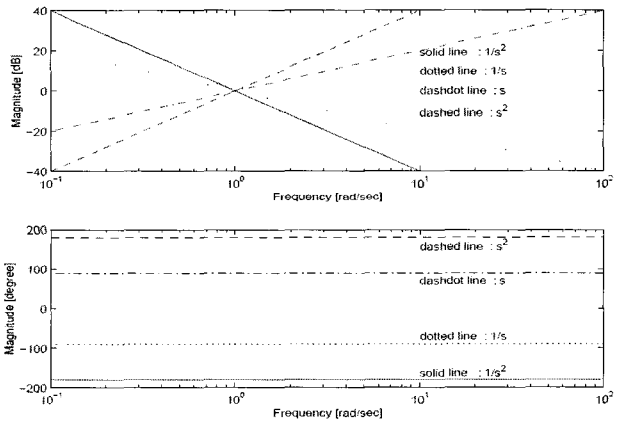

그림 10. $1/s^2$, $1/s$, s, s^2의 보데 선도.

- 영점의 보데 선도 : $G(jw)=1+jw/w_1$의 보데선도는

① $w \ll w_1$일 때는 $20\log|G(jw)|\approx 0, \angle G(jw)\approx 0^o$

② $w \gg w_1$일 때는 $20\log|G(jw)|\approx 20\log(w/w_1), \angle G(jw)\approx 90^o$의 근사를 써 그림 11을 얻을 수 있다.

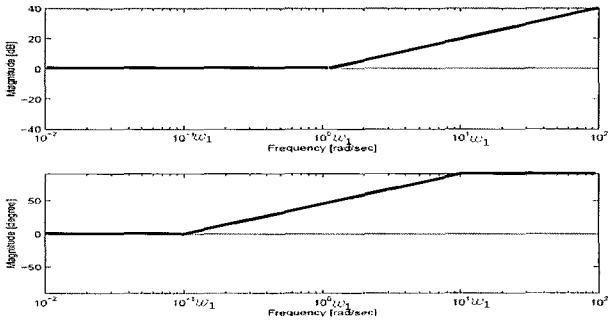

그림 11. 영점 $1+s/w_1$의 보데 선도 근사화.

• 극점의 보데 선도 : $G(jw)=1/(1+jw/w_1)$의 보데선도는

① $w \ll w_1$일 때 $20\log|G(jw)| \approx 0$, $\angle G(jw) \approx 0^o$

② $w \gg w_1$일 때는 $20\log|G(jw)| \approx -20\log(w/w_1)$,

$\angle G(jw) \approx -90^o$의 근사를 써 그림 12처럼 얻을 수 있다.

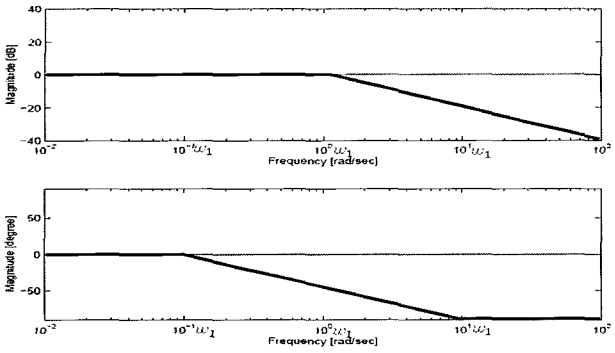

그림 12. 극점 $1/(1+s/w_1)$의 보데 선도 근사화.

• 2차 요소의 보데선도 : $G(jw)=\dfrac{1}{(1+2\zeta jw/w_n-(w/w_n)^2)}$

로 주어질 때 만약 $0.1<\zeta<1$인 경우 ① $w \ll w_n$일 때는 $20\log|G(jw)| \approx 0$, $\angle G(jw) \approx 0$, ② $w \gg w_n$일때는 $20\log|G(jw)| \approx -40\log(w/w_n)$, $\angle G(jw) \approx -180^o$의 근사를 써 그림 4.19을 얻을 수 있다.

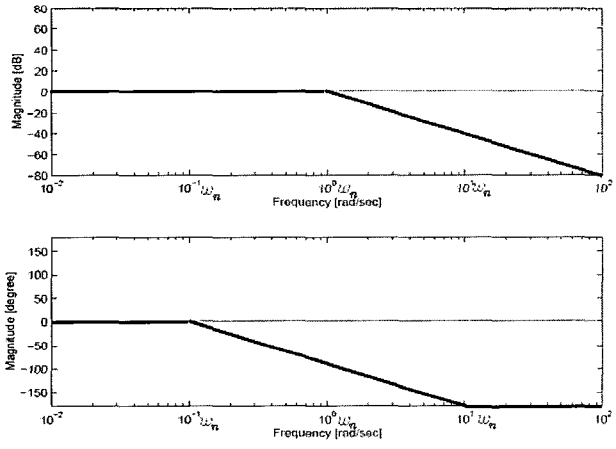

그림 13. 2차요소 $w_n^2/(s^2+2\zeta w_n s+w_n^2)$의 보데 선도 근사화 $(0.1<\zeta<1)$.

6.2. 보데 근사화 선도 그리는 법

① 전달함수 $G(jw)$를 기본 요소의 곱으로 표시한다.

② 기본 요소들의 절점 주파수 w_1혹은 w_n을 계산한다.

③ 각 기본 요소의 근사화 선도의 합을 구하여 전달함수의 보데 근사화 선도를 그린다.

6.3. 보데선도에서 주요값들

• DC 이득 : 주파수가 0일 때 오픈 루프 이득 즉 $|G(0)|$로 단위계단입력을 가했을 때 정상상태 응답과 같다.

• 대역폭 : 일반적으로 $G(jw)$의 크기가 $20\log|G(0)|-3$ [dB]가 되는 주파수 즉 $|G(jw)|$가 $|G(0)|/0.707$로 감소하는 주파수 값으로 대역폭이 넓으면 응답속도가 빠르다. 근사적으로 시간영역 성능에서 다루는 시상수 T의 역수이다.

• 공진 최대값 : 오픈 루프 이득의 최대값으로 보통 M_r로 표기한다. 감쇠비 ζ와 관련이 있어 2차 요소의 경우의 $0<\zeta<0.707$ 범위에서 $M_r=(2\zeta\sqrt{1-\zeta^2})^{-1}$로 주어진다. 공진 최대값이 커지면 오버슈트가 커지며 설계사양으로 자주 고려된다. 1.1-1.5 정도의 값이 적절하다.

• 공진주파수 : 오픈 루프 이득이 최대가 되는 주파수로 보통 w_r로 표기하며 2차요소의 경우 $0<\zeta<0.707$ 범위에서 공진이 존재하고 $w_r=w_n\sqrt{1-2\zeta^2}$의 관계식으로 표현된다.

• 위상 통과 주파수 : $G(jw)$의 위상이 -180도가 되는 주파수를 의미하며 w_p로 표기한다.

• 이득 통과 주파수 : $20\log|G(jw)|$가 0[dB]가 되는 주파수를 의미하며 w_g로 표기한다. 보통 이때의 위상은 -180도보다 커야 안정하다. $\angle G(jw_g)>-180^o$을 만족해야 일반적으로 안정하다.

