9.

Computer programs

9.1. Programs related to the study of the Gibraltar bridge without wings.

9.1.1. Main_Program_GIBRALTAR.m

```
%**************************************************************************
%  Main_Program_GIBRALTAR.m
%  It was built to search the critical wind speed
%  for certain dynamical characteristics of a bridge deck
%  This program was adapted from a source program developed by K. Wilde
%  ADAPTED FOR 3 LAMBDAS
%**************************************************************************
DATA_GIBRALTAR    %  Call dataGibratar
U=0.;                   %  Loop starts from zero wind velocity;
flag=1;
while flag==1
U=U+0.01;          %  Delta U = 0.01 is the step;
%  State matrix A
A = ASSEMBLE_GIBRALTAR(U,Bd,A1g,A0g,Dg,Fg,Gg,Md,Cd,Kd);
test = real(eig(A));
for i=1:7       % Is there any eigenvalue with a positive real part?
if test(i) > 0  % The critical speed is found as soon as the first
flag=0;         % eigenvalue with a positive real root is detected
              % (from one to 7 possible eigenvalues).
%**************************************************************************
Uc=U                      % Critical flutter wind speed was found
end
end
end
disp(['Results of program "Main_Program_Gibraltar" ']);
```

```
disp(['Critical wind speeed Uc = ', num2str(Uc), 'm/s']);
disp(['State Matrix 7x7 :'])
A
[Ome,Ksi]=eig(A)
damp(A)
PLOTS_OME_DAMP_GIBRALTAR
```

9.1.2. DATA_GIBRALTAR

```
% DATAGIBRALTAR
ro_a = 1.225;           % air density
%
%  Deck characteristic for a scaled model 1:200
%
Bd = 0.3 ;              % bridge width
mh= 0.9875 ;            % mass
I_a = 0.0167;           % rotational moment
delta_h = 2*pi*0.003;    % damping log. decrement of h mode
delta_a = 2*pi*0.0015;   % damping log. decrement of a mode
ksi_h = 0.003; ksi_a = 0.0015;
fre_h = 200*0.383/(2*pi);       % frequency of h mode
fre_a = 200*0.509/(2*pi);       % frequency of a mode
ome_h = 200*0.383;ome_a = 200*0.509;
%
% Control wing characteristic
%
Bw1 = 0.1*Bd;             % width of wing nr 1
Bw2 = 0.1*Bd;             % width of wing nr 2
e1 =  0.5*Bd;             % position of wing 1 from center of deck
e2 =  0.5*Bd;             % position of wing 2 from center of deck
m11 = 1;                  % actuator 1 mass
m22 = 1;                                        % actuator 2 mass
ksi_1 = 0.7;              % actuator 1 damping ratio
ksi_2 = 0.7;              % actuator 2 damping ratio
ome_1 = 6*2*pi;           % nat. freq. of actu. 1
ome_2 = 6.01*2*pi;         % nat. freq. of actu. 2
Md = [(mh)*Bd 0 ; 0 I_a];
Cd = [2*ksi_h*ome_h*mh*Bd 0; 0 2*ksi_a*ome_a*I_a ];
```

```
Kd = [ome_h*ome_h*mh*Bd 0; 0 ome_a*ome_a*I_a];
%       Flutter derivatives
A0 = [0.3058600E+00  0.3093970E+01 ; 0.1373977E+01 -0.5468954E+00];
A1 = [0.2459046E+01 -0.1403846E+00; -0.2400085E+00 -0.2141308E+00];
Dd = [0.1021054E+02  0.1787100E+01 -0.4713930E+00;
        -0.4298823E+00  0.1149574E+01  0.1083369E+01];
Ed= [-0.1027703E-01 -0.3480843E+00 -0.1946212E+01;
         0.4017426E-01 -0.2610986E+00  0.2132517E+01]';
lamb1= 0.5164275E+00; lamb2= 0.1346202E+01; lamb3= 0.1971455E+01;
miBd = 0.5*ro_a*Bd;
Vf = miBd*[-1 0; 0 Bd];                    % Vf must be *U*U
A1g = [Bd*Vf*A1] ;                         % A1a = U*A1a
A0g = [Vf*A0] ;                            % A1a = U*U*A1a
Dg =[ Vf*Dd ];                                  % Da = U*U*Da
Fg = -diag([lamb1/Bd; lamb2/Bd;lamb3/Bd]);
Gg = [Ed/Bd];
```

9.1.3. PLOTS_OME_DAMP_GIBRALTAR

```
%*********************************************************************
% subrotine  rl_damp_condeGIBRALTAR(Bd,A1Gg,A0g,Dg,Fg,Gg,Ms,Cs,Ks)
%*********************************************************************
write_mes(' Ploting rootlocus of frequency and damping')
%  Root locus analysis of closed loop system
subplot(111)
clear graph
flag = 1;
Umin = 0.0;
Umax = 60.01; % max speed = 80 m/s = 288 km/h
deltaU = 0.01;
U = Umin;
fn = (Umax-U)/deltaU;
U_vec = [Umin:deltaU:Umax-deltaU]';
RE_ome = zeros(fn,7);
RE_ksi = zeros(fn,7);
for i=1:fn
A = ASSEMBLE_GIBRALTAR(U,Bd,A1g,A0g,Dg,Fg,Gg,Md,Cd,Kd);
 [Ome, Ksi] = damp(A);
```

```
if (any(real(Ksi) < 0.0)) && flag
Ucritico=U
flag = 0;
end
 n_ome = 7;
  U = U+deltaU;
  for j=1:n_ome
     RE_ome(i,j) = Ome(j);
     RE_ksi(i,j) = Ksi(j);
  end
end
axis('normal');
axis([Umin Umax -0.2 1.1]);
subplot(211); plot(U_vec,RE_ksi,'.')
title(['Plot of damping factors from ','Umin = ',num2str(Umin),...
' m/s to Umax = ',num2str(Umax),'m/s']); grid;
xlabel('Wind velocity U (m/s)')
ylabel('Damping factors Ksi')
subplot(212); plot(U_vec,RE_ome,'.')
axis([Umin Umax 0 150]);
title(['Plot of structural frequencies from ','Umin = ',...
num2str(Umin), 'm/s to Umax = ',num2str(Umax), 'm/s ']); grid;
xlabel('Wind velocity U (m/s)')
ylabel('Structural frequencies Omega (rad/s)')
ASSEMBLE_GIBRALTAR.m
%*************************************************************************
function Ap = ASSEMBLE_GIBRALTAR(U,Bd,A1g,A0g,Dg,Fg,Gg,Ms,Cs,Ks)
  A1s = U*A1g;   A0s = U*U*A0g;  Ds = U*U*Dg; Fs = U*Fg;
  Gs = U*Gg;
%   assembling state matrix
% U
Ap = [inv(Ms)*(-Cs+A1s) inv(Ms)*(-Ks+A0s) inv(Ms)*Ds;
     eye(2,2) zeros(2,2) zeros(2,3);
     zeros(3,2) Gs Fs];
```

9.1.4. write_mes.m

```
%*************************************************************************
```

```
function write_mes(message)
disp(' ')
disp(message)
disp(' ')
%*****************************************************************************
```

9.2. Programs related to the study of the Gibraltar bridge with stationary wings.

9.2.1. Main_Program GIBRALTAR.m

```
%*********************************************************************
%  Program rootlocus GIBRALTAR.m
%  It was built to search the critical wind speed
%  for certain dynamical characteristics of a bridge deck
%  This program was adapted from a source program developed by K. Wilde
%  ADAPTED FOR 3 LAMBDAS and stationary wings
DECK_AND_ST_WING_DATA
PREPROCESSOR
U=0.;                %  Loop starts from zero wind velocity;
flag=1;
while flag==1
U=U+0.10;          %  Delta U = 0.01 is the step;
%  State matrix A
A = asemb_conde(U,Bd,A1g,A0g,Dg,Fg,Gg,Ms,Cs,Ks);
test = real(eig(A));
for i=1:15       % Is there any eigenvalue with a positive real part?
if test(i) > 0  % The critical speed is found as soon as the first
flag=0;         % eigenvalue with a positive real root is detected
              % (from one to 7 possible eigenvalues).
Uc=U                      % Critical flutter wind speed was found
end
end
end
disp(['Results of program "RootlocusGibraltar" ']);
disp(['Critical wind speeed Uc = ', num2str(Uc), 'm/s']);
disp(['State Matrix 7x7 :'])
```

