Chapter 13
Analysis and Design of Parameter-Dependent Systems

The structured uncertainty models presented in Chap. 9 are not the unique way to represent systems with uncertain parameters. These models are usually appropriate to describe systems whose uncertainties are relatively small in size and may not be convenient to implement in the case of large parameter variations. In such cases it might be better to use models in the form of the so called *parameter-dependent systems* which are appropriate for use in the more general case of large uncertainties. The linear parameter-dependent systems, called also *Linear Parameter-Varying Systems* (LPV systems), arise in many applications, for instance in robotics and aerospace systems. In this chapter we show how to build models of parameter-dependent systems and how to use these models in robust stability analysis and controller design. Especially important is the so called *gain scheduling design* which allows to obtain satisfactory closed-loop performance in case of widely varying plant parameters.

The presentation in this chapter follows closely the corresponding chapters of [50].

13.1 Representation of Parameter-Dependent Systems

13.1.1 SYSTEM Matrix

In many cases the linear system models are obtained in the general form of the so called *descriptor systems* which are described by state-space equations of the form

$$E\frac{\mathrm{d}x}{\mathrm{d}t} = Ax(t) + Bu(t) \tag{13.1}$$

$$y(t) = Cx(t) + Du(t) \tag{13.2}$$

In these equations the matrix E is a square matrix that may be singular in the general case. If this matrix is invertible and well conditioned, (13.1), (13.2) are easily converted to the usual state-space description multiplying on the left both sides

DOI 10.1007/978-1-4471-4682-7_13,© Springer-Verlag London 2013

of (13.1) by the matrix E^{-1}. However, if the matrix E is singular or poorly conditioned this conversion is not possible and one has to use the descriptor system model (13.1), (13.2). The descriptor system models arise naturally in the description of many physical systems and are especially appropriate to represent systems with large parameter variations.

In the discrete-time case the descriptor systems are described by the state-space equations

$$Ex_{k+1} = Ax_k + Bu_k$$
$$y_k = Cx_k + Du_k$$

Consider as an example the mass–damper–spring system introduced in Sect. 9.2 and depicted in Fig. 9.11. The dynamics of this system is described by the 2nd-order differential equation

$$m\ddot{x}_s + c\dot{x}_s + kx_s = u \tag{13.3}$$

where x_s is the displacement of the mass block from the equilibrium position and u is the force acting on the mass, with m the mass, c the damper constant and k the spring constant. Denoting $x_1 = x_s$, $x_2 = \mathrm{d}x_s/\mathrm{d}t$, the system (13.3) may be described by the two first order differential equations

$$\dot{x}_1 = x_2 \tag{13.4}$$
$$m\dot{x}_2 = -kx_1 - cx_2 + u \tag{13.5}$$

Taking the output as $y = x_s$, (13.4), (13.5) are represented as a descriptor system in the form (13.1), (13.2) with matrices

$$E = \begin{bmatrix} 1 & 0 \\ 0 & m \end{bmatrix}, \qquad A = \begin{bmatrix} 0 & 1 \\ -k & -c \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \qquad C = \begin{bmatrix} 1 & 0 \end{bmatrix}, \qquad D = 0$$

In MATLAB®, a continuous-time or discrete-time descriptor system model is stored for convenience as a single matrix called `SYSTEM` matrix. This matrix has the form

$$\left[\begin{array}{cc|c} A + j(E - I) & B & n \\ & & 0 \\ C & D & \vdots \\ & & 0 \\ \hline 0 & 0 & -\mathrm{Inf} \end{array}\right]$$

where $j = \sqrt{-1}$, n is equal to the number of states of (13.1) and the entry Inf is used to distinguish `SYSTEM` matrices from the regular matrices.

The `SYSTEM` matrices are created by the function `ltisys`. For given matrices A, B, C, D, E of the descriptor system formulation (13.1), (13.2), the command line

```
sys = ltisys(A,B,C,D,E)
```

produces the SYSTEM matrix sys. When omitted, D and E are set to the default values $D = 0$ and $E = I$. The autonomous system $E\dot{x} = Ax$ is thus specified by the line

```
sys = ltisys(A,E)
```

Given a SYSTEM matrix sys created with ltisys, the function ltiss allows to extract the corresponding state-space realization. This is done by the command line

```
(A,B,C,D,E) = ltiss(sys)
```

It is important to note that when the output argument E is not referenced, ltiss returns the state-space realization $(E^{-1}A, E^{-1}B, C, D)$ provided the matrix E is nonsingular.

