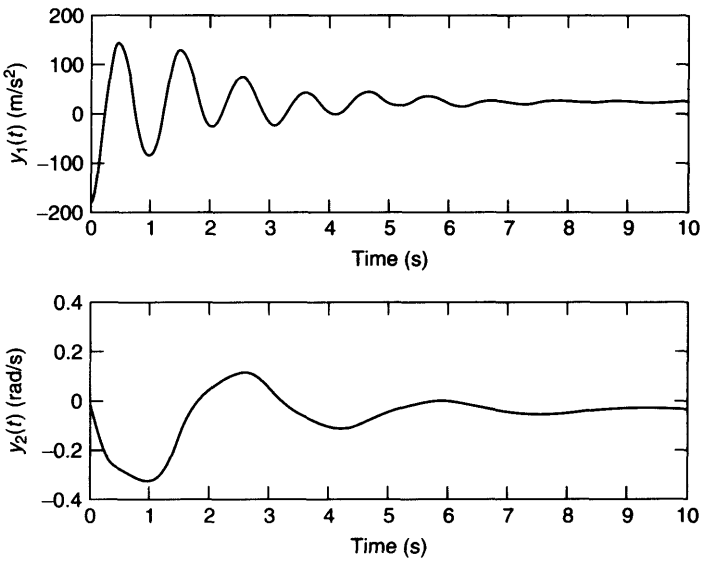

Figure 4.10 Response of the flexible bomber aircraft of Example 4.9 to elevator and canard impulse inputs of same magnitude but opposite signs

```
>>[y,X]=impgen(A,B,C,D,X0,t,c); <enter>
```

The computed outputs, $y_1(t)$ and $y_2(t)$, which are the first and second columns of the returned matrix, y, respectively, are plotted in Figure 4.10. Note that we could not get this response from the CST function *impulse*, which is confined to the case of $\mathbf{c} = [\,1;\quad 1\,]^T$.

4.5 Numerical Solution of Linear Time-Varying State-Equations

There is no general analytical procedure of solving the state-equations for linear *time-varying* systems (i.e. when the state coefficient matrices, **A**,**B**,**C**,**D**, are functions of time). Thus, numerical solution procedures using digital approximation methods, similar to those of the previous section, are required for solving the state-equations of such systems. For special time-varying systems in which the state-dynamics matrix, **A**, is a constant and the matrices **B**, **C**, and **D** are functions of time, the analytical solution given by Eq. (4.27) is valid with the following modification:

$$\mathbf{x}(t) = \exp\{\mathbf{A}(t - t_0)\}\mathbf{x}(t_0) + \int_{t_0}^{t} \mathrm{e}^{A(t-\tau)}\mathbf{B}(\tau)\mathbf{u}(\tau)d\tau; \quad (t \geq t_0) \tag{4.76}$$

where $\mathbf{x}(t_0)$ is the initial condition and $\mathbf{B}(\tau)$ indicates the controls coefficient matrix evaluated at time $t = \tau$. For linear, time-varying systems the output equation is given by:

$$\mathbf{y}(t) = \mathbf{C}(t)\mathbf{x}(t) + \mathbf{D}(t)\mathbf{u}(t) \tag{4.77}$$

For systems with a constant **A** matrix, the output can be calculated by Eq. (4.77), in which $\mathbf{x}(t)$ is calculated using Eq. (4.76). A digital approximation of Eq. (4.76) is given by

$$\mathbf{x}(n\Delta t) = \mathbf{A}_d\mathbf{x}((n-1)\Delta t) + \mathbf{B}_d((n-1)\Delta t)\mathbf{u}((n-1)\Delta t);\ (n = 1, 2, 3, \ldots) \quad (4.78)$$

where $\mathbf{A}_d = e^{A\Delta t}$ and $\mathbf{B}_d((n-1)\Delta t) = \mathbf{B}((n-1)\Delta t)\Delta t$. The solution given by Eq. (4.78) can be implemented in a computer program, such as *march.m*. However, since time-varying systems with a constant **A** matrix are rarely encountered, such a program will be rarely used. Instead, we need a general solution procedure which is applicable to a general time-varying system given by:

$$\mathbf{x}^{(1)}(t) = \mathbf{A}(t)\mathbf{x}(t) + \mathbf{B}(t)\mathbf{u}(t) \quad (4.79)$$

