# Quantitative Local Analysis of Nonlinear Systems

Ufuk Topcu
Control and Dynamical Systems
California Institute of Technology

September 17, 2009

# Advertisement

Another workshop on V&V

- September 23&24, 2009
- At Caltech

Speakers:

Karl-Erik Arzen, Gerard Holzmann, Brian Williams, Mani Chandy, Nancy Leveson, Rupak Majumdar, Paulo Tabuada, Sayan Mitra, Stavros Tripakis, Edmund Clarke, Eric Feron, Rajeev Alur, Allessandro Pinto, Andrew Packard, Ashish Tiwari, Domitilla Del Vecchio, Calin Belta, Koushik Sen, Andre Platzer.

Participants include researchers from the academia, NSF, NASA, Boeing, UTRC, Honeywell, Toyota, SRI.

http://www.cds.caltech.edu/~utopcu/VVworkshop.html

To attend, contact Ufuk Topcu at utopcu@cds.caltech.edu.

# Acknowledgements

- The presentation is based on the slides from a pre-conference workshop (ACC 2009) by Andrew Packard, Gary Balas, Peter Seiler, and Ufuk Topcu.

- All material, required code, and other examples are available through

  http://www.cds.caltech.edu/~utopcu/NLShortCourse.html

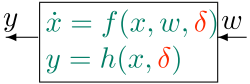
- Ryan Feeley, Evan Haas, George Hines, Zachary Jarvis-Wloszek, Erin Summers, Kunpeng Sun, Weehong Tan, Timothy Wheeler, Abhijit Chakraborty.

- AFOSR and NASA NRA.

# Tools for quantitative nonlinear robustness analysis

**Quantify** with **certificate** (and through an **automated** procedure)

$$\boxed{\dot{x} = f(x, \delta)}$$



$$\dot{x} = f(x, w, \delta)$$
$$y = h(x, \delta)$$

**Region-of-attraction (ROA)**

**Reachable set**
**Local input-output gain**

Repeat the above in the presence of parametric uncertainties $\delta \in \Delta$ (+ unmodeled dynamics)

**f and h are vectors of polynomials in** $x$ **and** $w$.

- If not polynomial, much harder.
  - Approximate and account for (extra) uncertainty

# General procedure to construct “certificates”

- System properties → Algebraic conditions
  - Lyapunov, dissipation inequalities.
- Algebraic conditions → Numerical optimization problems
  - Restrict the attention to polynomial vector fields, polynomial certificates,...
  - S-procedure like conditions (for set containment constraints)
  - Sum-of-squares (SOS) relaxations for polynomial nonnegativity
  - Pass to semidefinite programming (SDP) that are equivalent of SOS conditions
- Solve the resulting (linear or “bilinear”) SDPs
- Construct polynomial certificates

# Preliminaries

# Linear and Bilinear Matrix Inequalities

- Given matrices $\{F_i\}_{i=0}^{N} \subset \mathcal{S}^{n\times n}$, Linear Matrix Inequality (LMI) is a constraint on $\lambda \in \mathbb{R}^N$ of the form:

$$F_0 + \sum_{k=1}^{N} \lambda_k F_k \succeq 0$$

- Given matrices $\{F_i\}_{i=0}^{N}$, $\{G_j\}_{j=1}^{M}$, and $\{H_{k,j}\}_{k=1}^{N}{}_{j=1}^{M}$ $\subset \mathcal{S}^{n\times n}$, a Bilinear Matrix Inequality (BMI) is a constraint on $\lambda \in \mathbb{R}^N$ and $\gamma \in \mathbb{R}^M$ of the form:

$$F_0 + \sum_{k=1}^{N} \lambda_k F_k + \sum_{j=1}^{M} \gamma_k G_j + \sum_{k=1}^{N}\sum_{j=1}^{M} \lambda_k \gamma_j H_{k,j} \succeq 0$$

- Semidefinite program (?)

# Properties of SDPs

**SDPs with LMI constraints**

- "Easy" to solve.
- Public domain, efficient solvers: SeDuMi, SDPT3,...
- Link to SeDuMi through

  `http://www.cds.caltech.edu/~utopcu/NLShortCourse.html`

**SDPs with BMI constraints**

- Non-convex in general (our problems are specifically non-convex by counterexample).
- No general purpose solvers
- Global optimization methods, e.g. branch-and-bound.
- Local solvers, e.g. PENBMI.

# Optimizations with BMIs

$$\min_{\lambda \in \mathbb{R}^N, \gamma \in \mathbb{R}^M} c^T \lambda + d^T \gamma$$

subject to:

$$F_0 + \sum_{k=1}^{N} \lambda_k F_k + \sum_{j=1}^{M} \gamma_k G_j + \sum_{k=1}^{N} \sum_{j=1}^{M} \lambda_k \gamma_j H_{k,j} \succeq 0$$

- One useful property is that the constraint is an LMI if either $\lambda$ or $\gamma$ is held fixed.
- Coordinate-wise Iterations:
  1. Initialize a value of $\lambda$.
  2. Hold $\lambda$ fixed and solve for optimal $\gamma$. This is an SDP.
  3. Hold $\gamma$ fixed and solve for optimal $\lambda$. This is an SDP.
  4. Go back to step 2 and repeat until values converge.
- This is local search scheme. Not even guaranteed to converge to local optimal points but works well for our problems.

# Multipoly Toolbox (available through workshop web page)

- 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
q = x1^2
p*q =
    2*x1^6 + 2*x1^5*x2 - x1^4*x2^2 + 5*x1^2*x2^4
jacobian(p, [x1;x2]) =
    [ 8*x1^3 + 6*x1^2*x2 - 2*x1*x2^2 ,
        2*x1^3 - 2*x1^2*x2 + 20*x2^3 ]
```

- Algebraic manipulation, visualization (sublevel sets, etc.),...

# 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 $\{x_1, \ldots, x_n\}$ will be denoted $\mathcal{P}[x_1, \ldots, x_n]$ 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.

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 $\{x_1, \ldots, x_n\}$ will be denoted $\Sigma\left[x_1, \ldots, x_n\right]$ or $\Sigma\left[x\right]$.
- 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 $\mathcal{P}\left[x\right] = \Sigma\left[x\right]$ only for a) $n = 1$, b) $d = 2$, and c) $d = 4$, $n = 2$.
- $p$ is a SOS iff there exists $Q \succeq 0$ such that $p = z^T Q z$.

Reference: Choi, M., Lam, T., and Reznick, B., Sums of Squares of Real Polynomials, *Proceedings of Symposia in Pure Mathematics*, Vol. 58, No. 2, 1995, pp. 103 – 126.

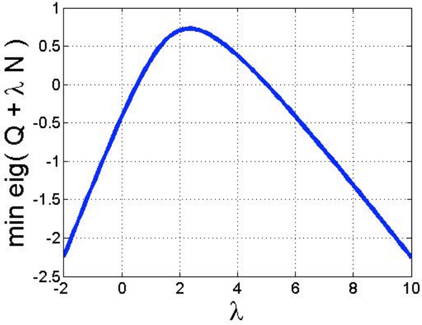
# SOS Example

All possible Gram matrix representations of

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

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

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



p is SOS iff

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

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

# SOS Test with issos

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

```
[feas,z,Q,f] = 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 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:

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

This SOS programming problem is an SDP.

- The cost is a linear function of $\alpha$.
- 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 $\{f_{j,k}\}_{j=1}^{N_s}{}_{k=0}^{m}$, 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 Synthesis Example (1)

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

$$f_0(x) := -x_1^4 + 2x_1^3 x_2 + 9x_1^2 x_2^2 - 2x_2^4$$
$$f_1(x) := x_1^4 + x_2^4$$

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

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

where

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

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 $\alpha$ 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,sossol]=sosopt(f0+alpha*f1,x,obj);

% s is f0+alpha*f1 evaluated at the minimal alpha
>> s = sossol{1};

% z and Q are the Gram matrix decomposition of s
>> z=sossol{2}; Q=sossol{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
```

# Set Containment Conditions

- Many nonlinear analysis problems can be formulated with set containment constraints.
- Need conditions for proving set containments:

  Given polynomials $g_1$ and $g_2$, define sets $S_1$ and $S_2$:

  $$\begin{aligned} S_1 &:= \{x \in \mathbb{R}^n \ : \ g_1(x) \leq 0\} \\ S_2 &:= \{x \in \mathbb{R}^n \ : \ g_2(x) \leq 0\} \end{aligned}$$

  Is $S_2 \subseteq S_1$?

- In control theory, the S-procedure is a common condition used to prove set containments involving quadratic functions. This can be generalize to higher degree polynomials.

# Polynomial S-Procedure

- Theorem: Let $g_1$ and $g_2$ be given polynomials. If there exists a positive semidefinite polynomial $\lambda \in \mathcal{P}[x]$ such that $-g_1(x) + \lambda(x) g_2(x) \in \mathcal{P}[x]$ is positive semidefinite, then $S_2 \subseteq S_1$.

- The PSD constraints are numerically difficult to handle. The theorem still holds if relaxed to SOS constraints:
  - If there exists a polynomial $\lambda \in \Sigma[x]$ such that $-g_1(x) + \lambda(x) g_2(x) \in \Sigma[x]$ then $S_2 \subseteq S_1$.

# Set Containment Maximization

- Given polynomials $g_1$ and $g_2$, the set containment maximization problem is:

$$\gamma^* = \max_{\gamma \in \mathbb{R}} \gamma$$
$$\text{s.t.: } \{x \in \mathbb{R}^n \ : \ g_2(x) \leq \gamma\} \subseteq \{\, x \in \mathbb{R}^n \ : \ g_1(x) \leq 0\}$$

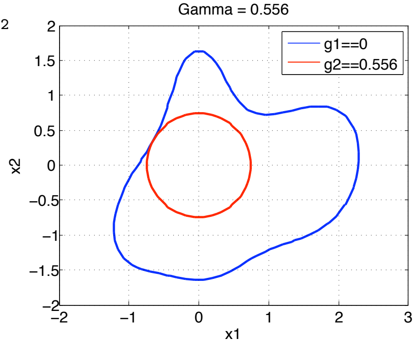
- The polynomial S-procedure can be used to relax the set containment constraint:

$$\gamma_{lb} = \max_{\gamma \in \mathbb{R}, s \in \Sigma[x]} \gamma$$
$$\text{s.t.: } -g_1 + (g_2 - \gamma)s \in \Sigma\,[x]$$

- The solution of this optimization satisfies $\gamma_{lb} \leq \gamma^*$.

# pcontain Example

```
% Maximize size of a disk inside
% the contour of a 6th degree poly
pvar x1 x2;
x = [x1;x2];

% S1 := { x : g1(x)<= 0}
g1 =  0.3*x1^6 + 0.05*x2^6 - 0.5*x1^5 - 1.4*x1^3*x2
    + 2.3*x1^2*x2^2 - 0.9*x1^3 + 2.6*x1^2*x2 - 1;

% S2 := { x : g2(x)<= gamma}
g2 = x'*x;

% Define monomials for s
z = monomials(x,0:2);

% Use pcontain to maximize gamma s.t. S2 \in S1
% gbnds gives lower/upper bounds on optimal gamma
% sopt is the optimal multiplier
[gbnds,sopt]=pcontain(g1,g2,z)
gamma = gbnds(1);
gbnds =
    0.5560    0.5569

sopt =
   1.4483*x1^4 + 0.055137*x1^3*x2 + 0.44703*x1^2*x2^2 - 0.043336*x1*x2^3
   + 1.2961*x2^4 - 0.21988*x1^3 - 0.26998*x1^2*x2 - 0.050453*x1*x2^2
   + 0.13586*x2^3 + 1.6744*x1^2 - 0.41955*x1*x2 + 1.4875*x2^2
   - 0.49756*x1 + 0.50148*x2 + 1.2679

