Chapter 7

Trim, Linearization, and LQR Controller Design

In Chapter 5 notes and Project 2 Assignment, we have learned how to use the nonlinear aircraft flight dynamics model together with a trim computer program to find a desired aircraft flight equilibrium and obtain the linearized dynamics model at the flight equilibrium. The linearized dynamics model has provided tremendous insight and explanations on why a manual control of an open-loop aircraft flight is an extreme challenging task. The uncompensated aircraft flight dynamics basically is a very lightly damped system, which can enter a long period oscillation easily and not much a pilot can do to manually stop or alleviate it.

The good news is that the linearized flight dynamics model can be employed to design a closed-loop controller that will fundamentally transform the aircraft dynamics into a much more stable one.

Since the aircraft flight control system is a multi-input multi-output system, we will design a feedback controller using a state-space LQR optimization approach to minimize the effect of disturbances/turbulences on the aircraft and keep the system more stable.

directory:
C:\ …. \Documents\MATLAB\F18_model_Teach\2024Summer_R2015b\4_F18full_DUtrim_stab_R2015b

this folder includes:

aerodynamics_coefficients_rev2.mat
f18stab.mat
runfgWindowsXP_F18.bat
f18full_DUtrim.mdl
Trim_stab_f18fullDU_a.m

Run Trim_stab_f18fullDU_a.m to

1. **trim the aircraft at a straight level flight with 10 degree angle of attack,**
2. **find the linearized model at the trim,**
3. **use the linearized model to design an LQR optimization controller,**
4. **conduct stability and performance analysis of the closed-loop system of the linearized model with the LQR controller.**

5. conduct the simulation and comment on the flight performance of nonlinear closed-loop system. **(to be covered in Chapter 8)**

```
% Trim_stab_f18fullDU_a.m
% Objectives:
% This file trims the aircraft at a desired flight
% condition, find the linearized model at this trim,
% and design an LQR stabilizing controller for the trim.
```

