

Optimal Co-design of Local and Global Controllers

In this file, we will test a solution for optimal co-design of local and global controllers.

Demonstration

Test the co-design approach with 3 parameters (pHat,pBar and gammaHat):

```
clear all
close all
clc
```

```
% To help local design:      Larger pHat, Smaller pBar, and Larger gammaHat
% To help global design:    Smaller pHat, Larger pBar, and Smaller gammaHat
```

```
%  $0 < \text{pHat} < p < \text{pBar}$ 
pHat = 0.10
```

```
pHat = 0.1000
```

```
pBar = 0.11
```

```
pBar = 0.1100
```

```
%  $0 < \text{gammaHat} < \text{gamma} < \text{gammaBar}$ 
gammaHat = 2.5
```

```
gammaHat = 2.5000
```

```
load TempNet3.mat
platoonObj = net.platoons(1)
```

```
platoonObj =
  Platoon with properties:
```

```
    platoonIndex: 1
    numOfVehicles: 6
         vehicles: [1x6 Vehicle]
        topology: [1x1 Topology]
    graphics1: [40.0184 42.0184 44.0184 46.0184 48.0184 50.0184 52.0184 54.0184 58.0131 60.0131 62.0131 64.0129]
    graphics2: [41.0184 43.0184 45.0184 47.0184 49.0184 51.0184 53.0184 55.0184 59.0131 61.0131 63.0131 65.0129]
```

```
for i = 1:1:1
    disp(['Iteration ',num2str(i)])

    [statusL,nuVal,rhoVal,LVal] = synthesizeLocalControllers(gammaHat,pHat,pBar)
    if statusL == 0
        pHat = pHat + 0.1;
        pBar = pBar - 0.1;
        gammaHat = gammaHat + 0.01;
        continue
    end
end
```

```

[statusG,gammaVal,Kval] = synthesizeGlobalRobustControllers(platoonObj,gammaHat,pHat,pBar,n
if statusG == 0
    pHat = pHat - 0.1;
    pBar = pBar + 0.1;
    gammaHat = gammaHat - 0.01;
    continue
end

pHat = pHat - 0.01;
pBar = pBar + 0.1;
gammaHat = gammaHat - 1;
end

```

```

Iteration 1
Local Synthesis Success
statusL = logical
1
nuVal = -22.5647
rhoVal = 10.2122
LVal = 1×3
    -594.2726 -658.9661 -52.9221
Global Synthesis Success
statusG = logical
1
gammaVal = 2.3600
Kval = 5×5 cell

```

	1	2	3	4	5
1	[0,0,0;0...	[0,0,0;0...	[0,0,0;0...	[0,0,0;0...	[0,0,0;0...
2	[0,0,0;0...	[0,0,0;0...	[0,0,0;0...	[0,0,0;0...	[0,0,0;0...
3	[0,0,0;0...	[0,0,0;0,0,...	[0,0,0;0...	[0,0,0;0...	[0,0,0;0...
4	[0,0,0;0...	[0,0,0;0,0,...	[0,0,0;0,0,...	[0,0,0;0...	[0,0,0;0...
5	[0,0,0;0,0,...	[0,0,0;0...	[0,0,0;0...	[0,0,0;0...	[0,0,0;0...

Test the co-design approach with 1 parameter (pVal)

(this approach is also called the "compact" co-design approach):

```

clear all
close all
clc

load TempNet3.mat
platoonObj = net.platoons(1)

```

```

platoonObj =
    Platoon with properties:

    platoonIndex: 1
    numOfVehicles: 6
    vehicles: [1×6 Vehicle]
    topology: [1×1 Topology]

```

```
graphics1: [40.0184 42.0184 44.0184 46.0184 48.0184 50.0184 52.0184 54.0184 58.0131 60.0131 62.0131 64.0129]
graphics2: [41.0184 43.0184 45.0184 47.0184 49.0184 51.0184 53.0184 55.0184 59.0131 61.0131 63.0131 65.0129]
```

```
p = 0.15
```

```
p = 0.1500
```

```
[statusL,nuVal,rhoVal,LVal] = synthesizeLocalControllersCompact(p)
```

```
Local Synthesis Success with gammaSq=2.9519
```

```
statusL = logical
```

```
1
```

```
nuVal = -19.6793
```

```
rhoVal = 7.0547
```

```
LVal = 1x3
```

```
-75.5069 -93.6429 -16.8442
```

```
[statusG,gammaVal,KVal] = synthesizeGlobalRobustControllersCompact(platoonObj,nuVal,rhoVal)
```

```
Global Synthesis Success with gammaSq=3.0366
```

```
statusG = logical
```

```
1
```

```
gammaVal = 3.0366
```

```
KVal = 5x5 cell
```

