

Polynomials

- ▶ Given $\alpha \in \mathbb{N}^n$, a monomial in n variables is a function $m_\alpha : \mathbb{R}^n \rightarrow \mathbb{R}$ defined as $m_\alpha(x) := x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_n^{\alpha_n}$.
- ▶ The degree of a monomial is defined as $\deg m_\alpha := \sum_{i=1}^n \alpha_i$.
- ▶ A polynomial in n variables is a function $p : \mathbb{R}^n \rightarrow \mathbb{R}$ defined as a finite linear combination of monomials:

$$p := \sum_{\alpha \in \mathcal{A}} c_\alpha m_\alpha = \sum_{\alpha \in \mathcal{A}} c_\alpha x^\alpha$$

where $\mathcal{A} \subset \mathbb{N}^n$ is a finite set and $c_\alpha \in \mathbb{R} \forall \alpha \in \mathcal{A}$.

- ▶ The set of polynomials in n variables $\{x_1, \dots, x_n\}$ will be denoted $\mathbb{R}[x_1, \dots, x_n]$ or, more compactly, $\mathbb{R}[x]$.
- ▶ The degree of a polynomial f is defined as $\deg f := \max_{\alpha \in \mathcal{A}, c_\alpha \neq 0} \deg m_\alpha$.
- ▶ $\theta \in \mathbb{R}[x]$ will denote the zero polynomial, i.e. $\theta(x) = 0 \forall x$.

Multipoly Toolbox

- ▶ Multipoly is a Matlab toolbox for the creation and manipulation of polynomials of one or more variables.

- ▶ Example:

```
pvar x1 x2
```

```
p = 2*x1^4 + 2*x1^3*x2 - x1^2*x2^2 + 5*x2^4
```

- ▶ A scalar polynomial of T terms and V variables is stored as a $T \times 1$ vector of coefficients, a $T \times V$ matrix of natural numbers, and a $V \times 1$ array of variable names.

$$\text{p.coef} = \begin{bmatrix} 2 \\ 2 \\ -1 \\ 5 \end{bmatrix}, \quad \text{p.deg} = \begin{bmatrix} 4 & 0 \\ 3 & 1 \\ 2 & 2 \\ 0 & 4 \end{bmatrix}, \quad \text{p.var} = \begin{bmatrix} \text{x1} \\ \text{x2} \end{bmatrix}$$

Vector Representation

- ▶ If p is a polynomial of degree $\leq d$ in n variables then there exists a coefficient vector $c \in \mathbb{R}^{l_w}$ such that $p = c^T w$ where

$$w(x) := [1, x_1, x_2, \dots, x_n, x_1^2, x_1x_2, \dots, x_n^2, \dots, x_n^d]^T$$

l_w denotes the length of w . It is easy to verify $l_w = \binom{n+d}{d}$.

- ▶ Example:

$$p = 2*x1^4 + 2*x1^3*x2 - x1^2*x2^2 + 5*x2^4;$$

$$x = [x1;x2];$$

$$w = \text{monomials}(x,0:4);$$

$$c = \text{poly2basis}(p,w);$$

$$p - c'*w$$

$$[c \ w]$$

Gram Matrix Representation

- ▶ If p is a polynomial of degree $\leq 2d$ in n variables then there exists a $Q \in \mathcal{S}^{l_z \times l_z}$ such that $p = z^T Q z$ where

$$z := [1, x_1, x_2, \dots, x_n, x_1^2, x_1 x_2, \dots, x_n^2, \dots, x_n^d]^T$$

The dimension of z is $l_z = \binom{n+d}{d}$.

- ▶ Equating coefficients of p and $z^T Q z$ yields linear equality constraints on the entries of Q
 - ▶ Define $q := \text{vec}(Q)$ and $l_w := \binom{n+2d}{2d}$.
 - ▶ There exists $A \in \mathbb{R}^{l_w \times l_z^2}$ and $c \in \mathbb{R}^{l_w}$ such that $p = z^T Q z$ (i.e. $p(x) - z(x)^T Q z(x) \equiv \theta$) is equivalent to $Aq = c$.
- ▶ There are $h := \frac{l_z(l_z+1)}{2} - l_w$ linearly independent homogeneous solutions $\{N_i\}_{i=1}^h$ each of which satisfies $z^T N_i z = \theta(x)$.
- ▶ Summary: All solutions to $p = z^T Q z$ can be expressed as the sum of a particular solution and a homogeneous solution. The set of homogeneous solutions depends on n and d while the particular solution depends on p .