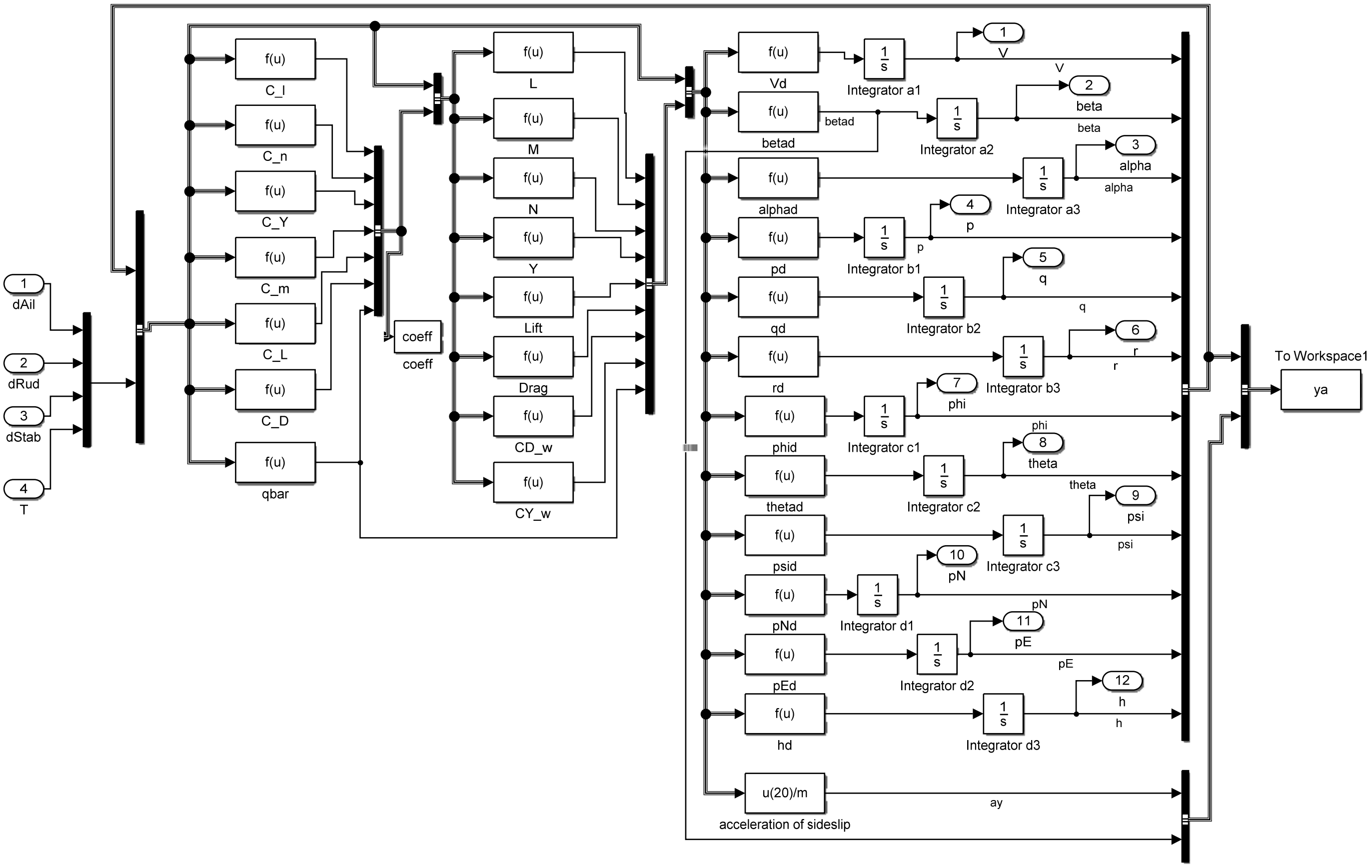
```
% Unit Conversion : Degree <--> Radian
d2r = pi/180;
r2d = 1/d2r;

h0=25000;
%=========================================================================
% F/A-18 data
%
% Aircraft Physical Paramters
% Reference: S. B. Buttrill and P. D. Arbuckle  and K. D. Hoffler
%            Simulation model of a twin-tail, high performance airplane
%            NASA 1992, NASA TM-107601

S = 400;                % ft^2
b =  37.4;                % ft
c =  11.52;             % ft
rho = 1.0660e-003;      % slugs/ft^3  --- 25C / 25000 ft
Ixx = 23000;            % slugs*ft^2
Iyy = 151293;           % slugs*ft^2
Izz = 169945;           % slugs*ft^2
Ixz = - 2971;           % slugs*ft^2
m = 1034.5;             % slugs
g = 32.2;               % ft/s^2

%=========================================================================
% Statename , Inputname and Outputname
%
% Baseline and Revised has different feedback channels
% So, their output equations are different

statenames = {'V (ft/s)','Beta (rad)','Alpha (rad)','Roll Rate (rad/s)',...
    'Pitch Rate (rad/s)','Yaw Rate (rad/s)','Phi (rad)','Theta (rad)',...
    'Yaw (rad)','pN (ft)','pE (ft)','h (ft)'}'
inputnames = { 'Aileron (rad)','Rudder (rad)','Stabilator (rad)', 'T (lb)'}'
%
%=========================================================================
% load aerodynamics coefficents
% the aerodynamic data is resulted from the Minnesotadata rev2.
load aerodynamics_coefficients_rev2

%=========================================================================
% Trimming initial conditions
%=========================================================================

%-------------------------------------------
% Initialization
% Initialize to find TrimA condition

% State Initial Value
V       =  300;         % Airspeed , ft/s  Guess, not fixed
beta    =  0*d2r;       % Sideslip Angle, rad   Desired to be zero
alpha   =  10*d2r;       % Angle-of-attack, rad   Desired to be 10 deg

p       =  0*d2r;       % Roll rate, rad/s   Desired to be zero
q       =   0*d2r;       % Pitch rate, rad/s   Desired to be zero
r       =  0*d2r;       % Yaw rate, rad/s   Desired to be zero
```

```
phi     =  0*d2r;       % Roll Angle, rad   Desired to be zero
theta   =  10*d2r;       % Pitch Angle, rad   Desired to be 10 deg so that gamma can be
zero
psi     =  0*d2r;       % Yaw Angle, rad   Guess, not fixed

pN      =  0;           % Position North,  ft   Not fixed
pE      =  0;           % Position East,  ft   Not fixed
h       =  25000;           % Altitude,  ft   Not fixed

% Stack Initial Condition for State
x_init = [V;beta;alpha;p;q;r;phi;theta;psi;pN;pE;h];
x = x_init;

% Initialize Input Value

d_STAB   = 0*d2r;    %Not fixed
d_AIL    = 0*d2r;    %Not fixed
d_RUD    = 0*d2r;    %Not fixed
T = 5000;    %Not fixed

u_init = [d_AIL; d_RUD; d_STAB; T];
%u=u_init
disp('initial values')
x_init(1)
x_init(2:9)*r2d
x_init(10:11)
x_init(12)
u_init(1:3)*r2d
u_init(4)

%===========================================================================
% Operating Point Specificaton Setup

open('f18full_DUtrim')
opys = operspec('f18full_DUtrim');

opys.States(1).Known  =  0;
opys.States(2).Known  =  1;
opys.States(3).Known  =  1;
%opys.States(2).Known  =  0;
%opys.States(3).Known  =  0;

opys.States(4).Known  =  1;
opys.States(5).Known  =  1;
opys.States(6).Known  =  1;
%opys.States(4).Known  =  0;
%opys.States(5).Known  =  0;
%opys.States(6).Known  =  0;

opys.States(7).Known  =  1;
opys.States(8).Known  =  1;
opys.States(9).Known  =  1;
%opys.States(7).Known  =  0;
%opys.States(8).Known  =  0;
%opys.States(9).Known  =  0;

opys.States(10).Known  =  0;
opys.States(11).Known  =  0;
```

```
opys.States(12).Known  =  0;


opys.States(1).steadystate = 1;
opys.States(2).steadystate = 1;
opys.States(3).steadystate = 1;
opys.States(4).steadystate = 1;
opys.States(5).steadystate = 1;
opys.States(6).steadystate = 1;
opys.States(7).steadystate = 1;
opys.States(8).steadystate = 1;
opys.States(9).steadystate = 1;
opys.States(10).steadystate = 0;
opys.States(11).steadystate = 0;
opys.States(12).steadystate = 0;


%Setting the Input value

opys.inputs(1).known = 0;
opys.inputs(2).known = 0;
opys.inputs(3).known = 0;
opys.inputs(4).known = 0;

opys.inputs(3).u = d_STAB;
opys.inputs(2).u = d_RUD;
opys.inputs(1).u = d_AIL;
opys.inputs(4).u = T;

opys.inputs(4).min = 0;
opys.inputs(4).max = 38000;
%===========================================================================
% Finding Trim/ Operating point
opt1 = optimset('MaxFunEvals',1e+04);
opt = linoptions('OptimizationOptions',opt1);
[ysop,rep] = findop('f18full_DUtrim',opys,opt);
get(ysop)



%===========================================================================
% Extracting Trim Point

x_trim = [ysop.States(1).x;  ysop.States(2).x; ysop.States(3).x;...
          ysop.States(4).x;  ysop.States(5).x; ysop.States(6).x;...
          ysop.States(7).x;  ysop.States(8).x; ysop.States(9).x;...
          ysop.States(12).x]

u_trim = [ysop.Inputs(1).u; ysop.Inputs(2).u; ysop.Inputs(3).u; ysop.Inputs(4).u]

disp('Trimmed Value')
x_trim(1)
x_trim(2:9)*r2d
x_trim(10)
u_trim(1:3)*r2d
u_trim(4)

Trim.T      = u_trim(4); % lb
Trim.elev   = u_trim(3); % -0.022 rad =  -1.2616 deg
```

```
Trim.ail    = u_trim(1); %rad
Trim.rud    = u_trim(2); %rad

Trim.V      = x_trim(1);   %ft/s
Trim.alpha  = x_trim(3);   %rad
Trim.q      = 0;        %rad/s
Trim.theta  = x_trim(8);   %rad
Trim.h      = h0;      %ft

Trim.beta   = 0;   %rad
Trim.p      = 0;   %rad/s
Trim.r      = 0;   %rad/s
Trim.phi    = 0;   %rad

Trim.psi    = 0;   %rad/s
Trim.pN     = 0;   %ft
Trim.pE     = 0;    %ft

%==========================================================================
% Creating Open Loop Linearized Model

[A ,B ,C, D] = linmod('f18full_DUtrim',x_trim,u_trim);
A_trim = A([1:8], [1:8]);
B_trim = B([1:8],:);
C_trim = C([1:8], [1:8]);
D_trim = D([1:8],:);

A_longltrl = A([1 3 5 8 2 4 6 7], [1 3 5 8 2 4 6 7])
B_longltrl = B([1 3 5 8 2 4 6 7], [3 4 1 2])
C_longltrl = C([1 3 5 8 2 4 6 7], [1 3 5 8 2 4 6 7])
D_longltrl = D([1 3 5 8 2 4 6 7], [3 4 1 2])



%==========================================================================
% Decoupled longitudinal
% Longitudinal states [ V  alpha q theta ]
% Longitudinal controls [d_STAB   T]
display('Longitudnal states: [V alpha q theta ]')
display('Longitudinal controls [d_STAB   T]')

A_x = A_longltrl([1:4], [1:4])
B_x = B_longltrl([1:4], [1:2])

%{
C_x = eye(4)
D_x = zeros(4,2)
sys_x = ss(A_x,B_x,C_x,D_x)
[Wn,Zeta,P] = damp(sys_x)
%}

% Lateral states [beta p r phi]
% Lateral controls [d_AIL  d_RUD]
display('Lateral states: [beta p r phi]')
display('Lateral controls [d_AIL  d_RUD]')

A_y = A_longltrl([5:8], [5:8])
B_y = B_longltrl([5:8], [3:4])
```

```
display('eigenvalues of A_longltrl')
eig(A_longltrl)

display('eigenvalues of A_x')
eg_x = eig(A_x)
wn_x = abs(eg_x(4))
ze_x = -real(eg_x(4))/wn_x

display('eigenvalues of A_y')
eg_y = eig(A_y)
wn_y = abs(eg_y(1))
ze_y = -real(eg_y(1))/wn_y

%Checking controllability
  disp('checking longitudinal controllability');
  Contp_x=[B_x A_x*B_x A_x^2*B_x A_x^3*B_x];
  RankOfCont_x=svd(Contp_x)

  disp('checking lateral controllability');
  Contp_y=[B_y A_y*B_y A_y^2*B_y A_y^3*B_y A_y^4*B_y];
  RankOfCont=svd(Contp_y)