• 이득 여유(Gm : Gain Margin) : w_p에서의 이득값과 0 [dB]의 차이로 $GM=-20\log G(jw_p)$로 정의된다. 보통 Gm이 양수이면 $G(jw)$는 안정하고 음수이면 불안정하다. 설계시 보통 6~20 [dB]를 갖도록 한다.

• 위상 여유(Pm : Phase Margin) : $Pm=\angle G(jw_g)+180$으로 보통 Pm이 양수이면 안정하다. 보통 Pm은 감쇠율 ζ과 비례하여 대략적으로 $Pm \approx 100\zeta$의 관계식을 만족시킨다. 설계시 보통 30~60도를 갖도록 한다.

• 적절한 보데 선도의 모양 : 그림 8과 같이 주어지는 일반적인 궤환제어시스템에서 폐루프 시스템 $GK/(1+GK)$의 특성이 좋으려면 외란 $d(s)$과 기준신호 $r(s)$는 낮은 주파수 성분을 갖고 잡음 $n(s)$는 높은 주파수 성분을 가지므로 저주파수에서는 루프이득 GK가 커야 외란제거와 추적 성능을 좋게 할 수 있고 고주파수에서는 GK를 작게 해야 잡음 제거 성능을 좋게 할 수 있다 중간 주파수에서는 이득여유 Gm이 6~20 [dB]를 갖고 위상여유 Pm이 30~60도를 갖고 $20\log|G(jw)K(jw)|$의 감소비율은 w_g근처에서 -20 [dB/dec]이 되도록 하면 좋다.

6.4. Nyquist 선도 (Polar 선도)

- 나이키스트 선도 : $G(jw)$의 크기와 위상을 동시에 극좌표상에 표시하여 주파수가 0에서 무한대로 변할 때의 $G(jw)$를 극좌표상에 표현한 것으로 극좌표 선도(polar plot)이라고도 한다.
- 1차 요소의 나이키스트 선도 : $G(s)=1/(1+s/w_1)$의 나이키스트 선도는 그림 14의 왼쪽과 같이 복소평면상의 점 (0.5, 0)을 중심으로 하는 반지름 0.5의 반원이 된다.
- 2차 요소의 나이키스트 선도 : 2차 요소의 전달함수가 $G(s)=1/(1+2\zeta s/w_n+(s/w_n)^2)$로 주어지며 $\zeta>0$일때 다양한 ζ값의 변화에 따른 나이키스트 선도가 그림 14의 오른쪽에 주어졌다. 주파수가 0일 때 점 $1\angle 0^o$로 출발하여 주파수가 무한대일 때 $0\angle -180^o$로 끝난다. ζ가 커지면 점점 반원모양에 가까워짐을 알 수 있다. $\zeta \gg 1$이면 거의 1차 요소처럼 된다.

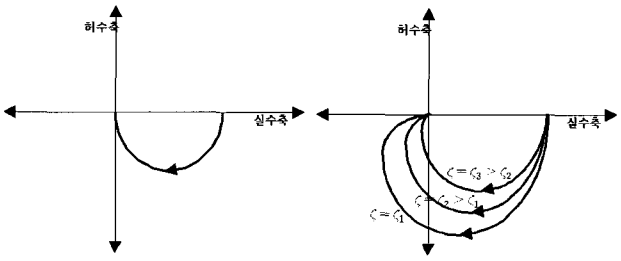

그림 14. $1/(1+jw/w_1)$ (왼쪽)과 $1/(1+2\zeta jw/w_n-(w/w_n)^2)$ (오른쪽)의 나이키스트 선도.

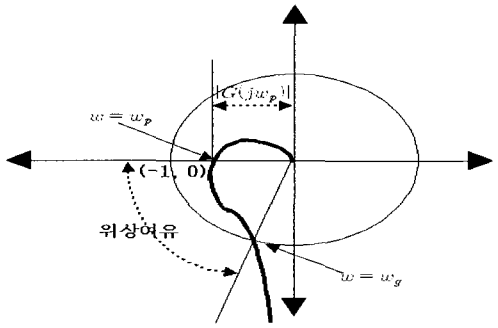

그림 15. 나이키스트 선도에서의 위상여유와 이득 여유.

- 일반적인 나이키스트 선도 : 1차 요소나 2차 요소에 $1/s$가 곱해져 0에 위치한 극점이 한 개 더해지면 나이키스트 선도는 90도 반시계방향으로 회전하고 음의 허수축 방향에서 무한대의 크기를 가지고 출발한다. 만약 $1/s^2$가 곱해지면 180도 반시계방향으로 회전하고 음의 실수축 방향에서 무한대의 크기를 가지고 출발한다. 1차 요소나 2차 요소에 $1/(1+s/w_2)$가 곱해져 좌반평면에 위치한 극점이 한 개 더해지면 나이키스트 선도는 출발점은 바뀌지 않고 주파수가 점점 커지면서 90도 반시계방향으로 나이키스트 선도가 확대된다.
- 이득 여유와 위상 여유 : 나이키스트 선도 상에서 이득여유와 위상여유를 구할 수 있는데 그림 15를 참조하면 원점을 중심으로 하는 반지름 1의 단위원과 만나는 점에서 주파수가 이득 통과 주파수 w_g로 $G(jw_g)$와 음의 허수축이 이루는 각도가 위상 여유 Pm이며 음의 허수축과 나이키스트 선도가 만나는 점에서의 주파수가 위상 통과 주파수 w_p로 그 때의 크기 $|G(jw_p)|$를 이용하여 이득 여유를 구할 수 있다.
- 나이키스트 안정도 판별법 : 그림 15와 같은 궤환 제어시스템에서 반시계 방향으로 극좌표상의 점 (-1, 0)을 둘러싸는 수와 양의 실수를 갖는 루프이득 $G(s)K(s)$의 극점의 수가 같으면 궤환 제어시스템은 안정하다. 그리고 상대적인 안정도는 위상여유와 이득여유의 크고 작음을 통해 판별할 수 있다.
- 보데선도와 나이키스트 선도의 비교 : 보데 선도를 통한 해석이나 설계는 수학적 모델 없이 실험적으로 얻어진 주파수 응답을 사용할 수 있다는 장점으로 많은 다양한 시스템 특히 극좌표상에서 우반면에 위치하는 극점이나 영점이 없는 시스템(이득과 위상여유가 양수인 시스템)의 안정성을 결정하고 PID, 진상, 지상 제어기 등 고전적인 제어기 설계에 성공적으로 쓰여 왔으나 극좌표상에서 우반면에 위치하는 극점이나 영점이 존재하는 시스템의 안정성을 위해서는 사용하기 어려운데 반해 나이키스트 선도는 상관없이 사용될 수 있다.

6.5. Nichols 선도

- 니콜스 선도 : 주파수 응답 $G(jw)$의 log 크기를 y 축으로 위상을 x축으로 하여 한꺼번에 그린 선도이다.
- 니콜스 선도의 장점 : 니콜스 선도를 사용하면 손쉽게 위상여유와 이득여유를 알 수 있다. 또한 그림 8처럼 주어지는 궤환 시스템에서 궤환이득이 $K(s)=1$인 경우 즉 폐루프 궤환 시스템 이득이 $G/(1+G)$로 주어지는 경우의 이득에 대한 일정한 크기 궤적과 일정한 위상궤적이 보조적으로 그려져 있어 개루프 주파수 응답 $G(jw)$으로부터 쉽게 폐루프 주파수 응답을 결정할 수 있어 보상기 설계에 유용하다.