	1	2	3	4	5
1	[0,0,0;0...	[0,0,0;0...	[0,0,0;0...	[0,0,0;0...	[0,0,0;0...
2	[0,0,0;0...	[0,0,0;0...	[0,0,0;0...	[0,0,0;0...	[0,0,0;0...
3	[0,0,0;0...	[0,0,0;0,0,...	[0,0,0;0...	[0,0,0;0...	[0,0,0;0...
4	[0,0,0;0...	[0,0,0;0,0,...	[0,0,0;0,0,...	[0,0,0;0...	[0,0,0;0...
5	[0,0,0;0,0,...	[0,0,0;0...	[0,0,0;0...	[0,0,0;0...	[0,0,0;0...

```
[feasibility,Ceq] = synthesizeControllersCompactFeasibility(platoonObj,p)
```

```
Local Synthesis Success with gammaSq=2.9519
```

```
Global Synthesis Success with gammaSq=3.0366
```

```
feasibility = 3x1
```

```
0
```

```
0
```

```
0
```

```
Ceq =
```

```
[]
```

Finding the Optimal Parameters (Using "fmincon")

For the co-design approach with 3 parameters:

```
clear all
```

```
close all
```

```
clc
```

```
% To help local design:      Larger pHat, Smaller pBar, and Larger gammaHat
```

```
% To help global design:    Smaller pHat, Larger pBar, and Smaller gammaHat
```

```
% 0 < pHat < p < pBar
pHat = 0.1
```

```
pHat = 0.1000
```

```
pBar = 1
```

```
pBar = 1
```

```
% 0 < gammaHat < gamma < gammaBar
gammaHat = 20
```

```
gammaHat = 20
```

```
load TempNet3.mat
platoonObj = net.platoons(1)
```

```
platoonObj =  
Platoon with properties:
```

```
    platoonIndex: 1
```

```
    numOfVehicles: 6
```

```
        vehicles: [1x6 Vehicle]
```

```
        topology: [1x1 Topology]
```

```
    graphics1: [40.0184 42.0184 44.0184 46.0184 48.0184 50.0184 52.0184 54.0184 58.0131 60.0131 62.0131 64.0129
```

```
    graphics2: [41.0184 43.0184 45.0184 47.0184 49.0184 51.0184 53.0184 55.0184 59.0131 61.0131 63.0131 65.0129
```

```
% x0 = [pHat,pBar,gammaHat];
x0 = [0.1003    0.1072    2.1107]
```

```
x0 = 1x3  
    0.1003    0.1072    2.1107
```

```
f = @(x)synthesizeControllers(platoonObj,x);
```

```
[x,fval] = fmincon(f,x0)
```

```
Local Synthesis Success  
Global Synthesis Success  
Local Synthesis Success  
Global Synthesis Success  
Local Synthesis Success  
Global Synthesis Success  
Local Synthesis Success  
Global Synthesis Success  
Local Synthesis Failed  
Local Synthesis Failed  
Local Synthesis Failed  
Local Synthesis Failed  
Local Synthesis Failed  
Local Synthesis Failed  
Local Synthesis Failed  
Local Synthesis Success  
Global Synthesis Success
```


[illegible]

[illegible]

[illegible]

```

Local Synthesis Success
Global Synthesis Success
Local Synthesis Success
Global Synthesis Success
Local Synthesis Success
Global Synthesis Success

```

Local minimum possible. Constraints satisfied.

fmincon stopped because the size of the current step is less than the value of the step size tolerance and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>

```

x = 1×3
    0.0783    0.2565    8.6906
fval = 2.8046

```

```

% x = [0.3816    1.0063    4.8777] % L is too large
% x = [0.3816    1.0063    4.9013]
% Lets use norm(L) in the objective, with a changing magnitude
% x = [0.1003    0.1072    2.1107]
% x = [0.1001    0.1036    2.1481]
% x = [0.1028    0.1046    2.2569]
% x = [0.1047    0.1047    2.2575]
% x = [0.1047    0.1047    2.2575]
% x = [0.1047    0.1047    2.2575]
% With an intersection motivating term in the objective function
% x = [0.1080    0.1082    2.5054]
% x = [0.1003    0.1072    2.1107]
% x = [0.1003    0.1072    2.1107]
% x = [0.0611    0.0611    2.2250]