% Design of F_x and F_y

% Nominal stabilizing controller design for longitudinal control
% Design an LQR state feedback gain F_x based on X_x
%  Q_x=diag([0.01 1 1 10]);
%  R_x=diag([100  0.1]);
%Q_x=diag([0.01 1 1 1]);   %2nd design
%R_x=diag([100  0.1]);      % 2nd design
Q_x=diag([0.01 1 1 1]);   % 4th  design
R_x=diag([1000  0.001]);   %  4th design

  X_x=are(A_x,B_x*inv(R_x)*B_x',Q_x);
  F_x=-inv(R_x)*B_x'*X_x
  disp('longitudinal eigenvalues of A_x+B_x*F_x');
  eg_xABF = eig(A_x+B_x*F_x)
  wn_xABF = abs(eg_xABF(4))
  ze_xABF = -real(eg_xABF(4))/wn_xABF

% Nominal stabilizing controller design for lateral control
% Design the state feedback gain F_y based on X_y
  Q_y=diag([10 100 10 10]);
  R_y=diag([100 10]);
  X_y=are(A_y,B_y*inv(R_y)*B_y',Q_y);
  F_y=-inv(R_y)*B_y'*X_y
  disp('lateral eigenvalues of A_y+B_y*F_y');
  eg_yABF = eig(A_y+B_y*F_y)
  wn_yABF = abs(eg_yABF(2))
  ze_yABF = -real(eg_yABF(2))/wn_yABF

save f18stab Trim A_x B_x A_y B_y F_x F_y
```




```
>> Trim_stab_f18fullDU_a
statenames =
    'V (ft/s)'
    'Beta (rad)'
    'Alpha (rad)'
    'Roll Rate (rad/s)'
    'Pitch Rate (rad/s)'
    'Yaw Rate (rad/s)'
    'Phi (rad)'
    'Theta (rad)'
    'Yaw (rad)'
    'pN (ft)'
    'pE (ft)'
    'h (ft)'
inputnames =
    'Aileron (rad)'
    'Rudder (rad)'
    'Stabilator (rad)'
    'T (lb)'
initial values
ans =
   300
ans =
     0
    10
     0
```

```
     0
     0
     0
    10
     0
ans =
     0
     0
ans =
       25000
ans =
     0
     0
     0
ans =
        5000
> In linoptions (line 131)
  In Trim_stab_f18fullDU_a (line 157)

Local minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the default value of the function tolerance,
and constraints are satisfied to within the default value of the constraint tolerance.

<stopping criteria details>


 Operating Point Search Report:
---------------------------------

 Operating Report for the Model f18full_DUtrim.
 (Time-Varying Components Evaluated at time t=0)

Operating point specifications were successfully met.
States:
----------
(1.) f18full_DUtrim/Integrator a1
      x:            436      dx:     -4.56e-08 (0)
(2.) f18full_DUtrim/Integrator a2
      x:              0      dx:             0 (0)
(3.) f18full_DUtrim/Integrator a3
      x:          0.175      dx:     -1.02e-06 (0)
(4.) f18full_DUtrim/Integrator b1
      x:              0      dx:             0 (0)
(5.) f18full_DUtrim/Integrator b2
      x:              0      dx:      8.83e-09 (0)
(6.) f18full_DUtrim/Integrator b3
      x:              0      dx:             0 (0)
(7.) f18full_DUtrim/Integrator c1
      x:              0      dx:             0 (0)
(8.) f18full_DUtrim/Integrator c2
      x:          0.175      dx:             0 (0)
(9.) f18full_DUtrim/Integrator c3
      x:              0      dx:             0 (0)
(10.) f18full_DUtrim/Integrator d1
      x:              0      dx:           436
(11.) f18full_DUtrim/Integrator d2
      x:              0      dx:             0
```

```
(12.) f18full_DUtrim/Integrator d3
      x:       2.5e+04      dx:            0

Inputs:
----------
(1.) f18full_DUtrim/dAil
      u:              0    [-Inf Inf]
(2.) f18full_DUtrim/dRud
      u:              0    [-Inf Inf]
(3.) f18full_DUtrim/dStab
      u:         -0.022    [-Inf Inf]
(4.) f18full_DUtrim/T
      u:       5.47e+03    [0 3.8e+04]

Outputs:
----------
(1.) f18full_DUtrim/V
      y:            436    [-Inf Inf]
(2.) f18full_DUtrim/beta
      y:              0    [-Inf Inf]
(3.) f18full_DUtrim/alpha
      y:          0.175    [-Inf Inf]
(4.) f18full_DUtrim/p
      y:              0    [-Inf Inf]
(5.) f18full_DUtrim/q
      y:              0    [-Inf Inf]
(6.) f18full_DUtrim/r
      y:              0    [-Inf Inf]
(7.) f18full_DUtrim/phi
      y:              0    [-Inf Inf]
(8.) f18full_DUtrim/theta
      y:          0.175    [-Inf Inf]
(9.) f18full_DUtrim/psi
      y:              0    [-Inf Inf]
(10.) f18full_DUtrim/pN
      y:              0    [-Inf Inf]
(11.) f18full_DUtrim/pE
      y:              0    [-Inf Inf]
(12.) f18full_DUtrim/h
      y:        2.5e+04    [-Inf Inf]

      Model: 'f18full_DUtrim'
     States: [12x1 opcond.StatePoint]
     Inputs: [4x1 opcond.InputPoint]
       Time: 0
    Version: 2
x_trim =
   1.0e+04 *
    0.0436
         0
    0.0000
         0
         0
         0
         0
    0.0000
         0
    2.5000
u_trim =
```

```
   1.0e+03 *
         0
         0
   -0.0000
    5.4705
Trimmed Value
ans =
  435.9249
ans =
     0
    10
     0
     0
     0
     0
    10
     0
ans =
       25000
ans =
         0
         0
   -1.2616
ans =
   5.4705e+03
> In dlinmod (line 195)
  In linmod (line 59)
  In Trim_stab_f18fullDU_a (line 203)
Warning: Extra states are being set to zero.
> In DAStudio.warning (line 28)
  In dlinmod (line 217)
  In linmod (line 59)
  In Trim_stab_f18fullDU_a (line 203)
A_longltrl =
   -0.0239  -28.3172         0  -32.2000         0         0         0         0
   -0.0003   -0.3621    1.0000         0         0         0         0         0
    0.0000   -2.2115   -0.2532         0         0         0         0         0
         0         0    1.0000         0         0         0         0         0
         0         0         0         0   -0.0374    0.1736   -0.9848    0.0727
         0         0         0         0   -8.5430   -0.8883    0.8762         0
         0         0         0         0    0.8860    0.0399   -0.1895         0
         0         0         0         0         0    1.0000    0.1763         0
B_longltrl =
   -3.8114    0.0010         0         0
   -0.0515   -0.0000         0         0
   -2.8791         0         0         0
         0         0         0         0
         0         0   -0.0149    0.0207
         0         0    8.3321    0.9541
         0         0   -0.0420   -0.6277
         0         0         0         0
C_longltrl =
     1     0     0     0     0     0     0     0
     0     1     0     0     0     0     0     0
     0     0     1     0     0     0     0     0
     0     0     0     1     0     0     0     0
     0     0     0     0     1     0     0     0
     0     0     0     0     0     1     0     0
     0     0     0     0     0     0     1     0
```

```
     0     0     0     0     0     0     0     1
D_longltrl =
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
Longitudnal states: [V alpha q theta ]
Longitudinal controls [d_STAB   T]
A_x =
   -0.0239  -28.3172         0  -32.2000
   -0.0003   -0.3621    1.0000         0
    0.0000   -2.2115   -0.2532         0
         0         0    1.0000         0
B_x =
   -3.8114    0.0010
   -0.0515   -0.0000
   -2.8791         0
         0         0
Lateral states: [beta p r phi]
Lateral controls [d_AIL  d_RUD]
A_y =
   -0.0374    0.1736   -0.9848    0.0727
   -8.5430   -0.8883    0.8762         0
    0.8860    0.0399   -0.1895         0
         0    1.0000    0.1763         0
B_y =
   -0.0149    0.0207
    8.3321    0.9541
   -0.0420   -0.6277
         0         0
eigenvalues of A_longltrl
ans =
  -0.3094 + 1.4799i
  -0.3094 - 1.4799i
  -0.0101 + 0.1008i
  -0.0101 - 0.1008i
  -0.2873 + 1.4530i
  -0.2873 - 1.4530i
  -0.4888 + 0.0000i
  -0.0518 + 0.0000i
eigenvalues of A_x
eg_x =
  -0.3094 + 1.4799i
  -0.3094 - 1.4799i
  -0.0101 + 0.1008i
  -0.0101 - 0.1008i
```