Gram Matrix Example

```
p = 2*x1^4 + 2*x1^3*x2 - x1^2*x2^2 + 5*x2^4;  
[z,c,A,w] = gramconstraint(p);  
p-c'*w  
Q = full(reshape(A\c,[3 3]));  
p-z'*Q*z
```

% Q is a particular solution in vectorized form

% Each column of N is a homogenous solution in vectorized form.

```
[z,Q,N] = gramsol(p);  
Q = full(reshape(Q,[3 3]));  
N = full(reshape(N,[3 3]));  
p-z'*Q*z  
z'*N*z
```

$$z = \begin{bmatrix} x_1^2 \\ x_1 x_2 \\ x_2^2 \end{bmatrix}, \quad Q = \begin{bmatrix} 2 & 1 & -0.5 \\ 1 & 0 & 0 \\ -0.5 & 0 & 5 \end{bmatrix}, \quad N = \begin{bmatrix} 0 & 0 & -0.5 \\ 0 & 1 & 0 \\ -0.5 & 0 & 0 \end{bmatrix}$$

Positive Semidefinite Polynomials

- ▶ $p \in \mathbb{R}[x]$ is positive semi-definite (PSD) if $p(x) \geq 0 \forall x$. The set of PSD polynomials in n variables $\{x_1, \dots, x_n\}$ will be denoted $\mathcal{P}[x_1, \dots, x_n]$ or $\mathcal{P}[x]$.
- ▶ Testing if $p \in \mathcal{P}[x]$ is NP-hard when the polynomial degree is at least four.
 - ▶ For a general class of functions, verifying global non-negativity is recursively undecidable.
- ▶ Our nonlinear analysis tools (to be presented) require a computational procedure for constructing polynomials that are PSD.
- ▶ Objective: Given $p \in \mathbb{R}[x]$, we would like a polynomial-time sufficient condition for testing if $p \in \mathcal{P}[x]$.

Reference: Parrilo, P., *Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization*, Ph.D. thesis, California Institute of Technology, 2000. (Chapter 4 of this thesis and the reference contained therein summarize the computational issues associated with verifying global non-negativity of functions.)

Sum of Squares Polynomials

- ▶ p is a sum of squares (SOS) if there exist polynomials $\{f_i\}_{i=1}^N$ such that $p = \sum_{i=1}^N f_i^2$.
- ▶ The set of SOS polynomials in n variables $\{x_1, \dots, x_n\}$ will be denoted $\Sigma[x_1, \dots, x_n]$ or $\Sigma[x]$.
- ▶ If p is a SOS then p is PSD.
 - ▶ The Motzkin polynomial, $p = x^2y^4 + x^4y^2 + 1 - 3x^2y^2$, is PSD but not SOS.
 - ▶ Hilbert (1888) showed that $\mathcal{P}[x] = \Sigma[x]$ only for a) $n = 1$, b) degree= 2, and c) degree= 4, $n = 2$.
- ▶ p is a SOS iff there exists $Q \succeq 0$ such that $p = z^T Q z$.

Proof:

$$\begin{aligned} p \text{ is SOS} &\leftrightarrow \exists \text{ polynomials } \{f_i\}_{i=1}^N \text{ such that } p = \sum_{i=1}^N f_i^2 \\ &\leftrightarrow \exists \{L_i\}_{i=1}^N \subseteq \mathbb{R}^{l_z} \text{ such that } p = \sum_{i=1}^N (L_i z)^2 \\ &\leftrightarrow \exists L \in \mathbb{R}^{N \times l_z} \text{ such that } p = z^T L^T L z \\ &\leftrightarrow \exists Q \succeq 0 \text{ such that } p = z^T Q z \end{aligned}$$

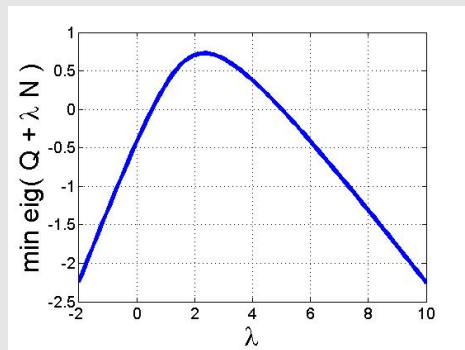
SOS Example (1)

