Performance Certification of a Nonlinear Interconnected System

In this example the ADMM algorithm described in Section 6.2 is used to certify a bound on the L_2 gain of an interconnection of randomly generated nonlinear subsystems. For demonstrative purposes the subsystems and interconnection matrix are constructed such that the interconnected system has an L_2 gain less than or equal to 1. **Each time this example is run the number of iterations required of the ADMM algorithm will vary depending on the random subsystem and interconnection data.**

Contents

Requirements

This script requires the CVX toolbox: http://cvxr.com/cvx/ and the SOSAnalysis toolbox: http://www.aem.umn.edu/~AerospaceControl/

Polynomial Subsystems

The state equations for each subsystem are

$$\dot{x}_1(t) = x_2(t), \quad \dot{x}_2(t) = -ax_2(t) - bx_1^3(t) + u_2(t)$$

where a and b are parameters. The output is

$$y(t) = x_2(t)$$

It can be shown that the L_2 gain of this system for all values of b is less than or equal to $1/a$. This fact will be used to construct an interconnected system with L_2 gain less than or equal to 1.

Interconnection and Subsystem Data

```
% Number of subsystems
```

```
N = 20;

% Number of states, inputs, and outputs for each subsystem
nx = 2;
nu = 1;
ny = 1;
```

Random Input and Output Scalings

Multiplicative scalings are applied to the input and output of the subsystems and the interconnection. These scalings do not effect the properties of the interconnected system, but disguise the construction of the system making the performance certification more difficult.

```
% Scaling randomly chosen between 0.25 and 4. Different scalings can be
% used, but they will effect the number of iterations required for the
% algorithm to certify performance.
u_scale = 3.75*rand(N,1)+0.25;
y_scale = 3.75*rand(N,1)+0.25;
```

Generate Polynomial Subsystems

The parameters a and b for each subsystem are chosen randomly. So that each subsystem has L_2 gain less than 1, the parameter a is chosen from a uniform distribution between 1 and 2.

```
% Initialize State and Input variables
x = mpvar('x', nx, N);
u = mpvar('u', nu, N);

% Generate random subsystem parameters
a = rand(N,1)+1;
b = rand(N,1);

% Initialize Polynomials for the Subsystem Dynamics and Output
f = polynomial(zeros(2,N));
h = polynomial(zeros(1,N));

% Create Subsystems
for k = 1:N
    % Subsystem Dynamics with input scaling
    f(1,k) = x(2,k);
    f(2,k) = -a(k)*x(2,k)-b(k)*x(2,k)^3+u(k)/u_scale(k);

    % Subsystem Output with output scaling
    h(k) = x(2,k)/y_scale(k);
end
```

Interconnection Matrix

The interconnection matrix is constructed to have norm equal to 1. Since each subsystem has L_2 gain less than 1, the resulting interconnected system will also have L_2 gain less than 1.

```
nd = 1; % Number of disturbances
```

```
ne = 1; % Number of exogenous outputs

% Generate random interconnection matrix
M = randn(N+nd, N+ne);

% Scale the interconnection matrix such that it has norm equal to 1
M = M/norm(M);

% Scale the input and output of the interconnection matrix
M = blkdiag(diag(u_scale), eye(nd))*M*blkdiag(diag(y_scale), eye(ne));
```

Performance Criteria

We want to certify the interconnected systems has L_2 gain less than or equal to 1. This property can be characterized by a quadratic supply rate parameterized by the matrix W as defined below.

```
W = [eye(nd) zeros(nd,ne); zeros(ne,nd) -eye(ne)];
```

Define and Initialize variables for the ADMM Algorithm

```
% Allocate variables to store supply rate matrices for each subsystem
X = zeros(nu+ny, nu+ny, N);
Z = zeros(nu+ny, nu+ny, N);
S = zeros(nu+ny, nu+ny, N);

% Initialize Z and S to be identity matrices. X does not need to be
% initialized.
for k = 1:N
    Z(:,:,k) = eye(nu+ny);
    S(:,:,k) = eye(nu+ny);
end
```