The dominant poles are $\begin{matrix} p_1 \\ p_2 \end{matrix} = -\alpha \pm j\omega = -0.0101 \pm j0.1008$

where $\alpha = \varsigma\omega_n$ and $\omega = \omega_n\sqrt{1-\varsigma^2} \quad \rightarrow \quad \omega_n = \sqrt{\alpha^2+\omega^2} = 0.1013, \quad \varsigma = \alpha/\omega_n = 0.1000$

```
wn_x =
    0.1013
ze_x =
```

```
    0.1000

eigenvalues of A_y
eg_y =
  -0.2873 + 1.4530i
  -0.2873 - 1.4530i
  -0.4888 + 0.0000i
  -0.0518 + 0.0000i
wn_y =
    1.4811
ze_y =
    0.1940

checking longitudinal controllability
Rank0fCont_x =
  193.3076
    9.4651
    2.6578
    0.0564
checking lateral controllability
Rank0fCont =
   23.7527
   22.4108
    1.2936
    0.8713
F_x =
   -0.0028   -0.2725    0.1370    0.3315
   -0.0326   -0.7685    0.9373    1.4266
longitudinal eigenvalues of A_x+B_x*F_x
eg_xABF =
  -0.3344 + 1.4927i
  -0.3344 - 1.4927i
  -0.1702 + 0.1126i
  -0.1702 - 0.1126i
```

