Analysis of Linear Control Systems

Topics:

- Properties of linear dynamic systems
- Time-domain analysis of linear systems
- Numerical simulation of linear system
- Root-locus of linear systems
- Frequency-domain analysis of linear systems
- Introduction to model Reduction techniques

Properties of linear dynamic systems

- Linear systems obey the superposition principle
- Stability analysis
  - Bounded-input bounded-output (BIBO) stability
    - A system is stable if all its poles have negative real-parts
      - Poles are the roots of the denominator of the system's transfer function G(s)
      - Zeros (transmission zeros) are the roots of the numerator of the system's transfer function G(s)
    - Poles on the imaginary axis with multiplicity one are critically stable
    - Poles on the imaginary axis with multiplicity more than one are unstable
- Matlab commands
    - Find system poles using commands pole() and eig()
    - Find system zeros using command zero()
    - Sketch system's poles and zeros using command pzmap()
    - Use command isstable() to check system's stability, returns 1=stable, 0=unstable

Example: Check the stability of system $G(s) = \frac{s^3+7s^2+24s+24}{s^4+10s^3+35s^3+50s+24}$

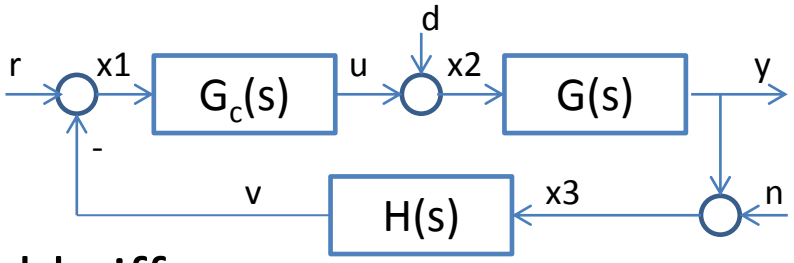
```
>> G=tf([1,7,24,24],[1,10,35,50,24]); eig(G); pzmap(G);
```

Or

```
>> s=tf('s'); G=(s^3+7*s^2+24*s+24)/(s^4+10*s^3+35*s^2+50*s+24);  isstable(G);
```

Properties of linear dynamic systems …

Internal stability



- The system shown is internally stable iff:
  1) The transfer function of 1+H(s)G(s)Gc(s) has no zeros in RHP
  2) The loop transfer function H(s)G(s)Gc(s) has no pole-zero cancellation in RHP

Matlab command: intstable()

- Syntax: [V,c]=intstable(G,Gc,H);
  - If system is internally stable , V=0 and c is empty
  - If system is I/O unstable, V=1 and c holds the unstable closed-loop poles
  - If system is I/O stable, but not internally stable, V=2 and c holds the cancelled unstable poles
- Matlab command minreal() gives the simplified transfer function, after pole-zero cancellation, which may not be internally stable

Properties of linear dynamic systems …

Controllability and observability analysis

- Controllability:
- The state $x_i(t)$ is said to be controllable if there exists an input that in finite time can drive it to any specified value $x_i(t_f)$ from the initial value $x_i(0)$
- The system is fully controllable if all its states are controllable
  - Full controllability of the system depends on the A and B matrices of its state-space model
    - An nth-order system is fully controllable if its controllability matrix T_c=[B, AB, …, A^{n-1}B] has full rank

Matlab commands: ctrb(), ctrbf(), rank()

- Tc=ctrb(A,B); returns the controllability matrix Tc
- [Ac,Bc,Cc,Tc]=ctrbf(A,B,C); returns the equivalent state-space matrices (Ac,Bc,Cc) in the stair-case form $A_c = \begin{bmatrix} \hat{A}_{\bar{c}} & 0 \\ \hat{A}_{21} & \hat{A}_c \end{bmatrix}$, $B_c = \begin{bmatrix} 0 \\ \hat{B}_c \end{bmatrix}$, $C_c = [\hat{C}_{\bar{c}} \quad \hat{C}_c]$, where ($\hat{A}_c$, $\hat{B}_c$, $\hat{C}_c$) represent the controllable subsystem $G(s) = \hat{C}_c(sI - \hat{A}_c)^{-1}\hat{B}_c + D$
- rank(Tc); generates the rank of the matrix Tc