A practical procedure for solving Eq. (4.79) is the application of the time-marching approach of Section 4.5, assuming that $\mathbf{A}(t)$ is constant *during each time step*, Δt, but

Table 4.4 Listing of the M-file *vmarch.m*

vmarch.m

```
function [y,X] = vmarch(tvfun,X0,t,u,method)
% Time-marching solution of linear, time-varying
% state-space equations using the matrix exponential.
% X0= initial state vector; t= time vector;
% u=matrix with the ith input stored in the ith column, and jth row

% corresponding to the jth time point
% y= returned matrix with the ith output stored in the ith column,
  and jth row
% corresponding to the jth time point
% X= returned matrix with the ith state variable stored in the ith column,
% and jth row corresponding to the jth time point
% method= method of digital interpolation for the inputs (see 'c2dm')
% copyright(c)2000 by Ashish Tewari
n=size(t,2);
dt=t(2)-t(1);
% initial condition:-
X(1,:)=X0';
% function evaluation of time varying state coefficient matrices
% using the M-file 'tvfun.m' for initial time t=t(1):-
[A,B,C,D]=feval(tvfun,t(1));
% outputs for t=t(1):-
y(1,:)=X(1,:)*C'+u(1,:)*D';
% beginning of the time-loop:-
for i=1:n-1
% function evaluation of time varying state coefficient matrices
% using the M-file 'tvfun.m' for t=t(i):-
[A,B,C,D]=feval(tvfun,t(i));
% digital approximation of the continuous-time system:-
[ad,bd,cd,dd]=c2dm(A,B,C,D,dt,method);
% solution of the digital state and output equations:-
X(i+1,:)=X(i,:)*ad'+u(i,:)*bd';
y(i+1,:)=X(i+1,:)*C'+u(i,:)*D';
end
```

varies as we proceed from *one time step to the next*. In essence, this procedure applies a *zero-order hold* to $\mathbf{A}(t)$. Thus, the approximate digital solution for a general time-varying system can be written as follows:

$$\mathbf{x}(n\Delta t) = \mathbf{A}_d((n-1)\Delta t)\mathbf{x}((n-1)\Delta t) + \mathbf{B}_d((n-1)\Delta t)\mathbf{u}((n-1)\Delta t); \quad (n = 1, 2, 3, \ldots) \tag{4.80}$$

where $\mathbf{A}_d = \exp\{\mathbf{A}((n-1)\Delta t)\Delta t\}$ and $\mathbf{B}_d((n-1)\Delta t) = \mathbf{B}((n-1)\Delta t)\Delta t$. A MATLAB M-file can be written based on Eqs. (4.80) and (4.77) for computing the time marching output of a general linear, time-varying system. Such an M-file, called *vmarch.m* is given in Table 4.4.

Example 4.10

Consider the following linear, time-varying system:

$$\mathbf{A}(t) = \begin{bmatrix} -0.1\sin(t) & 0 \\ 0 & -0.7\cos(t) \end{bmatrix}; \quad \mathbf{B}(t) = \begin{bmatrix} 0.2\sin(t) \\ 0 \end{bmatrix} \tag{4.81}$$

$$\mathbf{C}(t) = [1 \quad -1]; \quad \mathbf{D}(t) = -0.05\cos(t) \tag{4.82}$$

Let us calculate the output, $\mathbf{y}(t)$, when the initial condition is $\mathbf{x}(0) = [0; \quad -1]^T$ and the input is a unit setp function, $\mathbf{u}(t) = u_s(t)$. The time vector, initial condition, and input are specified by the following MATLAB command:

```
>>t = 0:0.1:10; X0 = [0 -1]'; u = ones(size(t,2), 1); <enter>
```

The time-varying coefficient matrices are calculated by the M-file *timv.m* tabulated in Table 4.5. Then, the solution to the time-varying state-equations and the output are obtained by calling *vmarch.m* as follows:

```
>>[y,X] = vmarch('timv',X0,t,u,'zoh'); <enter>
```

The resulting output, $y(t)$, is plotted in Figure 4.11.