```

x

```

x = 1×3
    0.0783    0.2565    8.6906

```

```
pHat = x(1)
```

```
pHat = 0.0783
```

```
pBar = x(2)
```

```
pBar = 0.2565
```

```
gammaHat = x(3)
```

```
gammaHat = 8.6906
```

```

% For: nuBar < nu < nuHat < 0
nuBar = -gammaHat/pBar

```

```
nuBar = -33.8846
```

```
% For: 0 < rhoHat1,rhoHat2 < rho < rhoBar
```

```
rhoHat1 = pBar/(4*gammaHat)
```

```
rhoHat1 = 0.0074
```

```
rhoHat2 = 1/pHat
```

```
rhoHat2 = 12.7732
```

```
[statusL,nuVal,rhoVal,LVal] = synthesizeLocalControllers(gammaHat,pHat,pBar)
```

Local Synthesis Success

```
statusL = logical
```

```
1
```

```
nuVal = -31.7173
```

```
rhoVal = 13.3949
```

```
LVal = 1x3
```

```
-209.4463 -239.2597 -25.9213
```

```
[statusG,gammaVal,KVal] = synthesizeGlobalRobustControllers(platoonObj,gammaHat,pHat,pBar,nuVal)
```

Global Synthesis Success

```
statusG = logical
```

```
1
```

```
gammaVal = 2.4856
```

```
KVal = 5x5 cell
```

	1	2	3	4	5
1	[0,0,0;0...	[0,0,0;0...	[0,0,0;0...	[0,0,0;0...	[0,0,0;0,0,...
2	[0,0,0;0,0,...	[0,0,0;0...	[0,0,0;0,0,...	[0,0,0;0...	[0,0,0;0...
3	[0,0,0;0...	[0,0,0;0,0,...	[0,0,0;0...	[0,0,0;0,0,...	[0,0,0;0...
4	[0,0,0;0...	[0,0,0;0,0,...	[0,0,0;0,0,...	[0,0,0;0...	[0,0,0;0...
5	[0,0,0;0,0,...	[0,0,0;0...	[0,0,0;0...	[0,0,0;0,0,...	[0,0,0;0...

For the co-design approach with 1 parameter (pVal)

```
clear all
```

```
close all
```

```
clc
```

```
load TempNet3.mat
```

```
platoonObj = net.platoons(1)
```

```
platoonObj =
```

```
Platoon with properties:
```

```
platoonIndex: 1
```

```
numOfVehicles: 6
```

```
vehicles: [1x6 Vehicle]
```

```
topology: [1x1 Topology]
```

```
graphics1: [40.0184 42.0184 44.0184 46.0184 48.0184 50.0184 52.0184 54.0184 58.0131 60.0131 62.0131 64.0129
```

```
graphics2: [41.0184 43.0184 45.0184 47.0184 49.0184 51.0184 53.0184 55.0184 59.0131 61.0131 63.0131 65.0129
```

```
f = @(x)synthesizeControllersCompact(platoonObj,x);
```

```
nonlcon = @(x)synthesizeSizeControllersCompactFeasibility(platoonObj,x);
```

```
A = [];
```

```
b = [];
```

```
Aeq = [];
```

```
beq = [];
```

```
lb = [0];
```

```
ub = [inf];
```

```
p0 = 0.15;
```

```
[p,fval] = fmincon(f,p0,A,b,Aeq,beq,lb,ub,nonlcon)
```

```
Local Synthesis Success with gammaSq=2.9519  
Global Synthesis Success with gammaSq=3.0366  
Local Synthesis Success with gammaSq=2.9519  
Global Synthesis Success with gammaSq=3.0366  
Local Synthesis Success with gammaSq=2.9519  
Global Synthesis Success with gammaSq=3.0366  
Local Synthesis Success with gammaSq=2.9519  
Global Synthesis Success with gammaSq=3.0366  
Local Synthesis Success with gammaSq=2.7325  
Global Synthesis Success with gammaSq=2.6184  
Local Synthesis Success with gammaSq=2.7325  
Global Synthesis Success with gammaSq=2.6184  
Local Synthesis Success with gammaSq=2.7325  
Global Synthesis Success with gammaSq=2.6184  
Local Synthesis Success with gammaSq=2.7325  
Global Synthesis Success with gammaSq=2.6184  
Local Synthesis Success with gammaSq=2.9913  
Global Synthesis Success with gammaSq=2.9807  
Local Synthesis Success with gammaSq=2.9913  
Global Synthesis Success with gammaSq=2.9807  
Local Synthesis Success with gammaSq=2.8798  
Global Synthesis Success with gammaSq=2.8021  
Local Synthesis Success with gammaSq=2.8798  
Global Synthesis Success with gammaSq=2.8021  
Local Synthesis Success with gammaSq=2.7429  
Global Synthesis Success with gammaSq=2.6459  
Local Synthesis Success with gammaSq=2.7429  
Global Synthesis Success with gammaSq=2.6459  
Local Synthesis Success with gammaSq=2.6894  
Global Synthesis Success with gammaSq=2.5882  
Local Synthesis Success with gammaSq=2.6894  
Global Synthesis Success with gammaSq=2.5882  
Local Synthesis Success with gammaSq=2.6894  
Global Synthesis Success with gammaSq=2.5882  
Local Synthesis Success with gammaSq=2.6894  
Global Synthesis Success with gammaSq=2.5882  
Local Synthesis Success with gammaSq=2.7035  
Global Synthesis Success with gammaSq=2.598  
Local Synthesis Success with gammaSq=2.7035  
Global Synthesis Success with gammaSq=2.598  
Local Synthesis Success with gammaSq=2.6892  
Global Synthesis Success with gammaSq=2.5862  
Local Synthesis Success with gammaSq=2.6892  
Global Synthesis Success with gammaSq=2.5862  
Local Synthesis Success with gammaSq=2.6892  
Global Synthesis Success with gammaSq=2.5862  
Local Synthesis Success with gammaSq=2.6892  
Global Synthesis Success with gammaSq=2.5862  
Local Synthesis Success with gammaSq=2.6886
```