The number of states, inputs, and outputs of a linear system are extracted from the SYSTEM matrix sys with the function sinfo,

```
sinfo(sys)
```

The time and frequency response plots of descriptor systems represented by SYSTEM matrices may be obtained by the function splot.

Series and parallel interconnections of SYSTEM matrices are performed by the functions sadd and smult and feedback interconnection can be done by the function sloop.

13.1.2 Affine Parameter-Dependent Models

Many physical systems may be described by linearized equations in descriptor form

$$E(p)\dot{x} = A(p)x + B(p)u \tag{13.6}$$

$$y = C(p)x + D(p)u \tag{13.7}$$

where $A(.), \dots, E(.)$ are known functions of some parameter vector $p = (p_1, \dots, p_s)$. The parameters $p_1, \dots, p_s$ may vary slowly with the time within some prescribed boundaries. In the case of *affine parameter-dependent systems* the matrices of the state-space description are affine functions of the parameters, which means that these matrices may be represented as

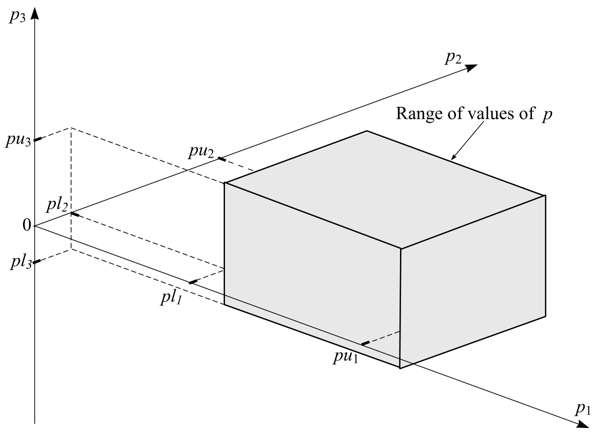
$$A(p) = A_0 + p_1A_1 + \cdots + p_sA_s, \qquad B(p) = B_0 + p_1B_1 + \cdots + p_sB_s$$

$$C(p) = C_0 + p_1C_1 + \cdots + p_sC_s, \qquad D(p) = D_0 + p_1D_1 + \cdots + p_sD_s$$

$$E(p) = E_0 + p_1E_1 + \cdots + p_sE_s$$

where $A_0, \dots, A_s$, $B_0, \dots, B_s$ and so on are constant matrices that do not depend on the parameters $p_1, \dots, p_s$. Note that for a particular system some of these matrices may be zero matrices.

Fig. 13.1 Parameter box



At first glance, the affine parameter-dependent systems constitute a very small set of systems with varying parameters. However, many real world systems may be approximated with sufficient accuracy as affine parameter-dependent systems. The advantage of such a system model is the possibility to derive easily several results concerning closed-loop stability and controller design.

Using the notation

$$S(p) = \begin{bmatrix} A(p) + jE(p) & B(p) \\ C(p) & D(p) \end{bmatrix}, \qquad S_i = \begin{bmatrix} A_i + jE_i & B_i \\ C_i & D_i \end{bmatrix}, \quad i = 0, \dots, s$$

the affine parameter-dependent system (13.6), (13.7) is written compactly in `SYSTEM` matrix terms as

$$S(p) = S_0 + p_1 S_1 + \cdots + p_s S_s \tag{13.8}$$

The constant matrices $S_0, \dots, S_s$ may be considered as system coefficients and may have not physical meaning by themselves.

The parameter uncertainty range can be described as a box in the parameter space. This corresponds to cases where each uncertain or slowly varying parameter p_i ranges between two known bounds pl_i and pu_i,

$$p_i \in [pl_i, pu_i], \quad i = 1, \dots, s \tag{13.9}$$

If $p = (p_1, \dots, p_s)$ is the vector of all uncertain parameters, (13.9) constitutes a hyperrectangle in the parameter space called the *parameter box*. The parameter box is set by the function `pvec`. For instance, in case of three parameters ($s = 3$), the parameter box is specified by the command lines

```
range = [pl_1 pu_1,pl_2 pu_2,pl_3 pu_3]
p = pvec('box',range)
```

The parameter box for the case of three parameters is shown in Fig. 13.1.