A

[Eigenvectors,Eigenvalues]=eig(A)

damp(A)

PLOT_ST_GIBRALTAR

9.2.2. DECK_AND_ST_WING_DATA

```
% deck and control wing data for GIBRALTAR
ro_a = 1.225;           % Air density
%*****************************************************************
% Deck characteristics - Scale model 1:200
%*****************************************************************
Bd = 60/200;            % bridge width
mh= 39500/(200^2);      % mass
I_a = 26700000/(200^4);  % rotational moment
%*****************************************************************
ksi_h = 0.0030;
ksi_a = 0.0015;
delta_h = 2*pi*ksi_h;    % damping log. decrement of h mode
delta_a = 2*pi*ksi_a;    % damping log. decrement of a mode
ome_h = 200*0.383;
ome_a = 200*0.509;
fre_h = ome_h/(2*pi);    % frequency of h mode
fre_a = ome_a/(2*pi);    % frequency of a mode
%                   Control wing characteristics
%*****************************************************************
%Bw1 = 0.1*Bd;    % width of wing nr 1 = 6m (it seems to big)
%Bw2 = 0.1*Bd;    % width of wing nr 2 = 6m (it seems to big)
%*****************************************************************
Bw1 = 0.05*Bd;    % width of wing nr 1 = 3m (it seems reasonable)
Bw2 = 0.05*Bd;    % width of wing nr 2 = 3m (it seems reasonable)
%*****************************************************************
%Bw1 = 0.0333*Bd;    % width of wing nr 1 = 2m (it seems too small)
%Bw2 = 0.0333*Bd;    % width of wing nr 2 = 2m (it seems too small)
%*****************************************************************
e1 =  0.5*Bd;     % position of wing 1 from center of deck
e2 =  0.5*Bd;     % position of wing 2 from center of deck
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
m11 = 1;          % actuator 1 mass
m22 = 1;                    % actuator 2 mass
ksi_1 = 0.7;      % actuator 1 damping ratio
ksi_2 = 0.7;      % actuator 2 damping ratio
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%No
tice that in Wilde's example, f_w = 6 Hz (1:150 scale model).
%In the present case, f_w = 20 Hz (1:200 model)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%ome
_1 = 20*2*pi;        % nat. freq. of actu. 1
ome_2 = 20.01*2*pi;     % nat. freq. of actu. 2
%****************************************************************
```

9.2.3. PREPROCESSOR.m

```
%  Data preprocessor for   'condek'
%  GIBRALTAR
disp(' ')
disp(' processing  data ... ')
disp(' ')
%
%           deck characteristics
%
ksi_h = delta_h/(2*pi);
ksi_a = delta_a/(2*pi);
%
ome_h = fre_h*2*pi;
ome_a = fre_a*2*pi;
%
M = [mh*Bd 0; 0 I_a];
C = [2*ksi_h*ome_h*mh*Bd 0; 0 2*ksi_a*ome_a*I_a];
K = [ome_h*ome_h*mh*Bd 0; 0 ome_a*ome_a*I_a];
%
%       aerodynamic finite state model matrices of deck
%
A0 = [0.3058600E+00  0.3093970E+01 ; 0.1373977E+01 -0.5468954E+00];
A1 = [0.2459046E+01 -0.1403846E+00 ; -0.2400085E+00 -0.2141308E+00];
Dd = [0.1021054E+02  0.1787100E+01 -0.4713930E+00;
         -0.4298823E+00  0.1149574E+01  0.1083369E+01];
Ed= [-0.1027703E-01 -0.3480843E+00 -0.1946212E+01;
          0.4017426E-01 -0.2610986E+00  0.2132517E+01]';
lamb1= 0.5164275E+00; lamb2= 0.1346202E+01 ; lamb3= 0.1971455E+01;
%
%           control wing characteristics
%
Mw = [m11 0; 0 m22];
Cw = [2*ksi_1*ome_1 0; 0 2*ksi_2*ome_2];
Kw = [ome_1*ome_1 0; 0 ome_2*ome_2];
%
Ms = [M zeros(size(M)); zeros(size(M)) Mw];
Cs = [C zeros(size(C)); zeros(size(C)) Cw];
Ks = [K zeros(size(K)); zeros(size(K)) Kw];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Data for Bw = 6m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
lamb1w = 0.4898931E-01;
% lamb2w = 0.2672425E+00;
% A0w = [0.5742289E+00  0.4010688E+01 ; 0.1435572E+00  0.1002672E+01];
% A1w = [0.3817801E+01  0.2337816E+01 ; 0.9544502E+00 -0.2009442E+00];
% Dw = [0.5719828E+01  0.2422333E+01; 0.1429957E+01  0.6055831E+00];
% Ew = [-0.1395796E-03 -0.6151422E-01; 0.2926502E-02  0.2082658E+00]';
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Data for Bw = 3m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%lamb
1w = 0.3500000E-01 ; lamb2w = 0.1320000E+00;
A0w = [ 0.2098244E+00  0.5221252E+01 ; 0.5251235E-01  0.1305108E+01];
A1w = [ 0.4615840E+01 -0.5292162E+00 ; 0.1153550E+01 -0.9164623E+00];
```

```
Dw = [  0.4798194E+01  0.1821348E+01 ; 0.1197957E+01  0.4559187E+00];
Ew = [  0.1471817E-05 -0.1512478E-01 ; 0.1524671E-02  0.5906955E-01]';
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Td =
[1 0 0 0; 0 1 0 0];              % transformation matrices
Tw1 = [Bd/Bw1 -e1/Bw1 0 0; 0 1 1 0];    % fom q(4) to q(2) local
Tw2 = [Bd/Bw2 e2/Bw2 0 0; 0 1 0 1];
%
miBd = 0.5*ro_a*Bd; miBw1 = 0.5*ro_a*Bw1; miBw2 = 0.5*ro_a*Bw2;
%
Vf = miBd*[-1 0; 0 Bd];                  % Vf must be *U*U
Vfw1 = miBw1*[-1 0; 0 Bw1];              % Vf must be *U*U
Vfw2 = miBw2*[-1 0; 0 Bw2];              % Vf must be *U*U
%
S = [1 0 1 0 1 0; 0 1 -e1 1 e2 1];     % transformation Fall to Ftot
%
A1g = S*[ Bd*Vf*A1*Td;
       Bw1*Vfw1*A1w*Tw1;
       Bw2*Vfw2*A1w*Tw2] ;  % A1a = U*A1a
%
A0g = S*[Vf*A0*Td;
      Vfw1*A0w*Tw1;
      Vfw2*A0w*Tw2] ;        % A1a = U*U*A1a
%
Dg = S*[Vf*Dd zeros(size(Dw))  zeros(size(Dw));
     zeros(size(Dd)) Vfw1*Dw zeros(size(Dw));
     zeros(size(Dd)) zeros(size(Dw)) Vfw2*Dw] ; % Da = U*U*Da
%
Fg = -diag([lamb1/Bd; lamb2/Bd; lamb3/Bd;
         lamb1w/Bw2; lamb2w/Bw1;
         lamb1w/Bw2; lamb2w/Bw2]);
%
Gg = [Ed*Td/Bd; Ew*Tw1/Bw1; Ew*Tw2/Bw2];
%
% ------ end of data preprocessor  --------
```