Properties of linear dynamic systems …

- Observability
- The state $x_i(t)$ is said to be observable if for any $t_f>0$, the initial state $x_i(0)$ can be determined from the time history of the input u(t) and output y(t) in the interval $[0,t_f]$
- The system is fully observable if all system states are observable
  - Full observability of the system depends on the A and C matrices of its state-space model
    - An nth-order system is fully observable if its observability matrix $T_o = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix}$ has full rank
    - Dual of controllability

Matlab commands: obsv(), obsvf()

- To=obsv(A,C); returns the observability matrix To
- [Ao,Bo,Co,To]=obsvf(A,B,C); returns the equivalent state-space matrices (Ao,Bo,Co) in the stair-case form $A_o = \begin{bmatrix} \hat{A}_{\bar{o}} & \hat{A}_{12} \\ 0 & \hat{A}_o \end{bmatrix}$, $B_o = \begin{bmatrix} \hat{B}_{\bar{o}} \\ \hat{B}_o \end{bmatrix}$, $C_o = [0 \quad \hat{C}_o]$, where $(\hat{A}_o, \hat{B}_o, \hat{C}_o)$ represent the observable subsystem $G(s) = \hat{C}_o\left(sI - \hat{A}_o\right)^{-1}\hat{B}_o + D$

Properties of linear dynamic systems …

Controllability and observability Gramians

- Controllability and observability Gramians W_c and W_o show how controllable and observable a system is, where $W_c = \int_0^\infty e^{At} B B^T e^{A^T t} dt$ and $W_o = \int_0^\infty e^{A^T t} C^T C e^{At} dt$
  - W_c and W_o satisfy the Lyapunov equations $AW_c + W_c A^T = -BB^T$ and $A^T W_o + W_o A = -C^T C$
  - W_c and W_o are positive definite if and only if (A,B) is controllable and (A,C) is observable
  - The singular values of Wc indicate the contribution of the input signal to each state
  - The singular values of Wo indicate the contribution of each state to the output signal

Matlab commands: lyap(), svd(), gram()

- Wc=lyap(A,B*B’); returns the controllability Gramian matrix Wc
- Wo=lyap(A’,C’*C); returns the observability Gramian matrix Wo
- [U,S,V]=svd(Wc); produces a diagonal matrix S, of the same dimension as Wc and with nonnegative diagonal elements in decreasing order, and unitary matrices U and V so that Wc = U*S*V‘
- W=gram(G,type); returns the gramian matrix W for a system with state-space model G, where type is ‘c’ or ‘o’ for controllability or observability Gramians

Properties of linear dynamic systems …

Other Matlab commands:

- kalmdec(); produces Kalman decomposition of a given system
- timmomt(); produces time moments M_i of a given system, where $M_i = \int_0^\infty t^i g(t)dt$, and $g(t)$ is the impulse response of the system G(s)
- markovp(); produces the Markov parameters d_i of a given system, where $d_0 = CB + D$ and $d_i = CA^iB$, i=1,2,…

Norm measures of signals and systems

- Norm measures of signals
  - L_p-norm defines the size of a signal u(t) as $\|u(t)\|_p = \left(\int_{-\infty}^{\infty}|u(t)|^p dt\right)^{1/p}$, where p is a positive integer
    - The L1-norm is $\|u(t)\|_1 = \int_{-\infty}^{\infty}|u(t)|dt$
    - The L2-norm is $\|u(t)\|_2 = \left(\int_{-\infty}^{\infty}|u(t)|^2 dt\right)^{1/2}$, (the measure of signal power)
    - The L∞-norm is $\|u(t)\|_\infty = \sup_t|u(t)|$, (the least upper-bound of |u(t)|)