ADMM Algorithm

The ADMM algorithm as described in Section 6.2 is implemented below to search for storage functions and supply rates certifying performance of the interconnected system

```
% Initialize variables
max_iterations = 100;
feas = false;
iteration = 1;

% Run the ADMM algorithm until a feasible solution is found certifying the
% interconnected system is dissipative with repect to the supply rate
% parameterized by the matrix W.
while ~feas && iteration <= max_iterations

    % X - Update for each subsystem
    for k = 1:N
        X(:,:,k) = ADMM_X_Update(f(:,k), h(k), x(:,k), u(k), Z(:,:,k), S(:,:,k));
    end
```

```
    % Check feasibility of the constraint in Equation 6.6.
    [feas, evals] = ADMM_CheckFeasibility(X, W, M);

    % Display the largest eigenvalue of the matrix in Equation 6.6 at
    % each iteration. Dissipativity is certified when the maximum
    % eigenvalue is less than or equal to 0.
    disp([' Iteration: ' int2str(iteration)] )
    disp(['    Maximum Eigenvalue: ' num2str(max(evals)) ])

    % Z - Update simultaneously for all subsystems
    Z = ADMM_Z_Update(X, S, M, W);

    % U - Update for each subsystem
    for k = 1:N
        S(:,:,k) = X(:,:,k)-Z(:,:,k)+S(:,:,k);
    end
    iteration = iteration + 1;
end
```

```
 Iteration: 1
    Maximum Eigenvalue: 1.2101
 Iteration: 2
    Maximum Eigenvalue: 1.8463
 Iteration: 3
    Maximum Eigenvalue: 0.78472
 Iteration: 4
    Maximum Eigenvalue: 0.58471
 Iteration: 5
    Maximum Eigenvalue: 0.31475
 Iteration: 6
    Maximum Eigenvalue: 0.18705
 Iteration: 7
    Maximum Eigenvalue: 0.10384
 Iteration: 8
    Maximum Eigenvalue: 0.050276
 Iteration: 9
    Maximum Eigenvalue: 0.00493
 Iteration: 10
    Maximum Eigenvalue: -0.031215
```

Display Results

```
if feas
   disp(['Dissipativity of the interconnected system is certified in ' int2str(iteration-1) ' itera
tions using ADMM.'])
else
   disp('Failed to certify dissipativity. More iterations are required.')
end
```

```
Dissipativity of the interconnected system is certified in 10 iterations using ADMM.
```

Reinitialize Variables

```
% Allocate variables to store supply rate matrices for each subsystem
X = zeros(nu+ny, nu+ny, N);
Z = zeros(nu+ny, nu+ny, N);
S = zeros(nu+ny, nu+ny, N);

% Initialize Z and S to be identity matrices. X does not need to be
% initialiazed.
for k = 1:N
    Z(:,:,k) = eye(nu+ny);
    S(:,:,k) = eye(nu+ny);
end
```

ADMM Algorithm with Relaxed Exit Criterion

Now we apply the ADMM algorithm with the relaxed exit criterion as described in Section 6.2 to the same problem. The only modification to the ADMM algortihm is that the function 'ADMM_CheckFeasibility' in line 132 is replaced by the function 'ADMM_CheckFeasibility_RelaxedCrit'.

```
% Initialize variables
feas = false;
iteration = 1;

% Run the ADMM algorithm until a feasible solution is found certifying the
% interconnected system is dissipative with repect to the supply rate
% parameterized by the matrix W.
while ~feas && iteration <= max_iterations

    % X - Update for each subsystem
    for k = 1:N
        X(:,:,k) = ADMM_X_Update(f(:,k), h(k), x(:,k), u(k), Z(:,:,k), S(:,:,k));
    end

    % Check feasibility of the constraint in Equation 6.6 with the relaxed
    % exit criterion: allow each supply rate matrix to be scaled by a
    % nonnegative scalar.
    [feas, evals] = ADMM_CheckFeasibility_RelaxedCrit(X, W, M);

    % Display the largest eigenvalue of the matrix described in Equation
    % 6.6 with the relaxed exit criterion. Dissipativity is certified when
    % the maximum eigenvalue is less than or equal to 0.
    disp([' Iteration: ' int2str(iteration)] )
    disp(['    Maximum Eigenvalue: ' num2str(max(evals)) ])

    % Z - Update simultaneously for all subsystems
    Z = ADMM_Z_Update(X, S, M, W);

    % U - Update for each subsystem
    for k = 1:N
        S(:,:,k) = X(:,:,k)-Z(:,:,k)+S(:,:,k);
    end
    iteration = iteration+1;
end
```

```
Iteration: 1
   Maximum Eigenvalue: 1.2101
Iteration: 2
   Maximum Eigenvalue: 0.24113
Iteration: 3
   Maximum Eigenvalue: 0.22783
Iteration: 4
   Maximum Eigenvalue: 0.032969
Iteration: 5
   Maximum Eigenvalue: -0.75123
```

Display Results

```
if feas
    disp(['Dissipativity of the interconnected system is certified in ' int2str(iteration-1) ' iter
ations using ADMM with relaxed exit criterion.'])
else
   disp('Failed to certify dissipativity. More iterations are required.')
end
```

```
Dissipativity of the interconnected system is certified in 5 iterations using ADMM with relaxed exi
t criterion.
```

Conclusion

A nonlinear interconnected system was constructed such that it has L_2 gain less than 1. Input-output scalings were added to the subsystems and interconnection matrix to disguise this construction making performance certification of the interconnected system more difficult. The ADMM algorithm as described in Section 6.2 is used to determine supply rates certifying that the interconnected system has L_2 gain less than 1. Additionally, the ADMM algorithm with the relaxed exit criterion was also used. Typically, it requires significantly fewer iterations to certify the same performance.

Attribution

This example supplements the book "Networks of Dissipative Systems: Compositional Certification of Stability, Performance, and Safety" by Murat Arcak, Chris Meissen, and Andrew Packard.