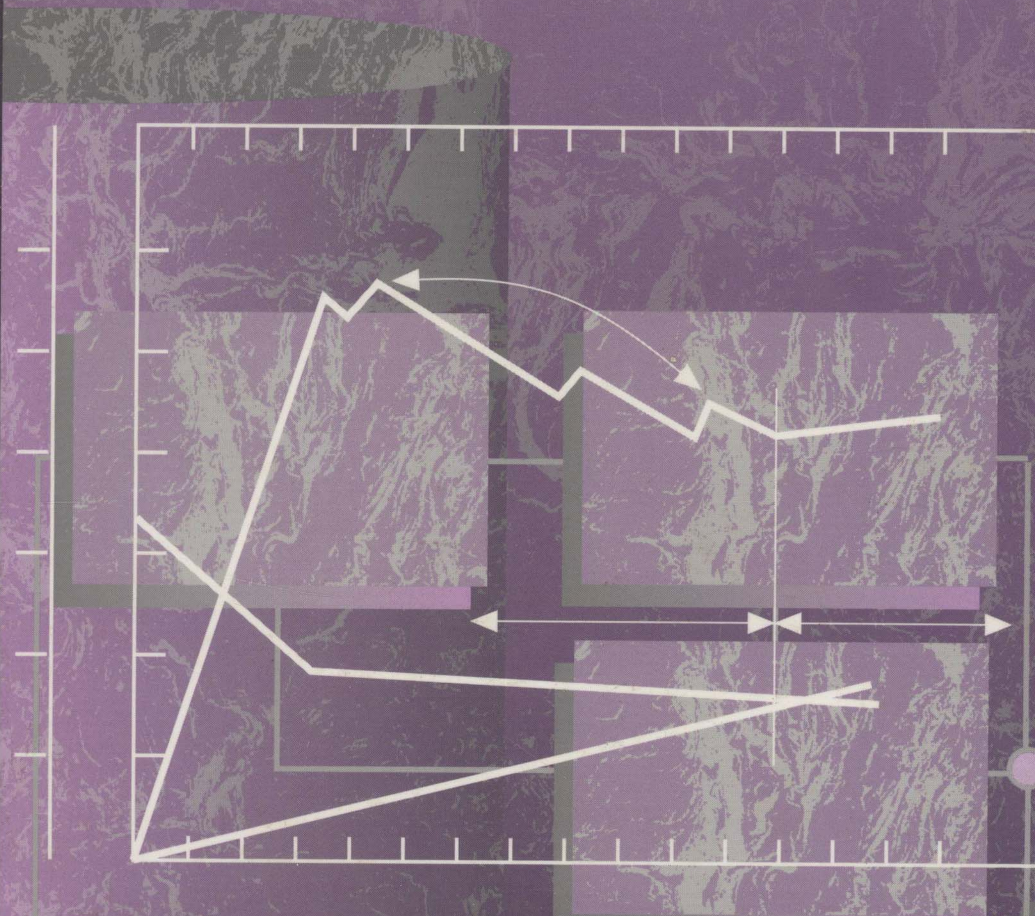
PROCESS CONTROL AND IDENTIFICATION

W. FRED RAMIREZ



PROCESS CONTROL AND IDENTIFICATION

W. Fred Ramirez

University of Colorado at Boulder
Department of Chemical Engineering
Boulder, Colorado

ACADEMIC PRESS, INC.
Harcourt Brace & Company, Publishers
Boston San Diego New York
London Sydney Tokyo Toronto

ACADEMIC PRESS, INC.
1250 Sixth Avenue, San Diego, CA 92101-4311

United Kingdom Edition published by
ACADEMIC PRESS LIMITED
24-28 Oval Road, London NWI 7DX

Library of Congress Cataloging-in-Publication Data

Ramirez, W. Fred, 1941-
Process control and identification / W. Fred Ramirez.
p. cm.
Includes bibliographical references and index.
ISBN 0-12-577240-8 (alk. paper)
1. Process control--Mathematical models. I. Title.
TS156.8.R36 1993
629.8'312--dc20 93-29779

Printed in the United States of America

93 94 95 96 CU 9 8 7 6 5 4 3 2 1

Errata

Several errors were noted in this edition. The following are corrections for these errors:

pp. 145-146: The order in which Figure 5.6 is shown is incorrect. The correct order is shown on the revised pages.

