EE480.3 Digital Control Systems

Part 7. Controller Design I.
- Pole Assignment Method

Kunio Takaya

Electrical and Computer Engineering

University of Saskatchewan

March 3, 2008

** Go to full-screen mode now by hitting CTRL-L
Contents

1  CONTROL SYSTEM DESIGN 1.  3

2  Ackermann’s Formula  20

3  Example of Ackermann’s Pole Assignment Method  26
1 CONTROL SYSTEM DESIGN 1.

Pole Assignment Method

Pole assignment method is a method to adjust the poles of a given system by adjusting the feedback gains from state variables. This method can be applied to such a system that is described by the state equations.
Discrete time state equation is

\[ \mathbf{x}(k + 1) = P\mathbf{x}(k) + q\mathbf{u}(k) \]

Since a linear combination of the state variables is fed back to the
differential node, the input $u(k)$ is replaced by

$$u(k) = r(k) - Kx(k).$$

Where, $K = [K_1, K_2, \cdots K_n]$, a row vector, assuming the system is of $n$th order. $x(k)$ is a column vector of size $n$. Substituting $u(k)$ in the state equation yields,

$$x(k + 1) = Px(k) + q[r(k) - Kx(k)]$$

$$= [P - qK]x(k) + qr(k)$$

Writing the $z$-transform of this equation, we have

$$zx(z) = [P - qK]x(z) + qr(z)$$

$$[zI - P + qK]x(z) = qr(z)$$

$$x(z) = [zI - P + qK]^{-1}qr(z)$$

Therefore, the poles of the closed loop system with the state
feedback is found from

$$\text{det } [zI - P + qK] = 0.$$ 

The solutions of this equation, $z = \lambda_1, z = \lambda_2, \cdots, z = \lambda_n$ are obtained from the expanded form of the determinant, i.e. characteristic equation,

$$(z - \lambda_1)(z - \lambda_2) \cdots (z - \lambda_n) = 0.$$ 

Pole assignment method determines the state feedback gain vector $K$ for a given set of poles at $z = \lambda_1, z = \lambda_2, \cdots, z = \lambda_n$. 
z-plane poles of 2nd order systems

Referring to the standard 2nd order transfer function,

\[ G_2(s) = \frac{\omega_n^2}{s^2 + 2\zeta \omega_n s + \omega_n^2}, \]

the poles of this system are located at

\[ s = -\zeta \omega_n \pm j\omega_n \sqrt{1 - \zeta^2}. \]

This equation defines damping ratio \( \zeta \) and damped natural frequency \( \omega_d \) as

\[ \zeta = \cos \theta \quad \text{and} \quad \omega_d = \omega_n \sqrt{1 - \zeta^2}. \]

In the z-plane, poles are (see textbook p.216):

\[ z = e^{sT} = e^{-\zeta \omega_n T} \angle \pm \omega_n T \sqrt{1 - \zeta^2} = r \angle \pm \theta \]
\[ \zeta \omega_n T = -\ln r \quad \text{from } e^{-\zeta \omega_n T} = r \]
\[ \omega_n T \sqrt{1 - \zeta^2} = \theta \]

Thus,
\[ \frac{\zeta}{\sqrt{1 - \zeta^2}} = \frac{-\ln r}{\theta} \]

damping ratio

\[ \zeta = \frac{-\ln r}{\sqrt{\ln^2 r + \theta^2}} \]

natural freq.

\[ \omega_n = \frac{1}{T} \sqrt{\ln^2 r + \theta^2} \]

time const.

\[ \tau = \frac{1}{\zeta \omega_n} = \frac{-T}{\ln r} \]
Pole Assignment for 2nd Order Systems

Consider the second order case of

\[ x(k + 1) = [P - qK]x(k) + qr(k). \]

Let

\[ P = \begin{bmatrix} p_{11}, & p_{12} \\ p_{21}, & p_{22} \end{bmatrix} \]

\[ q = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} \]

\[ K = [K_1, K_2] \]

\[ qK = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} [K_1, K_2] \]
Therefore, the characteristic equation is obtained as

\[
\text{det } [z I - P + qK] = 0
\]
The characteristic equation of the 2nd order system takes either of two real roots

\[(z - \gamma_1)(z - \gamma_2) = z^2 - (\gamma_1 + \gamma_2)z + \gamma_1\gamma_2 = 0\]

complex roots

\[(z - \alpha - j\beta)(z - \alpha + j\beta) = z^2 + 2\alpha z + \alpha^2 + \beta^2 = 0\]

Equating the characteristic equation of unknown \(K\) and that defined by two desired poles, we have a set of simultaneous equations.