[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

% p = 0.0449

```
[statusL, nuVal, rhoVal, LVal] = synthesizeLocalControllersCompact(p)
```

```
[statusG, gammaVal, KVal] = synthesizeGlobalRobustControllersCompact(platoonObj, nuVal, rhoVal)
```

35

	1	2	3	4	5
1	[0,0,0;0...	[0,0,0;0,0,...	[0,0,0;0...	[0,0,0;0...	[0,0,0;0,0,...
2	[0,0,0;0,0,...	[0,0,0;0...	[0,0,0;0,0,...	[0,0,0;0...	[0,0,0;0...
3	[0,0,0;0...	[0,0,0;0,0,...	[0,0,0;0...	[0,0,0;0,0,...	[0,0,0;0...
4	[0,0,0;0...	[0,0,0;0,0,...	[0,0,0;0,0,...	[0,0,0;0...	[0,0,0;0,0,...
5	[0,0,0;0,0,...	[0,0,0;0...	[0,0,0;0...	[0,0,0;0,0,...	[0,0,0;0...

Necessary Functions

Functions for co-design approach with 3 parameters:

For local synthesis:

```
function [status,nuVal,rhoVal,LVal] = synthesizeLocalControllers(gammaHat,pHat,pBar)
% Here we will synthesize the local controllers for local error
% dynamics to optimize the passivity properties

errorDynamicsType = 2;
if errorDynamicsType == 1
    A = [0,1,0;0,0,0;0,0,0]; % For error dynamics type 1
else
    A = [0,1,0;0,0,1;0,0,0]; % For error dynamics type 2
end
B = [0;0;1];
I = eye(3);
O = zeros(3);

% For: nuBar < nu < nuHat < 0
nuBar = -gammaHat/pBar;
% nuHat = -0.001;

% For: 0 < rhoHat1,rhoHat2 < rho < rhoBar
% rhoBar = 10;
rhoHat1 = pBar/(4*gammaHat);
rhoHat2 = 1/pHat;

% For: 0 < rhoTildeHat < rhoTilde < rhoTildeBar1,rhoTildeBar2
% rhoTildeHat = 1/rhoBar;
rhoTildeBar1 = 1/rhoHat1;
rhoTildeBar2 = 1/rhoHat2;

% Set up the LMI problem
solverOptions = sdpsettings('solver','mosek','verbose',0);
P = sdpvar(3,3,'symmetric');
K = sdpvar(1,3,'full');

rhoTilde = sdpvar(1,1,'full'); %Representing: 1/rho
```

```

nu = sdpvar(1,1,'full');

% Basic Constraints
con1 = P >= 0;

% Approach 4 with rho = prespecified, nu < 0 and nu is free to maximize
DMat = [rhoTilde*I];
MMat = [P, 0];
ThetaMat = [-A*P-P*A'-B*K-K'*B', -I+0.5*P; -I+0.5*P, -nu*I];
W = [DMat, MMat; MMat', ThetaMat];
con2 = W >= 0;

%%Constraints on resulting nu and rho from the local design

% nuBar < nu < nuHat < 0
con3 = nu >= nuBar; % Helps global design
%con4 = nu <= nuHat;

% 0 < rhoTildeHat < rhoTilde < rhoTildeBar1,rhoTildeBar2
% con5 = rhoTilde >= rhoTildeHat;
