HW 7, problems 6 and 7, matlab solution

Nathan Barton

October 20, 1999

1 Problem 6

Just the `matlab` code for the function is given here.

```
function [tau_max, eigval] = stress(sigma)
%
% given stress components as a 3x3 matrix, find the principal stresses
% and the maximum shear stress
%
[eig_vec, eigval_matx] = eig(sigma);
eigval = diag(eigval_matx);
tau_max = (max(eigval)-min(eigval))*0.5;
disp(sprintf('principal stresses are %16.8e %16.8e %16.8e\n', ...
eigval(1), eigval(2), eigval(3)));
disp(sprintf('maximum shear stress is %16.8\n',tau_max));
```

2 Problem 7

2.1 Driver file

`fan_driver.m`

```
% some of the values that are set in this file are not used until later
% parts of the problem

global r r2 rg lthg;
global iwk iwu1 iwu2 ium;
global w_data;
global gc g_by_gc;
global m ai;

% figure counter
ifig = 0;

% some of geometry of linkage
%
% lengths of bars in linkage (meters)
r = 0.01*[10.0 40.0 30.0 40.0]';
r2 = r*r';
% angle (relative to local coordinates on link) to the center of mass
```

```
lthg = [0.0 0.0 pi/6]';
% distance to centers of mass from starting point of link
rg = [r(1)*0.5 r(2)*0.5 r(3)/cos(lthg(3))]';

% masses, moments of inertia
m = [0.0 0.0 10.0]'; % in kg
% moment of inertia for uniform distribution of mass over links
ai = zeros(3,1);
ai(1) = m(1)*r2(1,1)/12.0;
ai(2) = m(2)*r2(2,2)/12.0;
diam_fan = 1.0;
ai(3) = m(3)*diam_fan*diam_fan/16.0;

% gravitation
g       = [0.0 9.8]'; % direction 2 is down, in m/s^2
gc      = 1.0; % metric, trust mass to be in kg
g_by_gc = g/gc;

th1_init = atan2(3,4);
th_init = fan_angles(r,r2,th1_init)

% data for initial angular velocities
w_data = [2.0*pi 0.0 0.0]';
% indices for known (iwk) and unknown (iwu*) angular velocities
% and index for unknown moment (ium)
iwk = 1; iwu1 = 2; iwu2 = 3; ium = 1;

tstart = 0.0; tstop = 2.0*pi/abs(w_data(iwk)); nt = 101;
tspan = linspace(tstart, tstop, nt);
dtime = (tstop - tstart)/(nt-1);



options = odeset('AbsTol',1e-10);
[t, th_num] = ode45('fan_solve',tspan,th_init,options);

p_num = zeros(nt,2,4);
for it = 1:nt;
p_num(it,:,:) = linkage_pos(th_num(it,:),r);
end;

ifig = ifig+1; figure(ifig);
plot(...
p_num(:,1,1),p_num(:,2,1),'*', ...
p_num(:,1,2),p_num(:,2,2), ...
p_num(:,1,3),p_num(:,2,3), ...
p_num(:,1,4),p_num(:,2,4),'*')
axis('equal');
xlabel('horizontal position'); ylabel('vertical position');
title('positions of pins, from numerical integration of angular velocities');
```

```
% get approximate maxiumum angle of tilt of fan blades from vertical
max_th3_num = max(th_num(:,3));
max_tilt_num = max_th3_num + pi/2;
disp(sprintf('max tilt of blades from vertical degrees is about %3.0f degrees\n', ...
     max_tilt_num*180/pi));
%
```

2.2 output from driver file

```
>> fan_driver
fan_driver

th_init =

    0.6435
    0.6435
   -1.5708

max tilt of blades from vertical degrees is about  42 degrees
```

2.3 m files

fan_solve.m

```
function [thdot] = fan_solve(t, th)
%
% given angles of components, find:
% thdot  -- angular velocities
%
global r r2 rg lthg;
global iwk iwu1 iwu2 ium;
global w_data;
global gc g_by_gc;
global m ai;
%
nlink = 3;
if (length(th) ~= 3);
disp('error, th of wrong length');
return;
end;
%
% get unit vectors
[nvec, tvec] = linkage_vec(th);
%
% solve for angular velocitiess
A = [r(iwu1)*tvec(:,iwu1) r(iwu2)*tvec(:,iwu2)];
y = [w_data(iwk)*r(iwk)*tvec(:,iwk)];
x = -A\y;
```

```
%
% set values in thdot
thdot = zeros(3,1);
thdot(iwk)  = w_data(iwk);
thdot(iwu1) = x(1);
thdot(iwu2) = x(2);
%
%
```