For the case of real roots,

\[
\begin{align*}
p_{11} - q_1 K_1 + p_{22} - q_2 K_2 &= \gamma_1 + \gamma_2 \\
(q_2 p_{12} - q_1 p_{22}) K_1 + (q_1 p_{21} - q_2 p_{11}) K_2 + (p_{11} p_{22} - p_{12} p_{21}) &= \gamma_1 \gamma_2
\end{align*}
\]
\[
\begin{align*}
q_1 K_1 + q_2 K_2 &= - (\gamma_1 + \gamma_2) + p_{11} + p_{22} \\
(q_2 p_{12} - q_1 p_{22}) K_1 + (q_1 p_{21} - q_2 p_{11}) K_2 &= \gamma_1 \gamma_2 - (p_{11} p_{22} - p_{12} p_{21})
\end{align*}
\]

\[
\begin{bmatrix}
q_1 & q_2 \\
q_2 p_{12} - q_1 p_{22} & (q_1 p_{21} - q_2 p_{11})
\end{bmatrix}
\begin{bmatrix}
K_1 \\
K_2
\end{bmatrix}
= \begin{bmatrix}
-2(\gamma_1 + \gamma_2) + p_{11} + p_{22} \\
\gamma_1 \gamma_2 - (p_{11} p_{22} - p_{12} p_{21})
\end{bmatrix}
\]
For the case of complex poles,

\[
\begin{align*}
p_{11} - q_1 K_1 + p_{22} - q_2 K_2 &= 2\alpha \\
(q_2 p_{12} - q_1 p_{22}) K_1 + (q_1 p_{21} - q_2 p_{11}) K_2 + (p_{11} p_{22} - p_{12} p_{21}) &= \alpha^2 + \beta^2
\end{align*}
\]

\[
\begin{bmatrix}
q_1 & q_2 \\
q_2 p_{12} - q_1 p_{22} & (q_1 p_{21} - q_2 p_{11})
\end{bmatrix}
\begin{bmatrix}
K_1 \\
K_2
\end{bmatrix}
= \begin{bmatrix}
-2\alpha + p_{11} + p_{22} \\
\alpha^2 + \beta^2 - (p_{11} p_{22} - p_{12} p_{21})
\end{bmatrix}
\]
The continuous time transfer function, \( G(s) = \frac{1}{s(s + 1)} \) can be described by the continuous time state equation,

\[
\dot{x}(t) = \begin{bmatrix} 0 & 1 \\ 0 & -1 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t)
\]

\[
y(t) = [1, 0] x(t)
\]

The discrete time model sampled at \( T = 0.1 \) is obtained as

\[
x(k + 1) = Px(k) + qu(k)
\]
\[
\begin{bmatrix}
1 & 0.0952 \\
0 & 0.905
\end{bmatrix} x(k) + \begin{bmatrix}
0.00484 \\
0.0952
\end{bmatrix} u(k)
\]

\[y(k) = [1, 0] x(k)\]

The state feedback equation is

\[u(k) = -K_1 x_1(k) - K_2 x_2(k)\]

\[
x(k + 1) = P x(k) + q u(k)
\]

\[
= \begin{bmatrix}
1 & 0.0952 \\
0 & 0.905
\end{bmatrix} x(k) - \begin{bmatrix}
0.00484 \\
0.0952
\end{bmatrix} [K_1 x_1(k) + K_2 x_2(k)]
\]

\[
= \begin{bmatrix}
1 - 0.00484 K_1 & 0.0952 - 0.00484 K_2 \\
-0.0952 K_1 & 0.905 - 0.0952 K_2
\end{bmatrix} x(k).
\]
The characteristic equation \(|zI - A_c| = 0\) becomes,

\[z^2 + (0.00480K_1 + 0.0952K_2 - 1.905)z + 0.00468K_1 - 0.0952K_2 + 0.905 = 0\]