Properties of linear dynamic systems …

Norm measures of systems

- The size of a system G(s) is generally measured in H_2-norm and H_∞-norm
  - The H_2-norm is $\|G(s)\|_2 = \left(\frac{1}{2\pi j}\int_{-j\infty}^{j\infty}|G(j\omega)|^2 d\omega\right)^{1/2} = \left(\int_0^\infty tr\left(g(t)^T g(t)\right)dt\right)^{1/2}$
    - (square-root of the integral-squared of the impulse-response $g(t)$ of the system)
  - The H_∞-norm is $\|G(s)\|_\infty = \sup_{u(t)\neq 0}\frac{\|y(t)\|_2}{\|u(t)\|_2} = \sup_\omega |G(j\omega)|$
    - (peak-value of the magnitude of the frequency-response of the system)
- Properties of L and H norms:
  - $\|y(t)\|_2 \leq \|G(s)\|_\infty \|u(t)\|_2$
  - $\|y(t)\|_\infty \leq \|G(s)\|_2 \|u(t)\|_\infty$
  - $\|G_1(s)G_2(s)\|_\infty \leq \|G_1(s)\|_\infty \|G_2(s)\|_\infty$

Matlab command: norm()

- norm(X,P); X is a matrix, P is norm type 1, 2, inf, fro (Frobenius)
- norm(V,P); V is a vector, P is norm type (1, 2, …, inf (max), and –inf (min))
- norm(G); H_2-norm of G(s)
- norm(G,inf); H_∞-norm of G(s)

Time-Domain Analysis of Linear Systems

Analytical solutions to continuous-time responses

- Consider the system $\begin{cases}\dot{x}(t) = Ax(t) + Bu(t)\\ y(t) = Cx(t) + Du(t)\end{cases}$, with initial condition $x(0) = x_0$
  - The system response to an arbitrary input u(t), for $t \geq 0$, is:
    - $\begin{cases}x(t) = e^{At}x_0 + \int_0^t e^{A(t-\tau)}Bu(\tau)d\tau\\ y(t) = C\left(e^{At}x_0 + \int_0^t e^{A(t-\tau)}Bu(\tau)d\tau\right) + Du(t)\end{cases}$
  - Laplace transform of the solution is: $\begin{cases}X(s) = (sI - A)^{-1}(x(0) + BU(s))\\ Y(s) = C(sI - A)^{-1}(x(0) + BU(s)) + DU(s)\end{cases}$

Analytical solutions to discrete-time responses

- Consider the system $\begin{cases}x(k+1) = Ax(k) + Bu(k)\\ y(k) = Cx(k) + Du(k)\end{cases}$, with sample-time T and $x(0) = x_0$
  - The system response to a sampled arbitrary input u(k), for $k = 0,1,2, \ldots$, is:
    - $\begin{cases}x(k+1) = e^{AT}x(k) + \int_0^T e^{A\tau}Bd\tau\ u(k)\\ y(k) = C\left(e^{AT}x(k) + \int_0^T e^{A\tau}Bd\tau\ u(k)\right) + Du(k)\end{cases}$, where $y(k) = \mathfrak{Z}^{-1}(G(z)U(z))$

Numerical Simulation of Linear Systems

Second-order system analysis

- Consider a second-order linear system $G(s) = \frac{{\omega_n}^2}{s^2+2\zeta\omega_n s+{\omega_n}^2}$
  - The step response of the system is:
    1. If $\zeta = 0$, $y(t) = 1 - \cos(\omega_n t)$
    2. If $0 < \zeta < 1$, $y(t) = 1 - e^{-\zeta\omega_n t}\frac{1}{\sqrt{1-\zeta^2}}\sin(\omega_d t + \theta)$

       where $\omega_d = \omega_n\sqrt{1-\zeta^2}$ and $\theta = tan^{-1}\frac{\sqrt{1-\zeta^2}}{\zeta}$
    3. If $\zeta = 1$, $y(t) = 1 - (1 + \omega_n \mathrm{t})e^{-\omega_n t}$
    4. If $\zeta > 1$, $y(t) = 1 - \frac{\omega_n}{2\sqrt{\zeta^2-1}}\left(\frac{e^{\lambda_1 t}}{\lambda_1} - \frac{e^{\lambda_2 t}}{\lambda_2}\right)$

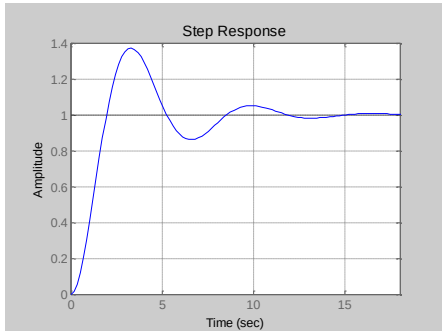
       where $\lambda_1 = -\zeta - \sqrt{\zeta^2-1}$ and $\lambda_2 = -\zeta + \sqrt{\zeta^2-1}$