% Plot contours of unit disk and maximal ellipse
plotdomain = [-2 3 -2 2];
pcontour(g1,0,plotdomain,'b') hold on;
pcontour(g2,gamma,plotdomain,'r')
axis equal; axis(plotdomain)
```



# ROA analysis using SOS optimization and solution strategies

# Region of Attraction

Consider the autonomous nonlinear dynamical system

$$\dot{x}(t) = f(x(t))$$

where $x \in \mathbb{R}^n$ is the state vector and $f : \mathbb{R}^n \to \mathbb{R}^n$.
Assume:

- $f \in \mathbb{R}[x]$
- $f(0) = 0$, i.e. $x = 0$ is an equilibrium point.
- $x = 0$ is asymptotically stable.

Define the region of attraction (ROA) as:

$$\mathcal{R}_0 := \{\xi \in \mathbb{R}^n \ : \ \lim_{t \to \infty} \phi(\xi, t) = 0\}$$

where $\phi(\xi, t)$ denotes the solution at time $t$ starting from the initial condition $\phi(\xi, 0) = \xi$.

Objective: Compute or estimate the ROA.

# Global Stability Theorem

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

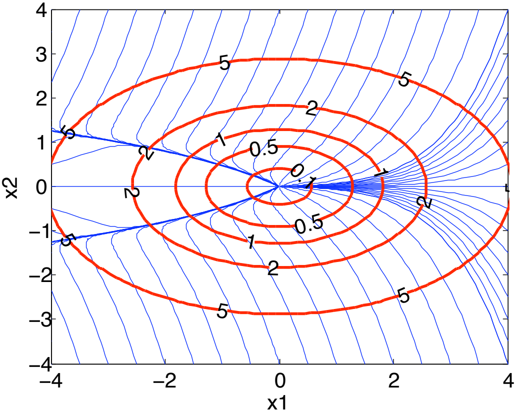
- $V(0) = 0$
- $V - l_1 \in \Sigma[x]$
- $-\nabla V \cdot f - l_2 \in \Sigma[x]$

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

Reference: Vidyasagar, M., *Nonlinear Systems Analysis*, SIAM, 2002.
(Refer to Section 5.3 for theorems on Lyapunov's direct method.)

# 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
```



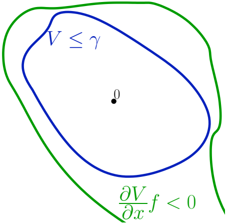
# Local Stability Theorem

Theorem: Let $l_1 \in \mathbb{R}[x]$ satisfy $l_1(0) = 0$ and $l_1(x) > 0 \; \forall x$.
If there exists $V \in \mathbb{R}[x]$ such that:

- $V(0) = 0$
- $V - l_1 \in \Sigma[x]$
- $\Omega_{V,\gamma} := \{x \in \mathbb{R}^n : V(x) \leq \gamma\} \subseteq \{\, x \in \mathbb{R}^n : \nabla V \cdot f < 0\} \cup \{\, 0\}$

Then $\Omega_{V,\gamma} \subseteq \mathcal{R}_0$.

Proof: The conditions imply that $\Omega_{V,\gamma}$ is bounded and hence the result follows from Lemma 40 in Vidyasagar.



# Local Stability via SOS Optimization

Idea: Let $\dot{x} = Ax$ be the linearization of $\dot{x} = f(x)$. If $A$ is Hurwitz then a quadratic Lyapunov function shows that $x = 0$ is locally asymptotically stable. Use the polynomial S-procedure to verify a quantitative estimate.

1. Select $Q \in \mathcal{S}^{n \times n}$, $Q > 0$ and compute $P > 0$ that satisfies the Lyapunov Equation: $A^T P + PA = -Q$
   - $V_{lin}(x) = x^T P x$ is a quadratic Lyapunov function proving $x = 0$ is locally asymptotically stable.
   - This step can be done with: `[Vlin,A,P]=linstab(f,x)`

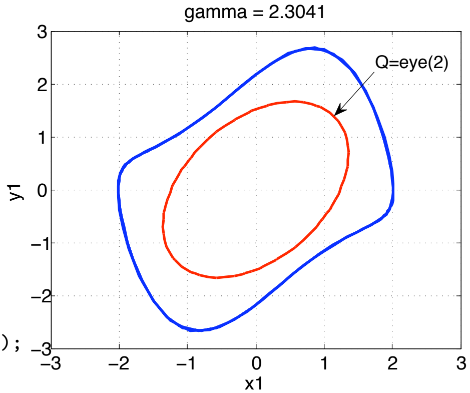
2. Define $l_2 \in \mathbb{R}[x]$ such that $l_2(0) = 0$ and $l_2(x) > 0 \; \forall x$. Solve the set containment maximization problem using `pcontain`:

$$\max_{\gamma \in \mathbb{R}} \gamma \text{ subject to } \Omega_{V,\gamma} \subset \{x \in \mathbb{R}^n : \nabla V_{lin} \cdot f - l_2 \leq 0\}$$

# Example: ROA Estimate for the Van der Pol Oscillator (1)

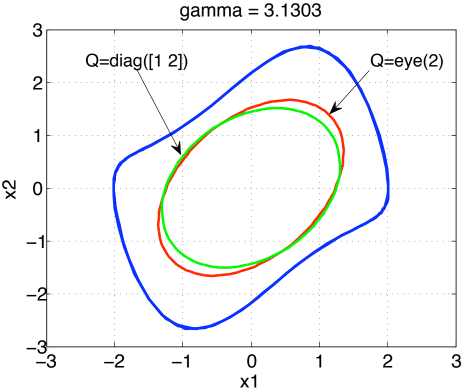
```
% Code from VDP_LinearizedLyap.m

% Vector field for VDP Oscillator
pvar x1 x2;
x = [x1;x2];
x1dot = -x2;
x2dot = x1+(x1^2-1)*x2;
f = [x1dot; x2dot];

% Lyap fnc from linearization
Q = eye(2);
Vlin = linstab(f,x,Q);

% maximize gamma
% subject to:
% {Vlin<=gamma} in {Vdot<0} U {x=0}
z = monomials(x, 1:2 );
L2 = 1e-6*(x'*x);
Vdot = jacobian(Vlin,x)*f;
[gbnds,s] = pcontain(Vdot+L2,Vlin,z);
Gamma = gbnds(1)
```

$$\dot{x}_1 = -x_2$$
$$\dot{x}_2 = x_1 + (x_1^2 - 1)x_2$$



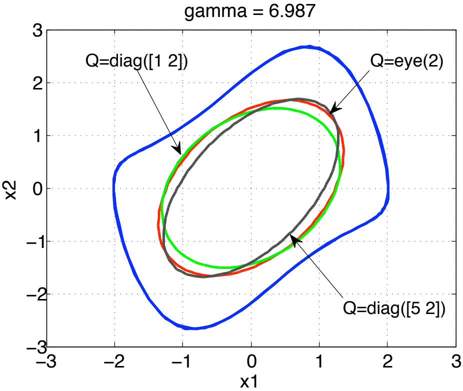
# Example: ROA Estimate for the Van der Pol Oscillator (2)

Choosing $Q = \left[\begin{smallmatrix} 1 & 0 \\ 0 & 2 \end{smallmatrix}\right]$ slightly increases $\Omega_{V,\gamma}$ along one direction but decreases it along another.



# Example: ROA Estimate for the Van der Pol Oscillator (3)

Choosing $Q = \left[\begin{smallmatrix} 5 & 0 \\ 0 & 2 \end{smallmatrix}\right]$ has the opposite effect on $\Omega_{V,\gamma}$.



# Increasing the ROA Estimate

For this problem, `pcontain` solves:

$$\max_{\gamma \in \mathbb{R}, s \in \Sigma[x]} \gamma$$

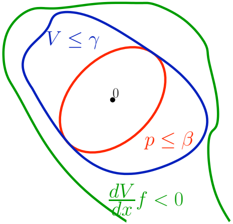
$$\text{s.t.:} \quad -\left(\nabla V_{lin} \cdot f + l_2 + s(\gamma - V_{lin})\right) \in \Sigma\left[x\right]$$

Objective: Increase the "size" of $\Omega_{V,\gamma}$ subject to the same constraints by searching over quadratic or higher degree Lyapunov functions.

Question: How should we measure the "size" of the ROA estimate?

Approach:
Introduce a shape factor $p$ which:

- is a positive definite polynomial
- captures the intent of the analyst
- (preferably) has simple sublevel sets



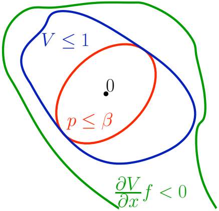
# Increasing the ROA Estimate

We increase the ROA estimate by increasing the shape function contained with a Lyapunov level set.

$$\beta^* = \max_{V \in \mathbb{R}[x],\ \beta \in \mathbb{R}} \beta$$

subject to:

$$\Omega_{p,\beta} \subseteq \Omega_{V,1}$$
$$\Omega_{V,1} \subseteq \{\nabla V \cdot f(x) < 0\} \cup \{0\}$$
$$V - l_1 \in \Sigma[x], V(0) = 0$$



How are the set containments verified?

# Increasing the ROA Estimate

Applying the polynomial S-procedure to both set containment conditions gives:

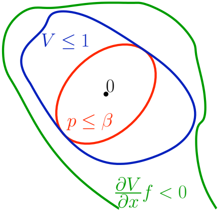
$$\max_{s_1, s_2 \in \Sigma[x],\ V \in \mathbb{R}[x],\ \beta \in \mathbb{R}} \beta$$

subject to:

$$-\left((V-1) + s_1(\beta - p)\right) \in \Sigma\left[x\right]$$

$$-\left((\nabla V \cdot f + l_2) + s_2(1 - V)\right) \in \Sigma\left[x\right]$$

$$V - l_1 \in \Sigma\left[x\right], V(0) = 0$$



This is not an SOS programming problem since the first constraint is bilinear in variables $s_1$ and $\beta$ and the second constraint is bilinear in variables $s_2$ and $V$.

# Solving the Bilinear ROA Problem

A coordinate-wise $V$-$s$ iteration is a simple algorithm to find a sub-optimal solution to this optimization.

- For fixed $V$, the constraints decouple into two subproblems

$$\begin{aligned}\gamma^* &= \max_{\gamma\in\mathbb{R},s_2\in\Sigma[x]} \gamma \;\text{ s.t. }\; -\left((\nabla V\cdot f + l_2) + s_2(1-V)\right)\in\Sigma\left[x\right]\\ &\le \max_{\gamma\in\mathbb{R}} \gamma \;\text{ s.t. }\; \Omega_{V,\gamma}\subseteq\{\nabla V\cdot f(x)<0\}\cup\{0\}\\ \beta^* &= \max_{\beta\in\mathbb{R},s_1\in\Sigma[x]} \beta \;\text{ s.t. }\; -\left((V-\gamma^*) + s_1(\beta-p)\right)\in\Sigma\left[x\right]\\ &\le \max_{\beta\in\mathbb{R}} \beta \;\text{ s.t. }\; \Omega_{p,\beta}\subseteq\Omega_{V,\gamma^*}\end{aligned}$$

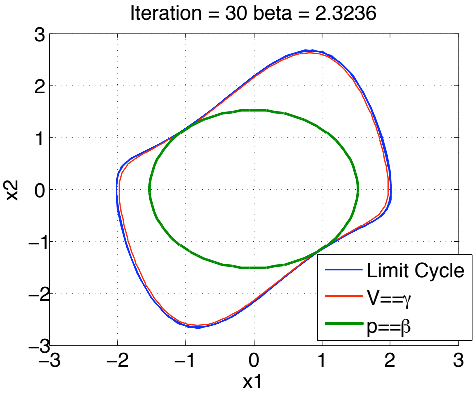
  `pcontain` can be used to compute $\gamma^*$ and $\beta^*$ as well as multipliers $s_1$ and $s_2$.

