Computer Implementation 8.5 ***(Matlab)*** *Transient analysis of a plane frame (p. 576)*

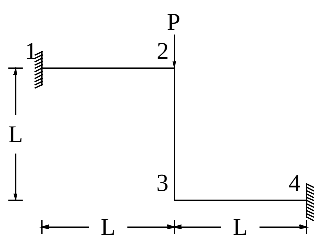
Consider solution of the plane frame. The load P varies with time as shown in the load-time history. The horizontal members carry a weight $200\,N/m$ in addition to their own weight. The other numerical data is as follows.

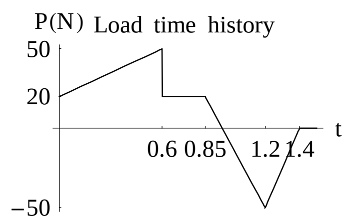
Use N-mm units.

$L = 600\,\text{mm}; \quad E = 200\,\text{GPa} = 200,000\,N/\text{mm}^2; \quad \rho = 7840\,\text{kg}\big/\,m^3 = 7.84\times10^{-6}\,\text{kg}/\text{mm}^3; \quad A = 240\,\text{mm}^2; \quad I$

Additional mass on horizontal members: $\frac{200}{9.81} = 20.3874\,\text{kg}/m = 0.0203874\,\text{kg}/\text{mm}$

f[t] = Which[t ≤ 0.6, 50 t + 20, t ≤ .85, 20, t ≤ 1.2, 190 – 200 t, t ≤ 1.4, 250 t – 350, t > 1.4, 0];





MatlabFiles\Chap8\TransientFrameEx.m

```
% Transient analysis of a plane frame
global Mf Kf Rf
L = 1000; e = 200000; rho = 7.84*10^(-6); a = 240;
inertia = 2000; ma = 0.0203874;
nodes = [0, 0; L, 0; L, -L; 2*L, -L];
conn = [1,2; 2,3; 3,4];
elems = size(conn,1);
lmm=[];
for i=1:elems
   lmm = [lmm; [3*conn(i,1)-2, 3*conn(i,1)-1, 3*conn(i,1),...
               3*conn(i,2)-2, 3*conn(i,2)-1, 3*conn(i,2)]];
end
debc = [1,2,3,10,11,12]; ebcVals=zeros(length(debc),1);
dof=3*size(nodes,1);
M=zeros(dof); K=zeros(dof);
R = zeros(dof,1); R(5)=1;
```

```
% Generate equations for each element and assemble them.
for i=1:2:elems
    con = conn(i,:);
    lm = lmm(i,:);
    [m, k, r] = TransientPlaneFrameElement(e, inertia, a, ...
        rho+ma/a, 0, 0, nodes(con,:));
    M(lm, lm) = M(lm, lm) + m;
    K(lm, lm) = K(lm, lm) + k;
    R(lm) = R(lm) + r;
end
for i=2
    con = conn(i,:);
    lm = lmm(i,:);
    [m, k, r] = TransientPlaneFrameElement(e, inertia, a, ...
        rho, 0, 0, nodes(con,:));
    M(lm, lm) = M(lm, lm) + m;
    K(lm, lm) = K(lm, lm) + k;
    R(lm) = R(lm) + r;
end

% Adjust for essential boundary conditions
dof = length(R);
df = setdiff(1:dof, debc);
Mf = M(df, df);
Kf = K(df, df);
Rf = R(df) - K(df, debc)*ebcVals;

% Setup and solve the resulting first order differential equations
u0 = zeros(length(Mf),1);
v0 = zeros(length(Mf),1);
[t,d] = ode23('FrameODE',[0,10],[u0; v0]);
plot(t,d(:,2)); xlabel('time'); ylabel('Disp');
title('Vertical displacement at node 2');
```

MatlabFiles\Chap8\FrameODE.m

```
function ddot = FrameODE(t, d)
% ddot = FrameODE(t, d)
% function to set up equations for a transient frame problem
global Mf Kf Rf
if t <= 0.6
    ft = 50*t + 20;
elseif t <= .85
        ft = 20;
    elseif t <= 1.2
            ft = 190 - 200*t;
        elseif t <= 1.4
```

```
            ft = 250*t - 350;
          else
            ft = 0;
          end
        end
      end
    end
    n=length(d);
    u = d(1:n/2);
    v = d(n/2+1:n);
    vdot = inv(Mf)*(Rf*ft - Kf*u);
    udot = v;
    % format short g
    % soln=[t, ft, u(2), udot(2), vdot(2)]
    ddot = [udot; vdot];
```

Executing the script file TransientFrameEx a plot of the vertical displacement at node 2. The script file can be modified to see other quantities of interest.

```
>> TransientFrameEx
```

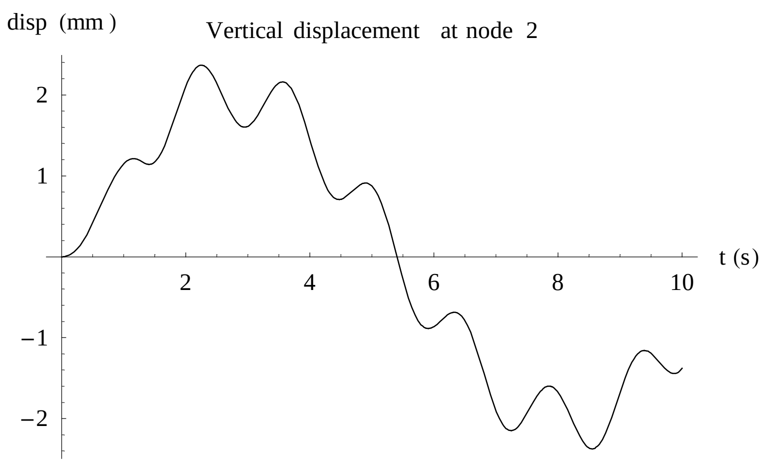


Figure 8.1. Displacement time history

Once the nodal displacements are known, any other quantity of interest, say bending moments in elements, can easily be computed using the same functions as those used in the static analysis case. Since the displacements are functions of time, obviously the element quantities are also functions of time.