Available online at www.sciencedirect.com
SCIENCE DIRECT

Journal of Sound and Vibration 271 (2004) 1099–1112

JOURNAL OF SOUND AND VIBRATION

www.elsevier.com/locate/jsvi

Author's reply ☆

Ebrahim Esmailzadeh[a,1], Nader Jalili[b,*]

[a] *Department of Mechanical Engineering, Sharif University of Technology, P.O. Box 11365-9567, Tehran, Iran*
[b] *Robotics and Mechatronics Laboratory, Department of Mechanical Engineering, Clemson University, Clemson, SC 29634-0921, USA*

Received 5 June 2003

1. Introduction and general response

Recently, we have received some comments [1] on our previously published paper in the Journal of Sound and Vibration [2]. The author has raised two issues with regards to: (a) the practical implementation and experimental verification of the modelling efforts presented in the paper as well as (b) the adequacy of the simple quarter car model for bridge–vehicle–passenger interaction problem. The author has also provided some analytical relationships between the maximum beam deflection and the speed of the moving vehicle. These findings are based on a recently published paper by the author in the Journal of Sound and Vibration [3].

Interestingly enough, our optimization results given in Ref. [2] are in full agreement with those new findings but only with minor differences due to the damping included in the model of Ref. [2]. Although the optimization problem given in Ref. [2] can be solved using the analytical solution provided by the author (Eq. (2) in Ref. [1]), such relationships typically hold true only with some simplifications and for certain assumptions that cannot be relaxed for many realistic cases, such as damping in the beam, damping in the vehicle suspension, non-linear behavior in the suspension components, and others. We strongly believe that with the capability of today's computational software and modern real-time control hardware together with the complexity of the models and realistic assumptions, numerical approaches are becoming more and more attractive. This was exactly the intention of the present authors to provide such innovative numerical and systematic-like procedure in Ref. [2] for finding the critical speed of vehicles travelling on suspension bridges (see pp. 626–629 in Ref. [2]).

Such comments ((a) and (b) above) are not new to us, as we have seen similar concerns during the review process of this paper. The purpose of this reply is to show that: (1) the numerical results given in Ref. [2] are indeed correct and are based on the true comparison between the simple

☆ Reply to doi:10.1016/j.jsv.2003.06.008

*Corresponding author. Tel.: +1-864-656-5642; fax: +1-864-656-4435.
E-mail addresses: Ebrahim.Esmailzadeh@uoit.ca (E. Esmailzadeh), jalili@clemson.edu (N. Jalili).
[1] Present address: Manufacturing Engineering Program, University of Ontario Institute of Technology, 2000 Simcoe Street North, Oshawa, Ontario, Canada L1H 7K4.

0022-460X/$ - see front matter
doi:10.1016/j.jsv.2003.06.001

quarter-car model and the half-car model, and (2) the focal point of our paper [2] is neither on the design and construction of bridges nor on the practical implementations and/or the suitability of our results in practice, since these are the focus of design investigations and structural studies, which we do not intend to approach at this stage.

The numerical results presented in our paper are merely to validate the assumptions made in deriving the governing equations of motion. As far as the author's comments on the adequacy of the simple quarter-car model are concerned, it appears that those comments are more of an opinion than detailed technical assessments, which are debatable. Next we shall demonstrate the inadequacy of the simple quarter-car model in analyzing the total vehicle dynamics in order to widen the horizon of the authors vision on vehicle suspension modelling.

2. Technical response on inadequacy of the SQC model

At this stage, we would like to draw the attention of the author to this fact as to how the present authors have concluded that "the use of a simple quarter car (SQC) model does not provide an adequate information for both vehicle dynamics and bridge characteristics". The author has objected, in Ref. [1], to the comparison of the results between the SQC and half-car models as illustrated in Figs. 9 and 10 in Ref. [2]. He has presented his own results (see Fig. 1 in Ref. [1]) in order to support his statements. It should be noted that in the same manner that this author has questioned the validity of our numerical results, we could in turn argue about the correctness of the results presented by the author in Ref. [1] since there are no numerical procedures given. To justify this argument, we have provided all the details of the programming codes that were used to generate the results shown in Figs. 9 and 10 in Ref. [2] (see Appendix A). While the author's viewpoint is on the suitability of the SQC model in practice as a simple rule of thumb, our intention is to demonstrate, mathematically and numerically, that there is quite a difference between those two models. How useful these comparisons are in oversimplified practical applications are not our intention to discuss here.