All possible Gram matrix representations of

$$p = 2x_1^4 + 2x_1^3x_2 - x_1^2x_2^2 + 5x_2^4$$

are given by $z^T (Q + \lambda N) z$ where:

$$z = \begin{bmatrix} x_1^2 \\ x_1x_2 \\ x_2^2 \end{bmatrix}, \quad Q = \begin{bmatrix} 2 & 1 & -0.5 \\ 1 & 0 & 0 \\ -0.5 & 0 & 5 \end{bmatrix}, \quad N = \begin{bmatrix} 0 & 0 & -0.5 \\ 0 & 1 & 0 \\ -0.5 & 0 & 0 \end{bmatrix}$$



p is SOS iff

$$Q + \lambda N \succeq 0$$

for some $\lambda \in \mathbb{R}$.

SOS Example (2)

p is SOS since $Q + \lambda N \succeq 0$ for $\lambda = 5$.

An SOS decomposition can be constructed from a Cholesky factorization:

$$Q + 5N = L^T L$$

where:

$$L = \frac{1}{\sqrt{2}} \begin{bmatrix} 2 & 1 & -3 \\ 0 & 3 & 1 \end{bmatrix}$$

Thus

$$\begin{aligned} p &= 2x_1^4 + 2x_1^3x_2 - x_1^2x_2^2 + 5x_2^4 \\ &= (Lz)^T (Lz) \\ &= \frac{1}{2} (2x_1^2 - 3x_2^2 + x_1x_2)^2 + \frac{1}{2} (x_2^2 + 3x_1x_2)^2 \in \Sigma[x] \end{aligned}$$

Gram Matrix Rank

- ▶ The number of terms in the SOS decomposition is equal to the rank of the Gram matrix.
 - ▶ In the previous example $Q + 5N \succeq 0$ has rank = 2 and the SOS decomposition has two terms.
 - ▶ For $\lambda = 2.5$, $Q + 2.5N \succ 0$ has rank = 3 and the SOS decomposition has three terms.
- ▶ Low rank Gram matrix solutions are positive semidefinite but not strictly positive definite.
- ▶ For some problems, the feasible solution set is low-dimension and consists only of low-rank Gram matrix solutions. This can cause some numerical difficulties.

Connection to LMIs

Checking if a given polynomial p is a SOS can be done by solving a linear matrix inequality (LMI) feasibility problem.

1. Primal (Image) Form:

- ▶ Find $A \in \mathbb{R}^{l_w \times l_z^2}$ and $c \in \mathbb{R}^{l_w}$ such that $p = z^T Q z$ is equivalent to $Aq = c$ where $q = \text{vec}(Q)$.
- ▶ p is a SOS if and only if there exists $Q \succeq 0$ such that $Aq = c$.

2. Dual (Kernel) Form:

- ▶ Let Q_0 be a particular solution of $p = z^T Q z$ and let $\{N_i\}_{i=1}^h$ be a basis for the homogeneous solutions.
- ▶ p is a SOS if and only if there exists $\lambda \in \mathbb{R}^h$ such that $Q_0 + \sum_{i=1}^h \lambda_i N_i \succeq 0$.

Complexity of SOS LMI Feasibility Problem

If p is a degree $2d$ polynomial in n variables then the dimensions of the primal or dual LMI test for $p \in \Sigma[x]$ can be computed with the function `sossize`:

```
n = 2;      % Number of variables in polynomial
deg = 10;   % Degree of polynomial
[lz,lw,h]=sossize(n,deg)
```

$l_z = \binom{n+d}{d}$	$2d=4$	6	8	10
$n = 2$	6	10	15	21
5	21	56	126	252
9	55	220	715	2002
14	120	680	3060	11628
16	153	969	4845	20349

$l_w = \binom{n+2d}{2d}$	$2d=4$	6	8	10
$n = 2$	15	28	45	66
5	126	462	1287	3003
9	715	5005	24310	92378
14	3060	38760	319770	1961256
16	4845	74613	735471	5311735

