Semidefinite Programming

Semidefinite programming sdptutorial.m

This example illustrates the definition and solution of a simple semidefinite programming problem.

Given a linear dynamic system $\dot{x} = Ax$, our goal is to prove stability by finding a symmetric matrix P satisfying

$$A^T P + PA \preceq 0$$
$$P \succeq 0$$

Define a stable matrix A and symmetric matrix P (remember: square matrices are symmetric by default)

```
A = [-1 2 0;-3 -4 1;0 0 -2];
P = sdpvar(3,3);
```

Having defined P, we are ready to define the semidefinite constraints.

```
F = [P >= 0, A'*P+P*A <= 0];
```

To avoid the zero solution on this homogeneous problem, we constrain the trace of the matrix (Of course, this is not the only way. We could have used, e.g., the dehomogenizing constraint $P \succeq I$ instead)

```
F = [F, trace(P) == 1];
```

At this point, we are ready to solve our problem. But first, we display the collection of constraints to see what we have defined.

```
F
++++++++++++++++++++++++++++++++++++++++++++++++++++
|   ID|      Constraint|                      Type|
++++++++++++++++++++++++++++++++++++++++++++++++++++
|   #1|   Numeric value|     Matrix inequality 3x3|
|   #2|   Numeric value|     Matrix inequality 3x3|
|   #3|   Numeric value|   Equality constraint 1x1|
++++++++++++++++++++++++++++++++++++++++++++++++++++
```

We only need a feasible solution, so one argument is sufficient when we call solvesdp to solve the problem.

```
solvesdp(F);
Pfeasible = double(P);
```

The resulting constraint satisfaction is easily investigated with checkset.

```
checkset(F)
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
|   ID|      Constraint|                 Type|   Primal residual|   Dual 
residual|
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
|   #1|   Numeric value|     Matrix inequality|           0.20138|
8.2785e-016|
|   #2|   Numeric value|     Matrix inequality|            1.1397|
3.6687e-016|
|   #3|   Numeric value|   Equality constraint|       -2.276e-015|
 -8.1801e-016|
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Minimizing, e.g., the top-left element of P is done by specifying an objective function.

```
F = [P >= 0, A'*P+P*A <= 0, trace(P)==1];
solvesdp(F,P(1,1));
```

We can easily add additional linear inequality constraints. If we want to add the constraint that all off-

diagonal elements are larger than zero, one approach is (remember, standard MATLAB indexing applies)

```
F = [P >= 0, A'*P+P*A <= 0, trace(P)==1, P([2 3 6])>=0];
solvesdp(F,P(1,1));
```

Since the variable P([2 3 6]) is a vector, the constraint is interpreted as a standard linear inequality, according to the rules introduced in the basic tutorial.