When the desired characteristic equation is,

\[\alpha_c(z) = (z - \lambda_1)(z - \lambda_2) = z^2 - (\lambda_1 + \lambda_2)z + \lambda_1\lambda_2 = 0\]

\[K_1 = 105[\lambda_1\lambda_2 - (\lambda_1 + \lambda_2) + 1.0]\]
\[K_2 = 14.67 - 5.34\lambda_1\lambda_2 - 5.17(\lambda_1 + \lambda_2)\]
The unity gain feedback $K = [K_1, K_2] = [1, 0]$ has its characteristic equation,

$$z^2 - 1.9z + 0.91 = 0.$$ 

The roots are $z = 0.964 \angle \pm 0.091 = r \angle \pm \theta$.

$$\zeta = \frac{-\ln r}{\sqrt{\ln^2 r + \theta^2}} = 0.46$$

$$\tau = \frac{1}{\zeta \omega_n} = \frac{-T}{\ln r} = 2.12\text{sec.}$$

$$\omega_n = \frac{1}{T} \sqrt{\ln^2 r + \theta^2}$$
The unity gain feedback gives satisfactory $\zeta$ but the time constant 2.12 sec. is too large. Try to reduce the time constant to 1 sec.

$$\ln r = \frac{-T}{\tau} = -0.1/1 = -0.1 \implies r = 0.905$$

$$\theta^2 = \frac{\ln^2 r}{\zeta^2} - \ln^2 r = (0.193)^2$$

From the characteristic equation, the desired poles are at $z = r \angle \pm \theta$. Thus,

$$(z - 0.888 - j0.173)(z - 0.888 + j0.173) = z^2 - 1.776z + 0.819 = 0.$$ 

This yields the pole assignment solution,

$$[K_1, K_2] = [4.52, 1.12]$$

The initial-condition response with $x_1(0) = 1.0$ and $x_2(0) = 0.0$ is shown below.
2 Ackermann’s Formula

Since it is difficult to extend the approach demonstrated in the second order case to a higher order system, the pole assignment must rely on a more practical procedure called Ackermann’s formula. When we know the desired poles, $z = \lambda_1, z = \lambda_2, \cdots z = \lambda_n$ for a $n$-th order system, write the characteristic equation,

$$\alpha_c(z) = (z - \lambda_1)(z - \lambda_2) \cdots (z - \lambda_n)$$

$$= z^n + \alpha_{n-1}z^{n-1} + \alpha_{n-2}z^{n-2} \cdots \alpha_1z + \alpha_0$$

$$= 0$$

Since the characteristic equation of state feedback systems is

$$\det [zI - P + qK] = 0,$$

$$\det [zI - P + qK] = \alpha_c(z).$$
In order to solve for the unknown gain vector $K$, form a matrix,

$$\alpha_c(\mathbf{P}) = \mathbf{P}^n + \alpha_{n-1}\mathbf{P}^{n-1} + \alpha_{n-2}\mathbf{P}^{n-2} \cdots \alpha_1\mathbf{P} + \alpha_0\mathbf{I}.$$  

Then, Ackermann’s formula for the gain matrix $K$ is given by

$$K = \left[ \begin{array}{c} 0, 0, 0, \cdots, 1 \end{array} \right] [ \mathbf{q}, \mathbf{P}\mathbf{q} \cdots \mathbf{P}^{n-2}\mathbf{q}, \mathbf{P}^{n-1}\mathbf{q} ]^{-1} \alpha_c(\mathbf{P}).$$

As defined earlier, $K$ is a row vector of $n$ elements.
Example

The continuous time transfer function, \( G(s) = \frac{1}{s(s + 1)} \) can be described by the continuous time state equation,

\[
\dot{x}(t) = \begin{bmatrix} 0 & 1 \\ 0 & -1 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t)
\]

\[ y(t) = [1, 0] x(t) \]

The discrete time model sampled at \( T = 0.1 \) is obtained as

\[
x(k + 1) = P x(k) + q u(k)
\]

\[
= \begin{bmatrix} 1 & 0.0952 \\ 0 & 0.905 \end{bmatrix} x(kt) + \begin{bmatrix} 0.00484 \\ 0.0952 \end{bmatrix} u(k)
\]

\[ y(k) = [1, 0] x(k). \]
The desired poles are $z = 0.8880 \pm j0.1745$. This determines

$$\alpha_c(z) = (z-0.8880+j0.1745)(z-0.8880-j0.1745) = z^2 - 1.776z + 0.819$$

Hence,

$$\alpha_c(P) = \begin{bmatrix} 1 & 0.0952 \\ 0 & 0.905 \end{bmatrix}^2 - 1.776 \begin{bmatrix} 1 & 0.0952 \\ 0 & 0.905 \end{bmatrix} + 0.819 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0.043 & 0.01228 \\ 0 & 0.03075 \end{bmatrix}$$