In order to convince the author, numerical comparisons between the SQC and half-car models are presented here, in Figs. 1–6, for the beam mid-span deflection (as an indicator of the bridge

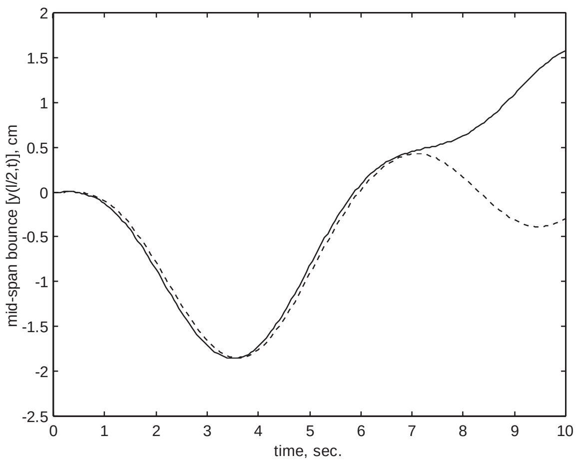

Fig. 1. Comparison between beam mid-span deflection for SQC (——) and half-car (- - - -) models for $V = 56\,\text{km/h}$.

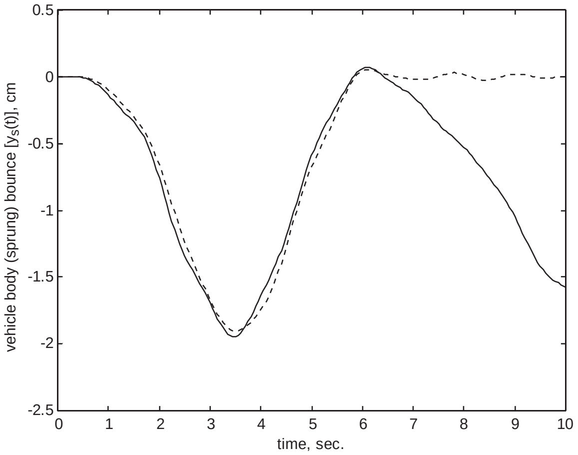

Fig. 2. Comparison between vehicle body bounce for SQC (——) and half-car (----) models for $V = 56\,\mathrm{km/h}$.

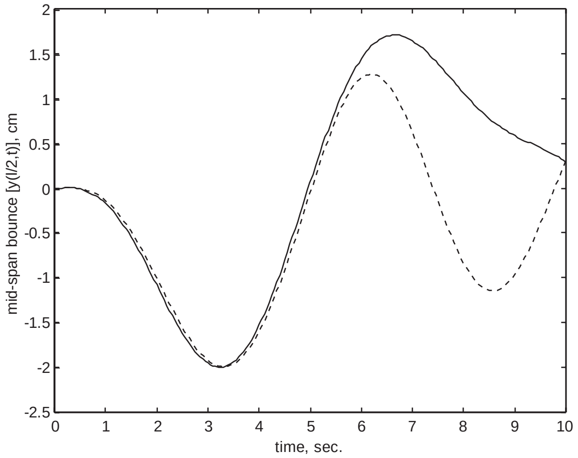

Fig. 3. Comparison between beam mid-span deflection for SQC (——) and half-car (----) models for $V = 72\,\mathrm{km/h}$.