  More concisely:

  - if $\zeta \neq 1$, $y(t) = 1 - \omega_n\, e^{-\zeta\omega_n t}\left(\frac{\cosh(\omega_d t)}{\omega_n} - \frac{\zeta\sinh(\omega_d t)}{\omega_d}\right)$

Numerical Simulation of Linear Systems

Qualitative specifications in step responses



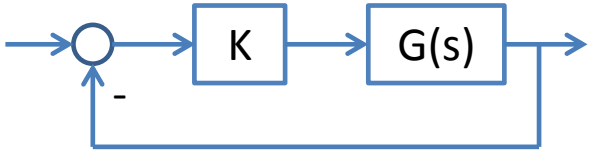
- Second-order linear system $G(s) = \frac{{\omega_n}^2}{s^2+2\zeta\omega_n s+{\omega_n}^2}$
  - Specifications of the step response (shown):
  - Steady-state value, y_{ss}: $y_{ss} = \lim_{s\to 0} sG(s)\frac{1}{s} = G(0)$
  - Rise-time, t_r: the time for the response to go from 10% to 90% of y_{ss}
  - Settling-time, t_s: the time when $y(t)$ enters and stays in the range $y_{ss} \pm \Delta y$
  - Overshoot (maximum peak), M_p: percent overshoot $M_p = \frac{y_{max}-y_{ss}}{y_{ss}} \times 100\%$

Matlab commands: step(), dcgain(), impulse(), lsim(), grid

- step(G); draws the step response of G(s)
- [y,t]=step(G); evaluate the step response of G(s) and not draw it
- K=dcgain(G); calculate the steady state value y_{ss} of the output y(t)
- impulse(G); draw the impulse response of G(s)
- lsim(G,u,t); draw the response of G(s) to an arbitrary input u(t)
- grid; superimpose a grid on the plot

Root Locus of Linear Systems

Unity feedback system



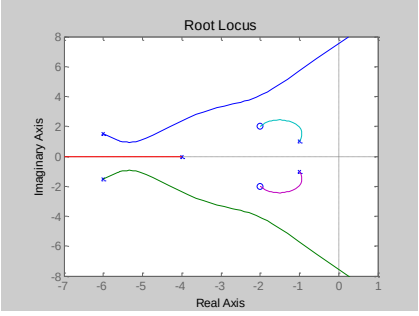
- Sketch the location of closed-loop poles
  - Roots location of $1 + KG(s) = 0$, where $-\infty < K < \infty$

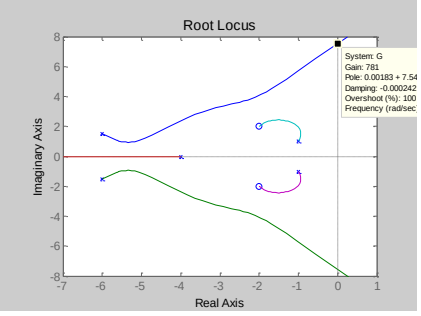
Matlab commands: rlocus(), grid

- rlocus(G); draws the root locus
- rlocus(G,K); draw root-locus for a given gain vector K
- [R,K]=rlocus(G); evaluate the closed-loop pole location R, not draw
- rlocus(G1,'-',G2,'-.b'G3,':r'); draw root-locus for several models

- Example: Draw the root-locus, where $G(s) = \frac{s^2+4s+8}{s^5+18s^4+120.3s^3+357.7s^2+478.5s+306}$

  >> num=[1,4,8]; den=[1,18,120.3,375.5,478.5,306];
  G=tf(num,den); rlocus(G);
  - Right click on the jw-axis crossing
    - Critical gain k≅781





Frequency-Domain Analysis of Linear Systems

Frequency-domain plots of G(s), s=jω

- Real and imaginary part representation

$$G(j\omega) = P(\omega) + jQ(\omega)$$

  - Nyquist plot draws $\mathrm{Q}(\omega) = Im(G(j\omega))$ v. $\mathrm{P}(\omega) = Re(G(j\omega))$