- For fixed $s_1$ and $s_2$, we could maximize $\beta$ with $V$ subject to the local ROA constraints. We obtain better results by re-centering $V$ to the analytic center of the LMI associated with:

$$\begin{aligned}&-\left((V-1)+s_1(\beta^*-p)\right)\in\Sigma\left[x\right]\\ &-\left((\nabla V\cdot f + l_2)+s_2(\gamma^*-V)\right)\in\Sigma\left[x\right]\\ &V-l_1\in\Sigma\left[x\right],\, V(0)=0\end{aligned}$$

# Example: V-s Iteration for the Van der Pol Oscillator

```
% Code from VDP_IterationWithVlin.m
pvar x1 x2;
x = [x1;x2];
x1dot = -x2;
x2dot = x1 + (x1^2-1)*x2;
f = [x1dot; x2dot];

% Create shape function and monomials vectors
p = x'*x;
zV = monomials( x, 2:6 );    % V has Deg = 6
z1 = monomials( x, 0:2 );
z2 = monomials( x, 1:2 );
L2 = 1e-6*(x'*x);

% Initialize Lyapunov Function
V = linstab(f,x);

% Run V-s iteration
opts.L2 = L2;
for i1=1:30;
    % gamma step
    Vdot = jacobian(V,x)*f;
    [gbnds,s2] = pcontain(Vdot+L2,V,z2,opts);
    gamma = gbnds(2);

    % beta step
    [bbnds,s1] = pcontain(V-gamma,p,z1,opts);
    beta = bbnds(1)

    % V step (then scale to roughly normalize)
    if i1~=30
        V = roavstep(f,p,x,zV,beta,gamma,s1,s2,opts);
        V = V/gamma;
    end
end
```

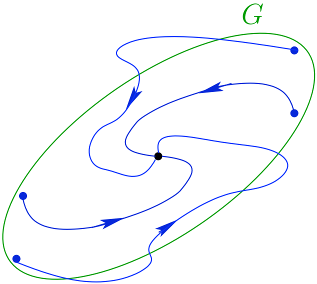


# Use of Simulation Data

- The performance of the $V$-$s$ iteration depends on the initial choice for $V$.
- Up to this point we have only started the iteration using the Lyapunov function obtained from linear analysis.
- It is also possible to used simulation data to construct initial Lyapunov function candidates for the iteration.
- The following slides explore this use of simulation data.

# Use of Simulation Data

- Given a set $G$, is $G \subset \text{ ROA }$?
- Run simulations starting in $G$.
- If any diverge, no.
- If all converge, "maybe yes."



**Fact**: A Lyapunov certificate would remove the "maybe".

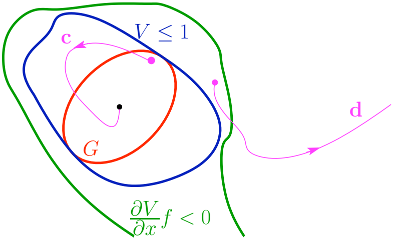
$$G \in \Omega_{V,\gamma=1} \subseteq \{x \in \mathbb{R}^n \ : \ \nabla V(x) \cdot f(x) < 0\}$$

**Question**: Can we use the simulation data to construct candidate Lyapunov functions for assessing the ROA?

# How can the simulation data be used?

If there exists $V$ to certify that $G$ is in the ROA through Lyapunov arguments, it is **necessary** that

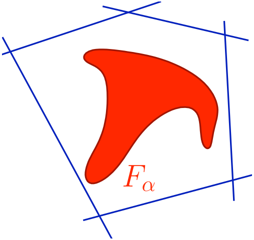
- $V > 0$
- $V \leq 1$ on converging trajectories starting in $G$
- $\dot{V} < 0$ on converging trajectories starting in $G$
- $V > 1$ on non-converging trajectories starting in the complement of $G$



The $V$ we are looking for (which may not even exist) **must** satisfy these constraints.

# Simulation-based constraints on $V$

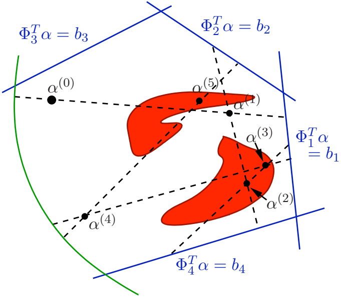
- Assume $V$ is linearly parameterized in some basis functions $V(x) = \alpha^T \phi(x)$, e.g. $\phi(x)$ can be a vector of monomials.
- Let $F_\alpha$ denote the set of coefficients $\alpha$ of Lyapunov functions which satisfy the constraints on some domain in the state space.
- Enforcing the constraints on the previous slide on the simulation trajectory points leads to LP constraints on $\alpha$.
- The collection of the LP constraints forms a polytope outer bound on the set $F_\alpha$ of coefficients.



# Set of Candidate $V$'s

- We can sample the polytope outer bound of $F_\alpha$ by solving an LP feasibility problem.
  - If the LP is infeasible then $F_\alpha$ is empty.
  - If the LP is feasible then we can test if $V = \alpha^T \phi$ is a Lyapunov function using SOS optimization methods.
- We can incorporate additional convex constraints on $\alpha$
  - $V - l_1 \in \Sigma\left[x\right] \Rightarrow$ LMI constraints on $\alpha$
  - The linear part of $f$ and quadratic part of $V$ must satisfy the Lyapunov inequality $\Rightarrow$ LMI constraints on $\alpha$.
- Let $\mathcal{Y}$ denote the set of $\alpha$ which satisfy the LP constraints from simulation data and the LMI constraints described above.

# Hit-and-run (H&R) algorithm



- As the number of constraints increases, the outer convex set $\mathcal{Y}$ becomes a tighter relaxation.
  $\Rightarrow$ Samples from $\mathcal{Y}$ become more likely to be in $F_\alpha$.

- Strategy: generate points in $\mathcal{Y}$, i.e., Lyapunov function candidates, and evaluate $\beta$ they certify.

- Generation of each point $\mathcal{Y}$ (after the initial feasible point) involves solving $4$ small LMIs and trivial manipulations.

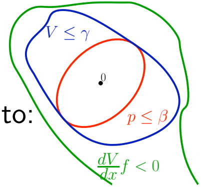
$$\bar{t}^{(k)} := \min \quad \left\{ \max_j \left\{ 0, \frac{b_j - \Phi_j^T\alpha^{(k)}}{\Phi_j^T\zeta^{(k)}} \right\}, \bar{t}_{SOS}^{(k)}, \bar{t}_{lin}^{(k)} \right\},$$

$$\underline{t}^{(k)} := \max \quad \left\{ \min_j \left\{ 0, \frac{b_j - \Phi_j^T\alpha^{(k)}}{\Phi_j^T\zeta^{(k)}} \right\}, \underline{t}_{SOS}^{(k)}, \underline{t}_{lin}^{(k)} \right\},$$

# Assessing the candidate: checking containments

**For a given** $V$,

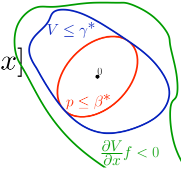
$\beta_V := \max\limits_{\beta,\gamma} \quad \beta$ subject to:



This can be solved in two steps solving **smaller "affine"** SDPs sequentially:

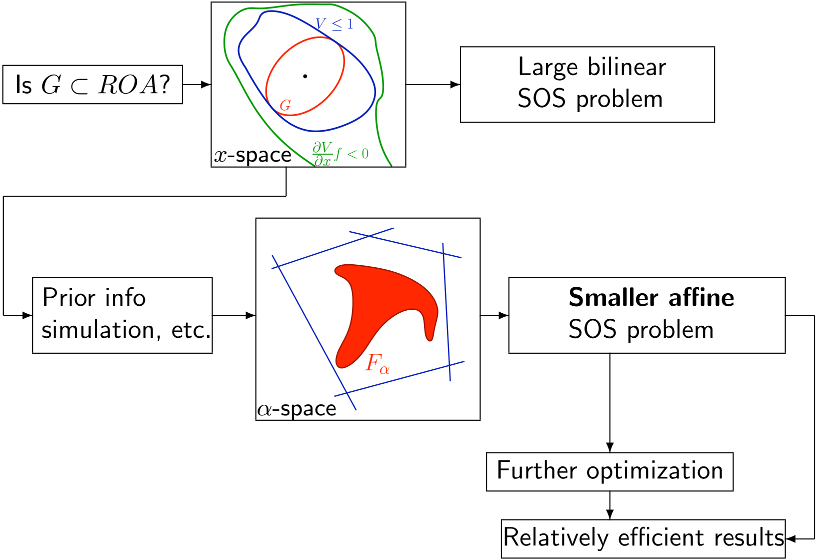
$$\begin{aligned} \gamma^* := \quad & \max \ \gamma \\ & -\left[(\gamma - V)s_2 + s_3 \tfrac{dV}{dx} f + l_2\right] \in \Sigma[x] \end{aligned}$$

$$\begin{aligned} \beta_V := \quad & \max \ \beta \\ & -\left[(\beta - p)s_1 + (V - \gamma^*)\right] \in \Sigma[x] \end{aligned}$$



These are the same $\gamma$ and $\beta$ steps from the $V$-$s$ iteration.

# Overview of the method



Is $G \subset ROA$?

Large bilinear SOS problem

Prior info simulation, etc.

**Smaller affine** SOS problem

Further optimization

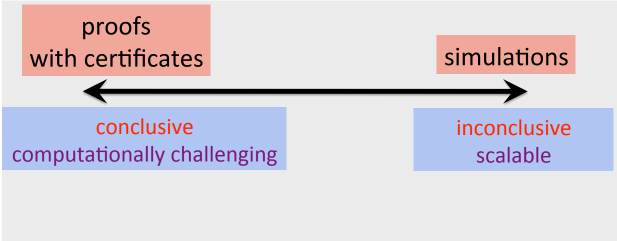
Relatively efficient results

# Properties of simulation-aided analysis

- Integration of simulation data yields higher reliability and better scalability
  - Balance between expressive power, computational complexity, and conservatism.
  - Not blind search, not hit-or-miss – Start collecting proofs from initial steps on and then refine.

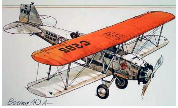
Most of the computation is trivially parallelizable.

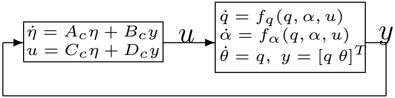
- We have automated this procedure (and more add-ons).



# Example: controlled aircraft [Short period pitch axis model]

- States: pitch rate ($q$), AoA ($\alpha$), and pitch angle ($\theta$).
- Control: elevator deflection (u)(2-state LTI).
- Cubic polynomial approximation (from Honeywell).
- $p(x) = x^T x$, [$x$: plant and controller states].




$\dot{\eta} = A_c\eta + B_c y$
$u = C_c\eta + D_c y$
$u$
$\dot{q} = f_q(q, \alpha, u)$
$\dot{\alpha} = f_\alpha(q, \alpha, u)$
$\dot{\theta} = q, \ y = [q \ \theta]^T$
$y$

Just sample-and-assess:

$\beta = 8.9$ (quartic)

$\beta = 6.2$ (quadratic)

Simulation-aided analysis + coordinate-wise affine iterations:

- deg($V$)$= 2 \Rightarrow \beta = 8.6$ in 2 minutes
- deg($V$)$= 4$:

$$\boxed{\begin{array}{ccccccc} 14.6 & < & \mathbf{15.3} & \le & \beta^{opt} & \le & \mathbf{16.1} \\ \uparrow & & \uparrow & & & & \uparrow \\ \text{30 min} & & \text{45 min} & & & & \text{div. traj.} \end{array}}$$

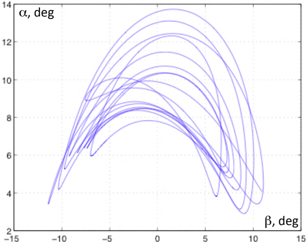
| | sim-aided | off-the-shelf solver |
|---|---|---|
| 5 states | 30 minutes | 38 hours |
| 5 states + 1st order Pade (in $u$) | 50 minutes | out-of-memory |

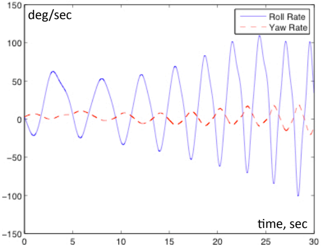
# Falling leaf mode in F/A-18 Hornet

Falling leaf motion: out-of-control

- oscillations in roll and yaw
- fluctuations in AoA and sideslip
  → loss of lift

Picture from http://www.classic-machines.com





Revised flight control law:

Extensive flight tests → suppression of the falling leaf mode.

