

# Formule LMI con codice Matlab YALMIP

Enrico Giordano

## 1 Descrizione del sistema

$$\dot{x} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -B_m/M_m & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -B_s/M_s \end{bmatrix} x + \begin{bmatrix} 0 & 0 \\ 1/M_m & 0 \\ 0 & 0 \\ 0 & 1/M_s \end{bmatrix} u + \begin{bmatrix} 0 & 0 \\ 1/M_m & 0 \\ 0 & 0 \\ 0 & 1/M_s \end{bmatrix} \begin{bmatrix} f_m \\ f_s \end{bmatrix} \quad (1)$$

$$e = \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{bmatrix} x \quad (2)$$

### 1.1 Valori scelti

$$M_s = 0.61$$

$$B_s = 11$$

$$K_v = 40$$

$$K_p = 40$$

$$M_m = 0.64$$

$$B_m = 12$$

### 1.2 Codice

```
1 Bs = 11;
2 Kv = 40;
3 Kp = 40;
4 Mn = 0.64;
5 Bm = 12;
6
7 A = [0 1 0 0; 0 (-Bm/Mn) 0 0; 0 0 0 1; 0 0 0 -(Bs/Ms)];
8 B = [0 0; 1/Mn 0; 0 0; 0 1/Ms];
9 B0 = [0 0; 1/Mn 0; 0 0; 0 1/Ms];
10 C0 = [1 0 -1 0; 0 1 0 -1];
11 C = [0 1 0 0; 0 0 0 1];
```

## 2 Passività 2x2

$$\begin{bmatrix} QA^T + M^T B^T + AQ + BM & B - QC^T \\ B^T - CQ & 0 \end{bmatrix} \prec 0 \quad (3)$$

$$Q \succ 0 \quad (4)$$

$$K = MP = MQ^{-1}$$

### 2.1 Codice

```

1 BlockUpLeft = Q*A'+M'*B0'+A*Q+B0*M;
2 BlockUpRight = B0-Q*C';
3 BlockDownLeft = B0'-C*Q;
4 BlockDownRight = zeros(size(BlockUpRight, 2), size(
    BlockDownLeft, 1));
5
6 LMI = [BlockUpLeft BlockUpRight; BlockDownLeft
    BlockDownRight];
7
8 F = [Q >= 0, LMI <= 0];
9
10 diagnostics = solvesdp(F);
11 disp(diagnostics.problem);
12 if diagnostics.problem == 0
13     disp('Feasible')
14     Q_s = value(Q);
15     M_s = value(M);
16
17     K = M_s * Q_s'
18
19 elseif diagnostics.problem == 1
20     disp('Infeasible')
21 else
22     disp('Something else happened')
23 end

```

## 3 Passività 3x3

$$\begin{bmatrix} I & 0 \\ A & B \\ C & D \end{bmatrix}^T \begin{bmatrix} 0 & X & 0 \\ X & 0 & 0 \\ 0 & 0 & P \end{bmatrix} \begin{bmatrix} I & 0 \\ A & B \\ C & D \end{bmatrix}^T \succ 0 \quad (5)$$

$$Q \succ 0 \quad (6)$$

### 3.1 Codice

```

1 n = size(A,1)
2 p = size(B,2)
3 P = sdpvar(size(A,1),size(A,1), 'symmetric');
4
5 F = [P >= 0, [[eye(n,n) zeros(n,p);A B; C D]'*[zeros(n,n)
    P zeros(n,p); P zeros(n,n) zeros(n,p); zeros(p,n)
    zeros(p,n) eye(p,p) ]*[eye(n,n) zeros(n,p);A B; C D]]
    >= 0];
6
7 diagnostics = solvesdp(F)
8 disp(diagnostics.problem)
9 if diagnostics.problem == 0
10     disp('Feasible')
11     solution = value(P)
12 elseif diagnostics.problem == 1
13     disp('Infeasible')
14 else
15     disp('Something else happened')
16 end