$h = \frac{l_z(l_z+1)}{2} - l_w$	$2d=4$	6	8	10
$n = 2$	6	27	75	165
5	105	1134	6714	28875
9	825	19305	231660	1912625
14	4200	192780	4363560	65649750
16	6936	395352	11003964	201739340

SOS Test with `issos`

The `issos` function tests if $p \in \Sigma[x]$ by converting to an LMI feasibility problem:

$$[\text{feas}, z, Q, f] = \text{issos}(p)$$

`feas=1` if $p \in \Sigma[x]$ and `feas=0` otherwise. If feasible, then

- ▶ `z` and `Q` provide a Gram matrix decomposition:

$$p = z' * Q * z,$$

where `z` is a vector of monomials and `Q` is a positive semidefinite matrix.

- ▶ `z` may not include the complete list of $\binom{n+d}{d}$ monomials since `issos` uses some simple heuristics to prune out un-needed monomials.
- ▶ `f` is a vector of polynomials providing the SOS decomposition:

$$p = f' * f,$$

SOS Example using issos

```
>> pvar x1 x2;  
>> p = 2*x1^4 + 2*x1^3*x2 - x1^2*x2^2 + 5*x2^4;  
>> [feas,z,Q,f]=issos(p);
```

```
% Verify feasibility of p \in SOS
```

```
>> feas
```

```
feas =
```

```
1
```

```
% Verify z and Q are a Gram matrix decomposition
```

```
>> p - z'*Q*z
```

```
ans =
```

```
-1.3185e-012*x1^4 + 6.5814e-013*x1^3*x2 - 2.3075e-012*x1^2*x2^2 +  
5.6835e-016*x1*x2^3 - 3.304e-013*x2^4
```

```
% Verify Q is positive semi-definite
```

```
>> min(eig(Q))
```

```
ans =
```

```
0.7271
```

```
% Verify SOS decomposition of p
```

```
>> p - f'*f
```

```
ans =
```

```
-1.3221e-012*x1^4 + 6.5148e-013*x1^3*x2 - 2.3106e-012*x1^2*x2^2 +  
1.3323e-015*x1*x2^3 - 3.3396e-013*x2^4
```

SOS Feasibility

SOS Feasibility: Given polynomials $\{f_k\}_{k=0}^m$, does there exist $\alpha \in \mathbb{R}^m$ such that $f_0 + \sum_{k=1}^m \alpha_k f_k$ is a SOS?

The SOS feasibility problem can also be posed as an LMI feasibility problem since α enters linearly.

1. Primal (Image) Form:

- ▶ Find $A \in \mathbb{R}^{l_w \times l_z^2}$ and $c_k \in \mathbb{R}^{l_w}$ such that $f_k = z^T Q z$ is equivalent to $Aq = c_k$ where $q = \text{vec}(Q)$.
- ▶ Define $C := -[c_1, c_2, \dots, c_m] \in \mathbb{R}^{l_w \times m}$.
- ▶ There is an $\alpha \in \mathbb{R}^m$ such that $f_0 + \sum_{k=1}^m \alpha_k f_k$ is a SOS iff there exists $\alpha \in \mathbb{R}^m$ and $Q \succeq 0$ such that $Aq + C\alpha = c_0$

2. Dual (Kernel) Form:

- ▶ Let Q_k be particular solutions of $f_k = z^T Q z$ and let $\{N_i\}_{i=1}^h$ be a basis for the homogeneous solutions.
- ▶ There is an $\alpha \in \mathbb{R}^m$ such that $f_0 + \sum_{k=1}^m \alpha_k f_k$ is a SOS iff there exists $\alpha \in \mathbb{R}^m$ and $\lambda \in \mathbb{R}^h$ such that $Q_0 + \sum_{k=1}^m \alpha_k Q_k + \sum_{i=1}^h \lambda_i N_i \succeq 0$.

SOS Programming

SOS Programming: Given $c \in \mathbb{R}^m$ and polynomials $\{f_k\}_{k=0}^m$, solve:

$$\min_{\alpha \in \mathbb{R}^m} c^T \alpha$$

subject to:

$$f_0 + \sum_{k=1}^m \alpha_k f_k \in \Sigma[x]$$

This SOS programming problem is an SDP.

- ▶ The cost is a linear function of α .
- ▶ The SOS constraint can be replaced with either the primal or dual form LMI constraint.