p. 156: There should be an additional line in the figure at the bottom of this page which reads "end."

pp. 165-167: The order in which Figure 5.17 is shown is incorrect. The correct order is shown on the revised pages.

p. 196: The figure caption for Figure 6.9 should read "*et al* ."

p. 221: The first 22 lines on the page (Figure 6.23) should not be there. The correct p. 221 is shown on the new pages.

p. 225: Figure 6.26 is missing a line. See the corrected page for where it should be.

p. 324: Both Figure 10.9 and Figure 10.10 are missing a line. See the corrected page for where they should be.

p. 325: There is an extra "C_{KI}" over the "20 minutes of time" in Figure 10.11. See corrected figure on new pages.

```
>> help lqr

 LQR    Linear quadratic regulator design for continuous systems.
        [K,S,E] = LQR(A,B,Q,R)  calculates the optimal feedback gain
        matrix K such that the feedback law  u = -Kx  minimizes the cost
        function:
                J = Integral {x'Qx + u'Ru} dt

        subject to the constraint equation:
                .
                x = Ax + Bu

        Also returned is S, the steady-state solution to the associated
        algebraic Riccati equation and the closed loop eigenvalues E:
                                  -1
                0 = SA + A'S - SBR  B'S + Q      E = EIG(A-B*K)

        [K,S,E] = LQR(A,B,Q,R,N) includes the cross-term N that relates
        u to x in the cost function.

                J = Integral {x'Qx + u'Ru + 2*x'Nu}

        The controller can be formed with REG.

        See also: LQRY, LQR2, and REG.

>> %input the a and b matrices
>> a =  [-0.05156  -0.05877;
          0.01503  -0.005887];
>> b = [0  -0.03384]';
>> help eye

EYE     Identity matrix. EYE(N) is the N-by-N identity matrix.
        EYE(M,N) is an M-by-N matrix with 1's on the diagonal and
        zeros elsewhere. EYE(A) is the same size as A.

>> % imput the q and r maatrices
>>
>> q=eye(2)

q =

     1     0
     0     1
>> r=1;
>> % compute the Kalman gain and Riccati  maatrix
>>
>>  [k,p] = lqr(a,b,q,r)

k =

    0.0505   -0.9241


p =

    9.2375   -1.4929
   -1.4929   27.3078

>> % use IMPLUSE to get the response of the linear system
>>
>> help impulse

 IMPULSE Impulse response of continuous-time linear systems.
```

Figure 5.6 MATLAB program for LQR of Nyquist–Ramirez reactor.

```
         IMPULSE(A,B,C,D,IU)  plots the time response of the linear system
                   .
                   x = Ax + Bu
                   y = Cx + Du
        to an impulse applied to the single input IU.  The time vector is
        automatically determined.

        IMPULSE(NUM,DEN) plots the impulse response of the polynomial
        transfer function  G(s) = NUM(s)/DEN(s)  where NUM and DEN contain
        the polynomial coefficients in descending powers of s.

        IMPULSE(A,B,C,D,IU,T) or IMPULSE(NUM,DEN,T) uses the user-supplied
        time vector T which must be regularly spaced.  When invoked with
        left hand arguments,
                [Y,X,T] = IMPULSE(A,B,C,D,...)
                [Y,X,T] = IMPULSE(NUM,DEN,...)
        returns the output and state time history in the matrices Y and X.
        No plot is drawn on the screen.  Y has as many columns as there
        are outputs and length(T) rows.  X has as many columns as there
        are states.

        See also: STEP,INITIAL,LSIM and DIMPULSE.

>> % redefine some matrices to put in proper form so that the immpulse is
>> % applied to both state variables with magnitude unity
>> aa = a-b*k

aa =

   -0.0516   -0.0588
    0.0167   -0.0372

>>
>> bb = [1 1]';
>> c = eye(2)

c =

     1     0
     0     1

>> d =  [0  0]'
>> t = [0:1:100];
>> y = impulse (aa,bb,c,d,1,t);
>> u = -k*y';
>> uu=u';
>> plot (t,y,t,uu)
>> title ('Plot of optimal state and control vs time')
>> xlabel ('Time, min')
>> ylabel ('State vector components and control variable')
>> meta plotl
>>% readjusting the R weighting
>> r=.1;
>> [k,p]=lqr(a,b,q,r)

k =

    0.7054   -3.3588


p =

    8.6073   -2.0845
   -2.0845    9.9255

>> aa
```