7. Matlab 명령어 소개

7.1. 일반적인 명령어 형태

- 명령어의 일반형태는 아래와 같다.

  [OUT1, OUT2, ..., OUTn] = CMDname(IN1, IN2, ... , INm)

  (예:[A,B,C,D] = tf2ss(num,den) % 전달함수 num/den의 상태공간 행렬 A,B,C,D를 돌려준다)

- OUTx : 전달 받을 데이터값 변수들은 [와]로 묶어 준다. 변수 argument 이름은 1개 이상의 영문자와 숫자의 조합이 가능하며 대문자와 소문자 구별을 하며 첫 글자는 반드시 영문자로 해야 한다. 당연히 명령어 등을 변수 이름에 사용하면 안된다. [OUT1, OUT2, ...] = 부분은 생략이 가능한데 생략하면 전달받을 여러 개의 변수 argument에서 첫 번째 변수값만 화면에 표시된다. 전달받을 변수가 하나인 경우 [와]는 생략 가능하다. 즉 OUT1 = CMDname(IN1, IN2, ... , INm) 도 가능하다.
- INx : 명령 함수에 전달해줄 변수 argument 이름을 의미한다.
- CMDname : 명령어 이름으로 help 명령을 치면 리스트를 알 수 있다.
- [OUT1, OUT2, ... , OUTn] = CMDname(IN1, IN2, ... , INm) ; (세미콜론) : 명령어 다음에 ;로 끝을 맺으면 결과값의 표시를 억제한다.

7.2. 기본적인 명령

- help : 명령 리스트를 표시해준다.
- help CMDname : 명령어 CMDname에 관한 정보를 표시해준다.
- lookfor name : name와 관련된 정보를 포함한 모든 명령어의 리스트를 보여준다.
- demo : 데모의 리스트를 표시해준다.
- diary FILEname : matlab 메인창에서 입력된 것과 표시된 것을 FILEname 에 기록한다. 데이터를 기록하지는 않는다.
- diary : matlab 메인창에서 입력된 것과 표시된 것을 matlab/bin 디렉토리에 diary라는 이름의 파일에 기록한다. diary 명령을 홀수번 입력하면 on되고 짝수번 입력하면 off 된다. off 하기 위해 diary off라고 입력해도 된다.
- % : 주석을 삽입할 때 사용하며 % 다음에 쓰인 것은 같은 줄에 있으면 무시된다.
- save FILEname : matlab 메인창에서 사용된 데이터를 FILEname라는 이름으로 binary matlab format으로 기록한다. 확장자명이 명기되지 않을 경우 FILEname.mat에 기록되고 FILEname을 명기하지 않으면 matlab.mat라는 파일에 기록된다.
- save FILEname VARname : FILEname에 변수값 데이터 VAREname를 기록한다.
- save FILEname /ascii : 데이터를 ascii format으로 long type으로 기록한다.
- load FILEname : FILEname에 저장된 데이터를 불러들인다.
- what : 현재 디렉토리의 matlab 파일과 mat 데이터 파일의 리스트를 보여준다.
- who : 변수명의 리스트를 보여준다.
- whos : 변수명, 차원, 메모리 할당 상황, 복소수 여부 emddmf vytlgksek.
- dir : 현재 디렉토리의 모든 파일의 리스트를 보여준다.
- chdir : dos의 chdir 명령과 동일하다.
- delete : dos의 del 명령과 동일하다.
- clear VARname : VARname 변수를 제거한다. 변수명을 명기하지 않으면 모든 변수를 지운다.
- ! : !다음에 os에 사용되는 명령을 치면 그 명령이 수행된다.
- exit , quit : matlab의 수행을 종료시킨다.

7.3. 벡터와 행렬

- VARname = [n1 : n2 : n3] : 변수명 VARname의 행벡터에 초기값 n1부터 최종값 n3까지 증감분 n2의 간격을 갖는 값을 할당한다. 증감분 n2 없이 초기값 n1과 최종값 n3만을 쓴 경우에는 1이 증가분이다. 증감분 n2가 양수인 경우 최종값 n3에 딱 떨어지지 않은 경우 최종값 n3보다 작은 최대값이 마지막 값이 된다. 증감분n2가 음수인 경우 최종값 n3보다 큰 최소값이 마지막 값이 된다.

  예 : x = [0 : 0.1 : 0.5]

  % x=[0 0.1 0.2 0.3 0.4 0.5] 와 동일한 명령

  예 : x = [0 : 3]; % x=[0 1 2 3]과 동일한 명령
- VARname = linspace(n1,n2, n) : 변수명 VARname으로 주어지는 행벡터에 n1 부터 n2까지 길이 n개를 갖는 값을 할당한다. n값을 쓰지 않으면 100개가 할당된다.
- VARname = logspace(n1,n2,n) : 변수명 VARname으로 주어지는 행벡터에 10의 n1승 값부터 10의 n2승 값까지 길이 n개를 갖는 값을 할당한다. n값을 쓰지 않으면 50개가 할당된다.
- 복소수의 표현 : 허수부분을 표현하기 위해 i나 j를 사용한다.
- pi , Inf, Nan : 원주율, 유한한 값을 0으로 나눈 값, 0 나누기 0처럼 무정의 값을 각각 의미한다.
- 행렬의 입력 : [로 시작해서 1행부터 입력을 시작한다. 각 값들은 공백이나 ,(쉼표)를 사용하여 구별한다. 각 행의 구별은 ;를 사용해서 한다. 마지막 행 값을 입력하고]로 끝낸다.

  예 : x = [1 2+2j; 2-2j 4];

  % x의 1행은 1과 2+2j가 할당되고 2행은 2-2j와 4가 할당됨
- ... : 매우 긴 명령이나 변수를 한 줄에 입력하지 못하고 다음 줄에 연속해서 입력할 때 사용한다.

  예 : x = [1 2 3 4 5 ...

  6 7 8 9]% x=[1 2 3 4 5 6 7 8 9] 와 동일한 명령
- '(어퍼스트로피) : 벡터나 행렬의 conjugate transpose 그리고 unconjugate transpose는 .'를 사용한다.

- size(VARname) : VARname의 열과 행수를 행벡터 형태로 돌려주는 함수
- length(VARname) : VARname의 길이값을 돌려주는 함수로 행렬인 경우 열과 행수에서 큰 값을 돌려준다.
- 승수의 표시 : e를 사용하여 한다. (예: 100은 1e2 혹은 1E2, 0.1은 1e-1 혹은 1E-1 등으로 표현가능)
- MTXname(m,n) : 행렬명 MTXname의 m행 n렬 값으로 m과 n은 양수이면 정수가 아니어도 상관없으며 사사오입된다. m과 n은 벡터여도 된다.
- MTXname(:) : 행렬을 벡터형태로 바꾼다.
  예 : x = [1 2; 3 4]; a= x(:);
  % a=[1 3 2 4]'의 형태가 된다.
- MTXname(:,i) : 행렬명 MTXname의 i렬 벡터값을 의미한다.
- MTXname(i,:) : 행렬명 MTXname의 i행 벡터값을 의미한다.
- MTXname(k:j,i) : 행렬명 MTXname의 i렬에서 k행부터 j행까지 값을 취한 열벡터값을 의미한다.
- MTXname(i,k:j) : 행렬명 MTXname의 i행에서 k열부터 j열까지 값을 취한 행벡터값을 의미한다.
- MTXname(k:j,i:l) : 행렬명 MTXname의 i렬부터 l렬의 k행부터 j행까지 취한 부분 행렬을 의미한다.
  예 : x=[1 2 3;4 5 6;7 8 9;0 1 2]; a= x(1:2,2:3);
  % a=[2 3;5 6]의 형태가 된다.