A more general SOS program can have many SOS constraints.

General SOS Programming

SOS Programming: Given $c \in \mathbb{R}^m$ and polynomials $\{f_{j,k}\}_{j=1}^{N_s} \quad m$
solve:

$$\min_{\alpha \in \mathbb{R}^m} c^T \alpha$$

subject to:

$$f_{1,0}(x) + f_{1,1}(x)\alpha_1 + \cdots + f_{1,m}(x)\alpha_m \in \Sigma[x]$$

\vdots

$$f_{N_s,0}(x) + f_{N_s,1}(x)\alpha_1 + \cdots + f_{N_s,m}(x)\alpha_m \in \Sigma[x]$$

There is freely available software (e.g. SOSTOOLS, YALMIP, SOSOPT) that:

1. Converts the SOS program to an SDP
2. Solves the SDP with available SDP codes (e.g. Sedumi)
3. Converts the SDP results back into polynomial solutions

SOS Programming with sosopt

- ▶ SOS programs can be solved with
`[info,dopt,ssosol] = sosopt(sosconstr,x,obj)`
 - ▶ `sosconstr` is a cell array of polynomials constrained to be SOS.
 - ▶ `x` is a vector of the independent (polynomial) variables.
 - ▶ `obj` is the objective function to be minimized. `obj` must be a linear function of the decision variables.
 - ▶ Feasibility of the problem is returned in `info.feas`.
 - ▶ Decision variables are returned in `dopt`.
 - ▶ `ssosol` provides a Gram decomposition for each constraint.
- ▶ Use `Z=monomials(vars,deg)` to generate a vector of all monomials in specified variables and degree.
- ▶ Use `p=polydecvar(dstr,Z,type)` to create a polynomial decision variable `p`.
 - ▶ If `type='vec'` then `p` has the form $p = D * Z$ where `D` is a column vector of decision variable coefficients.
 - ▶ If `type='mat'` then `p` has the form $p = Z * D * Z$ where `D` is a symmetric matrix of decision variable coefficients.
 - ▶ Note: For efficient implementations, only use the `'mat'` if `p` is constrained to be SOS. `p` must then be included in `sosconstr`. Do not use the `'mat'` form if `p` is not SOS constrained.

SOS Synthesis Example (1)

Problem: Minimize α subject to $f_0 + \alpha f_1 \in \Sigma[x]$ where

$$f_0(x) := -x_1^4 + 2x_1^3x_2 + 9x_1^2x_2^2 - 2x_2^4$$

$$f_1(x) := x_1^4 + x_2^4$$

For every $\alpha, \lambda \in \mathbb{R}$, the Gram Matrix Decomposition equality holds:

$$f_0 + \alpha f_1 = z^T (Q_0 + \alpha Q_1 + \lambda N_1) z$$

where

$$z := \begin{bmatrix} x_1^2 \\ x_1x_2 \\ x_2^2 \end{bmatrix}, Q_0 = \begin{bmatrix} -1 & 1 & 4.5 \\ 1 & 0 & 0 \\ 4.5 & 0 & -2 \end{bmatrix}, Q_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, N_1 = \begin{bmatrix} 0 & 0 & -0.5 \\ 0 & 1 & 0 \\ -0.5 & 0 & 0 \end{bmatrix}$$

If $\alpha = 2$ and $\lambda = 0$ then $Q_0 + 2Q_1 + 9N_1 = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 9 & 0 \\ 0 & 0 & 0 \end{bmatrix} \succeq 0$.

SOS Synthesis Example (2)

Use sosopt to minimize α subject to $f_0 + \alpha f_1 \in \Sigma[x]$

```
% Problem set-up with polynomial toolbox and sosopt
```

```
>> pvar x1 x2 alpha;
```

```
>> f0 = -x1^4 + 2*x1^3*x2 + 9*x1^2*x2^2 - 2*x2^4;
```

```
>> f1 = x1^4 + x2^4;
```

```
>> x = [x1;x2];
```

```
>> obj = alpha;
```

```
>> [info,dopt,ssol]=sosopt(f0+alpha*f1,x,obj);
```

```
% s is f0+alpha*f1 evaluated at the minimal alpha
```

```
>> s = ssol{1};
```

```
% z and Q are the Gram matrix decomposition of s
```

```
>> z=ssol{2}; Q=ssol{3};
```