- Magnitude and Phase representation in separate plots

$$G(j\omega) = A(\omega)e^{-j\phi(\omega)}$$

  - Bode plot draws the magnitude in decibels ($\mathrm{M}(\omega) = 20log(A(\omega))$) and phase in degrees versus the frequency in log scale

- Magnitude and Phase representation in a single plot

$$G(j\omega) = A(\omega)e^{-j\phi(\omega)}$$

  - Nichols chart draws the magnitude versus phase

Matlab commands: nyquist, bode(), nichols(),

- nyquist(G); draws the Nyquist plot
- [R,I,K]=nyquist(G); evaluate the Nyquist data, not draw
- bode(G); draw the Bode plot of G(s)
- [A,ϕ,ω]=bode(G); evaluate the Bode data, not draw
- nichols(G); draw the Nichols chart of G(s)
- [A,ϕ,ω]=bode(G); evaluate the Nichols data, not draw

Frequency-Domain Analysis of Linear Systems

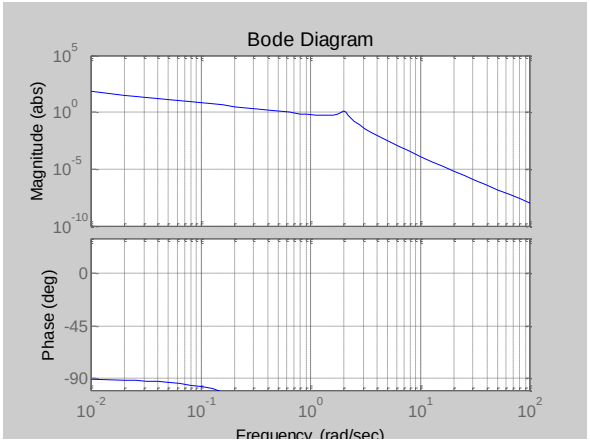
Example: Draw the Bode plot, Nyquist plot and Nichols chart for

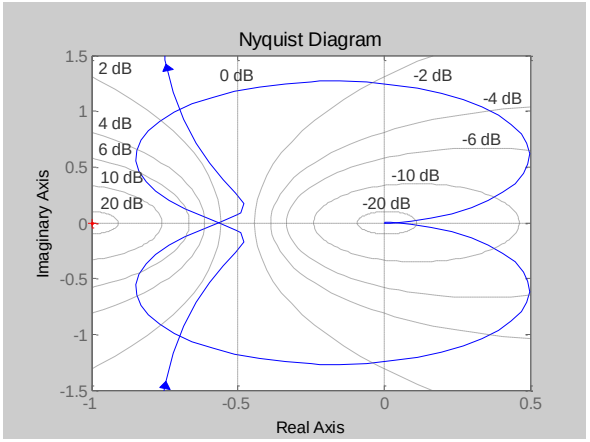
$$G(s) = \frac{s+8}{s(s^2+0.2s+4)(s+1)(s+3)}$$

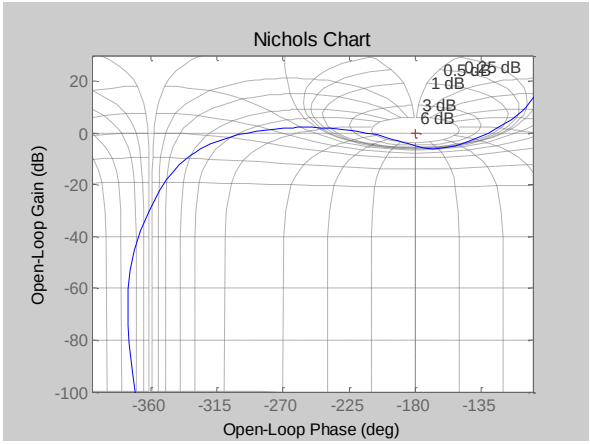
Matlab code:

```
close all; s=tf('s'); G=(s+8)/(s*(s^2+0.2*s+4)*(s+1)*(s+3));
figure(1), bode(G);
figure(2), nyquist(G); set(gca,'Ylim',[-1.5,1.5]);
figure(3), nichols(G); set(gca,'Xlim',[-400,-100],'Ylim',[-100,30]);
```