7.4. 특수한 행렬

- eye(m) : $m \times m$ 항등행렬
- eye(m,n) : m과 n중 작은 값과 같은 항등행렬과 나머지부분은 0으로 채워지는 $m \times n$행렬을 정의
  예 : a=eye(2,3);% a=[1 0 0;0 1 0]의 형태가 된다.
- ones(m) : 모든 엔트리가 1인 $m \times m$행렬
- ones(m,n) : 모든 엔트리가 1인 $m \times n$행렬
- zeros(m) : 모든 엔트리가 0인 $m \times m$행렬
- zeros(m,n) : 모든 엔트리가 0인 $m \times n$행렬
- rand(m,n) : 임의로 뽑은 엔트리값을 갖는 $m \times n$행렬
- randn(m,n) : 정규분포를 갖는 엔트리값을 갖는 $m \times n$행렬
- diag(VECname) : 벡터명 VECname의 값들을 다이아고널에 갖는 다이아고널 행렬의 정의
  예 : a=[1 2 3]; b=diag(a)
  % b=[1 0 0;0 2 0;0 0 3]의 형태가 된다.
- diag(MTXname) : MTXname를 갖는 행렬의 다이아고널 값을 열벡터의 형태로 돌려준다.
  예 : x=[1 2 3;4 5 6;7 8 9;0 1 2]; a= diag(x);
  % a=[1;5;9]의 열벡터가 된다.
- diag(MTXname,m) : m이 양수이면 MTXname를 갖는 행렬의 m번째 수퍼다이아고널 값을 열벡터의 형태로 돌려준다. m이 음수이면 -m번째 서브다이아고널 값을 열벡터의 형태로 돌려준다.
  예 : x=[1 2 3;4 5 6;7 8 9;0 1 2]; a= diag(x,1); %
  a=[2;6]의 열벡터가 된다.
- triu(MTXname) : 행렬 MTXname의 upper-triangular 행렬값을 돌려준다.
- tirl(MTXname) : 행렬 MTXname의 lower-triangular 행렬값을 돌려준다.
  예 : x=[1 2 3;4 5 6;7 8 9;0 1 2]; a= triu(x);
  % a=[1 2 3;0 5 6; 0 0 9;0 0 0]의 matrix가 된다.

7.5. 수학 연산자 및 함수

- *, +, - : 곱하기, 더하기, 빼기 연산자
- ^ : 승수 연산자 즉 x^-1.2 는 x라는 변수의 -1.2승을 의미한다.
- 나누기 연산자 /와 \ : 스칼라의 연산에서는 똑같다. 그러나 행렬간의 연산에서 A/B의 결과는 A*inv(B)와 같아 선형식 xB=A의 해값 x와 같다. A\B의 결과는 inv(A)*B와 같아 Ax=B의 해값 x와 같다.
- (* 혹은 /, \, ^) : 행렬연산의 경우 각 엔트리끼리 수행된다.
  예 : a=[1 2;3 4].*[1 0;0 1]
  % a=[1 2;3 4]가 아닌 a=[1 0;0 4]의 matrix가 된다.
- 기본 함수 : abs(x), acos(x), asin(x), atan(x), conj(x), cos(x), exp(x), angle(x), imag(x), real(x), log(x), log10(x), round(x), sqrt(x), sin(x), tan(x)는 각기 x라는 벡터나 행렬의 절대값, 아크코사인, 아크사인, 아크탄젠트, 공액복소수, 코사인, 지수함수, 위상각도, 허수부분, 실수부분, 로그, 상용로그, 반올림, 제곱근, 사인, 탄젠트 값을 되돌려 준다. 주의할 것은 x의 엔트리에 대하여 수행된다는 것이다.
  예 : a=[1 2;3 4]; b=cos(a)
  % b=[cos(1) cos(2); cos(3) cos(4)]의 행렬이 된다.
- 행렬 함수 : expm(A) , logm(A), sqrtm(A) 는 각각 주어진 사각행렬 A에 대하여 e^A, $\log A$, $\sqrt{A}$의 행렬함수 값을 연산해 되돌려준다.
- funm(MTXname,'FUNname') : 주어진 행렬 MTXname에 대한 기본함수 FUNname의 행렬함수값을 연산하여 되돌려 준다.
  예 : a=[1 2;3 4]; b=funm(a,'exp');
  % a=[1 2;3 4]; b=expm(a); 과 결과가 같다.
- 열벡터 분석용 함수 : min, max, mean, median, std, sum, prod, cumsum, cumprod 등 함수는 열벡터값을 해석하기 위한

함수로 행렬에 대해 적용하면 열에 대하여 계산하고 결과를 행벡터로 출력한다. 이들은 벡터의 최소값, 최대값, 평균값, 중간값, 표준편차, 합, 곱, 누적합, 누적곱을 각각 계산한다.

예 : a=[1 2;3 4]; b=min(a);

% b=[1 2] 로 할당된다.

- [Mp, tp] = max(mBYnMTX) : $m \times n$ 행렬 mBYnMTX의 각 열의 최고값을 행벡터의 형태로 Mp에 돌려주고 그때의 인덱스값을 tp에 행벡터의 형태로 돌려준다.
- [Mp, tp] = min(mBYnMTX) : $m \times n$ 행렬 mBYnMTX의 각 열의 최소값을 행벡터의 형태로 Mp에 돌려주고 그때의 인덱스값을 tp에 행벡터의 형태로 돌려준다.

예 : a=[1 2;3 4]; [c,d]=min(a);

% c=[1 2];d=[1 1] 로 할당된다.

- [sx,index] = sort(mBYnMTX) : $m \times n$ 행렬 mBYnMTX의 각 열을 엔트리값의 크기에 따라 오름차순으로 재정렬하여 $m \times n$ 행렬의 형태로 sx에 돌려주고 그 때 사용된 인덱스를 $m \times n$ 행렬의 형태로 index에 돌려준다.

예 : a=[2 4 ;3 1]; [x,id]=sort(a);

% s=[2 1;3 4];d=[1 2;2 1] 로 할당된다.

- yi=spline(xVec, yVec, xi) : 주어진 행벡터값 xVec와 그에 대응하는 yVec값으로부터 함수 yVec =f(xVec)를 추정하고 yi=f(xi)값을 추정하여 되돌려준다.

7.6. 다항식

- 다항식의 표현 : 계수들을 벡터의 형태로 나타낸다. 예로 다항식 s^2+2s+1은 x=[1 2 1]; 라는 명령으로 계수들만 사용해 x에 할당 할 수 있다.
- roots(x) : 벡터 x로 표현된 다항식의 근을 구해 열벡터형태로 되돌려준다.

예 : x=[1 2 1]; y=roots(x);

% y=[-1;-1] 로 할당된다.