The function pvec allows also to specify bounds on the rates of variation $\dot{p}_i$ of the parameter vector components $p_i(t)$. For instance, if the rates of the three parameters $p_1(t), p_2(t), p_3(t)$ vary in the ranges $[vl_1\ vu_1]$, $[vl_2\ vu_2]$, $[vl_3\ vu_3]$, respectively, these rates may be specified by adding a third argument rate to the calling list of pvec as shown below

```
rate = [vl_1 vu_1, vl_2 vu_2, vl_3 vu_3]
p = pvec('box',range,rate)
```

When the argument rate is omitted, all parameters are assumed to be time-invariant.

Provided the coefficient matrices $S_0, \ldots, S_s$ are known and the domains where the parameters and their rates vary are specified, the affine parameter-dependent model (13.8) may be obtained by the function psys. For instance, the system

$$S(p) = S_0 + p_1 S_1 + p_2 S_2 + p_3 S_3$$

depending on three parameters, is defined by

```
S0 = ltisys(A0,B0,C0,D0,E0)
S1 = ltisys(A1,B1,C1,D1,E1)
S2 = ltisys(A2,B2,C2,D2,E2)
S3 = ltisys(A3,B3,C3,D3,E3)
pds = psys(pv, [S0 S1 S2 S3])
```

where pv is the parameter description returned by the function pvec.

It is necessary to stress that, by default, the function ltisys sets the $E(p)$ matrix to the identity matrix I. That is why if the arguments E0, E1, E2, E3 in the above lines are omitted then the matrix $E(p)$ will be set to $E(p) = I + (p_1 + p_2 + p_3)I$. To specify an affine parameter-dependent system with a parameter-independent E matrix (e.g., $E(p) = I$), it is necessary to explicitly set $E_1 = E_2 = E_3 = 0$ by entering

```
S0 = ltisys(A0,B0,C0,D0)
S1 = ltisys(A1,B1,C1,D1,0)
S2 = ltisys(A2,B2,C2,D2,0)
S3 = ltisys(A3,B3,C3,D3,0)
```

Example 13.1 Consider the mass–damper–spring system whose state-space equations (13.4), (13.5) were obtained earlier in this section in descriptor form. The three uncertain parameters m, k, c of the system vary in the intervals

$$m \in [1.8, 4.2], \qquad k \in [1.4, 2.6], \qquad c \in [0.8, 1.2]$$

The parameter-dependent matrices of the mass–damper–spring system may be represented as affine functions of the uncertain parameters m, k, c as

$$A(m,k,c) = \begin{bmatrix} 0 & 1 \\ -k & -c \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} + m\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} + k\begin{bmatrix} 0 & 0 \\ -1 & 0 \end{bmatrix} + c\begin{bmatrix} 0 & 0 \\ 0 & -1 \end{bmatrix}$$

$$E(m,k,c) = \begin{bmatrix} 1 & 0 \\ 0 & m \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} + m \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} + k \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} + c \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

The affine model of the mass–damper–spring system is specified by the following command lines:

```
A_0 = [0 1;0 0]; B_0 = [0 1]'; C_0 = [1 0]; D_0 = 0;
E_0 = [1 0;0 0];
S_0 = ltisys(A_0,B_0,C_0,D_0,E_0);
%
A_m = [0 0;0 0]; B_m = [0 0]'; C_m = [0 0]; D_m = 0;
E_m = [0 0;0 1];
S_m = ltisys(A_m,B_m,C_m,D_m,E_m);
%
A_k = [0 0;-1 0]; B_k = [0 0]'; C_k = [0 0]; D_k = 0;
E_k = [0 0;0 0];
S_k = ltisys(A_k,B_k,C_k,D_k,0);
%
A_c = [0 0;0 -1]; B_c = [0 0]'; C_c = [0 0]; D_c = 0;
E_c = [0 0;0 0];
S_c = ltisys(A_c,B_c,C_c,D_c,0);
%
pv = pvec('box',[1.8 4.2;1.4 2.6;0.8 1.2]);
pds_mds = psys(pv,[S_0 S_m S_k S_c]);
```

The first `SYSTEM` matrix `S_0` contains the state-space data for $m = k = c = 0$ while `S_m, S_k, S_c` define the coefficient matrices of m, k, c. The range of parameter values in the parameter box is specified by the `pvec` command. The results obtained can be checked with the commands `psinfo` and `pvinfo`:

```
psinfo(pds_mds)

Affine parameter-dependent model with 3 parameters
    (4 systems)
    Each system has 2 state(s), 1 input(s), and
    1 output(s)

pvinfo(pv_mds)

Vector of 3 parameters ranging in a box
```

The command line

```
nom_sys = psinfo(pds_mds,'eval',[3.0 2.0 1.0])
```

allows to evaluate the parameter-dependent model for the nominal values $m = 3.0, k = 2.0, c = 1.0$.