Figure 5.6 continued.

```
aa =

   -0.0516   -0.0588
    0.0167   -0.0372

>> aa=a-b*k

aa =

   -0.0516   -0.0588
    0.0389   -0.1195

>> y=impulse (aa,bb,c,d,1,t);
>> u=-k*y';
>> plot (t,y,t,u')
>> meta plot2
>> title('Plot of optimal state and contro vs time')
>> ylabel('State vector components and control variable')
>> xlabel('Time, min')
>> meta plot2
>>% redifining the Q and R weighting matrices
>> q(1,1)=10;
>> r=1;
>> [k,p]=lqr (a,b,q,r)

k =

    1.0172   -1.9623


p =

   78.1773  -30.0600
  -30.0600   57.9868

>> aa=a-b*k

aa =

   -0.0516   -0.0588
    0.0495   -0.0723

>> y=impulse (aa,bb,c,d,1,t);
>> u = -k*y';
>> plot (t,y,t,u')
>> title('Plot of optimal state and control vs time')
>> ylabel('State vector components and control variable')
>> xlabel('Time,min')
>> meta plot3
>> quit

 17423 flop(s).
```

Figure 5.6 continued.

```
% input the a and b matrices
a = [-0.05156  -0.05877;
      0.01503  -0.005887];
b = [0  -0.03384]';
% input the w matrix
w = [0.0125  0;
     0        0.0125];

% define the Johnson transformation

aa= [a   eye(2);
     zeros(2) zeros(2)];
bb= [0;
     0;
     b];
% define the weighting matrices

q = [1  0;
     0  0];

r = 1;

% Johnson transformation weighting matrices

qq = [q      zeros(2);
      zeros(2) zeros(2)];

% calculate the feedback gain using recursive method of section 5.2

z = [aa      -bb*inv(r)*bb';
     -qq     -aa'          ];

% chose the integration time interval dt and calculate transition matrix

dt = 1;

theta = expm(z);

% partition the transition matrix

theta11 = theta(1:4,1:4);
theta12 = theta(1:4,5:8);
theta21 = theta(5:8,1:4);
theta22 = theta(5:8,5:8);

% define the final Raccati matrix p(T)

pt = zeros(4);

% define t max

tmax = 70;
%loop for calcuating feedback gain matrix k

for t = tmax:-1:1,
        pt1 = inv(theta22 - pt*theta12)*(pt*theta11 - theta21);
        kt1 = inv(r)*bb'*pt1;
%       this loop is to transform the  matrix into a vector
        for col = 1:4
                k(t,col) = kt1(1,col);
        end
        pt=pt1;
end
```

Figure 5.11 MATLAB program for the Johnson algorithm control of the Nyquist–Ramirez reactor.

```
>> % input the a, b, and w matrices
>> a = [-0.033181  -0.002511;
        0.05237    0.010216];
>> b = [0   0.0988]';
>> w = [0.030864  0;
       0          0.030864];
>>
>> % check the eigenvalues of a
>> eig(a)

ans =

   -0.0299
    0.0069

>> % the positive eigenvalue shows that the system is unstable
>>
>> % let's look at the response of the uncontrolled system
>>
>> % use a unit impulse on each state
>> % define appropriate matrices
>>
>> bb = [1   1]';
>> c = eye(2);
>> t = [0:1:100];
>> d = [0  0]';
>> y = impulse(a,bb,c,d,1,t);
>> x1 = y(1:101,1);
>> x2 = y(1:101,2);
>> plot (t,y)
>> title('Uncontrolled Response');
>> xlabel('Time');
>> ylabel('State Vector Components');
>> meta uncontroll
>>
>> % compute the Kalman feedback gain matrix for the system
>> % input the weighting matrices
>>
>> q = [1  0;
        0  1];
>> r = 1;
>> [k,p,e] = lqr(a,b,q,r)
k =

    0.4015    1.0985


p =

   19.0538    4.0642
    4.0642   11.1187


e =

   -0.0978
   -0.0337

>> % these results show that a stable optimal proportional feedback system
>> % has been designed
>>
>> % now let's display the results of the controlled system to an unit
>> % impulse disturbance to each state variable
>>
>> % define the closed loop response matrix
```