The dominant poles are $\begin{matrix} p_1 \\ p_2 \end{matrix} = -\alpha \pm j\omega = -0.1702 \pm j0.1126$

where $\alpha = \varsigma\omega_n$ and $\omega = \omega_n\sqrt{1-\varsigma^2} \quad \rightarrow \quad \omega_n = \sqrt{\alpha^2 + \omega^2} = 0.2041, \quad \varsigma = \alpha/\omega_n = 0.834$

```
wn_xABF =
    0.2041
ze_xABF =
    0.8340
F_y =
    0.8739   -0.8937   -0.3573   -0.3094
   -1.8130   -0.7223    2.6598   -0.1995
lateral eigenvalues of A_y+B_y*F_y
eg_yABF =
  -8.6583 + 0.0000i
  -0.9858 + 0.9773i
  -0.9858 - 0.9773i
  -0.3258 + 0.0000i
wn_yABF =
    1.3881
ze_yABF =
    0.7102
>>
```

Introduction to Control System Design

First Edition

Harry Kwatny and Bor-Chin Chang
Drexel University

cognella
SAN DIEGO

10.5 State-Feedback Control via Linear Quadratic Regulator Design

Consider a system described by the following state equation

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{10.34}$$

where $x \in R^n, u \in R^m$, and all the state variables in $x(t)$ are available for feedback. Recall that the behavior and performance of the system are mainly determined by the poles of the system, which are the eigenvalues of the A matrix. Using the state-feedback control to close the loop,

$$u(t) = Fx(t) \tag{10.35}$$

the state equation of the closed-loop system will become

$$\dot{x}(t) = (A + BF)x(t) \tag{10.36}$$

and then the behavior and performance of the closed-loop system will be dictated by the eigenvalues of $A+BF$.

If the system is controllable, or the rank of the controllability matrix

$$\mathscr{C} = [B \ AB \ A^2B \ \cdots \ A^{n-1}B]$$

is n, then the eigenvalues of $A+BF$ can be placed anywhere of the complex plane. The concept of the state feedback was first applied in Section 4.4.3 of the book to stabilize the originally unstable simple inverted pendulum system. More detailed discussions of the state-feedback pole placement approach were given later in Sections 7.5.1 and 7.5.2, respectively, for the direct approach and the transform approach. The direct approach is implemented according to Equations 10.35 and 10.36, which are conceptually simple but computationally can become very complicated for high-order systems. On the other hand, the transform approach requires the state equation be transformed into the companion form, but the effort of the transformation certainly is worthwhile since it has made the computation for high-order computations as easy as that for the low-order systems.

In addition to stabilizing a system at an originally unstable equilibrium, as demonstrated in Section 7.6.4, the state-space pole placement approach can work together with tracking/regulation theory and root locus design, as shown in Section 8.6.3 and Section 8.7, respectively, to achieve aircraft flight path angle tracking control and aircraft altitude regulation.

The reasoning behind the pole placement approach is that the performance of the system is quite related to the pole locations on the complex plane. For example, the typical second-order system underdamped step response is determined by the two complex poles $-\alpha \pm \omega$ or their associated damping ratio ς and natural frequency ω_n, as shown in Section 3.4, particularly demonstrated in Figures 3.10 and 3.11. However, in general, especially for high order systems, the pole locations may not precisely reflect the desired time-domain performance. Furthermore, control-input constraints are not explicitly considered in the pole placement design process. Hence, substantial simulations usually are required to verify the design.

The linear quadratic regulator (LQR) design is a time-domain performance index optimization approach in which the performance index consists of two parts: one accounting for the performance and the other representing the control-input effort. With a chosen weighting function, an optimal trade-off solution can be found by solving the optimization problem.

10.5.1 Performance Index and LQR State Feedback

Consider the system described by the state equation Equation 10.34. Assume the system is stabilizable; then the LQR control design problem is to determine a state-feedback control law $u(t) = Fx(t)$ so that the closed-loop system is stable and the performance index,

$$PI = \int_0^\infty \left(x^T(t)Qx(t) + u^T(t)Ru(t)\right) dt \tag{10.37}$$

is minimized, where $Q \in R^{n\times n}$ is positive semi-definite and $R \in R^{m\times m}$ is positive definite. The definitions of positive definite and positive semi-definite matrices are given in Appendix E.6.