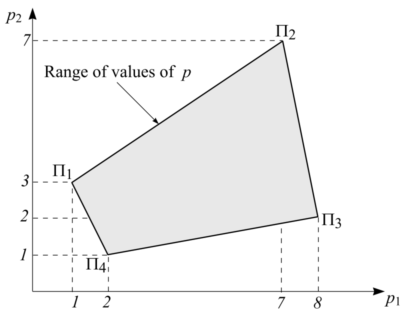


Fig. 13.2 Polytopic parameter range

Apart from the possibility to describe the parameter uncertainty range as a box, uncertain parameter vectors can be specified as ranging in a polytope of the parameter space. This possibility is illustrated in Fig. 13.2 for the simple case of $s = 2$. The polytope is characterized by the four vertices

$$\Pi_1 = (1,3), \qquad \Pi_2 = (7,7), \qquad \Pi_3 = (8,2), \qquad \Pi_4 = (2,1)$$

An uncertain parameter vector p with this range of values is defined by

```
pi1=[1,3], pi2=[7,7], pi3=[8,2], pi4=[2,1]
p = pvec('pol',[pi1,pi2,pi3,pi4])
```

The string `'pol'` indicates that the parameter range is defined as a polytope.

13.1.3 Polytopic Models

Polytopic system is a linear time-varying system

$$\begin{aligned} E(t)\dot{x} &= A(t)x + B(t)u \\ y &= C(t)x + D(t)u \end{aligned}$$

whose `SYSTEM` matrix

$$S(t) = \begin{bmatrix} A(t) + jE(t) & B(t) \\ C(t) & D(t) \end{bmatrix}$$

varies within a fixed polytope of matrices

$$P = \left\{ \sum_{i=1}^{q} \alpha_i S_i : \alpha_i \geq 0, \sum_{i=1}^{q} \alpha_i = 1 \right\}$$

In the above expression $S_1, S_2, \ldots, S_q$ are given *vertex systems*

$$S_i = \begin{bmatrix} A_i + jE_i & B_i \\ C_i & D_i \end{bmatrix}, \quad i = 1, \ldots, q$$

corresponding to the vertices of $S(t)$.

In this way the polytopic system $S(t)$ is a convex combination (linear combination with nonnegative coefficients) of the SYSTEM matrices $S_1, \dots, S_q$, i.e. $S(t) = \alpha_1 S_1 + \dots + \alpha_q S_q$. The nonnegative numbers $\alpha_1, \dots, \alpha_q$ are called the *polytopic coordinates* of $S(t)$.

Polytopic models are fully specified by the list of their vertex systems, i.e., by the SYSTEM matrices $S_1, \dots, S_q$ and may be created by the function `psys`. For instance, a polytopic model taking values in the convex combination of the three linear time-invariant systems `S1, S2, S3` is created by the line

```
polsys = psys([S1 S2 S3])
```

Affine parameter-dependent systems

$$S(p) = \begin{bmatrix} A(p) + jE(p) & B(p) \\ C(p) & D(p) \end{bmatrix}$$

may be converted easily to polytopic systems. If each parameter p_i varies in some interval $[pl_i, pu_i]$, $i = 1, \dots, s$, then the parameter vector $p = (p_1, \dots, p_s)$ takes values in a parameter box with $q = 2^s$ corners $\Pi_1, \Pi_2, \dots, \Pi_q$. According to the results of functional analysis, if the function $S(p)$ is affine in p, then it maps this parameter box to some polytope of SYSTEM matrices. Specifically, this polytope is the convex linear combination of the images $S_1 = S(\Pi_1), S_2 = S(\Pi_2), \dots, S_q = S(\Pi_q)$ of the parameter box corners $\Pi_1, \Pi_2, \dots, \Pi_q$. These images are the vertex systems of the polytopic model.

An affine parameter-dependent system is converted to an equivalent polytopic model using the function `aff2pol`. The syntax is of this function is

```
polsys = aff2pol(affsys)
```

where `affsys` is the affine model and `polsys` is the resulting polytopic model. The polytopic model consists of the instances of `affsys` at the vertices of the parameter box.