---

linkage_vec.m

```
function [nvec,tvec] = linkage_vec(th)
%
% get unit vectors along links (that is along direction given by angles)
%
nlink = length(th);
%
nvec = zeros(2,nlink);
%
for i_link = 1:nlink;
nvec(:,i_link) = [cos(th(i_link)) sin(th(i_link))]';
end;
%
if nargout > 1;
  %
  % also get vectors along k ^ n (where ^ indicates cross product)
  %
  tvec = zeros(2,nlink);
  %
  for i_link = 1:nlink;
tvec(:,i_link) = [-sin(th(i_link)) cos(th(i_link))]';
  end;
  %
end;
```

---

fan_angles.m

```
function th=fan_angles(r,r2,th1)
%
% This solution is much more complex than you need, given that the
% initial configuration puts the links in a right triangle; Note that the
% solution below works over a full cycle of the linkage -- giving the
% angles of all links if the angle of link 1 is known.
%
% calculate angles of components in 4 bar linkage
% r  :  lengths of components
% r2 : r'*r, lengths of compoeenents squared
%
```

```
% assumes that angle of ground link (link 4) is zero and angle of
% first link (theta_1) is given; assumes that th3 is in range 0 to pi
%

% use dirty (and expensive) trick to get th4 in the range 0 to pi
gam1 = acos(cos(th1));
%
[rm, rm2] = law_cos_len(r2(1,1), r2(4,4), r2(1,4), gam1);
gam2      = law_cos_ang(r2(4,4), r2(1,1), rm2, r(1)*rm);
gam3      = pi - gam1 - gam2;
%
gam4      = law_cos_ang(rm2, r2(2,2), r2(3,3), r2(2,3));
gam5      = law_cos_ang(r2(3,3), r2(2,2), rm2, r(2)*rm);
%
gam8      = pi - gam4 - gam5;
if (sin(th1) > 0.0e0)
gam6      = gam8 + gam3;
else
gam6      = gam8 - gam3;
end
th3 = -gam6;
%
%[rn, rn2] = law_cos_len(r2(4,4), r2(3,3), r2(4,3), gam6);
%gam7      = law_cos_ang(rn2, r2(1,1), r2(2,2), r(1,2));
%
th2 = pi - gam4 - gam6;
%
% set values
%
th = [th1 th2 th3]';
```

For `law_cos_len.m` and `law_cos_ang.m`, see previous solutions.

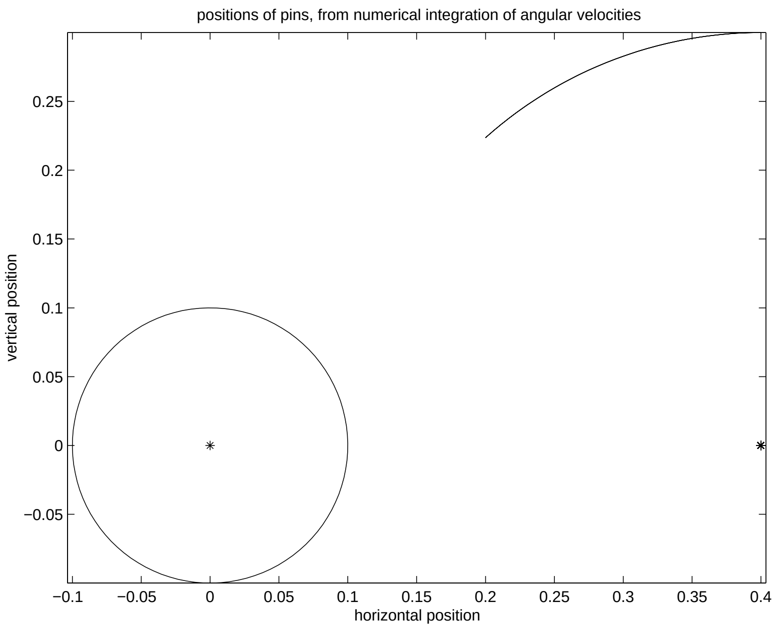

Figure 1: positions