Figure 5.17 MATLAB solution to regulation of unstable reactor.

```
>>
>> aa = a - b*k;
>>
>> y = impulse(aa,bb,c,d,1,t);
>> u = -k*y';
>> plot (t,y,t,u');
>> title('Optimal State and Control Variables');
>> xlabel ('Time');
>> ylabel ('State Vector Components and Control Variables');
>> meta optimal2_
>>
>> % implement the Johnson algorithm
>>
>> aa = [a            eye(2);
         zeros(2)     zeros(2)];
>> bb = [0;
         0;
         b];
>> q = [1    0;
        0    0];
>> qq = [q            zeros(2);
         zeros(2)    zeros(2)];
>>
>> % integrate the Riccati Equation backwards in time
>>
>> t0 = 0; tf = 200;
>> p0 =[0  0  0  0  0  0  0  0  0  0]';
>> [t,pv] = ode45('xprime',t0,tf,p0);
>> plot (t,pv)
>> title ('Riccati Matrix vs tf - t');
>> xlabel('tf - t'); ylabel('Riccati Gain Components');
>> meta riccati1
>>
>> % preparing for the response of the system to unit step disturbances
>>
>> % steady state values of the Riccati matrix elements
>>
>> pv(65,1:10)

ans =

   1.0e+03 *

  Columns 1 through 7

    0.0128   -0.0005    0.1786   -0.0032    0.0000   -0.0174    0.0006

  Columns 8 through 10

    6.1447   -0.1913    0.0110
>> p(1,1) = pv(65,1);
>> p(1,2) = pv(65,2);
>> p(1,3) = pv(65,3);
>> p(1,4) = pv(65,4);
>> p(2,1) = p(1,2);
>> p(2,2) = pv(65,5);
>> p(2,3) = pv(65,6);
>> p(2,4) = pv(65,7);
>> p(3,1) = p(1,3);
>> p(3,2) = p(2,3);
>> p(3,3) = pv(65,8);
>> p(3,4) = pv(65,9);
```

Figure 5.17 continued.

```
>> p(4,1) = p(1,4);
>> p(4,2) = p(2,4);
>> p(4,3) = p(3,4);
>> p(4,4) = pv(65,10);
>> k = inv(r)*bb'*p;
>> kp = k(1,3:4)

kp =

  -18.8991    1.0890

>> ki = k(1,1:2) -k(1,3:4)*a

ki =

   -1.0000   -0.0000

>> api = [a-b*kp  -b*ki;
          eye(2)   zeros(2)];
>> wi = [w        zeros(2);
         zeros(2) zeros(2)];
>> c = eye(4);
>> d = zeros(4);
>> t = [0:1:100];
>> xpi = step(api,wi,c,d,1,t);
>> x = xpi(1:101,1:2);
>> plot (t,x)
>> title('Johnson Algorithm State Response');
>> ylabel('Normalized Composition and Temperature');
>> xlabel('Time (min)');
>> meta xpi3;
```

Figure 5.17 continued

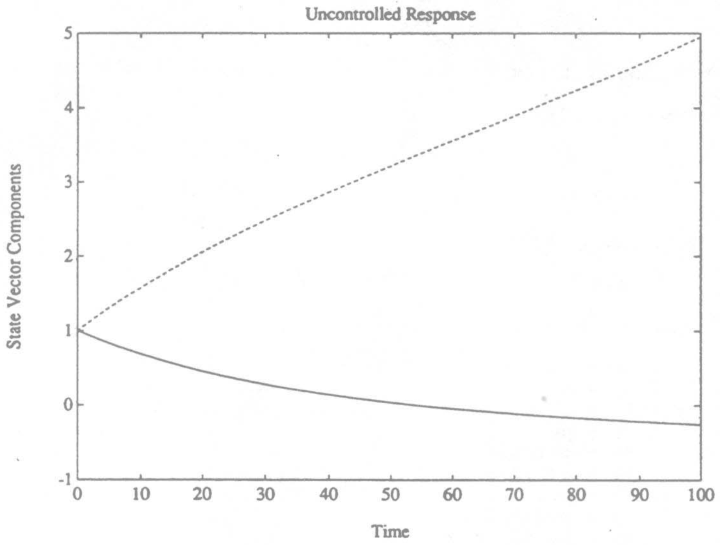


Figure 5.18 Uncontrolled system response. Temperature (- - - -), composition (—).