Linear analysis has not detected any performance issues for the baseline controller. What does nonlinear analysis say?

# Modeling Summary

- The reduced order, nonlinear $3$rd polynomial model captures the characteristics of the falling leaf motion.
- For analysis purpose, roll-coupled maneuvers that drive the aircraft to the falling leaf motion are considered.
- The velocity is assumed to be fixed at 250 ft/s.

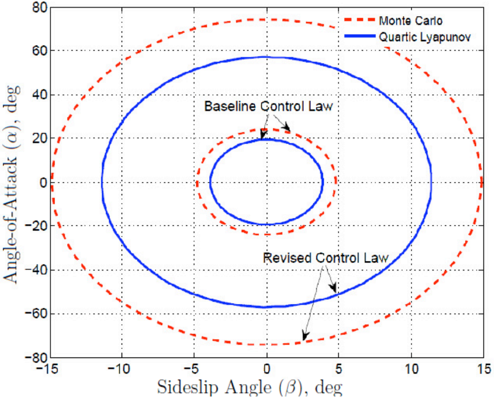
$$\dot{x} = f(x, u)\ ,\ \ y = h(x)$$

$$x = \begin{bmatrix} \text{angle-of-attack}(\alpha) \\ \text{sideslip angle}(\beta) \\ \text{roll rate}(p) \\ \text{yaw rate}(r) \\ \text{pitch rate}(q) \\ \text{bank angle}(\phi) \end{bmatrix},\ \ y = \begin{bmatrix} \text{angle-of-attack}(\alpha) \\ \text{roll rate}(p) \\ \text{yaw rate}(r) \\ \text{pitch rate}(q) \\ \text{lateral acceleration}(a_y) \\ \text{sideslip rate}(\dot{\beta}) \\ \text{sideslip angle}(\beta) \end{bmatrix}$$

$$u = \begin{bmatrix} \text{aileron deflection}(\delta_{ail}) \\ \text{rudder deflection}(\delta_{rud}) \\ \text{stabilator deflection}(\delta_{stab}) \end{bmatrix}$$

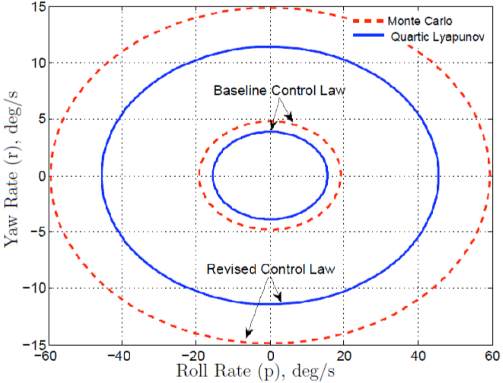
# Nonlinear Region-of-Attraction Analysis

## Results on Estimating ROA: $\alpha$ vs. $\beta$



# Nonlinear Region-of-Attraction Analysis

## Results on Estimating ROA: $p$ vs. $r$



# Nonlinear Region-of-Attraction Analysis (cont'd)

## Computational Aspects

- Computational time for estimating both lower and upper bound are as follows:

| Analysis | Iteration Steps | **Baseline** | **Revised** |
|---|---|---|---|
| **V-s Iteration**$^{(1)}$ | 40 | 6.8 Hrs | 4.7 Hrs |
| **Monte Carlo Upper Bound**$^{(2)}$ | 5 million | 96 Hrs | 96 Hrs |

(1) V-s iteration analysis performed on *Intel(R) Core(TM) i7 CPU 2.67GHz 8.00GB RAM*

(2) Monte Carlo analysis performed on *Intel(R) Core(TM)2 Duo CPU E65550 2.33GHz 3.00GB RAM*

# Robust ROA analysis with parametric uncertainty

# Systems with parametric uncertainty

System with parametric uncertainty governed by

$$\dot{x}(t) = f(x(t), \delta)$$

The parameter $\delta$ is

- constant
- unknown
- known to take values on the bounded set $\Delta$

Assumption:

- For each $\delta \in \Delta$, the origin is an equilibrium point, i.e.,

$$f(0, \delta) = 0 \qquad \text{for all } \delta \in \Delta.$$

# ROA analysis for systems with parametric uncertainty

System with **constant parametric** uncertainty governed by

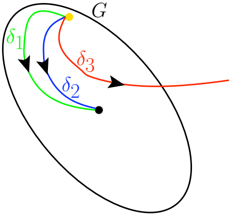
$$\dot{x}(t) = f(x(t), \delta)$$

**Question:** Given a set $G$,

- is $G$ in the ROA for each $\delta \in \Delta$?
- is $G$ a subset of the robust ROA, defines as

$$\bigcap_{\delta \in \Delta} \{\zeta \in \mathbb{R}^n \ : \ \lim_{t \to \infty} \varphi(\zeta, t; \delta) = 0\}?$$

[ $\varphi(\zeta, t; \delta)$ is the solution at time $t$ with initial condition $\zeta$ for $\delta$.]



# ROA analysis for $\dot{x} = f(x, \delta)$

**Theorem:** If there exists a continuously differentiable function $V$ such that

- $V(0) = 0$, and $V(x) > 0$ for all $x \neq 0$
- $\Omega_{V,1} = \{x \ : \ V(x) \leq 1\}$ is bounded
- For each $\delta \in \Delta$, the set containment

$$\{x \ : \ V(x) \leq 1\} \backslash \{0\} \subset \{x \ : \ \nabla V(x) f(x, \delta) < 0\}$$

holds, then $\{x \in \mathbb{R}^n \ : \ V(x) \leq 1\}$ is an invariant subset of the robust ROA.

**Proof:** Apply Lyapunov theory to each system ...

**A few issues:**

- "For each $\delta \in \Delta$..." there are infinite number of set containment conditions.
- $V$ does not depend on $\delta$, though $f$ does, will this be restrictive?

# ROA analysis: $f(x,\delta)$ affine in $\delta$

Affine uncertainty dependence & bounded, polytopic $\Delta$ (with vertices $\mathcal{E}$)

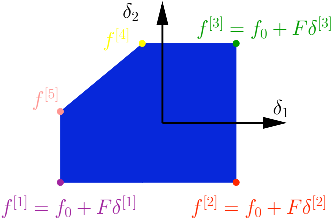
$$\dot{x}(t) = f_0(x(t)) + \sum_{i=1}^{m} f_i(x(t))\delta_i \quad = \quad f_0(x(t)) + F(x(t))\delta$$

**Theorem:** If $\Delta$ is a polytope, and for all $\delta \in \mathcal{E}$

$$\Omega_V \setminus \{0\} \subseteq \{\, x \in \mathbb{R}^n : \nabla V(x)(f_0(x) + F(x)\delta) < 0\} \,,$$

then the set containment holds for all $\delta \in \Delta$.

**Proof:**
For each $\tilde{\delta} \in \Delta$, $\nabla V(x) F(x) \tilde{\delta}$ is a convex combination of $\{\nabla V(x) F(x) \delta \ : \ \delta \in \Delta\}$.

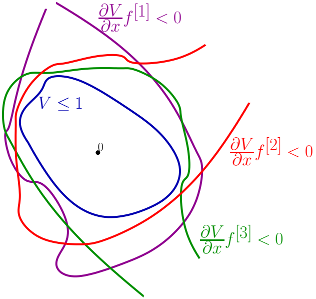

$\delta_2$
$f^{[4]}$
$f^{[3]} = f_0 + F\delta^{[3]}$
$f^{[5]}$
$\delta_1$
$f^{[1]} = f_0 + F\delta^{[1]}$
$f^{[2]} = f_0 + F\delta^{[2]}$

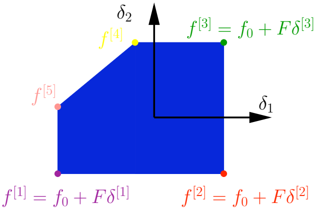
# ROA analysis with parameter-independent $V$ (2)

$$\dot{x}(t) = f_0(x(t)) + F(x(t))\delta$$

Impose at the vertices of $\Delta$, then they hold everywhere on $\Delta$.

$$\Omega_V \setminus \{0\} \subseteq \{\, x \in \mathbb{R}^n : \nabla V(x)(f_0(x) + F(x)\delta) < 0\}$$


$\frac{\partial V}{\partial x} f^{[1]} < 0$
$V \leq 1$
$0$
$\frac{\partial V}{\partial x} f^{[2]} < 0$
$\frac{\partial V}{\partial x} f^{[3]} < 0$


$\delta_2$
$f^{[4]}$
$f^{[3]} = f_0 + F\delta^{[3]}$
$f^{[5]}$
$\delta_1$
$f^{[1]} = f_0 + F\delta^{[1]}$
$f^{[2]} = f_0 + F\delta^{[2]}$

For every $i = 1, \ldots, N_{vertex}$ (index to elements of $\mathcal{E}$),

$$-\left[(1 - V)s_2 + s_3 \nabla V \cdot (f_0 + F\delta^{[i]}) + l_2\right] \text{ is SOS in } x \text{ (only)}$$

# SOS problem for robust ROA computation

$$\max_{0<\gamma,0<\beta,V\in\mathcal{V},s_1\in\mathcal{S}_1,s_{2\delta}\in\mathcal{S}_2,s_{3\delta}\in\mathcal{S}_3} \quad \beta \quad \text{subject to}$$

$$s_{2\delta} \in \Sigma[x], \text{ and } s_{3\delta} \in \Sigma[x]$$

$$-[(\gamma - V)s_{2\delta} + \nabla V(f_0 + F(x)\delta)s_{3\delta} + l_2] \in \Sigma[x] \quad \forall \delta \in \mathcal{E},$$

$$-[(\beta - p)s_1 + V - 1] \in \Sigma[x]$$

- Bilinear optimization problem
- SOS conditions:
  - only in $x$
  - $\delta$ does not appear, but...
  - there are a lot of SOS constraints ($\delta \in \mathcal{E}$)

# Example

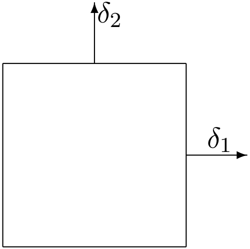
Consider the system with a single uncertain parameter $\delta$

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -x_2 - (\delta + 2)(x_1 - x_1^3) \end{aligned}$$

with $\delta \in [-1, 1]$.

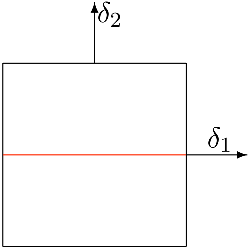
Codepad Demo: attached to the end of the slides.

# Dealing with conservatism: partition $\Delta$



For all $\delta \in \Delta$:

$$\{x : V_0(x) \leq 1\} \backslash \{0\} \subset \left\{x : \frac{\partial V_0}{\partial x} f(x, \delta) < 0\right\}$$



For all $\delta \in$ upper half of $\Delta$:

$$\{x : V_1(x) \leq 1\} \backslash \{0\} \subset \left\{x : \frac{\partial V_1}{\partial x} f(x, \delta) < 0\right\}$$

For all $\delta \in$ lower half of $\Delta$:

$$\{x : V_2(x) \leq 1\} \backslash \{0\} \subset \left\{x : \frac{\partial V_2}{\partial x} f(x, \delta) < 0\right\}$$

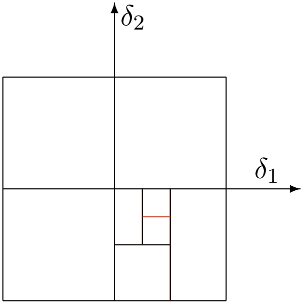
$V_1 := V_0$ and $V_2 := V_0$ are feasible for the right-hand side.
Improve the results by searching for different $V_1$ and $V_2$.