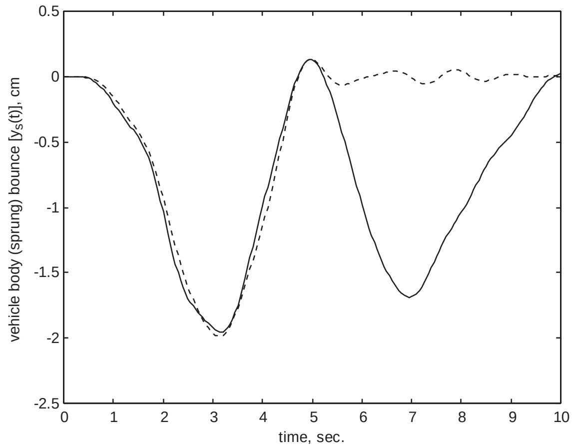

Fig. 4. Comparison between vehicle body bounce for SQC (——) and half-car (----) models for $V = 72\,\mathrm{km/h}$.

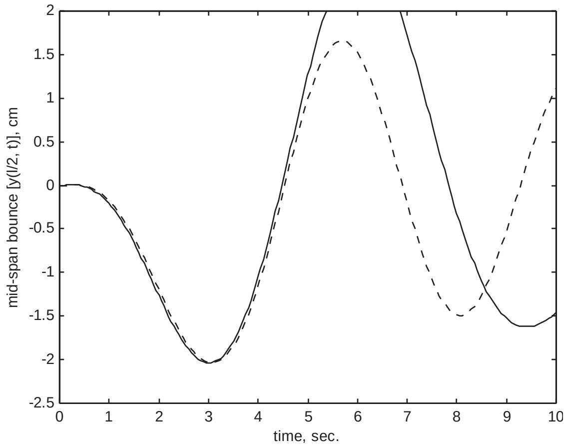

Fig. 5. Comparison between beam mid-span deflection for SQC (——) and half-car (- - - -) models for $V = 88\,\mathrm{km/h}$.

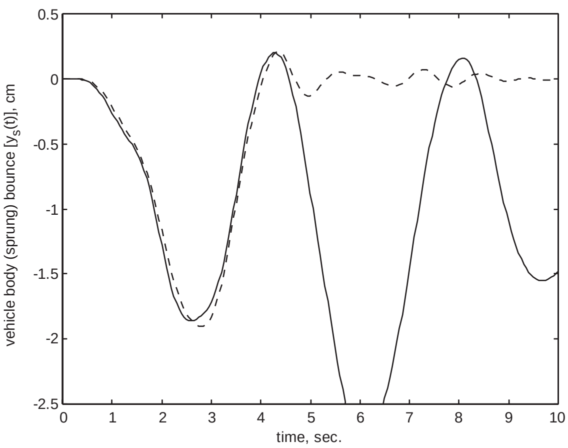

Fig. 6. Comparison between vehicle body bounce for SQC (——) and half-car (- - - -) models for $V = 88\,\mathrm{km/h}$.

vibration characteristics) and the vehicle body bounce (as an indicator of the vehicle ride comfort). The Matlab programming, with all the subroutines, are presented in Appendix A. It can be seen that, depending on vehicle speeds, the difference between these two models' responses (SQC and half-car) varies and in some cases it becomes significant. This also makes sense, physically and mathematically, since the governing differential equations are quite different from many viewpoints including the number of poles and zero-dynamics, order of the total system and the distribution of the natural frequencies. We must point out that the use of the SQM provide an oversimplification of the required accuracy and could well be sufficient for limited practical applications. However, the results presented in Ref. [2] reiterate that this conclusion is not mathematically correct and cannot be justified at least for the system parameters considered in Ref. [2]. Therefore, the numerical results presented in Ref. [2] are all correct as illustrated in Figs. 9 and 10 Ref. [2].