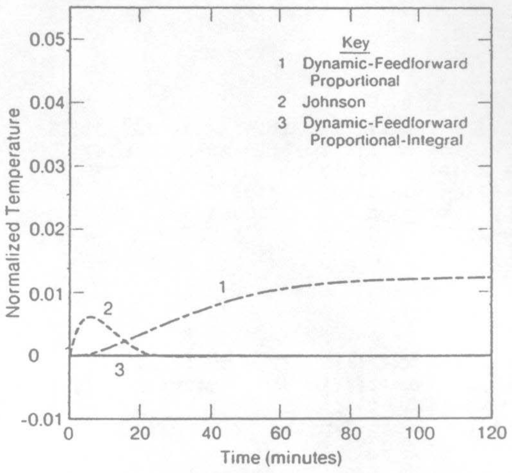


Figure 6.9 Outlet temperature regulated with only inlet temperature measurable. From Ramirez *et al.* (1987).

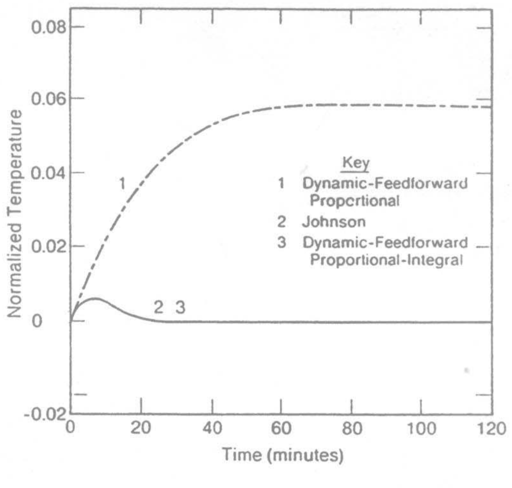


Figure 6.10 Outlet temperature regulated with only inlet concentration measurable. From Ramirez *et al.* (1987).

```
>> y4(12:27,1) = xx41(1:16,1) - xx41(1:16,2);
>> y4(12:27,2) = xx41(1:16,3) - xx41(1:16,4);
>> xx420 = xx41(16,1:16);
>> [t42,xx42] = ode45('xprime',3,50,xx420);
Singularity likely at t = 7.000081
>> size(xx42)

ans =

    18    16

>> y4(28:45,1) = xx42(1:18,1) - xx42(1:18,2);
>> y4(28:45,2) = xx42(1:18,3) - xx42(1:18,4);
>> xx430 = xx42(18,1:16);
>> [t43,xx43] = ode45('xprime',7,50,xx430);
>> size(xx43)

ans =

    36    16

>> y4(46:81,1) = xx43(1:36,1) - xx43(1:36,2);
>> y4(46:81,2) = xx43(1:36,3) - xx43(1:36,4);
>> tp4 = [t3;
          t41;
          t42;
          t43];
>> plot(tt,y3,tp4,y4)
>> plot(tt,y3)
>> meta y1
>> plot(tt,y3,tp4,y4)
>> meta y2
>> diary off
```

Figure 6.23 continued.

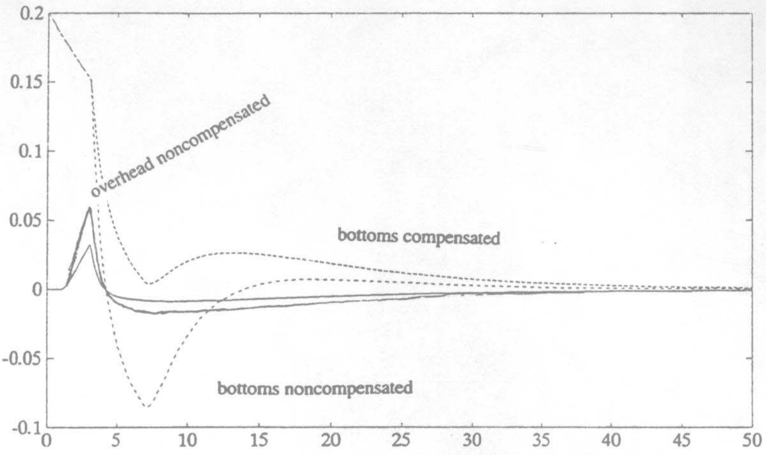


Figure 6.26 Output response to bottoms set point change of $-.2$ mole fraction.