You may right-click on any of the figures and modify its properties (labels, limits, units, style, etc)

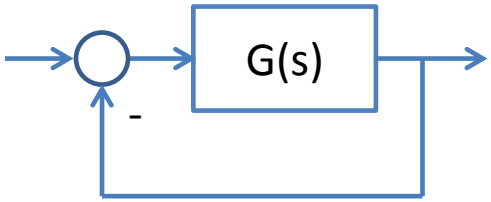






Frequency-Domain Analysis of Linear Systems

Stability analysis in frequency-domain

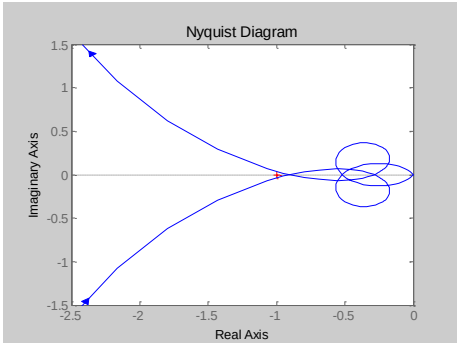


- The closed-loop system is stable if:
  - The Nyquist plot of G(s) encircles (counter-clockwise) the point (-1+0j) as many times as the number of RHP poles of G(s)

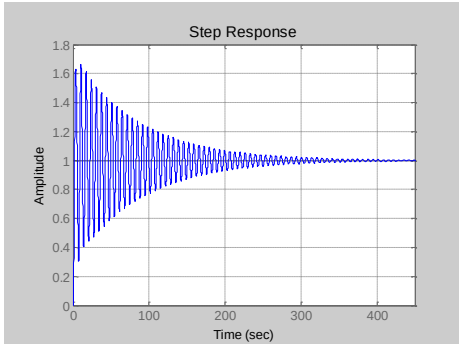
- Example: For $G(s) = \frac{2.7778(s^2+0.192s+1.92)}{s(s+1)^2(s^2+0.384s+2.56)}$ is the closed-loop system stable?

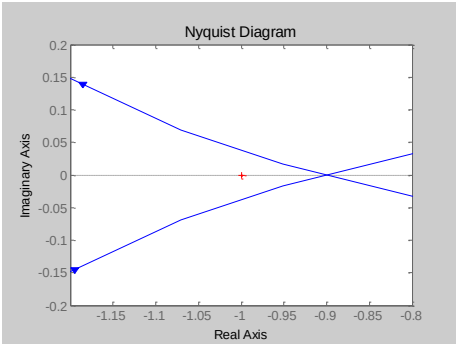
Matlab code:

```
s=tf('s'); G=2.7778*(s^2+0.192*s+1.92)/(s*(s+1)^2*(s^2+0.384*s+2.56));
figure(1), nyquist(G); axis([-2.5,0,-1.5,1.5]); grid;
figure(2), nyquist(G); axis([-1.2,-0.8,-0.2,0.2]); grid;
figure(3), step(feedback(G,1,-1));
```



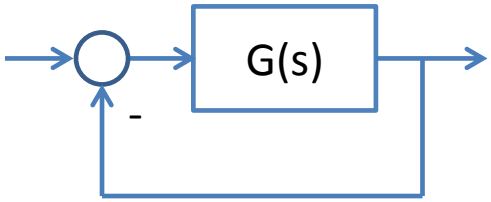
Stable, but too oscillatory!





Frequency-Domain Analysis of Linear Systems

Gain and phase margins of a system



- Gain margin: $G_m = \frac{1}{A(\omega_{cg})}$
  - $A(\omega_{cg})$ and ω_{cg} are the magnitude and frequency where the Nyquist plot intersects the negative real-axis
  - The smaller the gain-margin, the faster the closed-loop system response
  - If $G_m < 1$, the closed-loop system is unstable
- Phase margin: $P_m = \phi(\omega_{cp}) + 180$
  - $\phi(\omega_{cp})$ and ω_{cp} are the phase and frequency where the Nyquist plot intersects the unit circle
  - The larger the phase-margin, the less overshoot in closed-loop system response
  - If $P_m < 0$, the closed-loop system is unstable