Appendix A. Matlab programming codes for SQC and HCM cases

A.1. Simple Quarter Car Model

```
% ... Simulation of Vehicle travelling on beam using simple Quarter Car Model

% … Main Program
clear;
echo off; clf;

global N S zita M1 M2 K1 K2 C2 C c L

% ... Beam Constants Parameters ... (see pp. 621 of [2])
n           = 4;            % Number of modes
L           = 100;          % Beam length (m)
xd          = L/2.;         % Desired Position of beam in order to plot it's deflection of v(x,t)
E           = 2.07e11;      % Modulus of Elasticity of the beam (N/m^2)
I           = 0.2*0.87;     % cross section moment of inertia
EI          = E*I;
roh         = 20000;        % Linear Density of beam (Kg/m)
c           = 1750;         % viscous damping of beam (N.s/m^2)

% ... Vehicle properties ... (see pp. 623 of [2])
M2          = 1794.4+2*75;          % Mass of the unsprung (Kg)
M1          = 140.4+87.15;          % Mass of the sprung (Kg)
C           = 1190+1000;            % damping
C2          = 14.6+14.6;
K1          = 66824.2+18615.0;      % spring
K2          = 2*101115.;            % spring

% ... Trajectory (or IC) Constants Parameters ...
V0          = 45.                   % Initial velocity (mile per hr.)
V0          = V0*1.609/3.6;         % Initial velocity (m/s)
a           = 0;
x(2*n+4)    = 0.;
x0          = x;

% ... Solver Constants Parameters ...
options     = odeset('RelTol',1e-3);
tfinal      = 10;           % Final Time(sec.)
dt          = 0.05;         % Sampling time(sec)
t           = 0.;
t0          = 0.;
tf          = dt;

% .... Calculating the Normalized Coefficients of Mode Shapes of the Beam...
```

```
for j=1:n,

   bn(j)          = j*pi/L;
   N(j)           = roh;
   S(j)           = EI*bn(j)^4;

end;

% ... Solving the ODE's  ...
disp('integrating....');
for i=1:(tfinal/dt)+1,

   sum1           = 0;

   for j=1:n;
      sum1        = sum1 + sqrt(2/L)*sin(j*pi*xd/L)*x(4+j);

   end

   zita           = 0.5*a*t^2+V0*t;

% ... Storing outputs ....

   outmidspan(i)        = sum1;
   outsprung(i)         = x(1);
   outunsprung(i)       = x(3);
   time(i)              = t;

       tspan            = [t0 tf];
       [del,xdel]       = ode45('polveh_bridge_qcm_comp_hcm', tspan, x0, options);
       [column, row]    = size(xdel);
       t0               = t0+dt;
       tf               = tf+dt;
       t                = t+dt;

       for j=1:2*n+4,
       x(j)             = xdel(column,j);
       x0(j)            = x(j);
       end;

end;

% ... RESULTS ...
figure(1)
clf
plot(time,outmidspan*100,'k-');
```

```
hold on
axis([0 tfinal -2.5 2]);
%grid;
ylabel('mid-span bounce [y(l/2,t)], cm');
xlabel('time, sec.');

figure(2)
clf;
plot(time,outsprung*100,'k-');
hold on
axis([0 tfinal -2.5 0.5]);
%grid;
ylabel('vehicle body (sprung) bounce [y_s(t)], cm');
xlabel('time, sec.');

figure(3)
clf;
plot(time,outunsprung*100,'k-');
hold on
axis([0 tfinal -2 0.5]);
%grid;
ylabel('Unsprung bounce [y_u(t)], cm');
xlabel('time, sec.');

figure(4)
clf;
plot(time,outunsprung*100,'k-');
hold on
axis([0 tfinal -2 0.5]);
%grid;
ylabel('Unsprung bounce [y_u(t)], cm');
xlabel('time, sec.');
```