the general time delayed system given by

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}_0\boldsymbol{x}(t) + \sum_{i=1}^{\delta} \boldsymbol{A}_i\boldsymbol{x}(t-\alpha_i) + \boldsymbol{B}_0\boldsymbol{u}(t) + \sum_{j=1}^{\mu} \boldsymbol{B}_j\boldsymbol{u}(t-\beta_j) \tag{6.8.24}$$

For the quadratic performance functional of

$$J = \frac{1}{2}\int_0^\infty \left(\boldsymbol{x}^T(t)\boldsymbol{Q}\boldsymbol{x}(t) + \boldsymbol{u}^T(t)\boldsymbol{R}\boldsymbol{u}(t)\right) dt \tag{6.8.25}$$

the optimal controller takes the form

$$\begin{aligned} \boldsymbol{u} &= -\boldsymbol{R}^{-1}\left[\left(\boldsymbol{B}_0^T\boldsymbol{P}_0 + \boldsymbol{P}_3(0)\right)\boldsymbol{x} + \int_{-\alpha_\delta}^{0}\left(\boldsymbol{B}_0^T\boldsymbol{P}_1(s) + \boldsymbol{P}_5^T(s,0)\right)\boldsymbol{x}(t+s)ds \right.\\ &\quad \left. + \int_{-\beta_\mu}^{0}\left(\boldsymbol{B}_0^T\boldsymbol{P}_3(s) + \boldsymbol{P}_4(0,s)\right)\boldsymbol{u}(t+s)ds\right] \end{aligned} \tag{6.8.26}$$

and the Riccati matrices $\boldsymbol{P}_i$ are given by

$$\begin{aligned} \boldsymbol{0} &= -\boldsymbol{P}_0\boldsymbol{A}_0 - \boldsymbol{A}_0^T\boldsymbol{P}_0 - \boldsymbol{P}_1^T(0) - \boldsymbol{P}_1(0) \\ &\quad + \left(\boldsymbol{P}_0\boldsymbol{B}_0 + \boldsymbol{P}_3(0)\right)\boldsymbol{R}^{-1}\left(\boldsymbol{B}_0^T\boldsymbol{P}_0 + \boldsymbol{P}_3^T(0)\right) - \boldsymbol{Q} \end{aligned} \tag{6.8.27}$$

$$\begin{aligned} \boldsymbol{0} &= \frac{d\boldsymbol{P}_1}{ds} - \boldsymbol{A}_0^T\boldsymbol{P}_1(s) - \boldsymbol{P}_2(0,s) \\ &\quad + \left(\boldsymbol{P}_0\boldsymbol{B}_0 + \boldsymbol{P}_3(0)\right)\boldsymbol{R}^{-1}\left(\boldsymbol{B}_0^T\boldsymbol{P}_1(s) + \boldsymbol{P}_5^T(s,0)\right) \end{aligned} \tag{6.8.28}$$

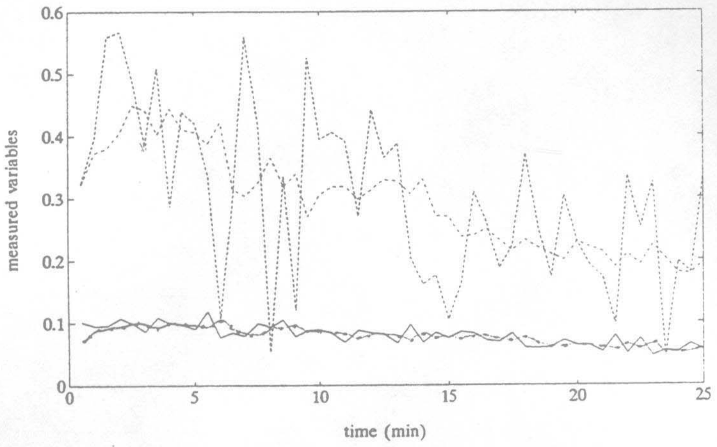


Figure 10.9 Comparison of actual and filtered measurements.