Also,

$$Pq = \begin{bmatrix} 1 & 0.0952 \\ 0 & 0.905 \end{bmatrix} \begin{bmatrix} 0.00484 \\ 0.0952 \end{bmatrix} = \begin{bmatrix} 0.0139 \\ 0.0862 \end{bmatrix}$$
Thus,

$$[q, Pq]^{-1} = \begin{bmatrix} 0.00484 & 0.0139 \\ 0.0952 & 0.0862 \end{bmatrix}^{-1}$$

$$= \begin{bmatrix} \begin{pmatrix} -95.13 \\ 105.1 \end{pmatrix} & \begin{pmatrix} 15.34 \\ -5.342 \end{pmatrix} \end{bmatrix}$$

Then, the gain vector is given as

$$K = [0, 1] \begin{bmatrix} -95.13 & 15.34 \\ 105.1 & -5.342 \end{bmatrix} \begin{bmatrix} 0.043 & 0.01228 \\ 0 & 0.03075 \end{bmatrix}$$

$$= [4.52, 1.12]$$
MATLAB Solution

This matrix calculations can be done with MATLAB by the following script.

```
% Ackermann’s formula example

order=2;
P=[1 0.0952
  0 0.905];
q=[0.00484
   0.9052];
alphac=[1 -1.776 0.819];
disp(' Desired roots in z-plane are:')
disp(' ')
roots(alphac)
AMB=q;
AMBT=q;
for n=2:order
    AMBT=P*AMBT;
```
for nn=1:order
    AMB(nn,n)=AMBT(nn,1);
end
end
AMBI=inv(AMB);
CC=polyvalm(alphac,P);
CCC=AMBI*CC;
disp(' The gain matrix K is: ')
disp(' ')
K=CCC(order,:)
----------------------------

Desired roots in z-plane are:
ans =
    0.8880 + 0.1745i
    0.8880 - 0.1745i
The gain matrix K is:
K =
    4.5155    1.1255
3 Example of Ackermann’s Pole Assignment Method

There is a DC servo motor system

\[ G(s) = \frac{1}{s(s^2 + s + 1)}. \]

This DC servo motor is subjected to digital control perhaps by some kind of digital compensator.

\[ \text{---> sampler ---> ZOH ---> } \frac{1}{s(s^2 + s + 1)} \text{ ---> sampler -->} \]

Problem here is to tune up the DC servo motor to become critical damping by state feedback. So, we attempt to move the two poles located at \( s = -0.5000 \pm j0.8660 \) to a location on the real axis. Since \( \frac{1}{s} \) i.e. a pole at the origin of \( s \)-plane is integration to change angular velocity to angular position (rotational angle), do not attempt to move this pole. The pole at \( s = 0 \) corresponds to \( z = 1 \).
in \( z \)-plane. The following MATLAB program uses Ackermann’s method to move the poles of this system to an assigned location. This program has four preassigned pole locations:

1. Case (1) \([1, 0.97+j*0.05, 0.97-j*0.05]\) under damped
2. Case (2) \([1, 0.98+j*0.05, 0.98-j*0.05]\) under damped
3. Case (3) \([1, 0.97, 0.97]\) critical damping
4. Case (4) \([1, 0.95, 0.95]\) critical damping

Pay attention to the gain (final position) for a pulse input that drive the motor to turn for a limited time duration. If the real part of the two poles are moved away from \( z = 1 \), more damping slows the motor and the number of turns (rotational angle) gets less.
Open loop response to pulse

Response without state feedback
State Feedback by Pole Assignment

Response for Case 1
Response for Case 2
Response for Case 3
State Feedback by Pole Assignment

Response for Case 4
% Example of Ackerman’s State Feedback, Pole Assignment Method
% K. Takaya, 2004