For illustration, if the function `aff2pol` is applied to the mass–damper–spring system, considered in Example 13.1, one obtains

```
pols_mds = aff2pol(pds_mds);
psinfo(pols_mds)

Polytopic model with 8 vertex systems
    Each system has 2 state(s), 1 input(s), and
    1 output(s)
```

In the given case the affine model depends on three parameters, which results in $q = 2^3 = 8$ vertex systems of the polytopic model.

13.2 Analysis of Parameter-Dependent Systems

In this section we consider the implementation of MATLAB® functions for stability analysis of parameter-dependent systems and evaluation of the transient response of such systems.

Stability analysis of parameter-dependent systems is done by the second method of Lyapunov. According to this method a linear time-varying system

$$E(t)\dot{x} = A(t)x \tag{13.10}$$

is asymptotically stable if there exists a positive definite quadratic Lyapunov function

$$V(x) = x^T P x$$

such that its derivative with respect to time is negative definite, i.e.

$$\frac{\mathrm{d}V(x(t))}{\mathrm{d}t} < 0$$

along all state trajectories. If it is possible to find such a Lyapunov function we say that the system (13.10) is *quadratically stable* .The existence of quadratic Lyapunov function guarantees stability for arbitrarily fast time variations of the system parameters which is a very stringent condition in practice.

The quadratic stability of affine or polytopic parameter-dependent system may be tested by the MATLAB® function `quadstab` which involves solution of several Linear Matrix Inequalities (LMI) in an attempt to find a suitable Lyapunov function $V(x)$. However, `quadstab` may produce pessimistic results due to the allowance of arbitrarily fast parameter variations. Less conservative results in case of constant or slowly varying parameters may be obtained by the function `pdlstab` which is considered next.

For a parameter-dependent system the function `pdlstab` seeks a parameter-dependent Lyapunov function to establish the stability of uncertain state-space models over some parameter range or polytope of systems. Specifically, for an affine parameter-dependent system

$$E(p)\dot{x} = A(p)x + B(p)u$$
$$y = C(p)x + D(p)u$$

with $p = (p_1, p_2, \ldots, p_s)$ `pdlstab` seeks a Lyapunov function of the form

$$V(x, p) = x^T Q^{-1} x, \quad Q(p) = Q_0 + p_1 Q_1 + \cdots + p_s Q_s$$

with symmetric matrices $Q_0, Q_1, \ldots, Q_s$ such that $\mathrm{d}V(x, p)/\mathrm{d}t < 0$ along all admissible trajectories.

For a time-invariant polytopic system

$$E\dot{x} = Ax + Bu$$

$$y = Cx + Du$$

with

$$\begin{bmatrix} A + jE & B \\ C & D \end{bmatrix} = \sum_{i=1}^{q} \alpha_i \begin{bmatrix} A_i + jE_i & B_i \\ C_i & D_i \end{bmatrix}, \quad \alpha_i \geq 0, \ \sum_{i=1}^{q} \alpha_i = 1 \tag{13.11}$$

`pdlstab` seeks a single Lyapunov function of the form

$$V(x,\alpha) = x^T Q(\alpha)^{-1} x, \quad Q(\alpha) = \alpha_1 Q_1 + \cdots + \alpha_q Q_q$$

such that $dV(x,\alpha)/dt < 0$ for all polytopic decompositions (13.11). Note that for this type of models one of the matrices A, E should be constant and the other uncertain.

Similarly to `quadstab`, the function `pdlstab` determines whether the sufficient LMI stability conditions are feasible.

The syntax of the function `pdlstab` is

```
[tau,Q0,Q1,...,Qs] = pdlstab(affsys,options)
```

for the case of affine parameter-dependent systems and

```
[tau,Q1,Q2,...,Qq] = pdlstab(polsys,options)
```

for the case of polytopic parameter-dependent systems.

Input arguments

– `affsys` is the affine system description specified with the function `psys`, which contains information about the range of values and rate of variation of each parameter p_i;
– `polsys` is the polytopic system description specified with the function `psys` or obtained by the function `aff2pol` from an affine model;
– `options`: optional parameter.
  - Setting `options(1) = 0` tests robust stability (default).
  - When `options(2) = 0`, `pdlstab` uses simplified sufficient conditions for faster execution. Set `options(2) = 1` to use the least conservative conditions.

