B. FORMAL MEASURES ANALYSIS – MATLAB PROGRAM

```
% ------------------------------------------------------------------------------------------------
% This program performs the calculations for the measures investigated in Chapter 6
% ------------------------------------------------------------------------------------------------

% INPUTS REQUIRED:
% ================
% steady        : 0 for dynamic analysis
%                 1 for steady state analysis
% w1            : Initial value of frequency
%                 Not required for steady state analysis
% nc            : Number of calculation points
%                 Use 1 for steady state analysis
% sc            : 0 if the system scaling matrices used
%                 1 if the system scaling matrices not used
% A,B,C,D       : System matrices
% T1,T2         : Scaling matrices [outputs-inputs] (if required)

%------------------------------------------------------------------------------------------------

% PROGRAM OUTPUTS:
% =================
% od            : Singular values
% ocd           : Condition number
% ocm           : Minimised condition number
% oth           : DIM
% ott           : PIM
% oIM           : muIM
```

```
% orga1        : diagonal RGA values
% orga2        : recommended RGA pairing
% oprmr1       : u-order of Moore Pairing
% oprmr2       : v-order of Moore Pairing

% -----------------------------------------------------------------------------------------------------------------------

% Determination of using steady state or dynamic calculation

w = 0.000 ;

if steady ~= 1
w2 = log10(w1)+1 ;
end

for f1=1:nc

        if steady == 1
        s = 0 ;
        else

% Frequency counter

        w = w + w1 ;
        w3 = 10^w2 ;
        w4 = w-w3 ;

        if abs(w4) < 1e-8
        w1=w3 ;
        w2=w2+1 ;
        end

        s = w*i ;

        end

% -----------------------------------------------------------------------------------------------------------------------

% Calculate the process gain matrix
```

```
Gn = C*inv(s*eye(size(A))-A)*B + D ;

        if sc == 1
        G = Gn ;
        else
        G = inv(T1)*Gn*T2 ;
        end

% ------------------------------------------------------------------------------------------------------------------------

% Relative Gain Array RGA
% ==================

rga = G.*inv(G') ;
[rga1,irga1] = min(abs(rga-1)) ;

        for f2=1:(size(rga,2))
        rga2(f2) = rga(irga1(f2),f2);
        end

prrga = [rga2' irga1'] ;

        rgaindex=0;
        for f2=1:(size(rga,2))
        rgaindex=rgaindex+f2;
        end

        if (sum(irga1)~=rgaindex)
        pwrga= 1 ;                     % invalid pairing
        else
        pwrga= 0 ;                     % valid pairing
        end

rgatotal1 = sum(rga2) ;
rgatotal2 = sum(diag(rga)) ;

% ------------------------------------------------------------------------------------------------------------------------
```

```
% Niederlinski Index
% ==============

NI = det(G) / prod(diag(G));

        if (NI<0)
        swni=-1 ;                    % Diagonal SISO controllers unstable
        else
        swni=1  ;                    % Diagonal SISO controllers may be stable
        end

% Relative Interaction Array RIA
% ======================

ria = 1./rga -1 ;
[ria1,iria1] = min(abs(ria)) ;

        for f2=1:(size(ria,2))
        ria2(f2) = ria(iria1(f2),f2);
        end

prria = [ria2' iria1'] ;

        if (sum(iria1)~=rgaindex)
        pwria = 1 ;                  % invalid pairing
        else
        pwria = 0 ;                  % valid pairing
        end

riatotal1 = sum(ria2) ;
riatotal2 = sum(diag(ria)) ;

% -----------------------------------------------------------------------------------------------------------------------

% Singular value decomposition
% =====================

[u,d,v] = svd(G) ;

% -----------------------------------------------------------------------------------------------------------------------
```

```
% Condition number
% ============

cn = cond(G) ;

% -----------------------------------------------------------------------------------------------------------------------

% Dynamic Interaction Measure DIM
% ========================

inn=0 ;
ind=0 ;
k=size(G,2) ;

        for f2=1:k
        ww = u(:,f2)*(v(:,f2))' ;
        mm = max(abs(ww)) ;
        ll = max(mm)*conj(max(mm)) ;
        hh = acos(sqrt(ll))*180/pi ;
        inn = inn + d(f2,f2)^2 * sqrt(ll)^2 ;
        ind = ind + d(f2,f2)^2 ;
        end

th = acos(sqrt(inn/ind))*180/pi ;

% Diagonal Dynamic Interaction Measure DIM
% ===============================

dinn=0 ;
dind=0 ;
dk=size(G,2) ;

        for f2=1:dk
        dww = u(:,f2)*(v(:,f2))' ;
        dmm = ww(f2,f2) ;
        dll = dmm*conj(dmm) ;
        dhh = acos(sqrt(ll))*180/pi ;
        dinn = dinn + d(f2,f2)^2 * sqrt(ll)^2 ;
        dind = dind + d(f2,f2)^2 ;
```

```
            end

dth = acos(sqrt(dinn/dind))*180/pi ;

% -----------------------------------------------------------------------------------------------------------------------

% Moore's Pairing
% ===========

[p1,ip1] = max(abs(u)) ;
[p2,ip2] = max(abs(v)) ;
prmr = [ip1' ip2'] ;

            if (sum(ip1)|sum(ip2)~=rgaindex)
            pwsvd = 1 ;                         % invalid pairing
            else
            pwsvd = 0 ;                         % valid pairing
            end

% -----------------------------------------------------------------------------------------------------------------------

% mu Interaction Measure
% =================

dd = diag(G) ;
E = (G-diag(dd))*inv(diag(dd)) ;
L = eig(abs(E)) ;
mu = max(L) ;
IM = 1/mu ;

% -----------------------------------------------------------------------------------------------------------------------

% Minimised Condition Number
% ====================

n1 = norm(rga,1) ;
n2 = norm(rga,inf) ;
cm = 2*max(n1,n2) ;

% -----------------------------------------------------------------------------------------------------------------------
```

```
% Performance Interaction Measure
% ========================

ww2 = u*v' ;
nw = (ww2-eye(size(ww2))) ;
nd = diag(svd(nw)) ;
tt = nd(1,1) ;
aww = abs(ww2-1) ;

% ---------------------------------------------------------------------------------------------------------------------

% Program outputs
% ============

of(f1) = f1;
ow(f1) = w ;

        for f2=1:k
        orga1(f2,f1) = rga(f2,f2) ;
        orga2(f2,f1) = irga1(:,f2) ;
        od(f2,f1) = d(f2,f2) ;
        oaww(f2,f1) = aww(f2,f2) ;
        oprmr1(f2,f1) = prmr(f2,1) ;
        oprmr2(f2,f1) = prmr(f2,2) ;
        end

        ocn(f1) = cn ;
        oth(f1) = th ;
        odth(f1) = dth;
        oIM(f1) = IM ;
        ocm(f1) = cm ;
        ott(f1) = tt ;

        opwrga(f1) = pwrga;
        opwria(f1) = pwria;
        opwsvd(f1) = pwsvd;
        oswni(f1) = swni;

        end
```

```
rgav = [ow' opwrga'];
riav = [ow' opwria'];
svdv = [ow' opwsvd'];
niv = [ow' oswni'];

% ---------------------------------------------------------------------------------------------------------------------
```