- poly(x) : 벡터 x를 근으로 하는 다항식의 계수를 구해 행벡터로 되돌려준다.
- poly(MTX) : 사각행렬 MTX의 특성방정식을 구해 계수를 행벡터 형태로 되돌려준다.
- [U,S,V]=svd(A) : 행렬 A에 대하여 singular 값 분해를 수행하여 S에는 singular 값이 주 대각선에 위치하는 행렬로 USV′=A를 만족시키는 orthogonal 행렬 U와 V를 함께 구하여 되돌려 준다.
- polyval(polynomial, point) : 벡터형태로 주어진 다항식 ploynomial이 point값에서 갖는 값을 되돌려준다.
- conv(x,y) : 다항식 x와 다항식 y의 곱을 구해 계수를 행벡터 형태로 되돌려준다.
- [q,r]=deconv(a,b) : 다항식 a를 b다항식으로 나누어 몫은 q에 나머지는 r에 되돌려 준다.
- [r,p,k]=residue(num,den) : num/den의 부분분수 전개를 수행하여 계수, 극점, 몫을 r,p,k에 되돌려준다. [num,den] =residue(r,p,k)는 반대과정을 수행한다.

7.7. 관계 및 논리 연산자

- 〈 , 〈=, ==, 〉, 〉=, ~= : 비교연산자로 ==와 ~=는 실수부와 허수부 양쪽을 비교하고 다른 것은 실수부만 비교한다. ~=는 같지 않은지를 비교하는 것이다.
- C = A 비교연산자 B : 차원이 같은 A와 B 행렬의 각 엔트리 A(i,j) 와 B(i,j) 를 비교해서 참이면 1을 거짓이면 0을 C(i,j)에 할당한다.
- & , | , ~ : 논리 연산자로 각각 AND, OR, NOT 연산을 의미한다.
- C = A & B : A(i,j)와 B(i,j) 둘 중 하나가 0이면 C(i,j)는 0 아니면 1
- C = A | B : A(i,j)와 B(i,j) 둘 다 0이면 C(i,j)는 0 아니면 1
- C = ~A : A(i,j) 0이 아니면 C(i,j)는 1 아니면 0
- idx = find(x) : 주어진 벡터 x에서 0이 아닌 값의 인덱스를 idx에 돌려준다.

7.8. 순환문 및 조건문

- for 문의 일반적 형태 : for var= expression, statement1, ... , statementN, end; 가 일반적 형태로 var=expression 는 보통 i=1:n 등으로 주어져 인덱스 i가 1부터 n까지 증가하는 동안 그 아래 statement들을 반복 수행하라는 형태로 프로그래밍한다.
- while 문의 일반적 형태 : while condition, statement1, ... , statementN, end; 가 일반적 형태로 condition을 만족하는 동안 그 아래 statement들을 반복 수행시킬때 사용한다.
- if else 문의 일반적 형태 : if cond1, state1 , ... , elseif cond2, ... , else stateK, ... , end; 가 일반적인 형태로 각 조건들 (cond1, ...)에서 충족되는 경우 그 아래 주어진 statement들을 수행시킬 때 사용한다.
- break : end 문의 위치로 점프할 때 사용한다.

7.9. 함수 파일과 스크립트 파일

- script file : 매우 많은 여러 개의 명령어들을 filename.m의 파일에 미리 저장하고 filename이라고 단순히 matlab 메인창에서 치기만 하면 filename.m에 작성해 넣은 명령들을 수행시킬 수 있다.
- function file의 일반적인 형태 : function [OUT1, OUT2,

...., OUTn] = FILEname(IN1, IN2, ... , INm)가 첫 번째 줄의 형태이다. OUTx가 출력변수이고 INx가 입력변수로 패러메터의 전달이 가능하고 내부에서 지역변수를 사용할 수 있다. 두번째 줄에는 수행할 명령을 기록하면 된다. 반드시 FILEname.m으로 파일을 저장해야만 matlab의 다른 명령어처럼 사용할 수 있다.

- 주석 : function file(예를 들어 FILEname.m)의 첫번째 줄 다음에 %과 함께 주석문을 써넣으면 나중에 메인창에서 help 명령을 사용해서(예로 help Filename) 주석문을 볼 수 있다.
- input 명령어 : VARname = input('Enter VARname ='); 형태로 script file 혹은 function file 에서 사용할 패러메터의 값을 입력받을 때 사용한다. 문자열 'Enter VARname ='은 다른 것을 해도 무방하다.
- keyboard : script file 혹은 function file 에서 사용자에게 제어권을 넘겨줄 때 사용한다.

7.10. 그래프와 프린트

- clf : 현재 그림을 지울 때 사용한다.
- plot(xVec, yMtx) : xVec가 x축 변수로 하는데 xVec가 행벡터이면 행렬 yMtx의 각 행들을 y축 변수로 yMtx의 렬의 개수만큼의 그림을 동시에 그려준다. xVec가 열벡터일 때는 행렬 yMtx의 각 열들을 y축 변수로 하여 행의 개수만큼의 그림을 동시에 그려준다. x축 변수의 길이와 y축의 변수의 길이는 같아야 한다.
- plot(x1,y1,s1,x2,y2,s2,x3,y3, ... , xN, yN,sN) : x1, x2, ... , xN을 x축 변수로 y1, y2, ... , yN을 대응되는 y축 변수로 여러 개의 그림을 하나에 동시에 그려준다. 각 x축 변수의 길이는 달라도 되나 x축 변수의 길이와 대응하는 y축 변수의 길이는 같아야 한다. s1, ... sN은 각 그림을 구별하기 위한 색깔이나 선의 형태를 정의할 수있는 문자열로 '-'이면 실선으로 ':'이면 점선으로 '—'이면 대쉬로 표현되고 'y', 'r', 'g', 'b', 'w', 'k' 이면 각각 황, 적, 녹, 청, 백, 흑색이다, '.', 'o', 'x', '+', '*' 이면 해당 글자가 선에 더해져서 표현된다. 이들의 조합이 가능해 만약 '-yo'라면 황색 실선에 o기호가 선에 더해져서 그려진다.
- loglog : plot와 같으나 x축과 y축을 상용로그 스케일로 그린다.
- semilogx : plot와 같으나 x축을 상용로그 스케일로 그린다.
- semilogy : plot와 같으나 y축을 상용로그 스케일로 그린다.
- plot3 : plot와 비슷하나 3차원으로 그린다.
- subplot(mnp) : 여러 개의 그래프를 한 페이지에 나타내는데, 그래프를 m행으로 나누고 n열로 나눈 다음, 1행1열을 1번으로, 1행2열을 2번, 1행n열을 n번, 2행1열을 n+1번 식으로 번호를 붙일 때 p번째에 그래프를 그릴 준비를 시킨다.
- title('string') : 그림에 string이라는 문자열의 제목을 달아준다.
- xlabel('xstring'), ylabel('ystring'), zlabel('zstring') : 그림의 x,y,z 축에 xstring, ystring, zstring라는 라벨을 각각 달아준다.
- text(x, y, 'string') : 그림의 좌표 (x, y)에 string라는 라벨을 달아준다. 3차원의 경우는 text(x, y, z, 'string')
- grid : 그림에 grid 보조선을 표시해준다.
- gtext('string') : 2차원 그래프에서 마우스가 지정하는 곳에 string이라는 라벨을 달아준다.
- hold on : 그림의 축척이나 좌표축을 보존하여 다른 그림을 위에 덧그리는 것을 가능하게 한다.
- hold off : hold 기능을 off시킨다. (hold만을 사용해도 된다. 홀수번 입력되면 on, 짝수번 입력하면 off된다)
- axis([xmin xmax ymin ymax]) : 좌표축의 범위를 설정한다. 3차원의 경우는 axis([xmin xmax ymin ymax zmin zmax]) 형태로 하면 된다.
- mesh(x,y,z) : x와 y값에 따른 z값을 3차원으로 그려준다.
- mesh(z) : z가 $m \times n$행렬 형태라면 x=1:n; y=1:m; mesh(x,y,z)와 같다.
- [X,Y]=meshgrid(x,y) : z=f(x,y) 함수에 대한 3차원 그래프를 그리기 위한 변수값 행렬 X, Y를 구해준다.