% … Function (subroutine) Program

```
function xdot = polveh_bridge_qcm_comp_hcm(t,x)

global N S zita M1 M2 K1 K2 C2 C c L

D         = sqrt(2/L)*(sin(pi*zita/L)*x(5)+sin(2*pi*zita/L)*x(6)+sin(3*pi*zita/L)*x(7)+…
            sin(4*pi*zita/L)*x(8));
DD        = sqrt(2/L)*(sin(pi*zita/L)*x(9)+sin(2*pi*zita/L)*x(10)+sin(3*pi*zita/L)*x(11)+…
            sin(4*pi*zita/L)*x(12));

xdot(1)   = x(2);
xdot(2)   = (-K1*(x(1)-x(3))-C*(x(2)-x(4)))/M2;
```

```
xdot(3)         = x(4);
xdot(4)         = (K1*(x(1)-x(3))+C*(x(2)-x(4))-K2*(x(3)-D)-C2*(x(4)-DD))/M1;
xdot(5)         = x(9);
xdot(6)         = x(10);
xdot(7)         = x(11);
xdot(8)         = x(12);
xdot(9)         = (-S(1)*x(5)-c*x(9)-((M1+M2)*9.81+K2*(D-x(3))+C2*(DD-…
                    x(4)))*sqrt(2/L)*sin(pi*zita/L))/N(1);
xdot(10)        = (-S(2)*x(6)-c*x(10)-((M1+M2)*9.81+K2*(D-x(3))+C2*(DD-…
                    x(4)))*sqrt(2/L)*sin(2*pi*zita/L))/N(2);
xdot(11)        = (-S(3)*x(7)-c*x(11)-((M1+M2)*9.81+K2*(D-x(3))+C2*(DD-…
                   x(4)))*sqrt(2/L)*sin(3*pi*zita/L))/N(3);
xdot(12)        = (-S(4)*x(8)-c*x(12)-((M1+M2)*9.81+K2*(D-x(3))+C2*(DD-…
                   x(4)))*sqrt(2/L)*sin(4*pi*zita/L))/N(4);

xdot            = xdot';

return
```

A.2. Half-Car Model

```
% ... Simulation of Vehicle travelling on beam using Half-Car Model with passenger dynamics ...
% ... 6-DOF system, beam is also considered to be viscous damped (linear)
```