SOS Synthesis Example (3)

```
% Feasibility of sosopt result
>> info.feas
ans =
    1

% Minimal value of alpha
>> dopt
dopt =
    'alpha'    [2.0000]

% Verify s is f0+alpha*f1 evaluated at alpha = 2.00
>> s-subs( f0+alpha*f1, dopt)
ans =
    0

% Verify z and Q are the Gram matrix decomposition of s
>> s-z'*Q*z
ans =
    -2.4095e-010*x1^4 + 4.3804e-011*x1^3*x2 - 2.1894e-011*x1^2*x2^2
    + 9.2187e-016*x1*x2^3 - 2.6285e-010*x2^4

% Verify Q is positive semi-definite
>> min(eig(Q))
ans =
    1.3718e-010
```

Global Stability Theorem

Theorem: Let $l_1, l_2 \in \mathbb{R}[x]$ satisfy $l_i(0) = 0$ and $l_i(x) > 0 \forall x$ for $i = 1, 2$. If there exists $V \in \mathbb{R}[x]$ such that:

- ▶ $V(0) = 0$
- ▶ $V - l_1 \in \Sigma[x]$
- ▶ $-\nabla V \cdot f - l_2 \in \Sigma[x]$

Then $\mathcal{R}_0 = \mathbb{R}^n$.

Proof:

- ▶ The conditions imply that V and $-\nabla V \cdot f$ are positive definite.
- ▶ V is a positive definite polynomial and hence it is both decrescent and radially unbounded.
- ▶ It follows from Theorem 56 in Vidyasagar that $x = 0$ is globally asymptotically stable (GAS) and $\mathcal{R}_0 = \mathbb{R}^n$.
- ▶ V is a Lyapunov function that proves $x = 0$ is GAS.

Global Stability via SOS Optimization

- ▶ We can search for a Lyapunov function V that proves $x = 0$ is GAS. This is an SOS feasibility problem.
- ▶ Implementation:
 - ▶ V is a polynomial decision variable in the optimization and the user must select the monomials to include.
 - ▶ V can not include constant or linear terms.
 - ▶ A good (generic) choice for V is to include all monomials from degree 2 up to d_{max} :

`V = polydecvar('c', monomials(x, 2:dmax), 'vec');`

- ▶ l_1 and l_2 can usually be chosen as $\epsilon \sum_{i=1}^n x_i^{d_{min}}$ where d_{min} is the lowest degree of terms in V , e.g. $l_i = \epsilon x^T x$ for $d_{min} = 2$.
- ▶ The theorem only provides sufficient conditions for GAS.
 - ▶ If feasible, then V proves $\mathcal{R}_0 = \mathbb{R}^n$.
 - ▶ If infeasible, then additional monomials can be included in V and the SOS feasibility problem can be re-solved.
 - ▶ If $x = 0$ is not GAS then the conditions will always be infeasible. A local stability analysis is needed to estimate \mathcal{R}_0 .

Global Stability Example with sosopt

```
% Code from Parrilo1_GlobalStabilityWithVec.m
```

```
% Create vector field for dynamics
```

```
pvar x1 x2;  
x = [x1;x2];  
x1dot = -x1 - 2*x2^2;  
x2dot = -x2 - x1*x2 - 2*x2^3;  
xdot = [x1dot; x2dot];
```

```
% Use sosopt to find a Lyapunov function  
% that proves x = 0 is GAS
```

```
% Define decision variable for quadratic  
% Lyapunov function  
zV = monomials(x,2);  
V = polydecvar('c',zV,'vec');
```

```
% Constraint 1 : V(x) - L1 \in SOS  
L1 = 1e-6 * ( x1^2 + x2^2 );  
sosconstr{1} = V - L1;
```

```
% Constraint 2: -Vdot - L2 \in SOS  
L2 = 1e-6 * ( x1^2 + x2^2 );  
Vdot = jacobian(V,x)*xdot;  
sosconstr{2} = -Vdot - L2;
```

```
% Solve with feasibility problem
```

```
[info,dopt,sossol] = sosopt(sosconstr,x);  
Vsol = subs(V,dopt)
```

```
Vsol =  
0.30089*x1^2 + 1.8228e-017*x1*x2 + 0.6018*x2^2
```