9.2.4. PLOT_ST_GIBRALTAR.m

```
%Rootlocus for stationary wings
% subrotine  rl_damp_conde(B,ro_a,A1a,A0a,Da,Kw,Fg,Gg,Ms,Cs,Ks,Kp,Ug)
write_mes(' Ploting rootlocus of frequency and damping')
%I15=eye(15,15);
%  Root locus analysis of open loop system
DECK_AND_ST_WING_DATA
PREPROCESSOR
subplot(111)
clear graph
flag = 1;
Umin = 0.0;
```

```
Umax = 100.01;
deltaU = 0.01;
U = Umin;
fn = (Umax-U)/deltaU;
U_vec = (Umin:deltaU:Umax-deltaU)';
RE_ome = zeros(fn,15);
RE_ksi = zeros(fn,15);
for i=1:fn
A = asemb_conde(U,Bd,A1g,A0g,Dg,Fg,Gg,Ms,Cs,Ks);
[ome, ksi] = damp(A);
 n_ome = 15;
  U = U+deltaU;
  for j=1:n_ome
  RE_ome(i,j) = ome(j);
  RE_ksi(i,j) = ksi(j);
  end
  %vetorA=eig(A);
  %for j=1:n_ome
 % if imag(vetorA(j))~=0
  %RE_ome(i,j) = sqrt(real(vetorA(j))^2+imag(vetorA(j))^2);
 % RE_ome(i,j) = ome(j);
  %else
  %RE_ome(i,j) = -10.0;
  %end
  %end
end
axis('normal');
subplot(211); plot(U_vec,RE_ksi,'.')
axis([Umin Umax -0.2 1.1]);
title(['Root locus of damping factors from ','Umin = ',...
num2str(Umin), ' m/s to Umax = ',num2str(Umax),' m/s']);
grid;
xlabel('Wind velocity U (m/s)');
ylabel('Damping factors Ksi');
subplot(212); plot(U_vec,RE_ome,'.')
axis([Umin Umax 0 160]);
title(['Root locus of structural frequencies from ','Umin = ',...
```

```
num2str(Umin), ' m/s to Umax = ', num2str(Umax),' m/s']);
grid;
xlabel('Wind velocity U (m/s)'); grid;
ylabel('Structural frequencies Omega (rad/s)');grid;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

9.2.5. asemb_conde.m

```
function Ap = asemb_conde(U,Bd,A1g,A0g,Dg,Fg,Gg,Ms,Cs,Ks)
%
  A1s = [U*A1g; zeros(size(A1g))];
  A0s = [U*U*A0g; zeros(size(A0g))];
  Ds = [U*U*Dg; zeros(size(Dg))];
  Fs = U*Fg;
  Gs = U*Gg;
%
%   Assembling state matrices
%
  Ap = [inv(Ms)*(-Cs+A1s) inv(Ms)*(-Ks+A0s) inv(Ms)*Ds;
     eye(4,4) zeros(4,4) zeros(4,7);
     zeros(7,4) Gs Fs];
 asemb_conde.m
```

9.2.6. write_mes.m

```
function write_mes(message)
disp(' ')
disp(message)
disp(' ')
```

9.3. Programs related to the study of the Gibraltar bridge with moving wings.

9.3.1. Main_Program_GIBRALTAR.m

```
%*****************************************************************
% Active control of bridge deck - Variable gain approach
%*****************************************************************
clear
subplot(111); clear graph;
write_mes(' reading data ...')
%*****************************************************************
Deck_and_Wing_Data - Preprocessor
%*****************************************************************
write_mes(' calculating  matrices  A, B ... ')
I6=eye (6,6); I15 = eye(15,15);
B = [[zeros(size(Kw)); Kw]; zeros(4,2); zeros(7,2)];
Call = eye(15,15); C =[Call(1:2,:);Call(5:8,:)]; D = zeros(6,2);
U_p =  [0 40 49 58 67 76]';     % operating points
fr_p = [1 50. 100.0 300.0 500.0 925.]';
q_p =  [0.1 1. 5. 10.0  30.0 68.0]';
fw_p =[0.1 200. 175.0 160.0 150.0 100.0]';
poly_qp = polyfit(U_p, q_p, 3);
poly_fr = polyfit(U_p, fr_p, 2);
poly_fw = polyfit(U_p, fw_p,3);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Definition of the weighting functions as polynomials
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 subplot(311)
 u = [0:0.5:80]';
 fwip = polyval(poly_fw,u);
 plot(U_p, fw_p, 'o', u,fwip,'-')
 title(['Approximation of [fw-p(U)]:'  'Polynom = ',num2str(poly_fw(1)),...
    ' x^3  ',     num2str(poly_fw(2)),' x^2  ',num2str(poly_fw(3)),...
    ' x  ',num2str(poly_fw(4)),' ']); ...
    xlabel('U  wind speed');ylabel('[fw-p(U])'); grid
```

```
u = [0:0.5:80]';
frip = polyval(poly_fr,u);
subplot(312)
plot(U_p, fr_p, 'o', u,frip,'-')
title(['Approximation of [fr-p(U)]:'  'Polynom = ',num2str(poly_fr(1)),...
   ' x^2  ',num2str(poly_fr(2)),' x^1  ',num2str(poly_fr(3)),' ']);...
   xlabel('U  wind speed');ylabel('[fr-p(U)]'); grid
u = [0:0.5:80]';
fqip = polyval(poly_qp,u);
subplot(313)
plot(U_p, q_p, 'o', u,fqip,'-')
title(['Approximation of [q_p(U)] on Heaving:'  '  Polynom = ',...
   num2str(poly_qp(1)),' x^3  ',num2str(poly_qp(2)),' x^2  ',...
   num2str(poly_qp(3)),' x  ',num2str(poly_qp(4)),' ']);
xlabel('U  wind speed');ylabel('[q_p-p(U)] = Q on heaving'); grid
pause
%*********************************************************************
Ji = 0; H1i = 0; H2i = 0;
p_big = 50; p_end = 80; dp = 10;
p = [p_big:dp:p_end]'
nr_p = length(p);
disp(' setting initial dKi ...')
for j=1:nr_p;
  j;
  pj = p(j);
  [Qj, Rj, frj] = updateQR(pj, poly_qp, poly_fr);
  [Gj, Cj] = updateGC(pj,C,l6);
  A = asemb_conde(pj,Bd,A1g,A0g,Dg,Fg,Gg,Ms,Cs,Ks);
  %Ki=[1.0072e-005 -0.00013364 -0.016202 0.025298 0.011811 -0.014926...
  %    0.010369     0.021369    1.7632  -1.4496  -0.47056   0.49153;
  % 0.00025394 -3.7763e-005 0.0036974 -0.036227  -0.010416  0.011387...
  % -0.033389   0.00015955  -0.49006 3.1879  0.50664 0.010758];
Ki=[9.4273e-006 -0.00015221 -0.016647  0.024445 0.010493 -0.016121...
    0.010496 0.022489 1.7937 -1.3989 -0.39166 0.55875;
    0.00025054 -3.2332e-005  0.0038904  -0.035306 -0.0097225  0.011928...
   -0.033179  -0.00016458  -0.49878     3.1332    0.46887    -0.022267];
Kinicial=Ki;
```

```
  Aj = A-B*Ki*Cj;
  Lj = lyap(Aj,  I15);
  Pj = lyap(Aj',Aj,Qj+Cj'*Ki'*Rj*Ki*Cj);
  if any(eig(Lj) < -eps) || any(eig(Pj) < -eps)
            error('system is unstable for initial gain')
  end
  fwj =  polyval(poly_fw,pj); if fwj <= 0 fwj = 0.001; end;
  Ji = Ji+trace(fwj*Pj);
  H1i = H1i + fwj*frj*Cj*Lj*Cj';
  H2i = H2i + fwj*B'*Pj*Lj*Cj';
end
  dKi = H2i*inv(H1i)-Ki;
%*********************************************************************
disp(' minimizing varK ...');
%*********************************************************************
al = 1;
dJ = 1e8;
tol = 10;
al_tol = 1e-3;
iter_tol =36;
iter = 0;
improv = 0;
z = 1.1;
while abs(dJ) > tol
  iter = iter+1;
  % disp (' '); disp(['iter= ', num2str(iter), ' start of iteration'])
  Ji1 = 0; H1in = 0; H2in = 0;
  Ki1 = Ki+al*dKi;
  for j=1:nr_p
    pj = p(j);
  % disp(['   ','pj= ', num2str(pj)])
    [Qj, Rj, frj] = updateQR(pj, poly_qp, poly_fr);
    [Gj, Cj] = updateGC(pj,C,I6);
    A = asemb_conde(pj,Bd,A1g,A0g,Dg,Fg,Gg,Ms,Cs,Ks);
    Aj = A-B*Ki1*Cj;
    Lj = lyap(Aj,I15);                  %              A*X + X*A' = -C
    Pj = lyap(Aj',Aj,Qj+Cj'*Ki1'*Rj*Ki1*Cj);    %                  A*X + X*B = -C
```