Table 4.5 Listing of the M-file *timv.m*

timv.m

```
function [A,B,C,D]=timv(t);
% Linear time-varying state coefficient matrices for Example 4.10.
A=[-0.1*sin(t) 0;0 -0.7*cos(t)];
B=[0.2*sin(t);0];
C=[1 -1];
D=-0.05*cos(t);
```

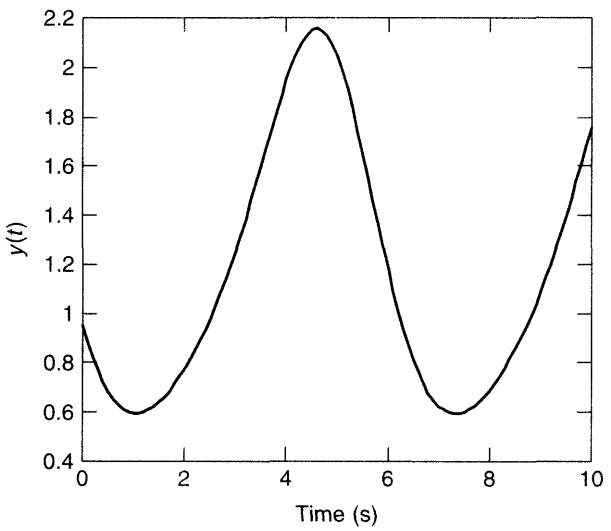

Figure 4.11 Output of the linear, time-varying system of Example 4.10

Example 4.11

Let us obtain the response of an interesting time-varying system. Consider a *surface-to-air missile* propelled by a rocket engine. Once launched, the rocket consumes the fuel rapidly, thereby changing the *mass, center-of-gravity*, and *moments of inertia* of the missile. Also, as the fuel is burned, the missile accelerates and changes altitude. The aerodynamic forces and moments acting on the missile are functions of the flight velocity and the altitude; therefore, the aerodynamic properties of the missile also keep changing with time. The motion of the missile is described by the following state variables: velocities, U, V, and W, along *three mutually perpendicular axes*, x, y, z, respectively, passing through the missile's *center of gravity*, and the rotation rates, p, q, and r, about x, y, and z, respectively. The state-vector is thus $\mathbf{x}(t) = [U(t); V(t); W(t); p(t); q(t); r(t)]^T$. All the state-variables are assumed to be the outputs of the system. A diagram of the missile showing the state-variables and the inputs is given in Figure 4.12. One input to the missile is the *rolling-moment*, ΔL, about the longitudinal axis of the missile caused by the deflection of an aerodynamic *control surface.* In addition, the *thrust* from the rocket engine can be *vectored* (i.e. deflected) by small angles, α and β, in the *longitudinal* (X, Z) plane and *lateral* (Y, Z) plane, respectively, of the missile (Figure 4.12). These thrust deflection angles constitute two additional inputs to the missile. The input vector is $\mathbf{u}(t) = [\alpha; \beta; \Delta L]^T$. The thrust of the missile is *assumed constant* until the rocket motor *burns-out* at $t = 20$ s, after which time the thrust is zero. The time-varying state-coefficient matrices representing the

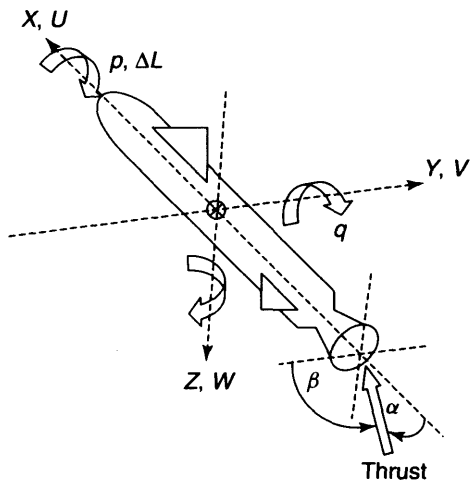

Figure 4.12 *State-variables and inputs for a surface-to-air missile equipped with thrust vectoring and aerodynamic roll control surfaces*

linearized motion of the missile are calculated by the M-file *misstimv.m* tabulated in Table 4.6.