# Dealing with conservatism: branch-and-bound in $\Delta$

Systematically refine the partition of $\Delta$:

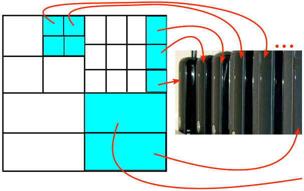
- Run an informal branch-and-bound (B&B) refinement procedure

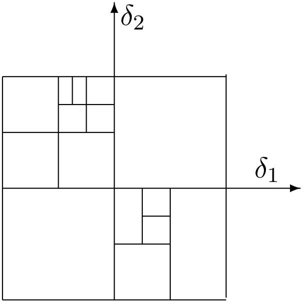
Sub-division strategy: Divide the worst cell into $2$ subcells.



# Properties of the branch-and-bound refinement

- Yields piecewise-polynomial, $\delta$-dependent $V$.
- Local problems are decoupled
  $\rightarrow$ parallel computing





- Organizes extra info regarding system behavior: returns a data structure with useful info about the system
  - Lyapunov functions, SOS certificates,
  - certified $\beta$,
  - worst case parameters,
  - initial conditions for divergent trajectories,
  - values of $\beta$ not achievable, etc.

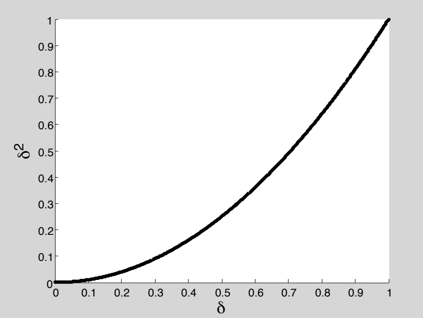
# Non-affine dependence on $\delta$

Let $g : \mathbb{R} \rightarrow \mathbb{R}$.

$$\begin{aligned} \dot{x}(t) &= f_0(x(t)) + \delta f_1(x(t)) + g(\delta) f_2(x(t)) \\ &= f_0(x(t)) + \delta f_1(x(t)) + \zeta f_2(x(t)) \end{aligned}$$

Treat $(\delta, g(\delta))$ as 2 parameters, whose values lie on a 1-dimensional curve. Then

$*$ Cover 1-d curve with 2-polytope
$*$ Compute ROA
$*$ Refine polytope into a union of smaller polytopes
$*$ Solve robust ROA on each polytope
$*$ Intersect ROAs → robust ROA



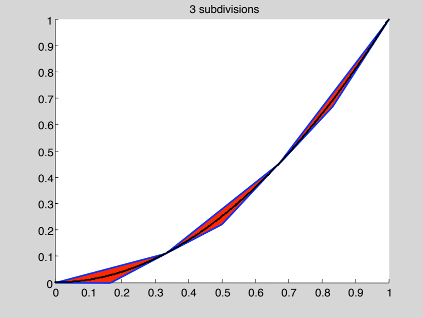
# Non-affine dependence on $\delta$

Let $g : \mathbb{R} \rightarrow \mathbb{R}$.

$$\begin{aligned} \dot{x}(t) &= f_0(x(t)) + \delta f_1(x(t)) + g(\delta) f_2(x(t)) \\ &= f_0(x(t)) + \delta f_1(x(t)) + \zeta f_2(x(t)) \end{aligned}$$

Treat $(\delta, g(\delta))$ as 2 parameters, whose values lie on a 1-dimensional curve. Then

$*$ Cover 1-d curve with 2-polytope
$*$ Compute ROA
$*$ Refine polytope into a union of smaller polytopes
$*$ Solve robust ROA on each polytope
$*$ Intersect ROAs → robust ROA



# Generalization of covering manifold

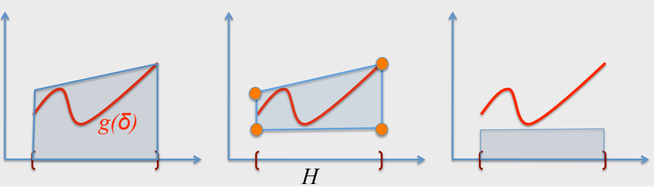
Given:

- polynomial $g(\delta)$ in many real variables, $\delta \in \mathbb{R}^q$
- domain $H \subseteq \mathbb{R}^q$, typically a polytope

Find a polytope that covers $\{(\delta, g(\delta)) : \delta \in H\} \subseteq \mathbb{R}^{q+1}$.

- Tradeoff between number of vertices, and
- excess "volume" in polytope

One approach: Find "tightest" affine upper and lower bounds to $g$ over $H$

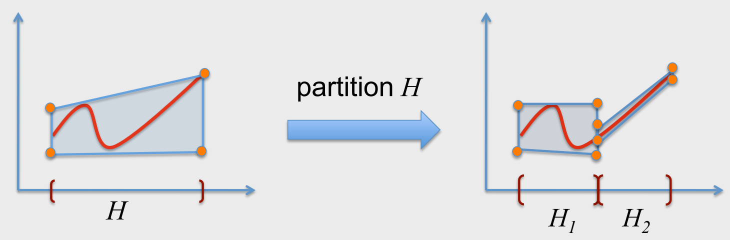


$$\min_{c_0,c} \int_H (c_0 + c^T\delta)d\delta \quad \text{subject to} \quad c_0 + c^T\delta \geq g(\delta) \;\; \forall \delta \in H$$

This optimization can be solved as a SOS program.

# Non-affine dependence on $\delta$ (2)

Covering $\{(\delta, g(\delta) \; : \; \delta \in H\}$ introduces extra conservatism.



B&B refinement reduces the conservatism due to covering by reducing the extra covered space.

# Multiple non-affine parametric uncertainty

For multivariable $g$,

$$\dot{x} = f_0(x) + \delta_1 f_1(x) + \cdots + \delta_q f_q(x) + \\ g_1(\delta) f_{q+1}(x) + \cdots + g_m(\delta) f_{q+m}(x)$$

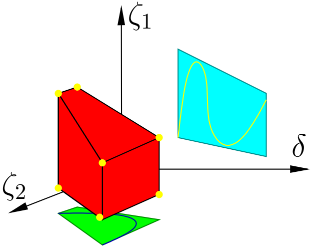
On $H$, bound each $g_i$ with affine functions $c_i$ and $d_i$

$$c_i(\delta) \leq g_i(\delta) \leq d_i(\delta) \quad \forall \delta \in H$$

Then (Amato, Garofalo, Gliemo) a polytope covering $\{(\delta, g(\delta)) : \delta \in H\}$ is

$$\left\{(\delta, v) \in \mathbb{R}^{q \times m} : \delta \in H, C(\delta) \leq v \leq D(\delta)\right\}$$

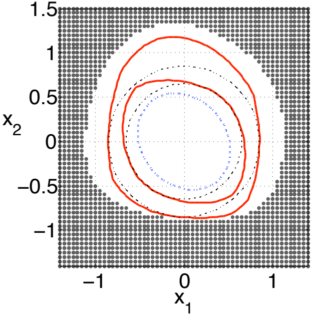
with $2^{q+m}$ easily computed vertices.



# Example: Interesting 2-state uncertain dynamics [Chesi, 2004]

$$\dot{x} = \begin{bmatrix} -x_1 \\ 3x_1 - 2x_2 \end{bmatrix} - \begin{bmatrix} 6x_2 - x_2^2 - x_1^3 \\ 10x_1 - 6x_2 - x_1x_2 \end{bmatrix} \delta + \begin{bmatrix} 4x_2 - x_2^2 \\ 12x_1 - 4x_2 \end{bmatrix} \delta^2,$$

- $\delta \in [0, 1]$.
- No common quadratic $V$ for uncertain linearized dyn.
- $p(x) = x^T x$.
- $50$ branch-and-bound refinements



Blue dotted curve: Result from Chesi, 2004.

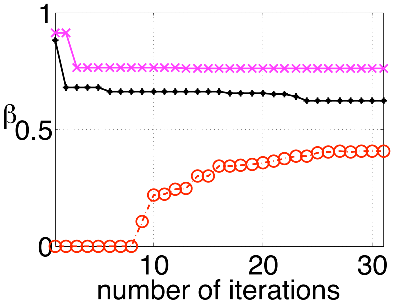
Red curves: Intersection of $\Omega_{V,1}$ for $V$'s obtained through the B&B refinement (inner for deg($V$) = 2 and outer for deg($V$) = 4)

Black dotted curves: Certified $\Omega_{p,\beta}$ for deg($V$) = 2 (inner) and for deg($V$) = 4 (outer)

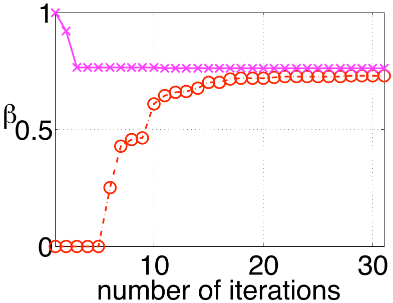
# Example: Interesting 2-state uncertain dynamics

**B&B iterations:** Divide the cell with the smallest $\beta$ into 2.

$\underline{\deg(V) = 2}$



$\underline{\deg(V) = 4}$



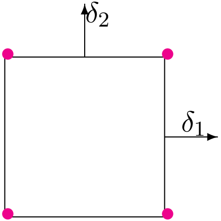
- Upper bounds from divergent trajectories
  - Upper bound does not depend on the complexity/degree of V
- Upper bounds from infeasibility of the affine relaxation
  - This bound shows how the basis choice for $V$ impacts what is certifiable.
- Certified values (using ideas from last previous 100+ slides)

# Dealing with large number of constraints

The SOS problem for the robust ROA includes the constraint:

$$-[(\gamma - V)s_{2\delta} + \nabla V(f_0 + F(x)\delta)s_{3\delta} + l_2] \in \Sigma[x] \quad \forall \delta \in \mathcal{E}$$

The number of vertices grows fast with the dimension of the uncertainty space.



Suboptimal procedure:

- Sample $\Delta$ with fewer points (fewer than in $\mathcal{E}$)
- Optimize $V$ for this restricted sampling
- Certify a value of $\beta$, using this $V$, at all vertices of $\Delta$

The last step involves solving decoupled smaller problems.

# Dealing with large number of constraints: 2-step procedure

- Call the Lyapunov function computed for a sample of $\Delta$ as $\tilde{V}$.
- For each $\delta \in \mathcal{E}$, compute

$$\gamma_\delta := \max_{0<\gamma, s_{2\delta}\in\mathcal{S}_2, s_{3\delta}\in\mathcal{S}_3} \gamma \quad \text{subject to}$$
$$s_{2\delta} \in \Sigma[x], \text{ and } s_{3\delta} \in \Sigma[x]$$
$$-[(\gamma - \tilde{V})s_{2\delta} + \nabla\tilde{V}(f_0 + F\delta)s_{3\delta} + l_2] \in \Sigma[x],$$

  and define

$$\gamma^{subopt} := \min\{\gamma_\delta \ : \ \delta \in \mathcal{E}\}.$$

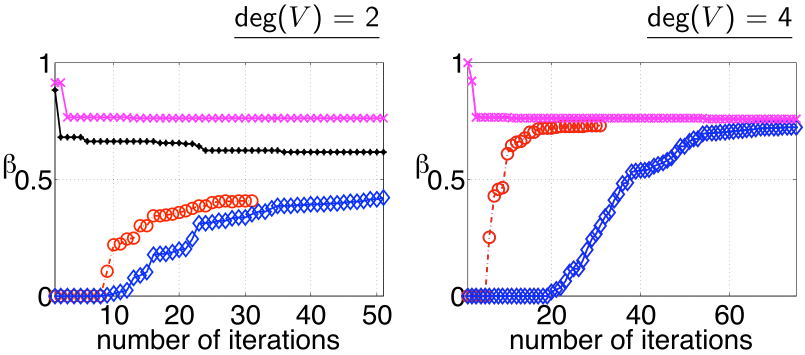
  $\Omega_{\tilde{V},\gamma^{subopt}}$ is an invariant subset of the robust ROA.