con6 = rhoTilde <= rhoTildeBar1; % Helps global design
con7 = rhoTilde <= rhoTildeBar2; % Helps global design

% Total Cost and Constraints
cons = [con1,con2,con3,con6,con7];
%costFun = 0*(-nu + rhoBar); % For stabilizing, set coefficient to 0
%costFun = 0.0000001*(-nu + rhoBar); % Otherwise set to 0.0000001.
costFun = 0;

% Solution
sol = optimize(cons,costFun,solverOptions);
status = sol.problem == 0; % sol.info;

PVal = value(P);
KVal = value(K);
LVal = KVal/PVal;

nuVal = value(nu);
rhoVal = 1/value(rhoTilde);

if status == 1
    disp(['Local Synthesis Success'])
else
    disp(['Local Synthesis Failed'])
end

end

```

For global synthesis:

```
function [status,gammaSqVal,K] = synthesizeGlobalRobustControllers(platoonObj,gammaHat,pHat,pB
```

```

% Number of follower vehicles
N = platoonObj.numOfVehicles-1;

% Creating the adjacency matrix, null matrix and cost matrix
G = platoonObj.topology.graph;
A = adjacency(G);
for i = 1:1:N
    for j = 1:1:N
        % Structure of K_ij (which is a 3x3 matrix) should be embedded here
        if i~=j
            if A(j+1,i+1)==1
                adjMatBlock{i,j} = [0,0,0; 0,0,0; 1,1,1];
                nullMatBlock{i,j} = [1,1,1; 1,1,1; 0,0,0];
                costMatBlock{i,j} = 1*[0,0,0; 0,0,0; 1,1,1];
            else
                adjMatBlock{i,j} = [0,0,0; 0,0,0; 0,0,0];
                nullMatBlock{i,j} = [1,1,1; 1,1,1; 0,0,0];
                costMatBlock{i,j} = 10*[0,0,0; 0,0,0; 1,1,1];
            end
        else
            adjMatBlock{i,j} = [0,0,0; 0,0,0; 1,1,1];
            nullMatBlock{i,j} = [1,1,1; 1,1,1; 0,0,0];
            costMatBlock{i,j} = 0*[0,0,0; 0,0,0; 1,1,1];
        end
    end
end
adjMatBlock = cell2mat(adjMatBlock);
nullMatBlock = cell2mat(nullMatBlock);
costMatBlock = cell2mat(costMatBlock);

% Set up the LMI problem
solverOptions = sdpsettings('solver','mosek','verbose',0);
I = eye(3*N);
I_n = eye(3);
O = zeros(3*N);

% Whether to use a soft or hard graph constraint
isSoft = 1;
normType = 2;

Q = sdpvar(3*N,3*N,'full');
P = sdpvar(N,N,'diagonal');
gammaSq = sdpvar(1,1,'full');

X_p_11 = [];
X_p_12 = [];
X_12 = [];
X_p_22 = [];
for i = 1:1:N