Let us calculate the response of the missile if the initial condition is zero, and the inputs are given by

$$\mathbf{u}(t) = \begin{bmatrix} 0.01\sin(t) \\ 0 \\ 0.01\cos(t) \end{bmatrix} \tag{4.83}$$

The time vector, initial condition, and input are specified by the following MATLAB command:

```
>>t = 0:0.2:30; X0 = zeros(6,1); u = [0.01*sin(0.1*t)' 0.01*cos(0.1*t)'
  zeros(size(t,2),1)]; <enter>
```

The output and state solution are then calculated by the following call to *vmarch.m*:

```
>>[y,X] = vmarch('misstimv',X0,t,u,'zoh'); <enter>
```

The calculated outputs, $U(t)$, $V(t)$, $W(t)$, and $p(t)$, are plotted in Figure 4.13, while the outputs $q(t)$ and $r(t)$ are shown in Figure 4.14. Note the discontinuity in the slopes of all the responses at $t = 20$ s, which is the rocket *burn-out* time.

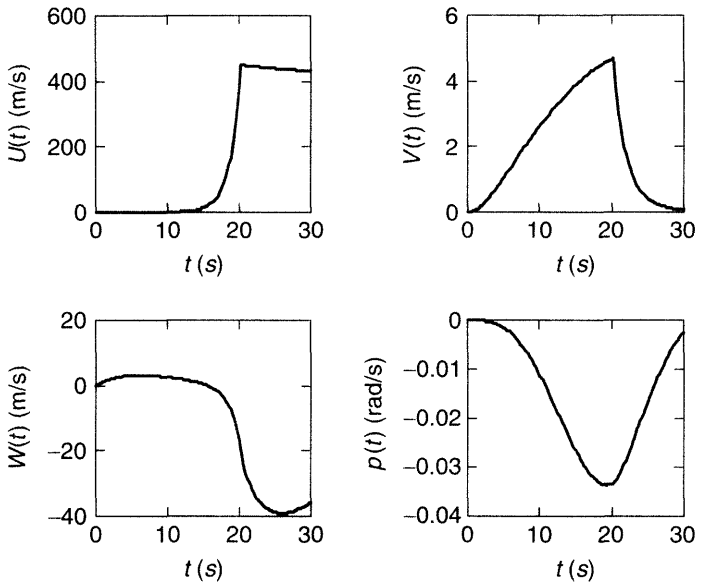

Figure 4.13 The outputs $U(t)$, $V(t)$, $W(t)$, and $p(t)$, for the missile of Example 4.11

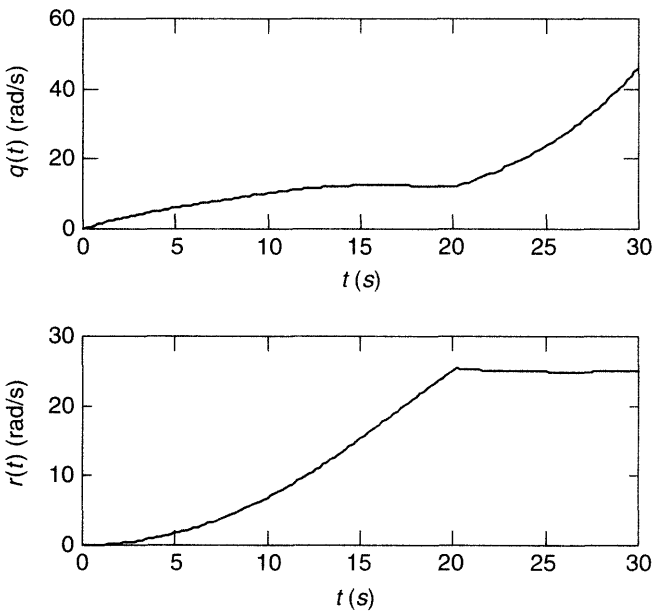

Figure 4.14 The outputs $q(t)$ and $r(t)$ for the missile of Example 4.11