```
    if any(eig(Lj) < -eps) || any(eig(Pj) < -eps)
      Ji1 = Ji+tol+1;
      break;
    end
    fwj =  polyval(poly_fw,pj); if fwj <= 0 fwj = 0.001; end;
    Ji1 = Ji1+trace(fwj*Pj);
    H1in = H1in + fwj*frj*Cj*Lj*Cj';  %fwj*
    H2in = H2in + fwj*B'*Pj*Lj*Cj';   %fwj*
  end
    if Ji1 < Ji
    % disp([' ',' Improvment']);
    improv = improv+1;
    iter;
    dJ = Ji1-Ji; disp(['   ','iter=', num2str(iter)...
        ,'    improvment ','   dJ=',num2str(dJ)]);
    Ki=Ki1;
    Ki1 = H2in*inv(H1in);
    dKi = Ki1-Ki;
    Ji=Ji1;
  else
    al = al/z;
    dJ = tol+1;
  disp(['     ','iter=', num2str(iter), ' dJ positive'])
   end
  if al < al_tol, break; end;
% if iter > iter_tol break; end;
end
varK = Ki;
disp(['   J = ',num2str(Ji)])
disp('Kinicial')
Kinicial
disp('Kfinal')
varK
disp('end of gain optimization');
pause
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Impulses
```

Movies
Plots
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

9.3.2. Deck_and_Wing_Data

```
% deck and control wing data for GIBRALTAR
ro_a = 1.225;           % Air density
%**************************************************************
% Deck characteristics - Scale model 1:200
%**************************************************************
Bd = 60/200;            % bridge width
mh= 39500/(200^2);      % mass
I_a = 26700000/(200^4);  % rotational moment
%**************************************************************
ksi_h = 0.0030; ksi_a = 0.0015;
delta_h = 2*pi*ksi_h;    % damping log. decrement of h mode
delta_a = 2*pi*ksi_a;    % damping log. decrement of a mode
ome_h = 200*0.383; ome_a = 200*0.509;
fre_h = ome_h/(2*pi);    % frequency of h mode
fre_a = ome_a/(2*pi);    % frequency of a mode
%                    Control wing characteristics
%*****************************************************************
%Bw1 = 0.1*Bd;    % width of wing nr 1 = 6m (it seems to big)
%Bw2 = 0.1*Bd;    % width of wing nr 2 = 6m (it seems to big)
%******************************************************************
Bw1 = 0.05*Bd;    % width of wing nr 1 = 3m (it seems reasonable)
Bw2 = 0.05*Bd;    % width of wing nr 2 = 3m (it seems reasonable)
%**************************************************************
%Bw1 = 0.0333*Bd;    % width of wing nr 1 = 2m (it seems too small)
%Bw2 = 0.0333*Bd;    % width of wing nr 2 = 2m (it seems too small)
%**************************************************************
e1 =  0.5*Bd;     % position of wing 1 from center of deck
e2 =  0.5*Bd;     % position of wing 2 from center of deck
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
m11 = 1;          % actuator 1 mass
m22 = 1;                    % actuator 2 mass
ksi_1 = 0.7;      % actuator 1 damping ratio
```

```
ksi_2 = 0.7;      % actuator 2 damping ratio
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Notice that in Wilde's example, f_w = 6 Hz (1:150 scale model).
%In the present case, f_w = 20 Hz (1:200 model)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ome_1 = 20*2*pi;        % nat. freq. of actu. 1
ome_2 = 20.01*2*pi;     % nat. freq. of actu. 2
%*****************************************************************
```

9.3.3. Preprocessor

```
%  Data preprocessor for GIBRALTAR
disp(' ')
disp(' processing  data ... ')
disp(' ')
%*****************************************************************
%  Deck characteristics
%*****************************************************************
ksi_h = delta_h/(2*pi); ksi_a = delta_a/(2*pi);
ome_h = fre_h*2*pi; ome_a = fre_a*2*pi;
M = [mh*Bd 0; 0 I_a];
C = [2*ksi_h*ome_h*mh*Bd 0; 0 2*ksi_a*ome_a*I_a];
K = [ome_h*ome_h*mh*Bd 0; 0 ome_a*ome_a*I_a];
%*****************************************************************
%  Aerodynamic finite state model matrices of deck
%*****************************************************************
A0 = [0.3058600E+00 0.3093970E+01 ; 0.1373977E+01 -0.5468954E+00];
A1 = [0.2459046E+01 -0.1403846E+00; -0.2400085E+00 -0.2141308E+00];
Dd = [0.1021054E+02  0.1787100E+01 -0.4713930E+00;
         -0.4298823E+00  0.1149574E+01  0.1083369E+01];
Ed= [-0.1027703E-01 -0.3480843E+00 -0.1946212E+01;
          0.4017426E-01 -0.2610986E+00  0.2132517E+01]';
lamb1= 0.5164275E+00; lamb2= 0.1346202E+01 ; lamb3= 0.1971455E+01;
%*****************************************************************
%  Control wing characteristics
%*****************************************************************
Mw = [m11 0; 0 m22];
Cw = [2*ksi_1*ome_1 0; 0 2*ksi_2*ome_2];
```

```
Kw = [ome_1*ome_1 0; 0 ome_2*ome_2];
Ms = [M zeros(size(M)); zeros(size(M)) Mw];
Cs = [C zeros(size(C)); zeros(size(C)) Cw];
Ks = [K zeros(size(K)); zeros(size(K)) Kw];
%*****************************************************************
% Data for Bw = 3m
%*****************************************************************
lamb1w = 0.3500000E-01 ; lamb2w = 0.1320000E+00;
A0w = [0.2098244E+00  0.5221252E+01 ; 0.5251235E-01  0.1305108E+01];
A1w = [0.4615840E+01 -0.5292162E+00 ; 0.1153550E+01 -0.9164623E+00];
Dw = [0.4798194E+01  0.1821348E+01 ; 0.1197957E+01  0.4559187E+00];
Ew = [0.1471817E-05 -0.1512478E-01 ; 0.1524671E-02  0.5906955E-01]';
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Td = [1 0 0 0; 0 1 0 0]; Tw1 = [Bd/Bw1 -e1/Bw1 0 0; 0 1 1 0];
Tw2 = [Bd/Bw2 e2/Bw2 0 0; 0 1 0 1];
miBd = 0.5*ro_a*Bd; miBw1 = 0.5*ro_a*Bw1; miBw2 = 0.5*ro_a*Bw2;
Vf = miBd*[-1 0; 0 Bd]; Vfw1 = miBw1*[-1 0; 0 Bw1];
Vfw2 = miBw2*[-1 0; 0 Bw2];
S = [1 0 1 0 1 0; 0 1 -e1 1 e2 1];
A1g = S*[ Bd*Vf*A1*Td;  Bw1*Vfw1*A1w*Tw1;  Bw2*Vfw2*A1w*Tw2] ;
A0g = S*[Vf*A0*Td; Vfw1*A0w*Tw1;  Vfw2*A0w*Tw2] ;
Dg = S*[Vf*Dd zeros(size(Dw))  zeros(size(Dw));
     zeros(size(Dd)) Vfw1*Dw zeros(size(Dw));
     zeros(size(Dd)) zeros(size(Dw)) Vfw2*Dw];
Fg = -diag([lamb1/Bd; lamb2/Bd; lamb3/Bd;
          lamb1w/Bw2; lamb2w/Bw1;
          lamb1w/Bw2; lamb2w/Bw2]);
Gg = [Ed*Td/Bd; Ew*Tw1/Bw1; Ew*Tw2/Bw2];
%*****************************************************************
```

9.3.4. Impulses