Theorem 10.20 (Linear Quadratic Regulator State Feedback)

Assume the system with the state equation Equation 10.34 is stabilizable, then the optimal state-feedback control that stabilizes the closed-loop system and minimizes the performance index PI of Equation 10.37 for any initial state $x(0) = x_0$ is

$$u = Fx \quad with \quad F = -R^{-1}B^T X \tag{10.38}$$

where $X = X^T \geq 0$ is the unique positive semi-definite stabilizing solution of the algebraic Riccati equation [Zhou et al., 1995, Kailath, 1980],

$$A^T X + XA - XBR^{-1}B^T X + Q = 0 \tag{10.39}$$

and the optimal performance index is $PI_{opt} = x_0^T X x_0$. ■

10.5.2 Stabilizing Solution of the Algebraic Riccati Equation

One way to solving for the stabilizing solution of the algebraic Riccati equation, Equation 10.39, is to use its corresponding Hamiltonian matrix [Zhou et al., 1995, Kailath, 1980],

$$H = \begin{bmatrix} A & -BR^{-1}B^T \\ -Q & -A^T \end{bmatrix} \tag{10.40}$$

The eigenvalue structure of the Hamiltonian matrix has the following interesting property.

Theorem 10.21 (Eigenvalues of the Hamiltonian Matrix)

The set of the $2n$ eigenvalues of the Hamiltonian matrix H are symmetric with respect to the imaginary axis. That is, λ is an eigenvalue of H if and only if $-\lambda$ is. ■

Assume the Hamiltonian matrix H has no eigenvalues on the imaginary axis; then H has n stable eigenvalues in the left half of the complex plane and another n unstable eigenvalues in the right half of the complex plane. The n eigenvectors associated with the stable eigenvalues can be stacked to form a $2n \times n$ complex matrix $T \in C^{2n \times n}$ with partition as follows:

$$T = \begin{bmatrix} T_1 \\ T_2 \end{bmatrix} \tag{10.41}$$

where T_1 and T_2 are both $n \times n$ complex matrices. If T_1 is invertible, then the stabilizing solution of the Riccati equation is

$$X = T_2 T_1^{-1} \tag{10.42}$$

The stabilizing solution X is real, symmetric, positive semi-definite, and unique.

Notice that **the stabilizing solution exists only if the two conditions are satisfied: (1) The Hamiltonian matrix has no eigenvalues on the imaginary axis, and (2) the matrix T_1 is invertible.** These two conditions are satisfied under the two assumptions of the LQR problem formulation, which are: **(1) (A, B) is stabilizable, and (2) the weighting matrices Q and R are positive semi-definite and positive definite, respectively.**

Example 10.22 (The Stabilizing Solution of an Algebraic Riccati Equation)

Consider the system

$$\dot{x}(t) = Ax(t) + Bu(t) = \begin{bmatrix} -2 & -2 \\ 2 & 3 \end{bmatrix} x(t) + \begin{bmatrix} -1 \\ 1 \end{bmatrix} u(t)$$

which was employed in Examples 7.27 and 7.28 to demonstrate the pole-placement state-feedback approaches. Here, we will use the LQR Riccati equation approach to find a state feedback $u(t) = Fx(t)$ so that the closed-loop system is stable and the performance index

$$PI = \int_0^\infty \left(x^T(t)Qx(t) + u^T(t)Ru(t)\right)\,dt$$

is minimized, where the weighting matrices are chosen as

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = 1$$

Note that the open-loop system has poles at -1 and 2; hence, it is unstable. (A, B) is stabilizable since it is controllable. The Hamiltonian matrix corresponding to the algebraic Riccati equation, Equation 10.39, is

$$H = \begin{bmatrix} A & -BR^{-1}B^T \\ -Q & -A^T \end{bmatrix} = \begin{bmatrix} -2 & -2 & -1 & 1 \\ 2 & 3 & 1 & -1 \\ -1 & 0 & 2 & -2 \\ 0 & -1 & 2 & -3 \end{bmatrix}$$

whose eigenvalues are

$$-2.4885, \quad -0.89856, \quad 2.4885, \quad 0.89856$$

and the eigenvectors associated with the stable eigenvalues are

$$T = \begin{bmatrix} T_1 \\ T_2 \end{bmatrix} = \begin{bmatrix} 0.28724 & 0.41001 \\ -0.2049 & -0.19405 \\ -0.3267 & 0.59762 \\ -0.8768 & 0.66112 \end{bmatrix}$$

hence, the stabilizing Riccati solution is

$$X = T_2 T_1^{-1} = \begin{bmatrix} 6.5737 & 10.81 \\ 10.81 & 19.433 \end{bmatrix}$$

The state-feedback gain is thus

$$F = -R^{-1}B^T X = \begin{bmatrix} -4.2361 & -8.6231 \end{bmatrix}$$

and the closed-loop system poles are now the eigenvalues of $A + BF$, which are

$$-2.4885, \quad -0.89856$$

exactly the same as the stable eigenvalues of the Hamiltonian matrix. ■

These numerical results are obtained by running the following MATLAB code:

```
% CSD Ex10.22 State feedback Riccati Hamiltonian
 A=[-2 -2;2 3]; B=[-1;1]; eig_A=eig(A), Q=eye(2); R=1;
 H=[A  -B*inv(R)*B'; -Q  -A'], [V,D]=eig(H), T1=V(1:2,1:2),
 T2=V(3:4,1:2), X=T2*inv(T1),
 Riccati_Eq_Check=A'*X+X*A-X*B*inv(R)*B'*X+Q,
 F=-inv(R)*B'*X, eig_ABF=eig(A+B*F)
```

A MATLAB command: `lqr` can also be employed to find an LQR solution:

```
>> [K,S,E]=lqr(A,B,Q,R)
   K =
     4.2361e+00   8.6231e+00
   S =
     6.5737e+00   1.0810e+01
     1.0810e+01   1.9433e+01
   E =
     -2.4885e+00
     -8.9856e-01
```

Note that $K = -F$, $S = X$, and E gives the eigenvalues of $A - BK$ or $A + BF$. If only the solution of the algebraic Riccati equation is of interest, the MATLAB command: `are` can be applied as follows:

```
>> X=are(A,B*inv(R)*B',Q)
   X =
     6.5737e+00   1.0810e+01
     1.0810e+01   1.9433e+01
```