- Example: For $G(s) = \frac{2.7778(s^2+0.192s+1.92)}{s(s+1)^2(s^2+0.384s+2.56)}$ find the closed-loop system margins

Matlab code:

```
s=tf('s'); G=2.7778*(s^2+0.192*s+1.92)/(s*(s+1)^2*(s^2+0.384*s+2.56));
[gm,pm,wg,wp]=margin(G);
```

$\Rightarrow$ gm = 1.1050; pm = 2.0985; wg = 0.9621; wp = 0.9261 (closed-loop is stable)

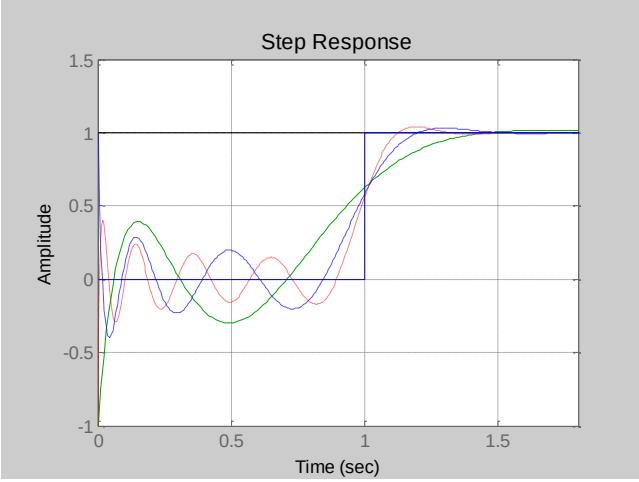
Introduction to Model Reduction Techniques

- Pade approximation to delay terms
  - A delay term $e^{-\tau s}$ can be approximated as rational transfer functions $\frac{n_p(s)}{d_p(s)}$

  Matlab Command: pade()
  - Syntax: [np,dp]=pade(τ,n); where n is the order of the Pade approximation
- Example: Find the Pade approximation of a pure delay $G(s) = e^{-s}$

  Matlab code:

```
tau=1;
[n1,d1]=pade(tau,3); G1=tf(n1,d1); % 3rd-order Pade approx
[n2,d2]=pade(tau,6); G2=tf(n2,d2); % 5th-order Pade approx
[n3,d3]=pade(tau,9); G3=tf(n3,d3); % 7th-order Pade approx
step(G1,'-g',G2,'-.b',G3,':r'); % Compare plots
line([0,1,1+eps,3],[0,0,1,1]);  % The step plot
```



Introduction to Model Reduction Techniques

State space model reduction

- Balanced realization method:
  1) Given a system model, find its unique balanced realization, where the controllability and observability Gramians are equal ($W = W_c = W_o$)
  2) Partition the balanced realization and eliminate the states with small Gramians

  Matlab Command: balreal(), modred()

- Example: For $G(s) = \frac{s^3+7s^2+24s+24}{s^4+10s^3+35s^2+50s+24}$ find a 2nd-order approximate model

  Matlab code:

  num=[1,7,24,24]; den=[1,10,35,50,24];

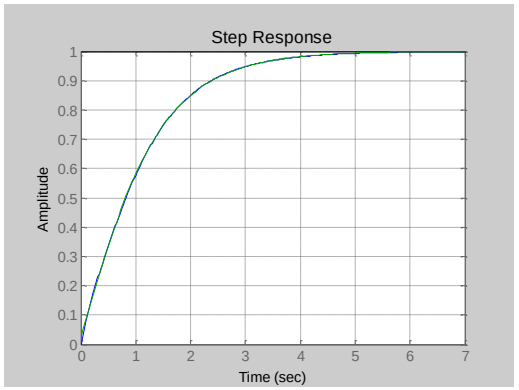
  G=tf(num,den); [Gb,g,T]=balreal(ss(G));