```
%*************************************************************
% Plot of impulses
%*************************************************************
write_mes(' Beginning of plots ')
begintime = 0; fintime = 0.5; numpoints = 500;
T = [linspace(begintime,fintime,numpoints)]';    % time vector
```

```
Uext = [zeros(size(2*pi*T))  zeros(size(T))];    % force - time history
pp =  [50  60  70   80]';                        % discret wind speed
nr_p = length(pp);
for j=1:nr_p
  pj = pp(j);
  [Gj, Cj] = updateGC(pj,C,I6);
  A = asemb_conde(pj,Bd,A1g,A0g,Dg,Fg,Gg,Ms,Cs,Ks);
  Ac = A-B*varK*Cj
  %pause
    X0 =[0 1*inv(M(2,2)) 0 0  0 0 0 0  0 0 0 0 0 0 0]'
eval(['[y_c,x_res_c',num2str(10*pj),'] = lsim(Ac,B,C,D,Uext,T,X0);'])
end
% RESPONSE TO AN INITIAL STATE X0
% Initial state X0 (impulse on (dalfa/dt))
subplot(411);
plot(T,x_res_c500(:,5), '-', T,x_res_c500(:,6),'--'...
    ,T,x_res_c500(:,7),'-.',T,x_res_c500(:,8), ':')
title(['Time Response to an unit impulse d(alfa)/dt=1/I_a, at U = '...
    ,num2str(50)]);
xlabel('Time  (sec)');ylabel('Dimensionless Amplitude'); grid
subplot(412);
plot(T,x_res_c600(:,5), '-', T,x_res_c600(:,6),'--'...
    , T,x_res_c600(:,7),'-.',T,x_res_c600(:,8), ':')
title(['Time Response to an unit impulse d(alfa)/dt=1/I_a, at U = '...
    ,num2str(60)]);
xlabel('Time  (sec)');ylabel('Dimensionless Amplitude'); grid
subplot(413);
plot(T,x_res_c700(:,5), '-', T,x_res_c700(:,6),'--',...
    T,x_res_c700(:,7),'-.', T,x_res_c700(:,8), ':')
title(['Time Response to an unit impulse d(alfa)/dt=1/I_a, at U = '...
    ,num2str(70)]);
xlabel('Time(sec)');ylabel('Dimensionless Amplitudes'); grid
subplot(414);
plot(T,x_res_c800(:,5), '-', T,x_res_c800(:,6),'--',...
    T,x_res_c800(:,7),'-.',T,x_res_c800(:,8), ':')
title(['Time Response to an unit impulse d(alfa)/dt=1/I_a, at U ='...
    ,num2str(80)]);
```

```
xlabel('Time  (sec)');ylabel('Dimensionless Amplitude'); grid
pause
subplot(411);
plot(T,x_res_c500(:,1), '-', T,x_res_c500(:,2),'--',...
   T,x_res_c500(:,3),'-.',T,x_res_c500(:,4), ':')
title(['Time Response to an unit impulse d(alfa)/dt=1/I_a, at U = '...
   ,num2str(50)]);
xlabel('Time  (sec)');ylabel('Dimensionless Velocities'); grid
subplot(412);
plot(T,x_res_c600(:,1), '-', T,x_res_c600(:,2),'--',...
   T,x_res_c600(:,3),'-.', T,x_res_c600(:,4), ':')
title(['Time Response to an unit impulse d(alfa)/dt=1/I_a, at U ='...
   ,num2str(60)]);
xlabel('Time  (sec)');ylabel('Dimensionless Velocities'); grid
subplot(413);
plot(T,x_res_c700(:,1), '-', T,x_res_c700(:,2),'--'...
   , T,x_res_c700(:,3),'-.', T,x_res_c700(:,4), ':')
title(['Time Response to an unit impulse d(alfa)/dt=1/I_a, at U ='...
   ,num2str(70)]);
xlabel('Time  (sec)');ylabel('Dimensionless Velocities'); grid
subplot(414);
plot(T,x_res_c800(:,1), '-', T,x_res_c800(:,2),'--'...
   , T,x_res_c800(:,3),'-.',T,x_res_c800(:,4), ':')
title(['Time Response to an unit impulse d(alfa)/dt=1/I_a, at U = '...
   ,num2str(80)]);
xlabel('Time  (sec)');ylabel('Dimensionless Velocities'); grid
pause
```

9.3.5. Movies

```
% subrotine  rl_dek_mov(B,ro_a,A1a,A0a,Da,Kw,Fg,Gg,Ms,Cs,Ks,Kp,varK)
%
disp(' ')
disp(' ploting root locus film ... ')
disp(' ')
%  Eigenvalue analysis of closed loop system
%
clear graph
```

```
I6 = eye(6,6); I14 = eye(14,14);I15 = eye(15,15);
Umin = 1.;
Umax = 81.;
deltaU = 5.;
flag = 1;
U = Umin;
fn = (Umax-U)/deltaU;
for i=1:fn
  [Gj, Cj] = updateGC(U,C,I6);
  A = asemb_conde(U,Bd,A1g,A0g,Dg,Fg,Gg,Ms,Cs,Ks);
  Ac = A-B*varK*Cj;
  Eg = eig(A-B*varK*Cj);     % for closed loop analysis
  if (any(real(Eg) > 0.0)) && flag
     disp(U)
     flag = 0;
  end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
subplot(4,4,i)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  plot(Eg,'o'); title([' U = ',num2str(U)]); grid
  [n, n1] = size(Eg);
  U = U+deltaU;
  for j=1:n
     REg(i,j) = Eg(j);
  end
end
pause
subplot(111)
plot(REg(1,1:n),'o');
plot(REg(1:fn,1:n),'.')
title([' Eigenvalues ','Umin = ',num2str(Umin), '   Umax = ',num2str(Umax)])
grid; axis;
% output for caleida graph
rl22_calgr = [real(REg(:,1)) imag(REg(:,1))];
for  i=2 : n
  rl22_calgr = [rl22_calgr; real(REg(:,i)) imag(REg(:,i))];
end
```

```
pause
axis;
 % save rl_K09_d.kal rl22_calgr /ascii
 %rl22_calgr = zeros(rl22_calgr);
```

9.3.6. Plots

```
%****************************************************************
% GIBRALTAR PLOTS
%****************************************************************
disp(' ')
disp(' Ploting root locus of frequency and damping ')
disp(' ')
clear graph
flag=1;
I6 = eye(6,6); I15 = eye(15,15);
Umin = 0;
Umax = 80;
deltaU = 0.01;
U = Umin;
fn = (Umax-U)/deltaU;
U_vec=[Umin:deltaU:Umax-deltaU]';
RE_ome=zeros(fn,15);
RE_Ksi=zeros(fn,15);
varK
for i=1:fn
  [Gj, Cj] = updateGC(U,C,I6);
  A = asemb_conde(U,Bd,A1g,A0g,Dg,Fg,Gg,Ms,Cs,Ks);
  Ac = A-B*varK*Cj;
  [Ome,Ksi]=damp(Ac);
 if (any(Ksi) < 0.0) && flag
     Ucritico=U
     disp(Ucritico)
     pause
     flag = 0;
  end
 n_ome = length(Ome);
  U = U+deltaU;
```

```
%************************************************************
%To print results for any U
%if ((U>55.6)&&(U<55.62))
%[Ome55_6,Ksi55_6]=damp(Ac)
%pause
%end
for j=1:n_ome
    RE_ome(i,j) = Ome(j);
    RE_Ksi(i,j) = Ksi(j);
end
  vetorAc=eig(Ac);
end
axis('normal');
subplot(211); plot(U_vec,RE_Ksi,'.')
axis([Umin Umax -0.1 1.0]);
title([' Plot  of damping factors',' Umin = ',num2str(Umin)...
   , '   Umax = ',num2str(Umax)]);
xlabel('Wind velocity');ylabel('Damping factor'); grid;
subplot(212); plot(U_vec,RE_ome,'.')
title([' Plot of structural frequencies',' Umin = '...
   ,num2str(Umin), '   Umax = ',num2str(Umax)]);
xlabel('Wind velocity');ylabel('Natural frequencies'); grid;
axis([Umin Umax 0 160]);
```

9.3.7. asemb_conde.m