% … Main Program

```
clear;
echo off; clf;

global N S c zita1 zita2 J Ms M1 M2 C1 C2 Ct1 Ct2 Cp1 Cp2 fg1 fg2
global K1 K2 Kt1 Kt2 Kp1 Kp2 Mp1 Mp2 b1 b2 d1 d2 L D1 D2

% ... Beam Constants Parameters ...(see pp. 261 of [2])
n               = 4;            % Number of modes
L               = 100;          % Beam length (m)
xd              = L/2.;         % Desired Position of beam in order to plot it's deflection of v(x,t)
E               = 2.07e11;      % Modulus of Elasticity of the beam (N/m^2)
I               = 0.2*0.87;     % cross section moment of inertia
EI              = E*I;
roh             = 20000;        % Linear Density of beam (Kg/m)
c               = 1750;         % viscous damping of beam (N.s/m^2)

% ... Vehicle properties ... (see pp. 261 of [2])
J               = 3443.05;      % Body inertia (Kg.m^2)
Ms              = 1794.4;       % Mass of the body (Kg)
M1              = 87.15;        % Mass of the front axle (Kg)
M2              = 140.4;        % Mass of the rear axle (Kg)
Mp1             = 75;           % Mass of the driver (Kg)
Mp2             = 75;           % Mass of the passenger (Kg)
C1              = 1190;         % front axle damping (N.s/m)
```

```
C2          = 1000;       % rear axle damping (N.s/m)
Ct1         = 14.6;       % front tire damping (N.s/m)
Ct2         = 14.6;       % rear tire damping (N.s/m)
Cp1         = 50.2;       % front seat damping (N.s/m)
Cp2         = 62.1;       % rear sear damping (N.s/m)
K1          = 66824.2;    % front main stiffness (N/m)
K2          = 18615.0;    % rear main stiffness (N/m)
Kt1         = 101115.;    % front tire stiffness (N/m)
Kt2         = 101115.;    % rear tire stiffness (N/m)
Kp1         = 14000;      % front seat stiffness (N/m)
Kp2         = 14000;      % rear seat stiffness (N/m)
b1          = 1.271;      % front dimension (m)
b2          = 1.713;      % front dimension (m)
d1          = 0.481;      % front passenger (m)
d2          = 1.313;      % rear passenger (m)

% .... Calculating the Normalized Coefficients of Mode Shapes of the Beam...
for j=1:n,

  bn(j)     = j*pi/L;
  N(j)      = roh;
  S(j)      = EI*bn(j)^4;

end;

% ... Trajectory Constants Parameters ...
V0          = 35.                % Initial velocity (mile per hr.)
Vf          = 55;                % final velocity (mph)
Vr          = 10;                % resolution (mph)
V0          = V0*1.609/3.6;      % Initial velocity (m/s)
Vf          = Vf*1.609/3.6;      % final velocity (m/s)
Vr          = Vr*1.609/3.6;      % resolution velocity (m/s)

for     ind_v = 1 : ((Vf-V0)/Vr)+1;

  V          = V0+(ind_v-1)*Vr
  V_(ind_v)  = V*3.6/1.609;
t1           = (b1+b2)/V;
t2           = L/V;
t3           = (L+b1+b2)/V;
a            = 0;

% ... Weight calculation, fg1 and fg2 ...
fg1          = (M1+(Ms*b2+Mp1*(b2+d1)+Mp2*(b2-d2))/(b1+b2))*9.81;
fg2          = (M2+(Ms*b1+Mp1*(b1-d1)+Mp2*(b1+d2))/(b1+b2))*9.81;
```

```
% ... Clearing all the variables ...
clear    x x0 out_midspan out_body out_ftire out_rtire out_driver out_passenger
clear    out_q1 out_q2 out_q3 out_q4 time
clear    z_ y M_ben YMAX INMAX YMAXp INMAXp y_max ind_max y_maxp ind_maxp
clear    MMAX INMMAX M_max ind_Mmax

% ... IC Constants Parameters ...
x(2*n+12)       = 0.;
x0              = x;

% ... Solver Constants Parameters ...
options = odeset('RelTol',1e-3);