10.5.3 Weighting Matrices Q and R in the Performance Index Integral

To employ the LQR state-feedback control design approach, usually we start with the knowledge of a nonlinear dynamics model of the system to be controlled, the desired operating equilibrium, disturbances and measurement noises, system model uncertainties, and actuator limitations. **To demonstrate the design process, we will use the simple inverted pendulum system discussed in Section 4.4 to demonstrate how to formulate a state-feedback control problem as an LQR optimization problem.**

The schematic of the simple inverted pendulum system is depicted in Figure 4.8, where the two state variables $x_1 = \theta$ and $x_2 = \dot{\theta}$ are the angular displacement and the angular velocity of the pendulum, respectively, and the control input $u = f_a$ is the external control force perpendicular to the gravity. The nonlinear dynamics model of the system is represented by the following nonlinear state equation:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ \frac{g}{\ell}\sin x_1 - \frac{b}{m\ell^2}x_2 + \frac{1}{m\ell}\cos x_1 \cdot u \end{bmatrix} \tag{10.43}$$

The equilibrium of interest is at $x^* = [0 \quad 0]^T$, which represents the stick at the upright position $\theta = 0$ with zero angular velocity $\dot{\theta} = 0$. Assume $g = 9.8\ \text{m/s}^2$, $\ell = 1.089$ m, $m = 0.918$ kg, $b = 0.551$ Ns so that

$$\frac{g}{\ell} = 9, \quad \frac{b}{m\ell^2} = 0.6, \quad \frac{1}{m\ell} = 1$$

hence, the state equation of the linearized model at the upright stick equilibrium $x^* = [0 \quad 0]^T$ can be found as

$$\dot{x}(t) = Ax(t) + Bu(t) = \begin{bmatrix} 0 & 1 \\ 9 & -0.6 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t) \tag{10.44}$$

Note that the uncompensated system is not stable at this equilibrium since the eigenvalues of the A matrix are 2.715 and -3.315. Recall that in Section 4.4.3, a state-feedback controller $u = Fx = [-34 \quad -7.4]x$ was designed based on the pole placement approach to place the closed-loop system poles at $-4 \pm j3$ so that the damping ratio and the natural frequency are $\varsigma = 0.8$ and $\omega_n = 5$ rad/s, respectively. A simulation that demonstrates the performance of this pole-placement state-feedback controller was shown in Figure 4.11.

In the following, we will employ the LQR approach to design a state-feedback controller to stabilize the system and minimize the performance index PI,

$$PI = \int_0^\infty \left(x^T(t)Qx(t) + u^T(t)Ru(t)\right)\, dt$$

where Q and R are positive semi-definite and positive definite, respectively. As long as (A, B) is stabilizable and the Q and R requirements are satisfied, any state-feedback controller designed based on the LQR approach theorem, Theorem 10.20, will stabilize the system. However, to achieve a desired closed-loop system performance the weighting matrices Q and R need to be chosen carefully according to the performance requirement and the control-input constraint.

In the following examples, we will not only design an LQR state-feedback controller to stabilize the system at the originally unstable equilibrium, $x^* = [0 \quad 0]^T$. **We would like the controller to be able to bring the system from a perturbed state, say $x(0) = x_0 = [0.2618 \quad 0]^T$, back to the equilibrium $x^* = [0 \quad 0]^T$ as quickly and smoothly as possible within control-input constraint.** Note that $x_1 = \theta = 0.2618$ rad is 15°.

Since the controllability matrix

$$[B \;\; AB] = \begin{bmatrix} 0 & 1 \\ 1 & -0.6 \end{bmatrix}$$

is of full rank, the system is controllable and thus stabilizable. There exists a state-feedback control strategy, $u(t) = Fx(t)$, so that the closed-loop system poles or the eigenvalues of $A + BF$, are in the left complex plane as well as the control-input constraints are satisfied. **In the following three examples, we will consider three sets of Q and R weighting matrices, respectively. For each set, there exists a unique stabilizing optimal controller. By evaluating the performance of each controller, we may see how the choice of Q and R would affect the performance of the system.**

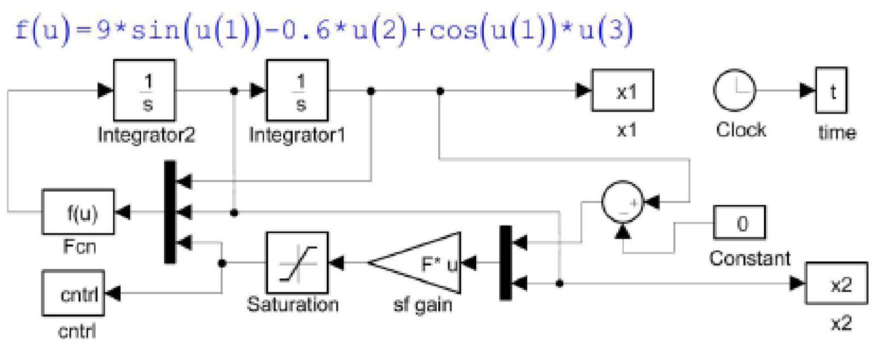


Fig. 10.19: Simulation diagram to evaluate LQR controller performance.

Although the controller is designed based on the linearized model, Equation 10.44, the nonlinear model Equation 10.43 is employed in the simulation, as shown in Figure 10.19. **The function block Fcn is defined by**

$$f(u) = 9 * sin(u(1)) - 0.6 * u(2) + cos(u(1)) * u(3)$$

where $u(1)$, $u(2)$, and $u(3)$, respectively, represent the state variables x_1, x_2, and the control-input u of Equation 10.43. The initial condition of the state variables are assigned as `x20=0` and

`x10=0.2618`, respectively, inside the integrator blocks. This initial condition means that the pendulum stick has deviated from the equilibrium by 15 degrees to the right with zero angular velocity.