```
function Ap = asemb_conde(U,Bd,A1g,A0g,Dg,Fg,Gg,Ms,Cs,Ks)
%GIBRALTAR 3 LAMBDAS
  A1s = [U*A1g; zeros(size(A1g))];
  A0s = [U*U*A0g; zeros(size(A0g))];
  Ds = [U*U*Dg; zeros(size(Dg))];
  Fs = U*Fg;
  Gs = U*Gg;
%%   asembling state matrices
%  Ap = [inv(Ms)*(-Cs+A1s) inv(Ms)*(-Ks+A0s) inv(Ms)*Ds;
    eye(4,4) zeros(4,4) zeros(4,7);
    zeros(7,4) Gs Fs];
asemb_conde.m
```

9.4. Inputs and Outputs of the FORTRAN program

9.4.1. Input to obtain the rational functions of Gibraltar Bridge

```
0.001  10000000  1 1
3  5  5
0.01  5.  5
0.5 1. 3.
0.1 1. 1.25
0.1 1.5 2.5
0.1 2.0 2.75
0.1 2.2 4.
1. 1. 1. 1.
0.1   1.225  0.3
.9875  0.0167
12.1913 16.202
.0188  .0094
0.38    0.183799      1.064112      0.072842      0.177870
0.56    0.236934      1.527047      0.134665      0.250908
0.75    0.318281      1.957090      0.216702      0.296837
0.94    0.328232      2.417619      0.303526      0.326467
1.13    0.348283      2.908290      0.396585      0.333030
0.38    2.794486      -0.232401 0.351720 -0.225599
0.57    2.689326      -0.229273 0.330179 -0.341035
0.75    2.632439      -0.168004 0.299683 -0.438172
0.94    2.634124      -0.108130 0.251713 -0.538879
1.13    2.647013      -0.058766 0.196737 -0.623428
```

9.4.2. Rational functions of Gibraltar Bridge

```
#  U     U/Bf       f      B*omega/U     amp       phase
 # 47.943  0.1094E+02  0.1461E+02  0.5743E+00  0.1219E+01  0.6895E+02
 #  min_err =  0.2086015E-01
 # er1..er4 =  0.2788576E-03  0.8643663E-04  0.2487346E-04  0.4497809E-04
 # optlamda =  0.5164275E+00  0.1346202E+01  0.1971455E+01
 #    a0    =  0.3058600E+00  0.3093970E+01  0.1373977E+01 -0.5468954E+00
 #    a1    =  0.2459046E+01 -0.1403846E+00 -0.2400085E+00 -0.2141308E+00
```

```
# d(2,nr) =  0.1021054E+02  0.1787100E+01 -0.4713930E+00
# d(2,nr) = -0.4298823E+00  0.1149574E+01  0.1083369E+01
# e(2,nr) = -0.1027703E-01 -0.3480843E+00 -0.1946212E+01
# e(2,nr) =  0.4017426E-01 -0.2610986E+00  0.2132517E+01
#    k     lzr(ex)     lzi(ex)    lzr(ap)    lzi(ap)
   0.3800000   0.1837990   1.0641120   0.1947426   1.0657586
   0.5600000   0.2369340   1.5270470   0.2491671   1.5198775
   0.7500000   0.3182810   1.9570900   0.2943955   1.9810034
   0.9400000   0.3282320   2.4176190   0.3272759   2.4333709
   1.1300000   0.3482830   2.9082900   0.3499478   2.8823142
#    k     ltr(ex)     lti(ex)    ltr(ap)    lti(ap)
   0.3800000   2.7944860  -0.2324010   2.7966005  -0.2471341
   0.5700000   2.6893260  -0.2292730   2.6875635  -0.2147423
   0.7500000   2.6324390  -0.1680040   2.6394958  -0.1594877
   0.9400000   2.6341240  -0.1081300   2.6296706  -0.1063802
   1.1300000   2.6470130  -0.0587660   2.6440576  -0.0682491
#    k     mzr(ex)     mzi(ex)    mzr(ap)    mzi(ap)
   0.3800000   0.0728420   0.1778700   0.0730344   0.1811874
   0.5600000   0.1346650   0.2509080   0.1348679   0.2478532
   0.7500000   0.2167020   0.2968370   0.2156104   0.2978017
   0.9400000   0.3035260   0.3264670   0.3047499   0.3257911
   1.1300000   0.3965850   0.3330300   0.3960575   0.3333503
#    k     mtr(ex)     mti(ex)    mtr(ap)    mti(ap)
   0.3800000   0.3517200  -0.2255990   0.3547994  -0.2249027
   0.5700000   0.3301790  -0.3410350   0.3304343  -0.3380438
   0.7500000   0.2996830  -0.4381720   0.2959162  -0.4396354
   0.9400000   0.2517130  -0.5388790   0.2502752  -0.5377694
   1.1300000   0.1967370  -0.6234280   0.1986069  -0.6251227
```

9.4.3. test3_fortran_G.m

```
%*****************************************************************
%   Program "PLOT_GIBRALTAR_3_LAMBDAS"
%   Plot of exact derivatives and the approximation functions
%*****************************************************************
clear
clf
lamb1 =   0.5164275E+00;
```

```
lamb2 =   0.1346202E+01;
lamb3 =   0.1971455E+01;
A0 = [  0.3058600E+00  0.3093970E+01 ; 0.1373977E+01 -0.5468954E+00];
A1 = [  0.2459046E+01 -0.1403846E+00 ; -0.2400085E+00 -0.2141308E+00];
Dd = [  0.1021054E+02  0.1787100E+01 -0.4713930E+00 ;
   -0.4298823E+00  0.1149574E+01  0.1083369E+01];
Ed = [ -0.1027703E-01 -0.3480843E+00 -0.1946212E+01 ;
   0.4017426E-01 -0.2610986E+00  0.2132517E+01]';
rf= [0.38 0.56 0.75 0.94 1.13]';
for inc2=1:2
for inc=1:1:5
%    Rational functions with Masukawa's results
Q(inc,2*inc2-1)=(A0(inc2,1)+i*rf(inc)*A1(inc2,1)+(Dd(inc2,1)*Ed(1,1)*...
1/(i*rf(inc)+lamb1))+(Dd(inc2,2)*Ed(2,1)*1/(i*rf(inc)+lamb2))...
+(Dd(inc2,3)*Ed(3,1)*1/(i*rf(inc)+lamb3)));
Q(inc,2*inc2)=(A0(inc2,2)+i*rf(inc)*A1(inc2,2)+(Dd(inc2,1)*...
Ed(1,2)*1/(i*rf(inc)+lamb1))+(Dd(inc2,2)*Ed(2,2)*1/(i*rf(inc)+lamb2))...
+(Dd(inc2,3)*Ed(3,2)*1/(i*rf(inc)+lamb3)));
end
%    Exact values
DD=[0.38 0.183799   1.064112    0.072842    0.177870;
0.56    0.236934    1.527047    0.134665    0.250908;
0.75    0.318281    1.957090    0.216702    0.296837;
0.94    0.328232    2.417619    0.303526    0.326467;
1.13    0.348283    2.908290    0.396585    0.333030;
0.38    2.794486    -0.232401   0.351720    -0.225599;
0.56    2.689326    -0.229273   0.330179    -0.341035;
0.75    2.632439    -0.168004   0.299683    -0.438172;
0.94    2.634124    -0.108130   0.251713    -0.538879;
1.13    2.647013    -0.058766   0.196737    -0.623428];
xx1=DD(1:5,2);
yy1=DD(1:5,3);
xx2=DD(1:5,4);
yy2=DD(1:5,5);
xx3=DD(6:10,2);
yy3=DD(6:10,3);
xx4=DD(6:10,4);
```

```
yy4=DD(6:10,5);
end
figure(1)
subplot(2,2,1),plot(real(Q(:,1)),imag(Q(:,1)),xx1,yy1,'o')
grid on
xlabel('real part','fontsize',12)
ylabel('imaginary part','fontsize',12)
title('Q11 - Lift due to heaving mode','fontsize',12)
subplot(2,2,2),plot(real(Q(:,2)),imag(Q(:,2)),xx3,yy3,'o')
grid on
xlabel('real part','fontsize',12)
ylabel('imaginary part','fontsize',12)
title('Q12 - Lift due to pitching mode','fontsize',12)
subplot(2,2,4),plot(real(Q(:,4)),imag(Q(:,4)),xx4,yy4,'o')
grid on
xlabel('real part','fontsize',12)
ylabel('imaginary part','fontsize',12)
title('Q22 - Moment due to pitching mode','fontsize',12)
subplot(2,2,3),plot(real(Q(:,3)),imag(Q(:,3)),xx2,yy2,'o')
xlabel('real part','fontsize',12)
ylabel('imaginary part','fontsize',12)
grid on
title('Q21 - Moment due to heaving mode','fontsize',12)
```

9.4.4. Input to obtain the approximation functions of the wings with 6m.