Output arguments

– `tau`: a scalar parameter which shows whether the LMI stability conditions are feasible. If `tau` is negative then the system is stable;
– `Q0,Q1,...,Qs`: contain the coefficient matrices Q_0, $Q_1, \ldots, Q_s$ of the matrix $Q(p)$ associated with the Lyapunov function $V(x, p)$ in case of affine model;
– `Q1,Q2,...,Qq`: contain the coefficient matrices Q_1, $Q_2, \ldots, Q_q$ of the matrix $Q(\alpha)$ associated with the Lyapunov function $V(x,\alpha)$ in case of polytopic model.

Example 13.2 Consider the mass–damper–spring system whose affine parameter-dependent model `pds_mds` was created in Example 13.1. For this model the command line

```
[tmin,Q0,Q1,Q2,Q3] = pdlstab(pds_mds)
```

produces

```
 Solver for LMI feasibility problems L(x) < R(x)
    This solver minimizes t subject to L(x) < R(x) + t*I
    The best value of t should be negative
    for feasibility

 Iteration   :    Best value of t so far

     1                        -0.024610

 Result:  best value of t:    -0.024610
          f-radius saturation:  0.000% of R = 1.00e+007

 This system is stable for the specified parameter
                                        trajectories

tmin =

   -0.0246

Q0 =

    1.6750   -0.3994
   -0.3994    1.1642

Q1 =

    0.8456   -0.0451
   -0.0451   -0.0271

Q2 =

    0.0543   -0.0989
   -0.0989    0.9498

Q3 =

    0.6545   -0.7452
   -0.7452   -0.0235
```

The results shows that the mass–damper–spring system remains stable for all parameter values that belong to the specified parameter box. Note that although the matrices Q_1 and Q_3 are not positive definite, the matrix $Q(p) = Q_0 + mQ_1 + kQ_2 + cQ_3$ remains positive definite for the whole parameter range.

The time response of an affine parameter-dependent system

$$\begin{aligned} E(p)\dot{x} &= A(p)x + B(p)u \\ y &= C(p)x + D(p)u \end{aligned}$$

along a given parameter trajectory $p(t)$ and for an input signal $u(t)$ is simulated by the function `pdsimul`. The parameter vector $p(t)$ is required to range in a box (type `'box'` of the function `pvec`). The function `pdsimul` also accepts the polytopic representation of affine parameter-dependent systems as returned by `aff2pol(pds)`.

The syntax of the function `pdsimul` is

```
pdsimul(pds,'traj',tf,'ut',xi,options)
```

or

```
[t,x,y] = pdsimul(pds,pv,'traj',tf,'ut',xi,options)
```

When invoked without output arguments, `pdsimul` plots the output response $y(t)$ of the affine parameter-dependent system `pds`. Otherwise, it returns the vector of integration time points t as well as the state and output responses x, y. The parameter trajectory and input signals are specified by two time functions `p = traj(t)` and `u = ut(t)`. If `'ut'` is omitted, the response to a step input is computed by default. The final time and initial state vector can be reset through `tf` and `xi` (their respective default values are 5 seconds and zero vector). Finally, the input argument `options` gives access to the parameters controlling the integration of the system differential equations.

Example 13.3 Consider the simulation of the step response of mass–damper–spring system for a period of 30 seconds. The parameter trajectory is chosen as

$$\begin{aligned} m(t) &= 3 + 1.2e^{-0.3t}\cos(10t) \\ k(t) &= 2 + 0.6e^{-0.3t}\sin(10t) \\ c(t) &= 0.8 + 0.01t \end{aligned}$$

and is specified in the file `traj_mds.m`.

```
function p = traj_mds(t)
%
p = [3.0 + 1.2*exp(-0.3*t)*cos(10*t); ...
     2.0 + 0.6*exp(-0.3*t)*sin(10*t); ...
     0.8 + 0.01*t];
```