  예 : $z = xe^{-x^2-y^2}$를 $-2 < x < 2, -2 < y < 2$ 범위에서 구하고 그래프를 그리려면 아래처럼 한다.

  ```
  [X,Y] = meshgrid(-2:.2:2, -2:.2:2);
  Z = X .* exp(-X.^2 - Y.^2); mesh(Z);
  ```

- print -ddevice filename : 그림을 filename으로 프린트한다. device는 여러가지가 있는데 만약 ps 형태로 하려면 -dps로 하고 eps 형태로 하려면 -deps 256색의 비트맵 포맷으로 하려면 -dbmp256 등을 사용하면 된다.

예제 2: 그림 10과 같은 그림을 그려서 fig10.ps에 저장하기 위한 matlab 프로그램은 다음과 같다.

```
w=logspace(-1, 2, 100); n=length(w);
y=ones(n,1); ww=(w.*w);
subplot(2,1,1);
semilogx(w, 20*log10(1./ww),'-', w,
20*log10(1./w),':',w, 20*log10(w),'-.', w,
20*log10(ww),'--');
axis([0.1 100 -40 40]);
text(10,20,'solid line    : {1/{s}^2}');
text(10,10,'dotted line  : {1/{s}}');
text(10,0,'dashdot line  : {s}');
```

```
text(10,-10,'dashed line  : {{s}^2}');
xlabel('Frequency [rad/sec]');
ylabel('Magnitude [dB]');
subplot(2,1,2);
semilogx(w, -180*y,'-', w,  -90*y,':',w,
90*y,'-.',w, 180*y,'--');
axis([0.1 100 -200 200]);
text(10,-160,'solid line  : {1/{s}^2}');
text(10,-70,'dotted line  : {1/{s}}');
text(10,70,'dashdot line  : {s}');
text(10,160,'dashed line  : {{s}^2}');
xlabel('Frequency [rad/sec]');
ylabel('Magnitude [degree]');
print -dps fig10.ps
```

7.11. 고전 제어 명령어

- y = step(n,d,t) : n/d로 주어지는 전달함수의 단위계단 응답을 y에 저장한다. t를 안 넣으면 자동으로 해주고 t를 넣는 경우에는 미리 정의되어야 한다.
- y = impluse(n,d,t) : 임펄스 응답을 구해 y에 되돌려 준다.
- y = lsim(n,d,u,t) : 입력 u에 대한 응답을 구해 y에 되돌려 준다.
- step , impulse, lsim : 명령어에 저장할 벡터를 설정해주지 않으면 그림을 그려준다.
- [n,d] = rmodel(n) : n차의 고유치가 0이하인 siso 시스템을 생성한다.
- [n,d] = rmodel(n,p) : n차의 출력이 p개인 고유치가 0이하인 SIMO 시스템을 생성한다.

예제 3: 그림 3를 얻어 fig3.ps 에 저장하기 위한 프로그램은 아래와 같다.

```
t=[0:0.1:20]'; n=length(t);y=zeros(n,1);
y(:,1)=step(1,[1 0.3 1],t);
y(:,2)=step(1,[1 0.5 1],t);
y(:,3)=step(1,[1 0.7 1],t);
y(:,4)=step(1,[1 0.9 1],t);
plot(t,y(:,1),'-',t,y(:,2),':',t,y(:,3),
'-.',t,y(:,4),'--');
axis([0 20 0 1.8]);
text(13,0.8,'solid line  : {\zeta =0.3}');
text(13,0.7,'dotted line  : {\zeta =0.5}');
text(13,0.6,'dashdot line  : {\zeta = 0.7}');
text(13,0.5,'dashed line  : {\zeta = 0.9}');
xlabel('Time(sec)');ylabel('Magnitude');
print -dps fig3.ps s
```

- rlocus(n,d,k) : k(벡터도 가능)값일 때의 근값을 그려준다. k를 할당하지 않으면 적당한 범위에 대하여 그린다.
- y=rlocus(n,d,k) : k(벡터도 가능)값일 때의 근값을 y에 할당한다.
- sgrid : 근궤적상에서 같은 ζ와 w_n을 갖는 보조선을 그려준다.
- sgrid(zeta, wn) : 근궤적상에 zeta의 감쇠율과 wn의 고유진동수를 갖는 보조선을 그려준다.
- rlocfind(n,d) : rlocus 명령으로 근궤적을 그린 후에 사용하는 명령으로 그려진 근궤적상에 마우스를 이용하여 지정한 극점을 갖도록 하는 이득값을 찾을 때 사용한다.
- [k, p] = rlocfind(n,d) : rlocus 명령으로 근궤적을 그린 후에 사용하는 명령으로 그려진 근궤적상에 마우스를 이용하여 지정한 점을 갖도록 하는 이득값을 k에 그 때의 극점을 p에 되돌려 준다.
- [k, p] = rlocfind(n,d,pd) : 오픈루프 이득이 n/d의 시스템의 근궤적에서 pd(벡터도 가능)에 가까이 위치하게 할 이득 k값들과 그 때의 극점들을 p에 되돌려 준다.
- bode(n,d, wmin, wmax) : 주파수값 wmin에서 wmax 범위에 대한 전달함수 n/d의 보데선도를 그려준다.
- bode(n,d,w) : 사용자가 정의해준 주파수값 벡터 w에 대한 전달함수 n/d의 보데선도를 그려준다. bode(n,d)의 형태이면 적당한 범위내에 자동적으로 생성해서 그려준다.
- [mag, ph, w] = bode(n,d) : 열벡터로 n/d의 크기와 위상과 주파수 값을 mag, ph, w에 되돌려준다.
- [mag, ph] = bode(n,d,w) : 사용자가 주파수값 w를 정의해주고 n/d의 크기와 위상값을 돌려 받는다.
- nyquist : 나이키스트 선도에 대한 명령어로 bode 명령어와 비슷하다.
- nichols : 니콜스 선도에 대한 명령어로 bode 명령어와 비슷하다.
- margin(n,d) : 전달함수 n/d의 보데선도를 그려주고 이득과 위상여유를 구해준다.
- margin(mag, ph, w) : [mag, ph, w] = bode(n,d)과 같은 명령으로 만들어준 mag, ph, w를 가지고 보데선도를 그려주고 이득과 위상여유를 인터폴레이션에 의해 구해준다.
- [gm , pm, wp, wg] = margin(n,d) : 주어진 n/d에 대하여 Gm, Pm, w_p, w_g를 구해 gm, pm, wp, wg에 되돌려 준다.
- [gm , pm, wp, wg] = margin(mag, ph, w) : [mag, ph, w] = bode(n,d)과 같은 명령으로 만들어준 mag, ph,

w를 가지고 인터폴레이션에 의해 Gm, Pm, w_p, w_g를 구해 gm, pm, wp, wg에 되돌려 준다.