```

```

    nu_i = nuVal;
    rho_i = rhoVal;

    X_p_11 = blkdiag(X_p_11,-nu_i*P(i,i)*I_n);
    X_p_12 = blkdiag(X_p_12,0.5*P(i,i)*I_n);
    X_12 = blkdiag(X_12,(-1/(2*nu_i))*I_n);
    X_p_22 = blkdiag(X_p_22,-rho_i*P(i,i)*I_n);
end
X_p_21 = X_p_12';
X_21 = X_12';

% Objective Function
% costFun = 1*norm(Q.*costMatBlock,normType);
costFun0 = sum(sum(Q.*costMatBlock));

% Budget Constraints (Not at this stage)
con0 = costFun0 >= 1;

% Basic Constraints
con1 = P >= 0;

DMat = [X_p_11, 0; 0, I];
MMat = [Q, X_p_11; I, 0];
ThetaMat = [-X_21*Q-Q'*X_12-X_p_22, -X_p_21; -X_p_12, gammaSq*I];
con2 = [DMat, MMat; MMat', ThetaMat] >= 0; % The real one

% For:  $0 < \gamma_{\text{Hat}} < \gamma < \gamma_{\text{Bar}}$ 
% gammaSqBar = 10000;
% con3 = gammaSq <= gammaSqBar;
% con4 = gammaSq >= gammaHat;

% For:  $0 < p_{\text{Hat}} < p_i < p_{\text{Bar}}$ 
% con5 = P >= pHat*eye(N);
% con6 = P <= pBar*eye(N);

% Structural constraints
con7 = Q.*(nullMatBlock==1)==0; % Structural limitations (due to the format of the control)
con8 = Q.*(adjMatBlock==0)==0; % Graph structure : hard constraint

% Total Cost and Constraints
if isSoft
    cons = [con0,con1,con2,con7]; % Without the hard graph constraint con7
    costFun = 1*costFun0 + 1*gammaSq; % soft
else
    cons = [con0,con1,con2,con7,con8]; % With the hard graph constraint con7
    costFun = 1*costFun0 + 1*gammaSq; % hard (same as soft)
end

sol = optimize(cons,[costFun],solverOptions);
status = sol.problem == 0; %sol.info;

```

```

costFunVal = value(costFun);
PVal = value(P);
QVal = value(Q);
X_p_11Val = value(X_p_11);
X_p_21Val = value(X_p_21);

gammaSqVal = value(gammaSq);

M_neVal = X_p_11Val\QVal;

% Obtaining K_ij blocks
M_neVal(nullMatBlock==1) = 0;
maxNorm = 0;
for i = 1:1:N
    for j = 1:1:N
        K{i,j} = M_neVal(3*(i-1)+1:3*i , 3*(j-1)+1:3*j); % (i,j)-th (3 x 3) block
        normVal = max(max(abs(K{i,j})));
        if normVal>maxNorm
            maxNorm = normVal;
        end
    end
end

% filtering out extremely small interconnections
for i=1:1:N
    for j=1:1:N
        if i~=j
            if isSoft
                K{i,j}(abs(K{i,j})<0.0001*maxNorm) = 0;
            else
                if A(i+1,j+1)==0
                    K{i,j} = zeros(3);
                end
            end
        end
    end

    K_ijMax = max(abs(K{i,j}(:)));
    K{i,j}(abs(K{i,j})<0.01*K_ijMax) = 0;

end
end

if status == 1
    disp(['Global Synthesis Success'])
else
    disp(['Global Synthesis Failed'])
end
end

```


The Overall Parametrized Cost Function

```
function gammaVal = synthesizeControllers(platoonObj,x)
    pHat = x(1);
    pBar = x(2);
    gammaHat = x(3);

    [statusL,nuVal,rhoVal,LVal] = synthesizeLocalControllers(gammaHat,pHat,pBar);
    if statusL == 0
        gammaVal = 10000;
        return
    end

    [statusG,gammaVal,KVal] = synthesizeGlobalRobustControllers(platoonObj,gammaHat,pHat,pBar);
    if statusG == 0
        gammaVal = 10000;
        return
    end

    interSum = 0;
    for i = 1:1:size(KVal,1)
        for j = 1:1:size(KVal,2)
            if i~=j
                interSum = interSum + norm(KVal{i,j});
            end
        end
    end

    gammaVal = gammaVal + 0.001*norm(LVal) - 0*interSum;
end
```

Functions for co-design approach with 3 parameters:

For local synthesis (compact):

```
function [status,nuVal,rhoVal,LVal] = synthesizeLocalControllersCompact(pVal)
    % Here we will synthesize the local controllers for local error
    % dynamics to optimize the passivity properties

    errorDynamicsType = 2;
    if errorDynamicsType == 1
        A = [0,1,0;0,0,0;0,0,0]; % For error dynamics type 1
    else
        A = [0,1,0;0,0,1;0,0,0]; % For error dynamics type 2
    end
    B = [0;0;1];
    I = eye(3);
    O = zeros(3);

    % Set up the LMI problem
```

```

solverOptions = sdpsettings('solver','mosek','verbose',0);
P = sdpvar(3,3,'symmetric');
K = sdpvar(1,3,'full');

nu = sdpvar(1,1,'full');
rhoTilde = sdpvar(1,1,'full'); %Representing: 1/rho

gammaSq = sdpvar(1,1,'full');

% For: nuBar < nu < nuHat < 0
nuBar = -gammaSq/pVal;

% For: 0 < rhoHat1,rhoHat2 < rho < rhoBar
% For: 0 < rhoTildeHat < rhoTilde < rhoTildeBar1,rhoTildeBar2
rhoTildeBar1 = 4*gammaSq/pVal;
rhoTildeBar2 = pVal;

% Basic Constraints
con1 = P >= 0;

% Approach 4 with rho = prespecified, nu < 0 and nu is free to maximize
DMat = [rhoTilde*I];
MMat = [P, 0];
ThetaMat = [-A*P-P*A'-B*K-K'*B', -I+0.5*P; -I+0.5*P, -nu*I];
W = [DMat, MMat; MMat', ThetaMat];
con2 = W >= 0;

%%Constraints on resulting nu and rho from the local design
% nuBar < nu < nuHat < 0
con3 = nu >= nuBar; % Helps global design

% 0 < rhoTildeHat < rhoTilde < rhoTildeBar1,rhoTildeBar2
con4 = rhoTilde <= rhoTildeBar1; % Helps global design
con5 = rhoTilde <= rhoTildeBar2; % Helps global design

% Total Cost and Constraints
cons = [con1,con2,con3,con4,con5];
%costFun = 0*(-nu + rhoBar); % For stabilizing, set coefficient to 0
%costFun = 0.0000001*(-nu + rhoBar); % Otherwise set to 0.0000001.
costFun = 0*gammaSq;

% Solution
sol = optimize(cons,costFun,solverOptions);
status = sol.problem == 0; % sol.info;

PVal = value(P);
KVal = value(K);
LVal = KVal/PVal;

```

```

nuVal = value(nu);
rhoVal = 1/value(rhoTilde);
gammaSqVal = value(gammaSq);

if status == 1
    disp(['Local Synthesis Success with gammaSq=',num2str(value(gammaSqVal))])
else
    disp(['Local Synthesis Failed'])
end

end

```

For global synthesis (compact):

```

function [status,gammaSqVal,K] = synthesizeGlobalRobustControllersCompact(platoonObj,nuVal,rhoVal)

% Number of follower vehicles
N = platoonObj.numOfVehicles-1;

% Creating the adjacency matrix, null matrix and cost matrix
G = platoonObj.topology.graph;
A = adjacency(G);
for i = 1:1:N
    for j = 1:1:N
        % Structure of K_ij (which is a 3x3 matrix) should be embedded here
        if i~=j
            if A(j+1,i+1)==1
                adjMatBlock{i,j} = [0,0,0; 0,0,0; 1,1,1];
                nullMatBlock{i,j} = [1,1,1; 1,1,1; 0,0,0];
                costMatBlock{i,j} = 1*[0,0,0; 0,0,0; 1,1,1];
            else
                adjMatBlock{i,j} = [0,0,0; 0,0,0; 0,0,0];
                nullMatBlock{i,j} = [1,1,1; 1,1,1; 0,0,0];
                costMatBlock{i,j} = 10*[0,0,0; 0,0,0; 1,1,1];
            end
        else
            adjMatBlock{i,j} = [0,0,0; 0,0,0; 1,1,1];
            nullMatBlock{i,j} = [1,1,1; 1,1,1; 0,0,0];
            costMatBlock{i,j} = 0*[0,0,0; 0,0,0; 1,1,1];
        end
    end
end
adjMatBlock = cell2mat(adjMatBlock);
nullMatBlock = cell2mat(nullMatBlock);
costMatBlock = cell2mat(costMatBlock);

% Some constants
I = eye(3*N);
I_n = eye(3);
O = zeros(3*N);