```

## 4 Sintesi

$$\begin{bmatrix} (AY + BM)^T + (AY + BM) & B_0 & (C_0Y + EM)^T \\ B_0^T & \gamma^2 I & D_0^T \\ (C_0Y + EM) & D_0 & -I \end{bmatrix} \prec 0 \quad (7)$$

$$Y \succ 0 \quad (8)$$

$$D_c = K$$

$$E = 0$$

$$D_0 = 0$$

### 4.1 Codice

```

1
2 n = size(A,1)
3 p = size(B,1)
4 q = size(B,2)
5
6 Q = sdpvar(n,n, 'symmetric');
7 M = sdpvar(q,p);
8 K = sdpvar(q,p);

```

```

9  E = zeros(q,q);
10 D0 = E;
11 Acl = A;
12
13 gamma = 0.001;% valore piu' piccolo possibile (es:
    0.000041)
14
15 b11 = (A*Q + B*M)' + (A*Q + B*M);
16 b12 = B0;
17 b13 = (C0*Q + E*M)';
18 b21 = B0';
19 b22 = -gamma^2 * eye(2);
20 b23 = D0';
21 b31 = (C0*Q + E*M);
22 b32 = D0;
23 b33 = -eye(2);
24
25 LMI = [b11 b12 b13; b21 b22 b23; b31 b32 b33];
26
27 F = [Q >= 0, LMI <=0];

```

## 5 Codice completo

```

1
2 Ms = 0.61;
3 Bs = 11;
4 Kv = 40;
5 Kp = 40;
6 Mn = 0.64;
7 Bm = 12;
8
9 s = tf('s');
10
11 A = [0 1 0 0; 0 (-Bm/Mn) 0 0; 0 0 0 1; 0 0 0 -(Bs/Ms)];
12 B = [0 0; 1/Mn 0; 0 0; 0 1/Ms];
13 B0 = [0 0; 1/Mn 0; 0 0; 0 1/Ms];
14 C0 = [1 0 -1 0; 0 1 0 -1];
15 C = [0 1 0 0; 0 0 0 1];
16
17 n = size(A,1)
18 p = size(B,1)
19 q = size(B,2)
20
21 Q = sdpvar(n,n, 'symmetric');
22 M = sdpvar(q,p);
23 K = sdpvar(q,p);

```

```

24 E = zeros(q,q);
25 D0 = E;
26 Acl = A;
27
28 gamma = 0.001;% valore piu' piccolo possibile (es:
    0.000041)
29
30 b11 = (A*Q + B*M)' + (A*Q + B*M);
31 b12 = B0;
32 b13 = (C0*Q + E*M)';
33 b21 = B0';
34 b22 = -gamma^2 * eye(2);
35 b23 = D0';
36 b31 = (C0*Q + E*M);
37 b32 = D0;
38 b33 = -eye(2);
39
40 LMI1 = [b11 b12 b13; b21 b22 b23; b31 b32 b33];
41
42
43 BlockUpLeft = Q*Acl' + M'*B0' + Acl*Q + B0*M;
44 BlockUpRight = B0 - Q*C';
45 BlockDownLeft = B0' - C*Q;
46 BlockDownRight = zeros(size(BlockUpRight, 2), size(
    BlockDownLeft, 1));
47
48 LMI2 = [BlockUpLeft BlockUpRight; BlockDownLeft
    BlockDownRight];
49
50 F = [Q >= 0, LMI1 <= 0, LMI2 <= 0];
51
52 diagnostics = solvesdp(F);
53 disp(diagnostics.problem);
54 if diagnostics.problem == 0
55     disp('Feasible')
56     Q_s = value(Q);
57     M_s = value(M);
58
59     K = M_s * Q_s'
60
61 elseif diagnostics.problem == 1
62     disp('Infeasible')
63 else
64     disp('Something else happened')
65 end

```