T_m	=	meas. temperature state (—)
R_m	=	meas. normalized reaction rate (- - - -)
$\hat{T}$	=	filtered temperature state (-·-)
$\hat{R}$	=	filtered normalized reaction rate (— · —·)

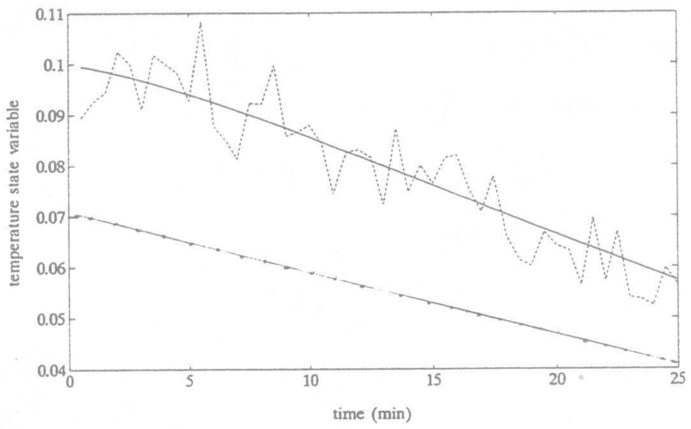


Figure 10.10 Comparison of actual, filtered, and model temperature profiles.

T	=	actual temperature state (—)
$\hat{T}$	=	filtered temperature state (- - - -)
$\overline{T}$	=	model temperature state (-·-)

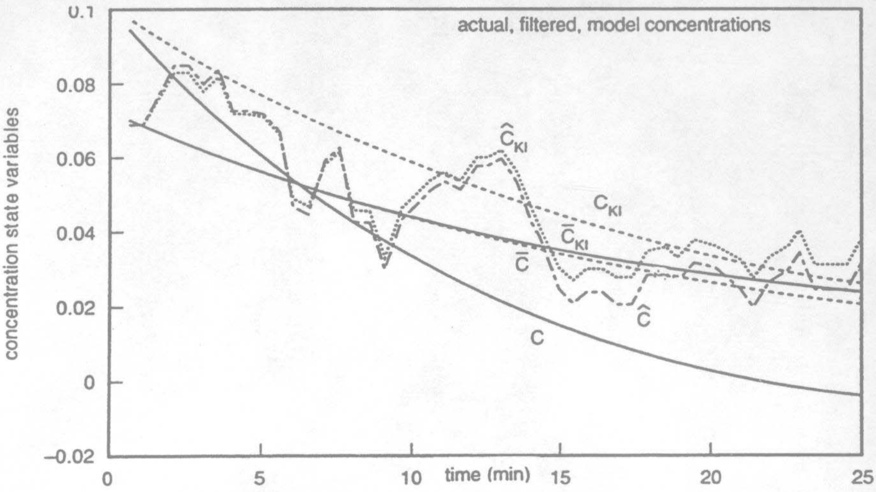


Figure 10.11 Comparison of actual, filtered, and model concentrations.

C	=	actual reactant conc. state	C_{KI}	=	actual catalyst conc. state
$\hat{C}$	=	filtered reactant conc. state	$\hat{C}$	=	filtered catalyst conc. state
$\overline{C}$	=	model reactant conc. state	$\overline{C}$	=	model catalyst conc. state

very weak. Finally there is more discrepancy in the reactant concentration due to the fact that the linearized model did not well represent that state's dynamic response (Figure 10.6). The tracking ability for the reactant concentration can be improved by performing the step by step updating of Equations (10.6.45) to (10.6.47).

10.8 Estimation of Model Parameters

Kalman filtering can be used effectively for the sequential estimation of unknown or uncertain model parameters. This is accomplished by introducing new state variables corresponding to the model parameters to be identified. If the parameters are expected to be constants or slowly varying over the process time domain, then the following state dynamic model is appropriate:

$$\dot{a} = 0 + w_1 \tag{10.8.1}$$

This model simply states that we expect the parameters to be constant and that they have an uncertainty that is characterized by the random variable w_1. The random variable vector w_1 has an expected value of zero and a covariance

$$\text{cov}\,(w_1) = Q_1 \tag{10.8.2}$$

PROCESS CONTROL AND IDENTIFICATION