```
0.001  10000000  1 1
2  5  5
0.001  2  5
0.035  0.12
0.035  0.12
0.035  0.12
0.035  0.12
0.01  0.12
1. 1. 1. 1.
0.1   1.225  0.03
9.875 0.167
```

12.1913 16.202

.0188 .0094

0.037644151 0.017064264 0.228469164 0.004266066 0.057117291

0.056508547 0.03349065 0.336682145 0.008372663 0.084170536

0.075337953 0.053076638 0.440662567 0.013269159 0.110165642

0.094144221 0.07492506 0.540724035 0.018731265 0.135181009

0.113027259 0.0984989 0.637641227 0.024624725 0.159410307

0.037644151 6.073446952 -0.337055941 1.518361738 -0.113829633

0.056508547 5.966447401 -0.419731268 1.49161185 -0.149314526

0.075337953 5.862413549 -0.476007707 1.465603387 -0.178172217

0.094144221 5.762302454 -0.512791645 1.440575614 -0.202138609

0.113027259 5.666106566 -0.534508354 1.416526642

9.4.5. Rational functions - wings with 6m.

U U/Bf f B*omega/U amp

0.6404E+03 0.1583E+02 0.9812E-02 0.8853E+01 0.4619E+01

min_err = 0.1516588E-02

er1..er4 = 0.2039881E-06 0.1042968E-05 0.1274919E-07 0.1040333E-05

optlamda = 0.4898931E-01 0.2672425E+00

a0 = 0.5742289E+00 0.4010688E+01 0.1435572E+00 0.1002672E+01

a1 = 0.3817801E+01 0.2337816E+01 0.9544502E+00 -0.2009442E+00

d(2,nr) = 0.5719828E+01 0.2422333E+01

d(2,nr) = 0.1429957E+01 0.6055831E+00

e(2,nr) = -0.1395796E-03 -0.6151422E-01

e(2,nr) = 0.2926502E-02 0.2082658E+00

k lzr(ex) lzi(ex) lzr(ap) lzi(ap)

0.0376442 0.0170643 0.2284692 0.0172549 0.2286043

0.0565085 0.0334906 0.3366821 0.0335236 0.3366582

0.0753380 0.0530766 0.4406626 0.0528598 0.4406863

0.0941442 0.0749251 0.5407240 0.0747374 0.5408348

0.1130273 0.0984989 0.6376412 0.0986799 0.6375001

k ltr(ex) lti(ex) ltr(ap) lti(ap)

0.0376442 6.0734470 -0.3370559 6.0765533 -0.3378168

0.0565085 5.9664474 -0.4197313 5.9642675 -0.4190943

0.0753380 5.8624135 -0.4760077 5.8610086 -0.4730264

0.0941442 5.7623025 -0.5127916 5.7628435 -0.5114252

0.1130273 5.6661066 -0.5345084 5.6660441 -0.5376987

```
 #   k     mzr(ex)     mzi(ex)     mzr(ap)     mzi(ap)
    0.0376442   0.0042661   0.0571173   0.0043137   0.0571511
    0.0565085   0.0083727   0.0841705   0.0083809   0.0841645
    0.0753380   0.0132692   0.1101656   0.0132150   0.1101716
    0.0941442   0.0187313   0.1351810   0.0186843   0.1352087
    0.1130273   0.0246247   0.1594103   0.0246700   0.1593750
 #   k     mtr(ex)     mti(ex)     mtr(ap)     mti(ap)
    0.0376442   1.5183617  -0.1138296   1.5191383  -0.1140198
    0.0565085   1.4916118  -0.1493145   1.4910669  -0.1491553
    0.0753380   1.4656034  -0.1781722   1.4652522  -0.1774269
    0.0941442   1.4405756  -0.2021386   1.4407109  -0.2017970
    0.1130273   1.4165266  -0.2223985   1.4165110  -0.2231961
```

9.4.6. Plot - Wings - 6m.

```
%   Program "test3_Fortran.m_WINGLETS_2_LAMBDAS"
%   Plot of exact derivatives and the approximation functions
clear
clf
lamb1 =   0.4898931E-01;
lamb2 =   0.2672425E+00;
A0 = [  0.5742289E+00  0.4010688E+01 ; 0.1435572E+00  0.1002672E+01];
A1 = [  0.3817801E+01  0.2337816E+01 ; 0.9544502E+00 -0.2009442E+00];
Dd = [  0.5719828E+01  0.2422333E+01 ;
   0.1429957E+01  0.6055831E+00];
Ed = [ -0.1395796E-03 -0.6151422E-01 ;
   0.2926502E-02  0.2082658E+00]';
rf= [0.0376442
0.0565085
0.0753380
0.0941442
0.1130273];
for inc2=1:2
for inc=1:1:5
%    Rational functions with Masukawa's results
Q(inc,2*inc2-1)=(A0(inc2,1)+i*rf(inc)*A1(inc2,1)...
   +(Dd(inc2,1)*Ed(1,1)*1/(i*rf(inc)+lamb1))...
   +(Dd(inc2,2)*Ed(2,1)*1/(i*rf(inc)+lamb2)));
```

```
Q(inc,2*inc2)=(A0(inc2,2)+i*rf(inc)*A1(inc2,2)+(Dd(inc2,1)*...
   Ed(1,2)*1/(i*rf(inc)+lamb1))...
   +(Dd(inc2,2)*Ed(2,2)*1/(i*rf(inc)+lamb2)));
end
%    Exact values
DD=[0.0376442   0.0170643   0.2284692   0.0042661   0.0571173 ;
   0.0565085   0.0334906   0.3366821   0.0083727   0.0841705  ;
   0.0753380   0.0530766   0.4406626    0.0132692   0.1101656;
   0.0941442   0.0749251   0.5407240    0.0187313   0.1351810;
   0.1130273   0.0984989   0.6376412    0.0246247   0.1594103;
   0.0376442   6.0734470  -0.3370559    1.5183617  -0.1138296 ;
   0.0565085   5.9664474  -0.4197313    1.4916118  -0.1493145;
   0.0753380   5.8624135  -0.4760077    1.4656034  -0.1781722 ;
   0.0941442   5.7623025  -0.5127916    1.4405756  -0.2021386;
   0.1130273   5.6661066  -0.5345084    1.4165266  -0.2223985];
xx1=DD(1:5,2);
yy1=DD(1:5,3);
xx2=DD(1:5,4);
yy2=DD(1:5,5);
xx3=DD(6:10,2);
yy3=DD(6:10,3);
xx4=DD(6:10,4);
yy4=DD(6:10,5);
end
figure(1)
subplot(2,2,1),plot(real(Q(:,1)),imag(Q(:,1)),xx1,yy1,'o')
grid on
xlabel('real part','fontsize',12)
ylabel('imaginary part','fontsize',12)
title('Q11 - Lift due to heaving mode','fontsize',12)
subplot(2,2,2),plot(real(Q(:,2)),imag(Q(:,2)),xx3,yy3,'o')
grid on
xlabel('real part','fontsize',12)
ylabel('imaginary part','fontsize',12)
title('Q12 - Lift due to pitching mode','fontsize',12)
subplot(2,2,4),plot(real(Q(:,4)),imag(Q(:,4)),xx4,yy4,'o')
grid on
```