- Determine the largest sublevel set of $p$ contained in $\Omega_{\tilde{V},\gamma^{subopt}}$

$$\max_{s_1\in\mathcal{S}_1,\beta} \beta \quad \text{subject to}$$
$$s_1 \in \Sigma[x]$$
$$-[(\beta - p)s_1 + \tilde{V} - \gamma^{subopt}] \in \Sigma[x].$$

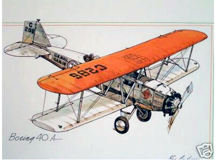
# Revisit *Chesi, 2004* with suboptimal $\Delta$ sampling

**B&B iterations:** Divide the cell with the smallest $\beta$ into 2.



- Upper bounds from divergent trajectories
- Upper bounds from infeasibility of the affine relaxation
- Lower bounds directly computing the robust ROA
- Lower bounds computing the robust ROA in two steps (sample $\Delta$ at cell center $\rightarrow$ optimize $V$ $\rightarrow$ verify at the vertices)

# Controlled aircraft [Short period pitch axis model]



Uncertain closed loop dynamics with

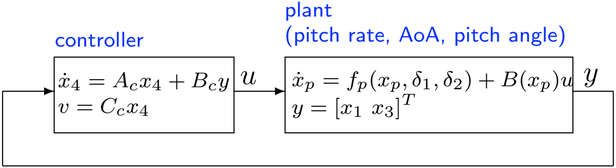
- $x = (x_p,\ x_4),\quad p(x) = x^T x$
- Cubic poly approx from Honeywell

$$\dot{x} = f_0(x) + f_1(x)\delta_1 + f_2(x)\delta_2 + f_3(x)\delta_1^2$$

- $\delta_1 \in [0.99, 2.05]$ (uncertainty in the center of gravity)
- $\delta_2 \in [-0.1, 0.1]$ (uncertainty in mass)

Implemented on a 9-processor cluster

- Problems for $9$ cells are solved at a time
- Trivial speed up as expected.



controller

$\dot{x}_4 = A_c x_4 + B_c y$
$v = C_c x_4$

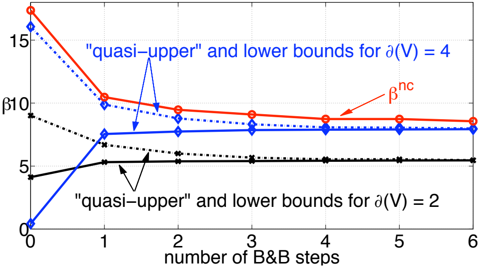
$u$

plant
(pitch rate, AoA, pitch angle)

$\dot{x}_p = f_p(x_p, \delta_1, \delta_2) + B(x_p)u$
$y = [x_1\ x_3]^T$

$y$

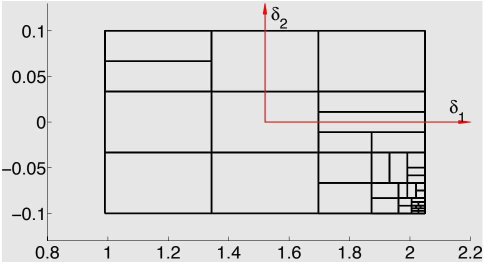
# Results - controlled aircraft dynamics



Strategy:

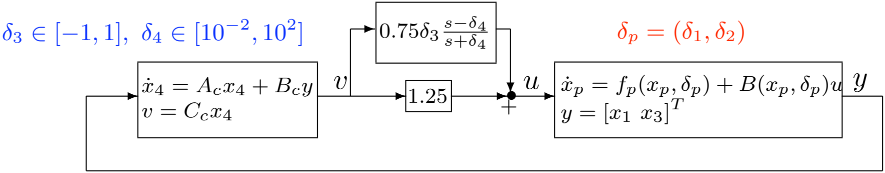
- Optimize at the center
- Verify at the vertices

Quasi upper bound: $\beta$ certified (by the SOS problem) for the "center system" in the first step.



$$\dot{x} = f_0(x) + f_1(x)\delta_1 + f_2(x)\delta_2 + f_3(x)\delta_1^2$$

# Controlled aircraft + 1st order unmodeled dynamics



$\delta_3 \in [-1,1],\ \delta_4 \in [10^{-2}, 10^2]$
$0.75\delta_3 \frac{s-\delta_4}{s+\delta_4}$
$\delta_p = (\delta_1, \delta_2)$
$\dot{x}_4 = A_c x_4 + B_c y$
$v = C_c x_4$
$v$
$1.25$
$u$
$+$
$\dot{x}_p = f_p(x_p, \delta_p) + B(x_p, \delta_p)u$
$y = [x_1\ x_3]^T$
$y$

$$\dot{x} = f_0(x) + \sum_{i=1}^{4} f_i(x)\delta_i + f_5(x)\delta_1^2 + f_6(x)\delta_1\delta_3 + f_7(x)\delta_2\delta_3$$

- First order LTI unmodeled dyn (state $x_5$)
- $p(x) = x^T x$, $x = \begin{bmatrix} x_p^T & x_4 & x_5 \end{bmatrix}^T$.

**Certified**

| param uncer / dyn uncer | with | without |
|---|---|---|
| with | 2.8 | 4.9 |
| without | 5.4 | 8.0 |

How about other uncertainty descriptions (e.g. unmodeled dynamics)?

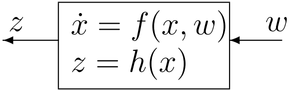
Coming up later

# Local input-output analysis

# What if there is external input/disturbance?

So far, only internal properties, no external inputs!

What if there are external inputs/disturbances?



$$f(0,0) = 0, \; h(0) = 0$$

If $w$ has **bounded energy/amplitude** and system starts from **rest**

- (reachability) how far can $x$ be driven from the origin?
- (input-output gain) what are bounds on the output energy/amplitude in terms of input energy?

# Notation

- For $u : [0, \infty) \to \mathbb{R}^n$, define the (truncated) $\mathcal{L}_2$ norm as

$$\|u\|_{2,T} := \sqrt{\int_0^T u(t)^T u(t) dt}.$$

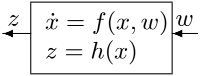
- For simplicity, denote $\|u\|_{2,\infty}$ by $\|u\|_2$.
- $\mathcal{L}_2$ is the set of all functions $u : [0, \infty) \to \mathbb{R}^n$ such that $\|u\|_2 < 0$.
- For $u : [0, \infty) \to \mathbb{R}^n$ and for $T \geq 0$, define $u_T : [0, \infty) \to \mathbb{R}^n$ as

$$u_T(t) : \begin{cases} u(t), & 0 \leq t \leq T \\ 0, & T < t \end{cases}$$

- $\mathcal{L}_{2,e}$ is the set of measurable functions $u : [0, \infty) \to \mathbb{R}^n$ such that $u_T \in \mathcal{L}_2$ for all $T \geq 0$.

# Upper bounds on "local" $\mathcal{L}_2 \rightarrow \mathcal{L}_2$ input-output gains

**Goal:** Establish relations between inputs and outputs:



$z$

$\dot{x} = f(x,w)$

$z = h(x)$

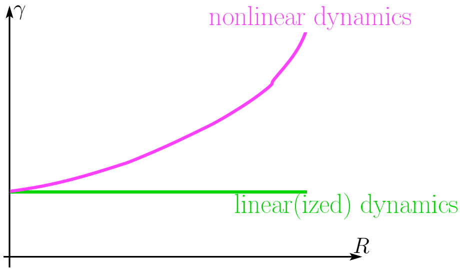
$w$

$$x(0) = 0 \;\&\; \|w\|_2 \leq R \qquad \Rightarrow \qquad \|z\|_2 \leq \gamma \|w\|_2.$$

- Given $R$, minimize $\gamma$
- Given $\gamma$, maximize $R$

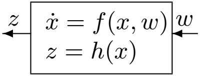
The $\mathcal{H}_\infty$ norm is a lower bound on the set of $\gamma$'s which satisfy inequalty.

Why "local" analysis?



# Upper bounds on "local" $\mathcal{L}_2 \to \mathcal{L}_2$ input-output gains

**Goal:** Establish relations between inputs and outputs:

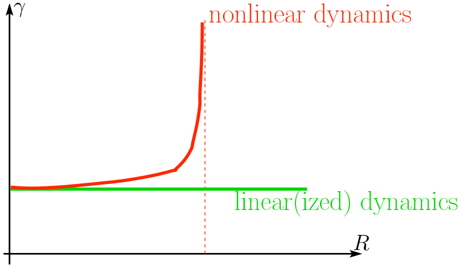


$$x(0) = 0 \ \& \ \|w\|_2 \leq R \qquad \Rightarrow \qquad \|z\|_2 \leq \gamma \|w\|_2.$$

- Given $R$, minimize $\gamma$
- Given $\gamma$, maximize $R$

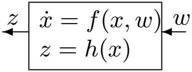
The $\mathcal{H}_\infty$ norm is a lower bound on the set of $\gamma$'s which satisfy inequalty.

Why "local" analysis?



# Local gain analysis

**Theorem:** If there exists a continuously differentiable function $V$ such that $V(0) = 0$, $V(x) > 0$ for all $x \neq 0$,



$$\dot{x} = f(x,w)$$
$$z = h(x)$$

- $\Omega_{V,R^2} := \{x \ : \ V(x) \leq R^2\}$ is bounded
- $\nabla V f(x,w) \leq w^T w - \frac{1}{\gamma^2} h(x)^T h(x)$ for all $x \in \Omega_{V,R^2}$ and $w \in \mathbb{R}^{n_w}$,

then

$$x(0) = 0, \ w \in \mathcal{L}_{2,e}, \ \& \ \|w\|_{2,T} \leq R \qquad \Rightarrow \qquad \|z\|_{2,T} \leq \gamma \|w\|_{2,T}.$$

- Note that algebraic condition on $(x,w) \in \mathbb{R}^n \times \mathbb{R}^{n_w}$ implies a relation between the signals $w \in \mathcal{L}_{2,e}$ and $z = h(x) \in \mathcal{L}_{2,e}$.
- Supply rate, $w^T w - \frac{1}{\gamma^2} h(x)^T h(x)$; Storage function, $V$.

# Bilinear SOS problem formulation for gain analysis

For given $\gamma > 0$ and positive definite function $l$, define $R_{\mathcal{L}_2}$ by

$$R_{\mathcal{L}_2}^2 := \max_{V \in \mathcal{V}_{poly}, R^2 > 0, s_1 \in \mathcal{S}_1} \quad R^2 \quad \text{subject to}$$

$$V(0) = 0, \quad s_1 \in \Sigma[(x, w)],$$

$$V - l \in \Sigma[x],$$

$$-\left[(R^2 - V)s_1 + \nabla V f(x, w) - w^T w + \gamma^{-2} z^T z\right] \in \Sigma[(x, w)].$$

Then,

$$x(0) = 0 \; \& \; \|w\|_2 \leq R_{\mathcal{L}_2} \qquad \Rightarrow \qquad \|z\|_2 \leq \gamma \|w\|_2.$$

- $\mathcal{V}_{poly}$ and $\mathcal{S}$'s are prescribed finite-dimensional subsets of $\mathbb{R}[x]$.
- $R_{\mathcal{L}_2}^2$ is a function of $\mathcal{V}_{poly}$, $\mathcal{S}$, and $\gamma$. This dependence will be dropped in notation.

- Similar problem for minimizing $\gamma$ for given $R$.