dt              = 0.05;              % Sampling Time(sec)
tfinal          = 10.;               % Final Time(sec.)
t               = 0.;
t0              = 0.;
tf              = dt;

% ... Solving the ODE's  ...
disp('integrating....');
for i=1:(tfinal/dt)+1,

  if (t<t3),

    if (t<t2),
      D1        = 1;
        if (t<t1),
        D2      = 0;
        else
        D2      = 1;
      end
    else
      D1        = 0;
      D2        = 1;
    end

  else
    D1 = 0;
    D2 = 0;
  end

  sum1= 0;

  for j=1:n;
    sum1        = sum1 + sqrt(2/L)*sin(j*pi*xd/L)*x(8+j);
  end
```

```
   zita1          = -b1+V*t+b1;
   zita2          = -b1+V*t-b2;

         % ... Storing outputs ....

   out_midspan(i)     = sum1;          % mid-span deflection (m)
   out_body(i)        = x(1);          % body bounce (m)
   out_ftire(i)       = x(5);          % front tire bounce (m)
   out_rtire(i)       = x(7);          % rear tire bounce (m)
   out_driver(i)      = x(17);         % driver bounce (m)
   out_passenger(i)   = x(19);         % passenger bounce (m)
   out_q1(i)          = x(9);          % time-domain generalized, 1st mode
   out_q2(i)          = x(10);         % time-domain generalized, 2nd mode
   out_q3(i)          = x(11);         % time-domain generalized, 3rd mode
   out_q4(i)          = x(12);         % time-domain generalized, 4tg mode

   time(i)            = t;             % time (sec.)

        tspan             = [t0 tf];
        [del,xdel]        = ode45('polveh_bridge_hcm6', tspan, x0, options);
        [column, row]     = size(xdel);
        t0                = t0+dt;
        tf                = tf+dt;
        t                 = t+dt;

        for   j=1:2*n+12,
     x(j)                 = xdel(column,j);
     x0(j)                = x(j);
        end;

end

y_l2(ind_v,:)         = out_midspan;
y_s(ind_v,:)          = out_body;
y_t1(ind_v,:)         = out_ftire;
y_t2(ind_v,:)         = out_rtire;
y_p1(ind_v,:)         = out_driver;
y_p2(ind_v,:)         = out_passenger;

end

V_km  = V_*1.609;

% ... RESULTS ...
figure(1)
clf;
```

```
plot(time,y_l2(1,:)*100,'k-');
hold on
plot(time,y_l2(2,:)*100,'k:');
plot(time,y_l2(3,:)*100,'k--');
axis([0 tfinal -2.5 2]);
%grid;
ylabel('mid-span bounce [y(l/2,t)], cm');
xlabel('time, sec.');

figure(2)
clf;
plot(time,y_p1(1,:)*100,'k-');
hold on
plot(time,y_p1(2,:)*100,'k:');
plot(time,y_p1(3,:)*100,'k--');
axis([0 tfinal -2.5 0.5]);
%grid;
ylabel('driver bounce [y_p1(t)], cm');
xlabel('time, sec.');

figure(3)
clf;
plot(time,y_p2(1,:)*100,'k-');
hold on
plot(time,y_p2(2,:)*100,'k:');
plot(time,y_p2(3,:)*100,'k--');
axis([0 tfinal -2.5 0.5]);
%grid;
ylabel('passenger bounce [y_p2(t)], cm ');
xlabel('time, sec.');

figure(4)
clf;
plot(time,y_s(1,:)*100,'k-');
hold on
plot(time,y_s(2,:)*100,'k:');
plot(time,y_s(3,:)*100,'k--');
axis([0 tfinal -2.5 0.5]);
%grid;
ylabel('vehicle body c.g. bounce [y_s(t)], cm');
xlabel('time, sec.');

figure(5)
clf;
plot(time,y_t1(1,:)*100,'k-');
hold on
plot(time,y_t1(2,:)*100,'k:');
```

```
plot(time,y_t1(3,:)*100,'k--');
axis([0 tfinal -2 0.5]);
%grid;
ylabel('front trie  bounce [y_t1(t)], cm');
xlabel('time, sec.');

figure(6)
clf;
plot(time,y_t2(1,:)*100,'k-');
hold on
plot(time,y_t2(2,:)*100,'k:');
plot(time,y_t2(3,:)*100,'k--');
axis([0 tfinal -2. 0.5]);
%grid;
ylabel('rear tire bounce [y_t2(t)], cm ');
xlabel('time, sec.');
```