Example 10.23 (Case 1 LQR State Feedback Controller for the Simple Inverted Pendulum)

Consider the state equation, Equation 10.44, which is the linearized state-space model of the nonlinear simple inverted pendulum dynamics of Equation 10.43 at the unstable equilibrium $x^* = [0 \quad 0]^T$. Let the weighting matrices Q and R be

$$Q = \begin{bmatrix} q_1 & 0 \\ 0 & q_2 \end{bmatrix}, \quad R = r_1$$

then

$$x^T Qx = q_1 x_1^2 + q_2 x_2^2, \quad u^T Ru = r_1 u^2$$

and the performance index (also called cost function) will be

$$PI = \int_0^\infty \left(q_1 x_1^2 + q_2 x_2^2 + r_1 u^2\right) dt = q_1 \int_0^\infty x_1^2\, dt + q_2 \int_0^\infty x_2^2\, dt + r_1 \int_0^\infty u^2\, dt$$

The three terms on the right-hand side of the equation are the weighted total energies of the angular displacement ($x_1 = \theta$), the angular velocity ($x_2 = \dot{\theta}$), and the control-input force ($u = f_a$), respectively. For Q to be positive semi-definite and R positive definite, q_1 and q_2 have to be greater or equal to zero and r_1 is required to be greater than zero. **The control-input weight r_1 cannot be zero since $r_1 = 0$ means that infinity feedback is allowed, which of course is not practically possible.** Making r_1 larger will put more constraint on the control-input energy consumption. **Similarly, the weight q_1 may need to be made larger if the reduction of the energy of x_1 would improve the performance.**

We will begin with the selection of the weights $q_1 = q_2 = 1$ and $r_1 = 1$, find the unique stabilizing controller that minimizes the performance index associated with this particular weight selection, evaluate the performance of the closed-loop system, and then revise the weight selection according to the performance evaluation.

With this Q, R selection,

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = 1$$

the unique stabilizing solution for the algebraic Riccati equation can be found as

$$X = \begin{bmatrix} 60.831 & 18.055 \\ 18.055 & 5.5213 \end{bmatrix}$$

The state-feedback gain is thus

$$F = -R^{-1}B^T X = \begin{bmatrix} -18.055 & -5.5213 \end{bmatrix}$$

and the closed-loop system poles are at -2.5018 and -3.6195. Since the initial state is $x(0) = x_0 = [0.2618 \quad 0]^T$, **the minimal performance index is**

$$PI_{opt} = x_0^T X x_0 = 4.1693$$

Based on Equation 10.43 and the simulation diagram in Figure 10.19, a Simulink program is assembled to conduct simulations to observe the time-domain responses due to the initial conditions of x_1 and x_2. The time-domain responses of the two state variables $x_1(t) = \theta(t)$ and $x_2(t) = \dot{\theta}(t)$ due to the initial conditions $x_1(0) = 0.2618$ rad and $x_2(0) = 0$ rad/s, are shown on the left-hand side of Figure 10.20. Meanwhile, the control-input action is recorded on the right graph of the figure. The pendulum initially is tilted to the right by 15° (0.2618 rad). The deviation of the pendulum position from the equilibrium $x^* = [0 \quad 0]^T$ prompted the control-input $u(t)$ to react immediately, changing **from 0 N to −4.7 N** and then gradually reducing to zero so that the angular velocity x_2 and the angular displacement x_1 can change accordingly to bring the pendulum back to the equilibrium. **It takes about 2.8 seconds to bring the pendulum back to the equilibrium.** ■

Comparing the simulation results of Case 1 LQR state-feedback controller with those of the pole placement state-feedback controller shown in Figure 4.11, we observe that the latter only **takes about 1.7 seconds to converge to the equilibrium while using more control-input, −8.8N.** In the next example, we will reduce the control-input weight r_1 to allow using more control-input energy and then observe if this change will improve the performance.

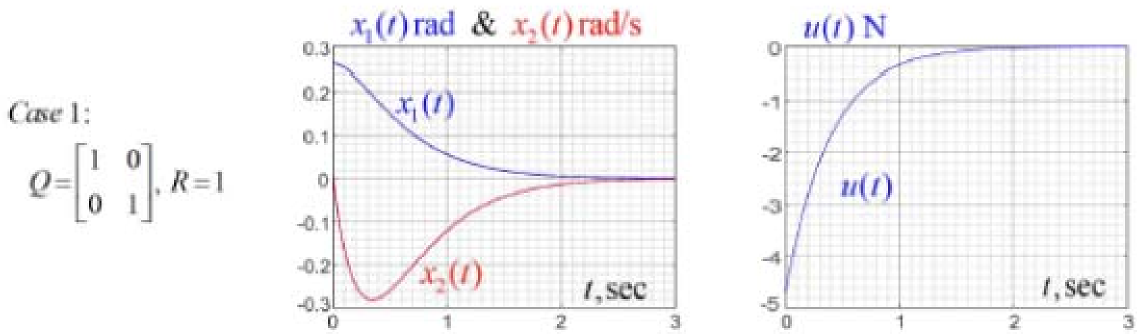


Fig. 10.20: Simulation results of the Case 1 LQR controller performance.

Before getting into the next example, it is interesting to see if multiplying Q and R by the same constant would change the outcome of the LQR state-feedback controller design. Let the Q and R in Case 1 design be multiplied by 10 to become

$$Q = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}, \quad R = 10$$

and repeat the LQR controller design in Example 10.23. We will see the stabilizing solution X to the algebraic Riccati equation becomes 10 times of the solution obtained in Example 10.23, and thus, the minimal performance index PI_{opt} will increase 10 times accordingly to

$$J_{opt} = x_0^T X x_0 = 41.693$$

However, the stabilizing state-feedback gain matrix F remains the same as

$$F = -R^{-1}B^T X = [-18.055 \; -5.5213]$$

since the same variation in R and X cancels each other. Therefore, **only the relative weights are important, and the value of the minimal performance index PI_{opt} does not reflect the performance** of the closed-loop system.