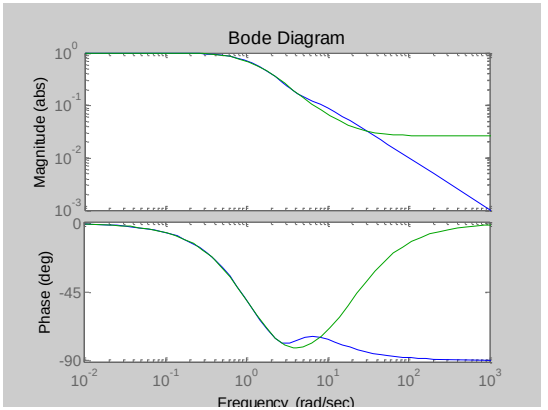
  Gr=modred(Gb,[3,4]); zpk(Gr),

  figure(1), bode(G,Gr), grid;

  figure(2), step(G,Gr);

$$G_r(s) = \frac{0.025974(s+22.36)(s+4.307)}{(s+1.078)(s+2.319)}$$



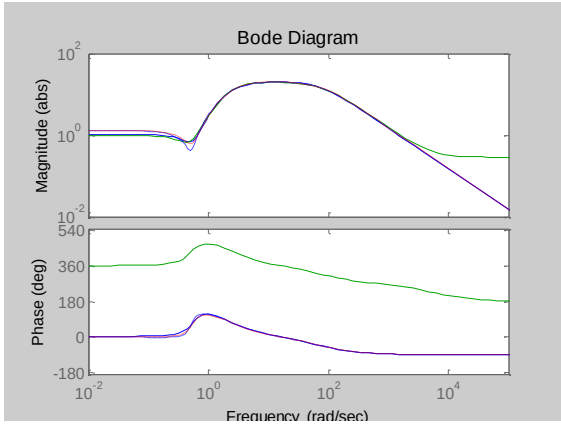


Introduction to Model Reduction Techniques

State space model reduction

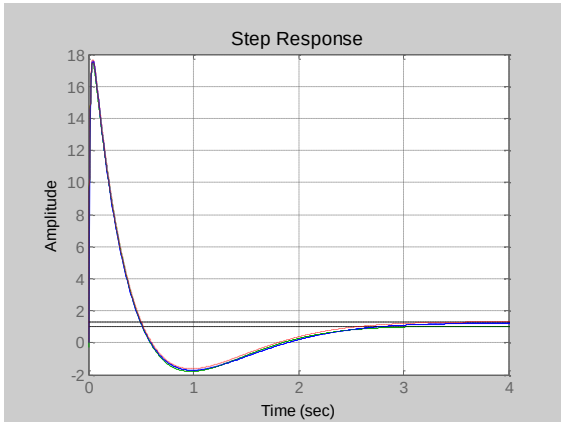
- Schur's balanced realization truncation method:
  - Similar to modred(), but can handle unstable systems
- Optimal Hankel norm approximation
  - Based on Hankel norm optimization technique

Matlab Commands: schmr(), ohklmr()



- Example: Find a 3rd-order approximate model for

$$G(s)=\frac{68.6131s^5+80.3787s^4+67.087s^3+29.9339s^2+8.8818s+1}{0.0462s^6+3.5338s^5+16.5609s^4+28.4472s^3+21.7611s^2+7.6194s+1}$$



Matlab code:

```
num=[68.6131,80.3787,67.087,29.9339,8.8818,1];
den=[0.0462,3.5338,16.5609,28.4472,21.7611,7.6194,1];
G=ss(tf(num,den)); [Gb,g,T]=balreal(G);
Gr=modred(Gb,[4,5,6]); zpk(Gr),
Gs=schmr(G,1,3); zpk(Gs),
Gh=ohklmr(G,1,3); zpk(Gh),
figure(1), bode(G,Gr, '-g',Gs,'-.b',Gh,':r'); grid;
figure(2), step(G,Gr, '-g',Gs,'-.b',Gh,':r');
```

$$G_r(s)=\frac{-0.28747(s-5398)(s^2+0.3576s+0.2519)}{(s+76.48)(s^2+3.853s+5.1111)}$$

$$G_s(s)=\frac{1485.3076(s^2+0.1789s+0.2601)}{(s+71.64)(s^2+3.881s+4.188)}$$

$$G_h(s)=\frac{1527.8048(s^2+0.2764s+0.2892)}{(s+73.93)(s^2+3.855s+4.585)}$$