```

```

% Whether to use a soft or hard graph constraint
isSoft = 1;
normType = 2;

% Set up the LMI problem
solverOptions = sdpsettings('solver','mosek','verbose',0);

Q = sdpvar(3*N,3*N,'full');
P = sdpvar(N,N,'diagonal');
gammaSq = sdpvar(1,1,'full');

X_p_11 = [];
X_p_12 = [];
X_12 = [];
X_p_22 = [];
for i = 1:1:N
    nu_i = nuVal;
    rho_i = rhoVal;
    X_p_11 = blkdiag(X_p_11,-nu_i*P(i,i)*I_n);
    X_p_12 = blkdiag(X_p_12,0.5*P(i,i)*I_n);
    X_12 = blkdiag(X_12,(-1/(2*nu_i))*I_n);
    X_p_22 = blkdiag(X_p_22,-rho_i*P(i,i)*I_n);
end
X_p_21 = X_p_12';
X_21 = X_12';

% Objective Function
% costFun0 = 1*norm(Q.*costMatBlock,normType);
costFun0 = sum(sum(Q.*costMatBlock));

% Minimum Budget Constraints
con0 = costFun0 >= 1;

% Basic Constraints
con1 = P >= 0;

DMat = [X_p_11, 0; 0, I];
MMat = [Q, X_p_11; I, 0];
ThetaMat = [-X_21*Q-Q'*X_12-X_p_22, -X_p_21; -X_p_12, gammaSq*I];
con2 = [DMat, MMat; MMat', ThetaMat] >= 0; % The real one

% Structural constraints
con3 = Q.*(nullMatBlock==1)==0; % Structural limitations (due to the format of the control)
con4 = Q.*(adjMatBlock==0)==0; % Graph structure : hard constraint

% Total Cost and Constraints
if isSoft
    cons = [con0,con1,con2,con3]; % Without the hard graph constraint con7
    costFun = 1*costFun0 + 1*gammaSq; % soft
else

```

```

    cons = [con0,con1,con2,con3,con4]; % With the hard graph constraint con7
    costFun = 1*costFun0 + 1*gammaSq; % hard (same as soft)
end

sol = optimize(cons,[costFun],solverOptions);
status = sol.problem == 0; %sol.info;

costFun0Val = value(costFun0);
costFunVal = value(costFun);
PVal = value(P);
QVal = value(Q);
X_p_11Val = value(X_p_11);
X_p_21Val = value(X_p_21);

gammaSqVal = value(gammaSq);

M_neVal = X_p_11Val\QVal;

% Obtaining K_ij blocks
M_neVal(nullMatBlock==1) = 0;
maxNorm = 0;
for i = 1:1:N
    for j = 1:1:N
        K{i,j} = M_neVal(3*(i-1)+1:3*i , 3*(j-1)+1:3*j); % (i,j)-th (3 x 3) block
        normVal = max(max(abs(K{i,j})));
        if normVal>maxNorm
            maxNorm = normVal;
        end
    end
end

% filtering out extremely small interconnections
for i=1:1:N
    for j=1:1:N
        if i~=j
            if isSoft
                K{i,j}(abs(K{i,j})<0.0001*maxNorm) = 0;
            else
                if A(i+1,j+1)==0
                    K{i,j} = zeros(3);
                end
            end
        end
    end

    K_ijMax = max(abs(K{i,j}(:)));
    K{i,j}(abs(K{i,j})<0.01*K_ijMax) = 0;
end

if status == 1

```

```

        disp(['Global Synthesis Success with gammaSq=',num2str(value(gammaSqVal))])
    else
        disp(['Global Synthesis Failed'])
    end
end
end

```

The Overall Parametrized Cost Function (Compact):

```

function gammaVal = synthesizeControllersCompact(platoonObj,pVal)

    [statusL,nuVal,rhoVal,LVal] = synthesizeLocalControllersCompact(pVal);
    if statusL == 0
        gammaVal = 1000000;
        return
    else
        [statusG,gammaVal,KVal] = synthesizeGlobalRobustControllersCompact(platoonObj,nuVal,rhoVal,LVal);
        if statusG == 0
            gammaVal = 1000000;
            return
        end
    end
end
end

function [C,Ceq] = synthesizeControllersCompactFeasibility(platoonObj,pVal)

    [statusL,nuVal,rhoVal,LVal] = synthesizeLocalControllersCompact(pVal);
    if statusL == 0
        statusG = 0;
    else
        [statusG,gammaVal,KVal] = synthesizeGlobalRobustControllersCompact(platoonObj,nuVal,rhoVal,LVal);
    end
    statusK = norm(LVal) <= 10000;

    C = [statusL-1; statusG-1; statusK-1];
    Ceq = [];

end

```