# Strategy to solve the bilinear SOS problem in gain analysis

**Coordinate-wise affine search:** Given a “feasible” $V$, alternate between

- maximize $R^2$ by choice of $s_1$ (requires bisection on $R$!)

$$R^2_{\mathcal{L}_2} := \max_{R^2>0, s_1 \in \mathcal{S}_1} \quad R^2 \quad \text{subject to}$$
$$s_1 \in \Sigma[(x,w)],$$
$$-\left[(R^2 - V)s_1 + \nabla V f(x,w) - w^T w + \gamma^{-2} z^T z\right] \in \Sigma[(x,w)].$$

- fix the multiplier and maximize $R^2$ by choice of $V$.

$$R^2_{\mathcal{L}_2} := \max_{V \in \mathcal{V}_{poly}, R^2>0} \quad R^2 \quad \text{subject to}$$
$$V(0) = 0, \quad V - l \in \Sigma[x],$$
$$-\left[(R^2 - V)s_1 + \nabla V f(x,w) - w^T w + \gamma^{-2} z^T z\right] \in \Sigma[(x,w)].$$

# Strategy to solve the bilinear SOS problem in gain analysis

**Finding initial "feasible" $V$:**

- Incorporate simulation data (requires to sample the input space!)
- Let $\gamma >$ gain of the linearized dynamics

$$\begin{aligned} \dot{\delta}_x &= A\delta_x + \delta_w \\ \delta_z &= C\delta_x \end{aligned}$$

and let $P \succ 0$ satisfy

$$\begin{bmatrix} A^TP + PA + \frac{1}{\gamma^2}C^TC & PB \\ B^TP & -I \end{bmatrix} \prec 0.$$

Then, there exists a small enough $R$ such that

$$x(0) = 0 \ \& \ \|w\|_2 \leq R \qquad \Rightarrow \qquad \|z\|_2 \leq \gamma\|w\|_2.$$

# Lower bound for $\mathcal{L}_2 \to \mathcal{L}_2$ gain

Let $\gamma$ and $R$ be obtained through the SOS based gain analysis. Then, for $T \geq 0$

$$\max_w \{\|z\|_{2,T} \;:\; x(0) = 0 \;\&\; \|w\|_{2,T} \leq R\} \leq \gamma R.$$

The first-order conditions for stationarity of the above finite horizon maximum are the existence of signals $(x, \lambda)$ and $w$ which satisfy

$$\begin{aligned}
\dot{x} &= f(x, w) \\
\|w\|_{2,T}^2 &= R^2 \\
\lambda(T) &= \left(\frac{\partial \|z\|_{2,T}^2}{\partial x}\right)^T \\
\dot{\lambda}(t) &= -\left(\tfrac{\partial f(x(t),w(t))}{\partial x}\right)^T \lambda(t) \\
w(t) &= \mu \left(\tfrac{\partial f(x(t),w(t))}{\partial w}\right)^T \lambda(t),
\end{aligned}$$

for $t \in [0, T]$, where $\mu$ is chosen such that $\|w\|_{2,T} = R$.
Tierno, et.al., propose a power-like method to solve a similar maximization.

# Gain Lower-Bound Power Algorithm

Adapting for this case yields: Pick $T > 0$ and $w$ with $\|w\|_{2,T}^2 = R^2$. Repeat the following steps until $w$ converges.

1. Compute $\|z\|_{2,T}$ (integration $\dot{x} = f(x, w)$ with $x(0) = 0$ forward in time).
2. Set $\lambda(T) = \left( \frac{\partial \|z\|_{2,T}^2}{\partial x} \right)^T$.
3. Compute the solution of $\dot{\lambda}(t) = -\frac{\partial f(x(t), w(t))}{\partial x}^T \lambda(t)$, $t \in [0, T]$ (integration backward in time).
4. Update $w(t) = \mu \left( \frac{\partial f(x(t), w(t))}{\partial w} \right)^T \lambda(t)$.

- Step (1) of each iteration gives a valid lower bound on the maximum (over $\|w\|_2 = R$) of $\|z\|_{2,T}$, independent of whether the iteration converges;
- (main point of Tierno) if dynamics are linear and $p$ quadratic, then the iteration is convergent power iteration for $\mathcal{H}_\infty$.

Implemented in `worstcase`.

# Adaptive Control: I/O Gain

**Plant:**

$$\dot{x} = -x + w + u$$
$$y = -1.8x + w + u$$

$x \in \mathbb{R}$ is the plant state, $u \in \mathbb{R}$ is the control input, $y \in \mathbb{R}$ is the output, and $w \in \mathbb{R}$ is a disturbance.

**Model-reference adaptive controller:**

$$\dot{x}_m = -x_m + r$$
$$\dot{z}_x = -x^2 + x x_m$$
$$\dot{z}_r = -xr + x_m r$$
$$u = (1 + z_r) r + z_x x$$

$x_m$ is the reference model state, $r$ is the reference signal, and $z_x$ and $z_r$ are feedback gains which are tuned by the adaptation.

**Question:** What is the gain from disturbance $w$ to output $y$?

# Results: Gain Bounds

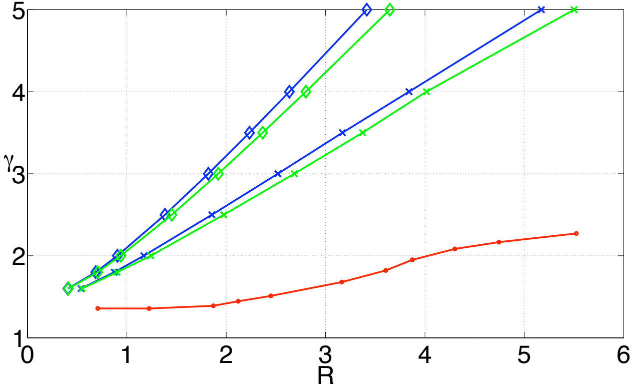


Figure: Upper bounds on $\|S\|_R$ for deg(V) = 2 (with $\diamond$) and deg(V) = 4 (with $\times$) before the refinement (blue curves) and after the refinement (green curves) along with the lower bounds (red curve).

# Upper bounds on the reachable set

$$\dot{x} = f(x, w) \qquad \text{with} \qquad f(0,0) = 0$$

- Find upper bounds on the reachable set from the origin for bounded $\mathcal{L}_2$ input norm
  - Denote the set of points reached from the origin with input signals $w$ such that $\|w\|_2 \le R$ by $\mathtt{Reach}_R$.

$$\mathtt{Reach}_R := \{x(t) \ : \ x(0) = 0, \ t \ge 0, \ \|w\|_2 \le R\}$$

**Goal:**

- Given a shape factor $p$ (positive definite, convex function with $p(0) = 0$), establish relations of the form

$$x(0) = 0 \ \& \ \|w\|_2 \le R \qquad \Rightarrow \qquad p(x(t)) \le \beta \ \ \forall t \ge 0.$$

- Two types of optimization
  - Given $R$, minimize $\beta$
  - Given $\beta$, maximize $R$

# A characterization of upper bounds on the reachable set

$$\dot{x} = f(x, w) \qquad \text{with} \qquad f(0,0) = 0$$

**Theorem:** If there exists a continuously differentiable function $V$ such that

- $V(x) > 0$ for all $x \neq 0$ and $V(0) = 0$
- $\Omega_{V,R^2} = \left\{\xi \ : \ V(\xi) \leq R^2\right\}$ is bounded
- $\nabla V f(x, w) \leq w^T w$ for all $x \in \Omega_{V,R^2}$ and for all $w \in \mathbb{R}^{n_w}$

then $\texttt{Reach}_R \subseteq \Omega_{V,R^2}$.

Given $R$, solve

$$\begin{aligned} &\min_{V,\beta} \quad \beta \\ &\text{s.t.} \quad \Omega_{V,R^2} \subseteq \Omega_{p,\beta} \\ &\text{V satisfies above conditions} \end{aligned}$$

OR

Given $\beta$, solve

$$\begin{aligned} &\max_{V,R^2} \quad R^2 \\ &\text{s.t.} \quad \Omega_{V,R^2} \subseteq \Omega_{p,\beta} \\ &\text{V satisfies above conditions} \end{aligned}$$

# Bilinear SOS problem formulation for reachability analysis

**Original Problem**

$$\max_{R^2, V} \quad R^2$$

subject to:

$$V(0) = 0, \quad V(x) > 0 \;\forall x \neq 0$$

$\left\{x \in \mathbb{R}^n \;:\; V(x) \leq R^2\right\}$ is bounded

$$\Omega_{V,R^2} \subseteq \Omega_{p,\beta}$$

$$\nabla V f(x,w) \leq w^T w \;\; \forall\, x \in \Omega_{V,R^2} \;\&\; w \in \mathbb{R}^{n_w}$$

$\Uparrow$ S-procedure - SOS

**Reformulation**

$$\max_{R^2, V, s_1, s_2} \quad R^2$$

subject to:

$-\left[(\beta - p) + (V - R^2)s_1\right]$ is SOS$[x]$,

$-\left[(R^2 - V)s_2 + \nabla V f(x,w) + w^T w\right]$ is SOS$[x,w]$,

$V - \epsilon x^T x$ is SOS$[x]$, $V(0) = 0,$ and

$s_1, s_2, s_3$ are SOS.

# Generalizations: dissipation inequalities

The system

$$\begin{aligned}\dot{x} &= f(x, w) \\ z &= h(x)\end{aligned}$$

with $f(0, 0) = 0$ and $h(0) = 0$ is said to be **dissipative** w.r.t. to the **supply rate** $r : (w, z) \mapsto \mathbb{R}$ if there exists a positive definite function $V$ such that $V(0) = 0$ and the following dissipation inequality (DIE) holds

$$\frac{\partial V}{\partial x} f(x, w) \leq r(w, z)$$

for all $x \in \mathbb{R}^n$ & $w \in \mathbb{R}^{n_w}$.

- $\mathcal{L}_2 \rightarrow \mathcal{L}_2$ **gain:** $r(w, z) = w^T w - z^T z$
- **Reachability:** $r(w, z) = w^T w$

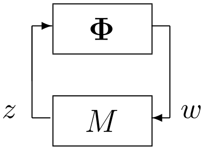
The system is said to be **locally** dissipative if the above DIE holds only for all $x \in \{x \;:\; V(x) \leq \gamma\}$ for some $\gamma > 0$.

# Robust ROA and performance analysis with unmodeled dynamics

# Recall: the small-gain theorem

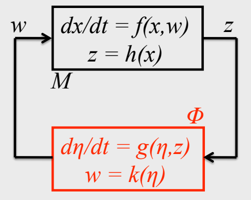
For stable $M$ and $\Phi$, the feedback interconnection is internally stable if

$$\gamma(M)\gamma(\Phi) < 1.$$



- $\gamma$ is an upper bound on the global $\mathcal{L}_2 \to \mathcal{L}_2$ gain.
- Extensively used in linear robustness analysis where $M$ is linear time-invariant (existence of global gains is guaranteed).
- How to generalize to nonlinear $M$ with possibly only local gain relations?

# Local small-gain theorems for stability analysis



Let $l$ be a positive definite function with $l(0) = 0$ e.g. $l(x) = \epsilon x^T x$ and $R > 0$.
Let $\tilde{l}$ be a positive definite function with $\tilde{l}(0) = 0$.

For $M$: There exists a positive definite function $V$ such that $\Omega_{V,R^2}$ is bounded and for all $x \in \Omega_{V,R^2}$ and $w \in \mathbb{R}^{n_w}$

$$\nabla V \cdot f(x,w) \leq w^T w - h(x)^T h(x) - l(x).$$

[$M$ is “locally strictly dissipative” w.r.t. the supply rate $w^T w - z^T z$ certified by the storage function $V$.]

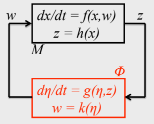
For $\Phi$: There exists a positive definite function $Q$ such that for all $\eta \in \mathbb{R}^{n_\eta}$ and $z \in \mathbb{R}^{n_z}$

$$\nabla Q \cdot g(\eta,z) \leq z^T z - k(\eta)^T k(\eta) - \tilde{l}(\eta).$$

[$\Phi$ is “strictly dissipative” w.r.t. $z^T z - w^T w$.]