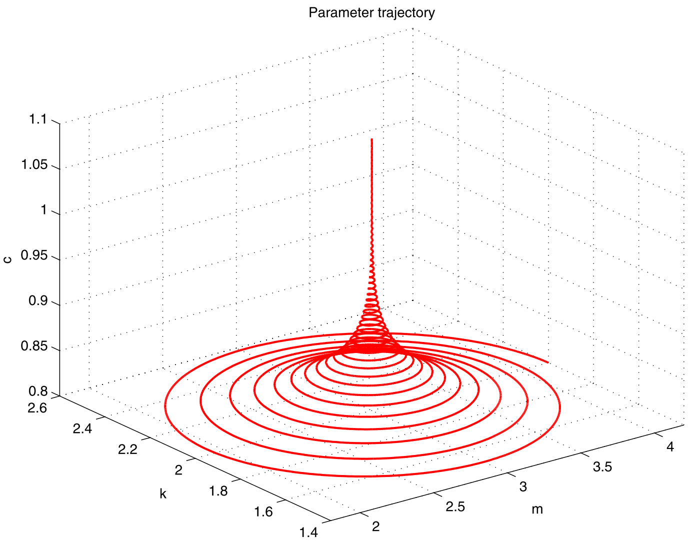


Fig. 13.3 Parameter trajectory for system simulation

The parameter trajectory is shown in Fig. 13.3.

The time response of the mass–damper–spring system for the given parameter trajectory along the time response of the nominal system are obtained by the lines

```
[t,x,y] = pdsimul(pds_mds,'traj_mds',30);
mds_nom = psinfo(pds_mds,'eval',[3 2 1]);
[a,b,c,d] = ltiss(mds_nom); mds_ss = ss(a,b,c,d);
tt = 0:0.01:30;
y_nom = step(mds_ss,tt);
figure(1)
plot(t,y(:,1),'b-',tt,y_nom,'r--'), grid
xlabel('Time (secs)')
ylabel('y')
title('System transient response')
legend('Parameter-dependent system','Nominal system')
```

Both time responses are shown in Fig. 13.4. Comparing with the step responses of the mass–damper–spring system for 20 random values of the uncertain elements, shown in Fig. 9.14, we see that the response of the parameter-dependent system is slightly different from the response of the nominal system. This is explained by the fact that in the latter case the parameters m, k, c correspond to a given trajectory in the parameter space and do not vary independently.

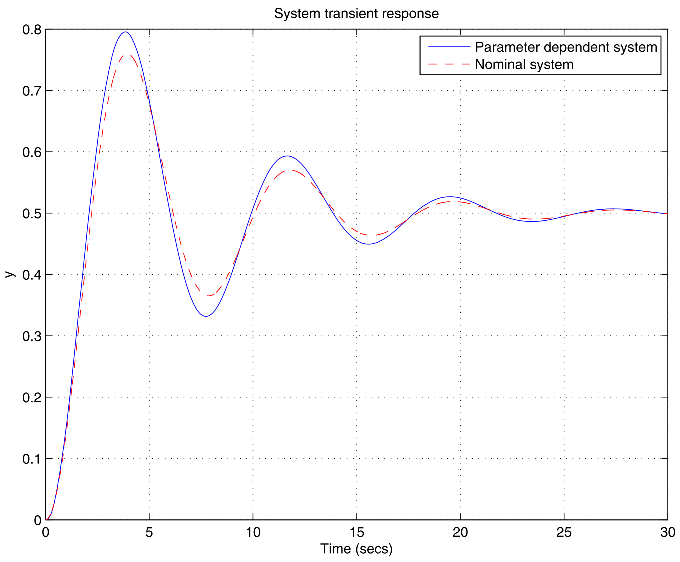


Fig. 13.4 Time response of the parameter-dependent system

13.3 Gain Scheduling Design for Parameter-Dependent Systems

When the system parameters undergo large variations it is often impossible to achieve high performance and even closed-loop stability over the entire operating range with a single time-invariant controller. In this case it is appropriate to apply a technique called *gain scheduling*, which is successfully used in the control of uncertain or time-varying systems. This technique involves implementation of a family of controllers designed for different regions of the parameter space so as to guarantee stability and performance in that region. During the system operation the controllers are changed according to a physical parameter measured in real time which detects in what region the system is working in the corresponding moment of the time. The change of controllers is done either by interpolation of certain parameters or by switching. Such controllers are said to be *scheduled* by the parameter measurements. In this section we consider implementation of gain-scheduled $\mathcal{H}_\infty$ controllers which ensures higher performance in the face of large variations in operating conditions and provides smooth changes between controllers.

The synthesis technique presented in this section is applicable to affine parameter-dependent plants with state-space description

$$P(.,p)\begin{cases}\dot{x}=A(p)x+B_1(p)w+B_2u,\\ z=C_1(p)x+D_{11}(p)w+D_{12}u\\ y=C_2x+D_{21}w+D22u\end{cases} \tag{13.12}$$