- printsys(n,d) : 전달함수를 메인창에 표시해준다.
- [z,p,k] = tf2zp(n,d) : 전달함수 n/d의 영점, 극점, DC 이득을 z, p, k에 되돌려 준다.
- [n,d] = zp2tf(z,p,k) : tf2zp의 반대
- [p,z] = pzmap(n,d) : 전달함수 n/d의 영점과 극점값을 z, p에 되돌려 준다.
- pzmap(n,d) : 전달함수 n/d의 영점과 극점을 o와 x를 사용해서 복소평면상에 그려준다.
- [n1,d1] = series(n2,d2,n3,d3) : 전달함수가 각각 G2=n2/d2와 G3=n3/d3인 두개의 시스템이 직렬 연결된 경우 등가 전달함수 G1=G2G3값을 계산해서 분자를 n1에 분모를 d1으로 되돌려 준다.
- [n1,d1] = parallel(n2,d2,n3,d3) : series와 비슷하나 병렬 연결된 경우의 등가 전달함수를 되돌려 준다.
- [n1,d1] = feedback(n2,d2,n3,d3, sign) : sign은 옵션 -1이나 1이 될 수 있다. 안 써주면 -1과 같다. -1인 경우 G2/(1+G2G3)를 계산하여 되돌려주고 1인 경우 G2/(1-G2G3)를 계산하여 되돌려 준다.

7.12. 현대 제어 명령어

- [a,b,c,d]=tf2ss(n,d) : 전달함수로부터 상태공간 행렬을 얻는다. a행렬의 첫번째 서브다이아고날이 1이 되는 observable 형태로 얻어진다.
- [a,b,c,d]=zp2ss(z,p,k) : 영점, 극점을 사용한 표현으로부터 상태공간 행렬을 얻는다. 전달함수의 분자가 상수형태이면 z는 Inf로 해야한다.
- [n,d]=ss2tf(a,b,c,d,ui) : 입력이 2개 이상인 경우 ui는 원하는 입력채널 번호 ui를 쓰지 않으면 1일 때와 같다. tf2ss 명령의 반대로 상태공간식으로부터 ui번째 입력에 대한 전달함수를 얻는다.
- [z,p,k]=ss2zp(a,b,c,d,ui) : zp2ss의 반대
- sys = ss(a,b,c,d) : 시스템 행렬값으로 a, b, c, d를 갖는 모델을 sys라는 이름에 할당한다.
- sys = tf(n,d) : n/d를 전달함수로 갖는 모델을 sys에 할당한다.
- sys = zpk(z,p,k) : ss와 비슷하나 영점, 극점, 시스템 이득 값으로 z, p, k를 갖는 모델을 sys에 할당한다.
- s=tf('s') : s를 라플라스 변수로 할당하여 직접 모델을 표현하기 쉽게 한다.

  예 : 전달함수 $1/(s^2+s+1)$를 sys1에 할당하려면 다음의 명령어를 사용하면 된다.

  s=tf('s'); sys1=1/(s^2+s+1);

  % sys1=tf([1],[1 1 1])과 동치
- [am,bm,cm,dm,T]=canon(a,b,c,d,'modal') : a행렬의 고유치가 다이아고날에 나타나는 modal 형태로 바꿔준다. a,b,c,d 에 대한 원래 상태가 x하고 am,bm,cm,dm에 대한 상태를 z라 하면 z=Tx의 관계가 있다.
- [am,bm,cm,dm,T]=canon(a,b,c,d,'companion') : 시스템행렬이 특성근이 다이아고날에 나타나는 modal 형태로 바뀐다. a,b,c,d 에 대한 원래 상태가 x, am,bm,cm,dm에 대한 상태를 z라 하면 z=Tx의 관계가 있다.
- [am,bm,cm,dm,T]=canon(a,b,c,d,'companion') : 시스템행렬은 첫번째 서브다이아고날이 1이 되는 observable 형태로 바뀐다. 원래 상태가 x 바뀐 상태를 z라 하면 z=Tx의 관계가 있다.
- [v,d]=eig(a) : a를 대각화한 행렬과 이를 위한 modal 행렬을 d와 v로 되돌려준다.
- eig(a) : a의 고유치를 되돌려 준다.
- damp(a) : a의 고유치, 감쇠율, 고유주파수를 되돌려 준다.
- [wn, z] = damp(a) : a의 고유주파수, 감쇠율을 wn, z로 되돌려 준다.
- bode, nyquist, nichols : cmdname(a,b,c,d,ui,w) 형태로 상태공간 모델에 대하여 또는 ss, tf, zpk 를 사용해 정의된 모델 sys에 대하여 cmdname(sys,w) 혹은cmdname (sys) 형태로 사용할 수 있다.
- step, impulse : cmdname(a,b,c,d,ui,t) 형태로 상태공간 모델에 대하여 또는 ss, tf, zpk 를 사용해 정의된 모델 sys에 대하여 cmdname(sys,t) 혹은 cmdname(sys) 형태로 사용할 수 있다.
- lsim(a,b,c,d,u,t,x0) : 사용자가 제공한 시간벡터 t,와 입력 u, 초기조건 x0에 대한 시스템의 응답을 그려준다. x0를 생략하면 0으로 한다.
- [y,x]=lsim(a,b,c,d,u,t,x0) : 사용자가 제공한 시간벡터 t와 입력 u, 초기조건 x0에 대한 시스템의 응답의 출력을 y에 상태를 x에 되돌려 준다. x0를 생략하면 0으로 한다.
- initial(a,b,c,d,x0,t) : 사용자가 제공한 시간벡터 t와 초기조건 x0에 대한 시스템의 영입력 응답을 그려준다. t를 생략하면 알아서 그려준다.
- [y,x] = initial(a,b,c,d,x0,t) : 사용자가 제공한 시간벡터 t와 초기조건 x0에 대한 시스템의 영입력 응답의 출력을 y에 상태를 x에 되돌려 준다. t를 생략하면 알아서 생성한다.
- rlocus, rlocfind : argument를 상태공간 행렬 (a,b,c,d)로 대체하여 쓸 수 있으나 siso시스템이어야 한다.
- [a,b,c,d]=rmodel(n) : n차의 출력이 고유치가 0이하인 siso 시스템을 생성한다.
- [a,b,c,d]=rmodel(n,p,m) : n차의 출력이 p개이고 입력

이 m개인 고유치가 0이하인 mimo 시스템을 생성한다.