% … Function (subroutine) Program

```
function xdot = polveh_bridge_hcm6(t,x)

global N S c zita1 zita2 J Ms M1 M2 C1 C2 Ct1 Ct2 Cp1 Cp2 fg1 fg2
global K1 K2 Kt1 Kt2 Kp1 Kp2 Mp1 Mp2 b1 b2 d1 d2 L D1 D2

D11        = sqrt(2/L)*(sin(pi*zita1/L)*x(9)+sin(2*pi*zita1/L)*x(10)+...
               sin(3*pi*zita1/L)*x(11)+sin(4*pi*zita1/L)*x(12));
D22        = sqrt(2/L)*(sin(pi*zita2/L)*x(9)+sin(2*pi*zita2/L)*x(10)+...
               sin(3*pi*zita2/L)*x(11)+sin(4*pi*zita2/L)*x(12));

Dd11       = sqrt(2/L)*(sin(pi*zita1/L)*x(13)+sin(2*pi*zita1/L)*x(14)+...
               sin(3*pi*zita1/L)*x(15)+sin(4*pi*zita1/L)*x(16));
Dd22       = sqrt(2/L)*(sin(pi*zita2/L)*x(13)+sin(2*pi*zita2/L)*x(14)+...
               sin(3*pi*zita2/L)*x(15)+sin(4*pi*zita2/L)*x(16));

xdot(1)    = x(2);
xdot(2)    = (-C1*(x(2)+b1*x(4)-x(6))-Cp1*(x(2)+d1*x(4)-x(18))-C2*(x(2)-b2*x(4)-x(8))-...
               Cp2*(x(2)-d2*x(4)-x(20))-K1*(x(1)+b1*x(3)-x(5))-Kp1*(x(1)+d1*x(3)-x(17))-...
                K2*(x(1)-b2*x(3)-x(7))-Kp2*(x(1)-d2*x(3)-x(19)))/Ms;
xdot(3)    = x(4);
xdot(4)    = (-C1*b1*(x(2)+b1*x(4)-x(6))-Cp1*d1*(x(2)+d1*x(4)-x(18))+C2*b2*(x(2)-…
               b2*x(4)-x(8))+Cp2*d2*(x(2)-d2*x(4)-x(20))-K1*b1*(x(1)+b1*x(3)-x(5))-….
               Kp1*d1*(x(1)+d1*x(3)-x(17))+K2*b2*(x(1)-b2*x(3)-x(7))+Kp2*d2*(x(1)-…
              d2*x(3)-x(19)))/J;
xdot(5)    = x(6);
xdot(6)    = (-C1*(x(6)-b1*x(4)-x(2))-Ct1*(x(6)-Dd11*D1)-K1*(x(5)-b1*x(3)-x(1))-…
                Kt1*(x(5)-D11*D1))/M1;
xdot(7)    = x(8);
```

```
xdot(8)       = (-C2*(x(8)+b2*x(4)-x(2))-Ct2*(x(8)-Dd22*D2)-K2*(x(7)+b2*x(3)-x(1))-…
                 Kt2*(x(7)-D22*D2))/M2;
xdot(9)       = x(13);
xdot(10)      = x(14);
xdot(11)      = x(15);
xdot(12)      = x(16);
xdot(13)      = (-S(1)*x(9)-c*x(13)-(fg1+Ct1*(Dd11*D1-x(6))+Kt1*(D11*D1-…
                x(5)))*D1*sqrt(2/L)*sin(pi*zita1/L)-(fg2+Ct2*(Dd22*D2-x(8))+Kt2*(D22*…
                D2-x(7)))*D2*sqrt(2/L)*sin(pi*zita2/L))/N(1);
xdot(14)      = (-S(2)*x(10)-c*x(14)-(fg1+Ct1*(Dd11*D1-x(6))+Kt1*(D11*D1-…
                x(5)))*D1*sqrt(2/L)*sin(2*pi*zita1/L)-(fg2+Ct2*(Dd22*D2-…
                x(8))+Kt2*(D22*D2-x(7)))*D2*sqrt(2/L)*sin(2*pi*zita2/L))/N(2);
xdot(15)      = (-S(3)*x(11)-c*x(15)-(fg1+Ct1*(Dd11*D1-x(6))+Kt1*(D11*D1-…
                x(5)))*D1*sqrt(2/L)*sin(3*pi*zita1/L)-(fg2+Ct2*(Dd22*D2-…
                x(8))+Kt2*(D22*D2-x(7)))*D2*sqrt(2/L)*sin(3*pi*zita2/L))/N(3);
xdot(16)      = (-S(4)*x(12)-c*x(16)-(fg1+Ct1*(Dd11*D1-x(6))+Kt1*(D11*D1-…
                x(5)))*D1*sqrt(2/L)*sin(4*pi*zita1/L)-(fg2+Ct2*(Dd22*D2-…
                x(8))+Kt2*(D22*D2-x(7)))*D2*sqrt(2/L)*sin(4*pi*zita2/L))/N(4);
xdot(17)      = x(18);
xdot(18)      = (-Cp1*(x(18)-d1*x(4)-x(2))-Kp1*(x(17)-d1*x(3)-x(1)))/Mp1;
xdot(19)      = x(20);
xdot(20)      = (-Cp2*(x(20)+d2*x(4)-x(2))-Kp2*(x(19)+d2*x(3)-x(1)))/Mp2;

xdot          = xdot';

return
```

References

[1] A.V. Pesterev, Comments on vehicle–passenger–structure interaction of uniform bridges traversed by moving vehicles, *Journal of Sound and Vibration* 271 (2004) 1093–1098, this issue; doi:10.1016/j.jsv.2003.06.008.

[2] E. Esmailzadeh, N. Jalili, Vehicle–passenger–structure interaction of uniform bridges traversed by moving vehicles, *Journal of Sound and Vibration* 260 (2003) 611–635.

[3] V. Pesterev, B. Yang, L.A. Bergman, C.A. Tan, Revisiting the moving force problem, *Journal of Sound Vibration* 261 (2003) 75–91.