```
xlabel('real part','fontsize',12)
ylabel('imaginary part','fontsize',12)
title('Q22 - Moment due to pitching mode','fontsize',12)
subplot(2,2,3),plot(real(Q(:,3)),imag(Q(:,3)),xx2,yy2,'o')
xlabel('real part','fontsize',12)
ylabel('imaginary part','fontsize',12)
grid on
title('Q21 - Moment due to heaving mode','fontsize',12)
```

9.4.7. Input to obtain the approximation functions of the wings with 6m

```
0.001  10000000  1 1
2  5  5
0.001  2  5
0.035  0.12
0.035  0.12
0.035  0.12
0.035  0.12
0.01  0.12
1. 1. 1. 1.
0.1   1.225  0.03
9.875 0.167
12.1913 16.202
.0188  .0094
0.0188 0.0052 0.1163 0.001289       0.029078
0.0283 0.0105 0.1730 0.002615       0.043261
0.0377 0.0171 0.2286 0.004271       0.057154
0.0471 0.0248 0.2831 0.006201       0.070767
0.0565 0.0335 0.3367 0.008374       0.084178
0.0188 6.180840       -0.215332 1.545210 -0.068616
0.0283 6.127152       -0.282550 1.531788 -0.092828
0.0377 6.073305       -0.337187 1.518326 -0.113882
0.0471 6.019721       -0.382250 1.504930 -0.132533
0.0565 5.966419       -0.419750 1.491605 -0.149323
Rational functions - wings with 3m
#  U      U/Bf        f       B*omega/U      amp       phase
 # ******  0.6444E+03  0.1582E+02  0.9750E-02  0.8987E+01  0.4567E+01
```

```
#  min_err =  0.1751344E-02
# er1..er4 =  0.9006185E-06  0.1031405E-05  0.5715989E-07  0.1078024E-05
# optlamda =  0.3500000E-01  0.1320000E+00
#    a0    =  0.2098244E+00  0.5221252E+01  0.5251235E-01  0.1305108E+01
#    a1    =  0.4615840E+01 -0.5292162E+00  0.1153550E+01 -0.9164623E+00
#  d(2,nr) =  0.4798194E+01  0.1821348E+01
#  d(2,nr) =  0.1197957E+01  0.4559187E+00
#  e(2,nr) =  0.1471817E-05 -0.1512478E-01
#  e(2,nr) =  0.1524671E-02  0.5906955E-01
#    k     lzr(ex)     lzi(ex)    lzr(ap)    lzi(ap)
   0.0188000   0.0052000   0.1163000   0.0054370   0.1158257
   0.0283000   0.0105000   0.1730000   0.0104243   0.1733059
   0.0377000   0.0171000   0.2286000   0.0169641   0.2290253
   0.0471000   0.0248000   0.2831000   0.0247729   0.2833648
   0.0565000   0.0335000   0.3367000   0.0335016   0.3362001
#    k     ltr(ex)     lti(ex)     ltr(ap)    lti(ap)
   0.0188000   6.1808400  -0.2153320   6.1823111  -0.2108570
   0.0283000   6.1271520  -0.2825500   6.1268702  -0.2842328
   0.0377000   6.0733050  -0.3371870   6.0715869  -0.3393993
   0.0471000   6.0197210  -0.3822500   6.0186070  -0.3829700
   0.0565000   5.9664190  -0.4197500   5.9680618  -0.4183198
#    k     mzr(ex)     mzi(ex)     mzr(ap)    mzi(ap)
   0.0188000   0.0012890   0.0290780   0.0013501   0.0289580
   0.0283000   0.0026150   0.0432610   0.0025986   0.0433286
   0.0377000   0.0042710   0.0571540   0.0042356   0.0572585
   0.0471000   0.0062010   0.0707670   0.0061903   0.0708430
   0.0565000   0.0083740   0.0841780   0.0083753   0.0840510
#    k     mtr(ex)     mti(ex)     mtr(ap)    mti(ap)
   0.0188000   1.5452100  -0.0686160   1.5455743  -0.0674639
   0.0283000   1.5317880  -0.0928280   1.5317197  -0.0932693
   0.0377000   1.5183260  -0.1138820   1.5179005  -0.1144467
   0.0471000   1.5049300  -0.1325330   1.5046532  -0.1327255
   0.0565000   1.4916050  -0.1493230   1.4920114  -0.1489480
```

9.4.8. Plot - Wings - 3m

```
%   Program "test3_Fortran.m_WINGLETS_2_LAMBDAS"
%   Plot of exact derivatives and the approximation functions
```

```
clear
clf
lamb1 =   0.3500000E-01;
lamb2 =   0.1320000E+00;
A0 = [   0.2098244E+00 0.5221252E+01 ; 0.5251235E-01  0.1305108E+01];
A1 = [  0.4615840E+01 -0.5292162E+00 ; 0.1153550E+01 -0.9164623E+00];
Dd = [  0.4798194E+01  0.1821348E+01 ; 0.1197957E+01  0.4559187E+00];
Ed = [ 0.1471817E-05 -0.1512478E-01 ; 0.1524671E-02  0.5906955E-01]';
rf= [0.0188000
0.0283000
0.0377000
0.0471000
0.0565000];
for inc2=1:2
for inc=1:1:5
%    Rational functions with Masukawa's results
Q(inc,2*inc2-1)=(A0(inc2,1)+i*rf(inc)*A1(inc2,1)...
   +(Dd(inc2,1)*Ed(1,1)*1/(i*rf(inc)+lamb1))...
   +(Dd(inc2,2)*Ed(2,1)*1/(i*rf(inc)+lamb2)));
Q(inc,2*inc2)=(A0(inc2,2)+i*rf(inc)*A1(inc2,2)+(Dd(inc2,1)*...
   Ed(1,2)*1/(i*rf(inc)+lamb1))...
   +(Dd(inc2,2)*Ed(2,2)*1/(i*rf(inc)+lamb2)));
end
%    Exact values
DD=[0.0188000   0.0052000   0.1163000   0.0012890   0.0290780  ;
   0.0283000   0.0105000   0.1730000  0.0026150   0.0432610 ;
   0.0377000   0.0171000   0.2286000    0.0042710   0.0571540 ;
   0.0471000   0.0248000   0.2831000  0.0062010   0.0707670 ;
   0.0565000   0.0335000   0.3367000  0.0083740   0.0841780 ;
   0.0188000   6.1808400  -0.2153320   1.5452100  -0.0686160;
   0.0283000   6.1271520  -0.2825500    1.5317880  -0.0928280 ;
   0.0377000   6.0733050  -0.3371870    1.5183260  -0.1138820;
   0.0471000   6.0197210  -0.3822500   1.5183260  -0.1138820;
   0.0565000   5.9664190  -0.4197500   1.4916050  -0.1493230 ];
xx1=DD(1:5,2);
yy1=DD(1:5,3);
xx2=DD(1:5,4);
```

```
yy2=DD(1:5,5);
xx3=DD(6:10,2);
yy3=DD(6:10,3);
xx4=DD(6:10,4);
yy4=DD(6:10,5);
end
figure(1)
subplot(2,2,1),plot(real(Q(:,1)),imag(Q(:,1)),xx1,yy1,'o')
grid on
xlabel('real part','fontsize',12)
ylabel('imaginary part','fontsize',12)
title('Q11 - Lift due to heaving mode','fontsize',12)
subplot(2,2,2),plot(real(Q(:,2)),imag(Q(:,2)),xx3,yy3,'o')
grid on
xlabel('real part','fontsize',12)
ylabel('imaginary part','fontsize',12)
title('Q12 - Lift due to pitching mode','fontsize',12)
subplot(2,2,4),plot(real(Q(:,4)),imag(Q(:,4)),xx4,yy4,'o')
grid on
xlabel('real part','fontsize',12)
ylabel('imaginary part','fontsize',12)
title('Q22 - Moment due to pitching mode','fontsize',12)
subplot(2,2,3),plot(real(Q(:,3)),imag(Q(:,3)),xx2,yy2,'o')
xlabel('real part','fontsize',12)
ylabel('imaginary part','fontsize',12)
grid on
title('Q21 - Moment due to heaving mode','fontsize',12)
```