- [as,bs,cs,ds]=series(a1,b1,c1,d1,a2,b2,c2,d2,outvector,inputvector) : S1=(a1,b1,c1,d1)과 S2=(a2,b2,c2,d2) 시스템의 직렬연결. outvector와 inputvector은 S1의 몇번째 출력이 S2의 몇번째 입력으로 연결되는가를 표현한다. 생략하면 S1의 모든 출력은 S2의 입력으로 연결된다.
- [as,bs,cs,ds] = series(a1,b1,c1,d1,a2,b2,c2,d2,in1,in2,out1,out2) : S1과 S2의 병렬연결. S2의 in2로 표시된 입력과 S1의 in1의 입력을 동일하게 사용하고 S2의 out2와 S1의 out1출력은 합쳐짐을 의미한다. 생략하면 모든 입력이 공통이고 모든 출력이 합쳐짐을 의미한다.
- [af,bf,cf,df] = feedback(a1,b1,c1,d1,a2,b2,c2,d2,sign) : 입력과 S2의 출력에 의한 오차신호를 S1의 입력으로 하고 (이때 S1의 출력을 S2의 입력으로 연결) sign이 -1일 때 오차신호는 입력과 S2의 출력의 차이이고 +1일 때는 합이다. sign을 생략한 경우에는 sign=-1이다.
- [af,bf,cf,df] = feedback(a1,b1,c1,d1,a2,b2,c2,d2,in1,out1) : S1의 출력중 out1으로 표시된 값만 S2의 입력으로 연결되고 S1의 입력채널에서 in1으로 표시된 채널로만 S2의 모든 출력이 궤환된다. 부궤환인 경우 in1에 -를 곱한다.

7.13. simulink

- simulink의 시작 : 동적인 시스템의 시뮬레이션을 위한 matlab의 확장도구로 메인창에서 simulink를 치면 그림 16과 같이 Simulink Library Browser라는 이름의 창이 뜬다.

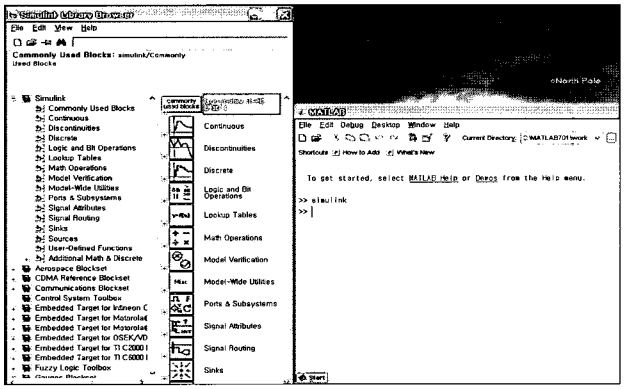

그림 16. Simulink Library Browser.

- Simulink 라이브러리 : Simulink Library Browser창의 중앙에는 시뮬레이션 모델을 위한 다양한 블록모델의 저장소가 위치한다. 주요한 것들은 아래와 같다.
  - Commonly Used Blocks : 적분기, 이득, 합산, 곱하기 등 자주 쓰이는 블록들이 위치하고
  - Continuous : 미분기, 적분기, 전달함수모델, 상태공간모델, 시간지연을 위한 블록들이 위치하고
  - Discontinuities : 백래쉬, 마찰력, 데드존, 양자화기, 릴레이, saturation 등을 위한 블록들이 위치하고
  - Sinks : 스코프, 그래프, 데이터 저장을 위한 블록들이 위치하고
  - Sources : 시계, 펄스 발생기, 함수발생기, 랜덤신호발생기 등의 블록이 위치한다.
- simulink를 통한 시뮬레이션 방법 : 새로운 시뮬레이션은 다음의 단계를 거쳐하면 쉽게 할 수 있다.
  ① 시작 : matlab의 메인창에서 simulink를 입력하여 Simulink Library Browser창을 띄운다.
  ② 새로운 file 만들기 : Simulink Library Browser창의 상단에 위치한 메뉴 [File->New->Model]을 선택한다. 그러면 untitled라는 이름의 창이 뜬다.
  ③ 시뮬레이션 모델 만들기 : Simulink Library Browser 창의 중앙에 Commonly Used Blocks, Continuous, Discontinuities, Discrete, Sinks, Sources 등의 블록모델이 정리된 곳으로부터 필요한 블록들을 untitled 창으로 끌어와 모델을 만든다. 블록을 더블클릭하면 입력해야할 패러메터값들(예로 적분기의 경우 초기값, 이득블록의 경우 이득값, 데이터 저장블록의 경우 파일명과 변수명)을 위한 창이 뜨는데 여기에 원하는 값을 입력하면 된다. 블록 연결하여 모델 완성하기 : 블록의 연결은 블록모델의 출력포트(기호 >로 표시되어 있다)에서 마우스를 갔다 대면 +기호가 나타나는데 버튼을 클릭하여 연결하고자 하는 블록의 입력포트(기호 >로 표시되어 있다.)까지 끌고가서 버튼을 놓으면 연결된다.
  ④ 시뮬레이션하기와 저장하기 : 완성된 모델이 존재하는 창의 [Simulation->Configuration Parameters]메뉴에서 시뮬레이션 시작시간, 최종시간, solver의 옵션 등을 설정하고 [Simulation->Start]메뉴를 선택하여 시뮬레이션을 한다. 시뮬레이션이 끝나면 '삑'소리가 난다. 모델의 저장을 위해서는 [Save As...]메뉴를 선택하여 Save As라는 이름의 새 창을 띄워 파일이름을 입력하면 된다.

8. 결론

선형 제어 시스템 이론에 대한 연재물의 첫 번째로 본고에서는 궤환 시스템의 성능과 안정도 판별에 관련된 고전 선형 제어이론과 이와 관련된 Matlab 명령어에 대하여 다루었다. 다음에는 진상, 지상 및 PID 제어기 설계에 대하여 다루겠다.

참고문헌

[1] B. Shahina and M. Hassul, *Control System Design Using Matlab*, Prentice Hall, 1993.

[2] S. Boyd, L. El Ghaoui, E. Feron and V. Balakrishnan, Linear Matrix Inequalities in system and Control Theory, SIAM, 1994, http://www.stanford.edu/~boyd/lmibook/

[3] W.C. Levine, *The Control Handbook*, CRC Press, 2000.

[4] C. Scherer and S. Weiland, Linear Matrix Inequalities in Control, http://www.cs.ele.tue.nl /SWeiland

[5] A. Damen and S. Weiland, Robust Control, http://www.cs.ele.tue.nl/SWeiland

[6] 김종식, 선형 제어 시스템 공학, 청문각, 1991년.

[7] G. Balas et al., *Robust Control Toolbox*, Mathworks, 2005.

[8] *Control System Toolbox for Use with Matlab : User's Guide*, Mathworks, 2005.

[9] 최한호, 로봇공학 강의 노트, http://home.dongguk.edu/user/hhchoi

저자약력

《최 한 호》

- 1966년 8월 25일생.
- 1988년 2월 서울대학교 제어계측공학과 (공학사).
- 1990년 2월 한국과학기술원 전기 및 전자공학과 (공학석사).
- 1994년 8월 한국과학기술원 전기 및 전자공학과 (공학박사).
- 1994년 9월~1998년 2월 대우전자 전략기술 연구소 연구원.
- 1998년 3월~2003년 2월 안동대학교 전자공학교육과 교수.
- 2003년 3월~현재 동국대학교 전기공학과 교수.
- 관심분야 : 강인제어이론, 마이콤 기반 제어, 가상현실 및 로보틱스.
- E-mail : hhchoi@dongguk.edu