# Local small-gain theorems for stability analysis (2)

**Conclusion:** $S := V + Q$ is a Lyapunov function for the closed-loop for the closed-loop dynamics ($\dot{\xi} = F(\xi)$).



$$\xi = \begin{bmatrix} x \\ \eta \end{bmatrix}$$

**Proof:**

$$\begin{aligned} \nabla V \cdot f(x,w) &\leq w^T w - z^T z - l(x) && \forall x \in \Omega_{V,R^2} \ \& \ w \in \mathbb{R}^{n_w} \\ \nabla Q \cdot g(\eta, z) &\leq z^T z - w^T w - \tilde{l}(\eta) && \forall \eta \in \mathbb{R}^{n_\eta} \ \& \ z \in \mathbb{R}^{n_z} \end{aligned}$$

$$\nabla V \cdot f(x, g(\eta)) + \nabla Q \cdot g(\eta, h(x)) \leq l(x) + \tilde{l}(\eta) \\ \forall (x,\eta) \in \left\{(x,\eta) \ : \ V(x) + Q(\eta) \leq R^2\right\}$$

$$\nabla S \cdot F(\xi) \leq -l(x) - \tilde{l}(\eta) = -L(\xi) \\ \forall (x,\eta) \in \left\{(x,\eta) \ : \ S(x,\eta) \leq R^2\right\}$$

**Corollary:**

- $\{(x,\eta) \ : \ V(x) + Q(\eta) \leq R^2\}$ is an invariant subset of the ROA for the closed-loop dynamics.

# Estimating the ROA (for $x$ states)

Let $p$ be a shape factor (as before) and $(\bar{V}, \bar{\beta}, \bar{R})$ be a solution to the above optimization

$$\max_{V \in \mathcal{V}, \beta \geq 0, R \geq 0} \quad \beta \quad \text{subject to}$$

$$V(x) > 0 \text{ for all } x \neq 0, \quad V(0) = 0,$$

$$\Omega_{p,\beta} \subseteq \Omega_{V,R^2},$$

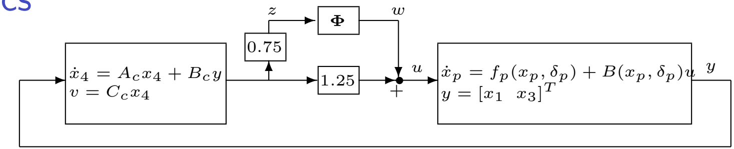
$$\Omega_{V,R^2} \text{ is bounded},$$

$$\nabla V f(x,w) \leq w^T w - z^T z - l(x) \quad \forall\ x \in \Omega_{V,R^2}, \ \ \forall\ w \in \mathbb{R}^{n_w}.$$

If $\Phi$ is strictly dissipative w.r.t. $z^T z - w^T w$ and $\eta(0) = 0$, then for any $x(0) \in \Omega_{p,\bar{\beta}}$,

- $x(t)$ stays in $\Omega_{\bar{V},\bar{R}^2}$
- $x(t) \to 0$ as $t \to \infty$.

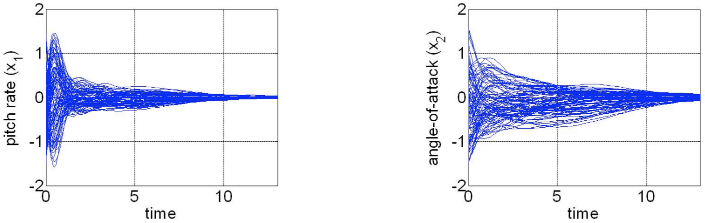
# Example: Controlled aircraft dynamics with unmodeled dynamics


$z$
$\mathbf{\Phi}$
$w$
$0.75$
$\dot{x}_4 = A_c x_4 + B_c y$
$v = C_c x_4$
$1.25$
$u$
$+$
$\dot{x}_p = f_p(x_p, \delta_p) + B(x_p, \delta_p) u$
$y = [x_1 \ \ x_3]^T$
$y$

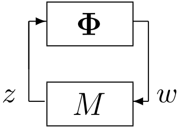
| | no $\delta_p$ | with $\delta_p$ |
|---|---|---|
| no $\Delta$ | 9.4 / 16.1 | 5.5 / 7.9 |
| with $\Delta$ | 4.2 / 6.7 | 2.4 / 4.1 |

In the table :
$(\partial(V) = 2/\partial(V) = 4)$

Closed-loop response with randomly generated first-order LTI $\Phi$:



# Generalization to generic supply rates



Results hold when the "$\mathcal{L}_2$-gain supply rate" is replaced by a general supply rate.

Suppose that

- $\Phi$ is strictly dissipative w.r.t. the supply rate $r_1(z,w)$ with the corresponding storage function $Q$
- $M$ satisfies

$$V f(x,w) \leq r_2(w,z) - l(x) \quad \forall x \in \Omega_{V,R^2} \;\&\; w \in \mathbb{R}^{n_w}$$

with

$$r_1(z,w) = -r_2(w,z) \quad \forall w, z.$$

Then, $\{(x,\eta) \;:\; V(x) + Q(\eta) \leq R^2\}$ is an invariant subset of the ROA for the closed-loop dynamics.

# General procedure to construct “certificates”

- System properties → Algebraic conditions
    - Lyapunov, dissipation inequalities.
- Algebraic conditions → Numerical optimization problems
    - Restrict the attention to polynomial vector fields, polynomial certificates,...
    - S-procedure like conditions (for set containment constraints)
    - Sum-of-squares (SOS) relaxations for polynomial nonnegativity
    - Pass to semidefinite programming (SDP) that are equivalent of SOS conditions
- Solve the resulting (linear or “bilinear”) SDPs
- Construct polynomial certificates

Recurring procedure for most computational analysis questions (that I know) for dynamical systems.

# Robust ROA calculations

dynamics:

x1dot = x2;

x2dot = -x2-2*x1+2*x1^3 + delta*(-x1+x1^3);

with delta \in [-1,1]

This example was also used in Topcu and Packard, IEEE TAC, 2009 (in the special issue on positive polynomials in controls (example 1 in the paper)

```
% Form the vector field
pvar x1 x2;
x = [x1;x2];
x1dot = x2;
x2dot = -x2-2*x1+2*x1^3;
```

Nominal system

```
f = [x1dot; x2dot];
```

Introduce an uncertain parameter

```
pvar d1
```

Specify its range

```
ini_cell = [-1 1];
```

Form the uncertain vector field

```
f = f + d1*[0; -x1+x1^3];


% Get the vertex system
[roaconstr,opt,sys] = GetRoaOpts(f, x);
[fNOM,fVER] = getf(sys,ini_cell);

% Generate the options, etc.
zV = monomials(x,2:4);
Bis.flag = 0;
Bis.r1deg = 4;


[roaconstr,opt,sys] = GetRoaOpts(fVER, x, zV, [], Bis);
sys.fWithDel = [];

opt.sim.NumConvTraj = 40;
opt.display.roaest = 1;
```

## Run the computations

```
outputs = wrapper(sys,[],roaconstr,opt);
```

```
------------------Beginning simulations
System 1: Num Stable = 0        Num Unstable = 1        Beta for Sims = 3.289   Beta UB = 3.289
System 1: Num Stable = 0        Num Unstable = 2        Beta for Sims = 1.390   Beta UB = 1.390
System 1: Num Stable = 2        Num Unstable = 3        Beta for Sims = 1.306   Beta UB = 1.306
System 1: Num Stable = 4        Num Unstable = 4        Beta for Sims = 0.913   Beta UB = 0.913
System 1: Num Stable = 6        Num Unstable = 5        Beta for Sims = 0.861   Beta UB = 0.861
System 1: Num Stable = 12       Num Unstable = 6        Beta for Sims = 0.818   Beta UB = 0.842
System 1: Num Stable = 18       Num Unstable = 7        Beta for Sims = 0.777   Beta UB = 0.808
System 2: Num Stable = 1        Num Unstable = 1        Beta for Sims = 1.476   Beta UB = 0.808
System 2: Num Stable = 3        Num Unstable = 2        Beta for Sims = 1.402   Beta UB = 0.808
System 2: Num Stable = 6        Num Unstable = 3        Beta for Sims = 1.114   Beta UB = 0.808
System 2: Num Stable = 6        Num Unstable = 4        Beta for Sims = 1.058   Beta UB = 0.808
System 2: Num Stable = 8        Num Unstable = 5        Beta for Sims = 1.000   Beta UB = 0.808
System 2: Num Stable = 10       Num Unstable = 6        Beta for Sims = 0.929   Beta UB = 0.808
System 2: Num Stable = 11       Num Unstable = 7        Beta for Sims = 0.882   Beta UB = 0.808
------------------End of simulations
------------------Begin search for feasible V
Try = 1          Beta for Vfeas = 0.882
Try = 2          Beta for Vfeas = 0.838
------------------Found feasible V
Initial V (from the cvx outer bnd) gives Beta = 0.173
-------------------Iteration = 1
Beta = 0.567 (Gamma = 0.535)
-------------------Iteration = 2
Beta = 0.665 (Gamma = 0.604)
-------------------Iteration = 3
Beta = 0.716 (Gamma = 0.640)
-------------------Iteration = 4
Beta = 0.739 (Gamma = 0.656)
```

## Extract the solution

```
[V,beta,gamma,p,multip,betaUpper] = extractSol(outputs);
```

```
beta
```

```
beta =

    0.7388
```

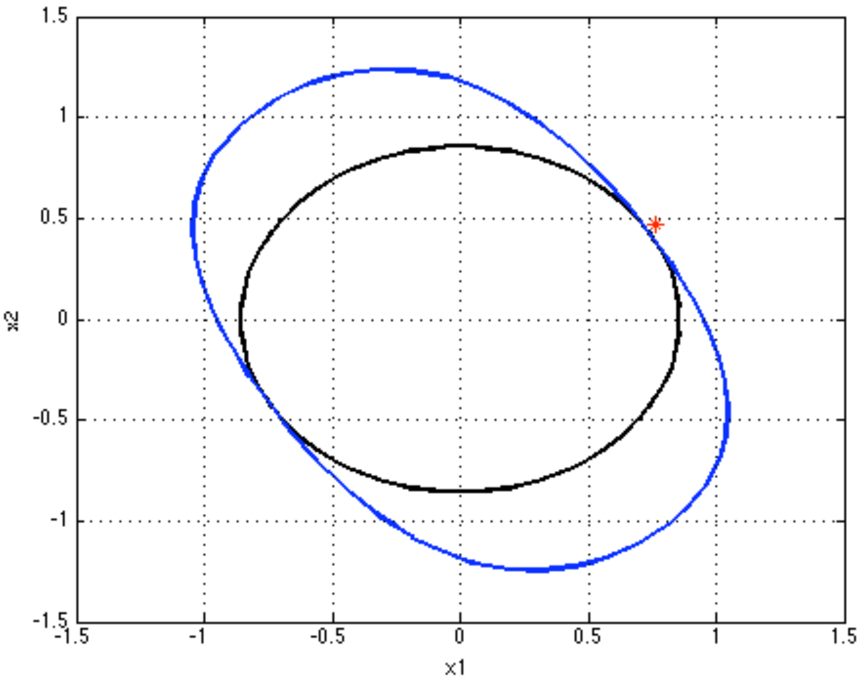
## Upper bound on beta

```
betaUpper
```

```
betaUpper =

    0.8822
```

## Plot the results

```
[Cp4,hp4] = pcontour(p,beta,[-2 2 -2 2],'k'); hold on;
set(hp4,'linewidth',2);
[CV4,hV4] = pcontour(V,gamma,[-2 2 -2 2],'b');
set(hV4,'linewidth',2);
set(gca,'xlim',[-1.5 1.5],'ylim',[-1.5 1.5]);

traj = outputs.RoaEstInfo.info.SimLFG.sim.Trajectories(1).unstab(end).state;
pval = peval(traj,p.coef,p.deg);
[aux,ind] = min(pval);
plot(traj(1,ind),traj(2,ind),'r*','markersize',8);
grid on;
```



*Published with MATLAB® 7.6*