% find the state space model of a transfer function in continuous time domain
% --> sampler --> ZOH --> 1/(s(s^2+s+1)) --> sampler -->
% This system is a DC motor that has two poles.
% 1/s means that output is in rotational angle.
% Do not attempt to move the pole at s=0 (or z=1 in z-plane)
% just model analog part only: 1/(s(s^2+s+1))
numA=[1]
denA=[1, 1, 1, 0]
[A, b, c, d]=tf2ss(numA,denA)
T=0.05;
% calculate discrete time (digital) state equation
[P, q]=c2d(A, b, T)
% find poles of this system
disp('Poles of the discrete time model');
poles=eig(P)

% simulate system response with a pulse input
N=500;
pulse=zeros(1,N);
pulse(N/20+1:N/20+N/20)=ones(1,N/20);
timebase=[0: T: (N-1)*T];
% use dlsim for simulation
[y,x]=dlsim(P, q, c, d, pulse);

% plot input and output
figure(1);
plot(timebase, pulse, timebase, y); title('Open loop response to pulse');

%verify the final DC level
stf=tf(numA,denA)
dft=c2d(stf,T,'zoh')
[zz,pp,kk]=zpkdata(dft,'v')
%Response to 25 impulses (to make a rectangular pulse)
disp('****DC level at the steady state is:');
Gdc=25*kk*sum(poly(zz))/sum(poly(pp(2:3)));

-----------------------------------------------------------
% apply Ackermann’s pole assignment method to do state feedback
-----------------------------------------------------------

cs=input('Enter case number 1 or 2 or 3 or 4 =','s');
switch cs
  case '1'
    newpoles=[1, 0.97+j*0.05, 0.97-j*0.05]
  case '2'
    newpoles=[1, 0.97+j*0.05, 0.97-j*0.05]
end
newpoles = [1, 0.98+j*0.05, 0.98-j*0.05]
case '3'
    newpoles = [1, 0.97, 0.97]
case '4'
    newpoles = [1, 0.95, 0.95]
end
alphac = poly(newpoles)
K = ackermann(P, q, alphac)
% state feedback compensated matrix of Pc
Pc = P - q*K
disp('new poles by state feedback K');
newpoles = eig(Pc)

% use dlsim for simulation of compensated system
[y1, x1] = dlsim(Pc, q, c, d, pulse);
% plot input and output
figure(2);
plot(timebase, pulse, timebase, y1); title('State feedback system by pole assignment');
% plot(timebase, y1);
title('State Feedback by Pole Assignment');

% Convert discrete time system back to transfer function to find DC gain
[num1, den1] = ss2tf(Pc, q, c, d, 1)
% check DC gain
%+++++ Write your code to calculate the steady state DC level
%+++++ of the pole assigned feedback system with K

% check if poles are the same
disp('check if poles of the transfer ftn are the same');
check_poles=roots(den1)
Subroutine for Ackermann’s method is given below.

```matlab
function K=ackermann(A, B, alphac)
% Ackermann’s formula discrete time version
%order=2;
%A=[1, 0.095; 0, 0.905];
%B=[0.00484; 0.0952];
%alphac=[1, -1.776, 0.819];
order=size(A,1);
AMB=B;
AMBT=B;
for n=2:order
    AMBT=A*AMBT;
    for nn=1:order
        AMB(nn,n)=AMBT(nn,1);
    end
end
AMBI=inv(AMB);
CC=polyvalm(alphac,A);
CCC=AMBI*CC;
%disp(' The gain matrix K is:')
%disp(' ')
K=CCC(order,:);
```
1. Design a state feedback control system that makes the following 3rd order underdamped system to be critically damped by the pole assignment method. (this system is similar to Figure 9-1 in page 338 in textbook, but 3rd order instead of 2nd order)

\[ \text{sampler} \rightarrow \text{ZOH} \rightarrow \frac{1}{s(s^2+s+1)} \rightarrow \text{sampler} \rightarrow \]

First convert this given transfer function to a state equation by \text{tf2ss} in analog domain, then use \text{c2d} to convert the obtained continuous time state equation into a discrete time state equation. Then, apply Ackermann’s design formula. Use \( T=0.05 \) second. MATLAB program for this problem is provided by the name PoleAssignment2008.m The specific assignment is, therefore, to run this program and interpret the
results. In particular, calculate the steady state DC value for the rectangular pulse of 25 ones for each case of the 4 state feedback systems resulted from the pole assignment method.