Lyapunov functions

- Consider a damped pendulum:

$$mgl\sin(\theta) - b\dot{\theta} + ml^2\ddot{\theta} = 0.$$

- No closed-form equation for trajectories, but we know they all converge to $\dot{\theta} = \theta = 0$. Why? Because the energy is always decreasing!
- **Lyapunov function**: a function V of the state x such that $\dot{V}(x) \leq 0$. Implies that if $V(x(0) < \rho$ then $V(x(t)) < \rho$ for all $t \geq 0$ (global asymptotic stability, assuming the sublevel sets of V are bounded).
  - If $\dot{x} = f(x)$, then $\dot{V}(x) = \frac{\partial V}{\partial x} f(x)$.

Variants on the Lyapunov condition

Guarantee	Conditions
convergence to origin	$\dot{V}(x) < 0$ if $x \neq 0, V(x) \geq V(0), V$ is continuous
exponential convergence	$\dot{V}(x) \leq -cV(x), V(x) \geq V(0), V$ is analytic
local stability	conditions only need to hold when $V(x) < \rho$

Connection to Bellman Equations

- Suppose we have a Markov chain with a cost $h(x, n)$ on being in state x at time n.
- The "cost-to-go" function is defined as

$$J(x, n) = \mathbb{E}\left[\sum_{m=n}^{N} h(x(m), m) \mid x(n) = x\right],$$

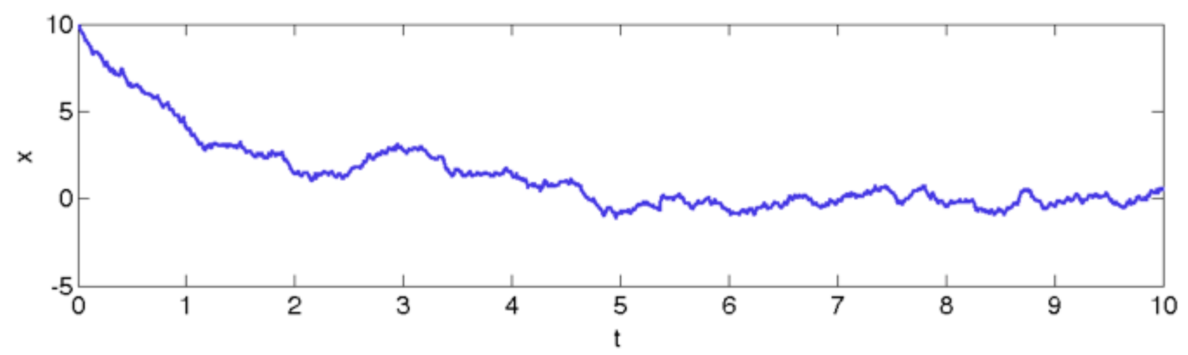
  where N is a time horizon.
- The cost-to-go function is the unique solution to the *Bellman equations*: $J(x, n) = \mathbb{E}[J(x(n+1), n+1) \mid x(n) = x]$.
- If the $=$ is replaced with a $\geq$, then J is instead an upper bound on the cost-to-go. If $J(x, n) \geq \mathbb{E}[J(x, n+1)] - c$, then $J(x, n) + c(N - n)$ is an upper bound.
- If we set cost to probability of failure, then we get back to the martingale condition, and obtain a proof of Theorem 1

Martingales

- Stochastic analogue of Lyapunov function
- Non-negative function V such that $\mathbb{E}[\dot{V}(x)] \leq c$
- Define $\mathbb{E}[\dot{V}(x(t))]$ as $\lim_{\Delta t \downarrow 0} \frac{\mathbb{E}[V(x(t+\Delta t))|x(t)] - V(x(t))}{\Delta t}$
- If $dx(t) = f(x)dt + g(x)dw(t)$, where $dw(t)$ is a standard Wiener process, then

$$\mathbb{E}[\dot{V}(x(t))] = \frac{\partial V}{\partial x} f(x) + \frac{1}{2}\operatorname{Tr}\left(g(x)^T \frac{\partial^2 V}{\partial x^2} g(x)\right)$$

Why $\mathbb{E}[\dot{V}(x)] \leq c$ instead of $\mathbb{E}[\dot{V}(x)] \leq 0$?



- Consider the equation $dx(t) = -xdt + dw(t)$ (above)
- Trajectory decays towards origin, then bounces around
- $\mathbb{E}[\dot{V}(x)] \leq 0$ will be too strict in this case
- Relaxing to $\mathbb{E}[\dot{V}(x)] \leq c$ allows us to handle noise at the origin (improvement over previous work)

Semidefinite and SOS Programming

- Key idea: to find a good Lyapunov function/martingale, phrase the problem as a constrained polynomial optimization problem.
- We will use tools from semidefinite and sum-of-squares (SOS) programming to solve the optimization problem.
- Semidefinite programming: while not all optimization problems are tractable, a special subclass known as *semidefinite programs* can be solved relatively efficiently. Example: maximizing $x + 2y$ subject to the constraint that $\begin{bmatrix} 3+2x & y \\ y & 1 \end{bmatrix} \succeq 0$.
- Sum-of-squares programming: suppose that I want to check whether the polynomial $4x^2y^2 + x^2 + 16xy + 2x + 4y^2 + 4y + 10$ is nonnegative. This might be difficult, but if I told you that it was equal to $(x + 2y + 1)^2 + (2xy + 3)^2$, then it would be clearly nonnegative. Sum-of-squares programming generalizes this idea to transform polynomial optimization problems into semidefinite programs.

Verification of Stochastic Systems

Motivation

- As robots move from factory floors to more demanding environments, they will have to cope with increasingly complex uncertainty.
  - Perceptual uncertainty from stereo vision or cluttered environments.
  - Dynamical uncertainty from rough terrain, wind gusts, or grasping soft fabrics.
- Classical approach: robust control.
  - If my uncertainty stays bounded in a certain region, then I am guaranteed to reach my goal.
  - Problems: heavy-tailed noise, conservative due to worst-case planning
- Goal: develop algorithms to deal with explicitly-modeled uncertainty.

Background

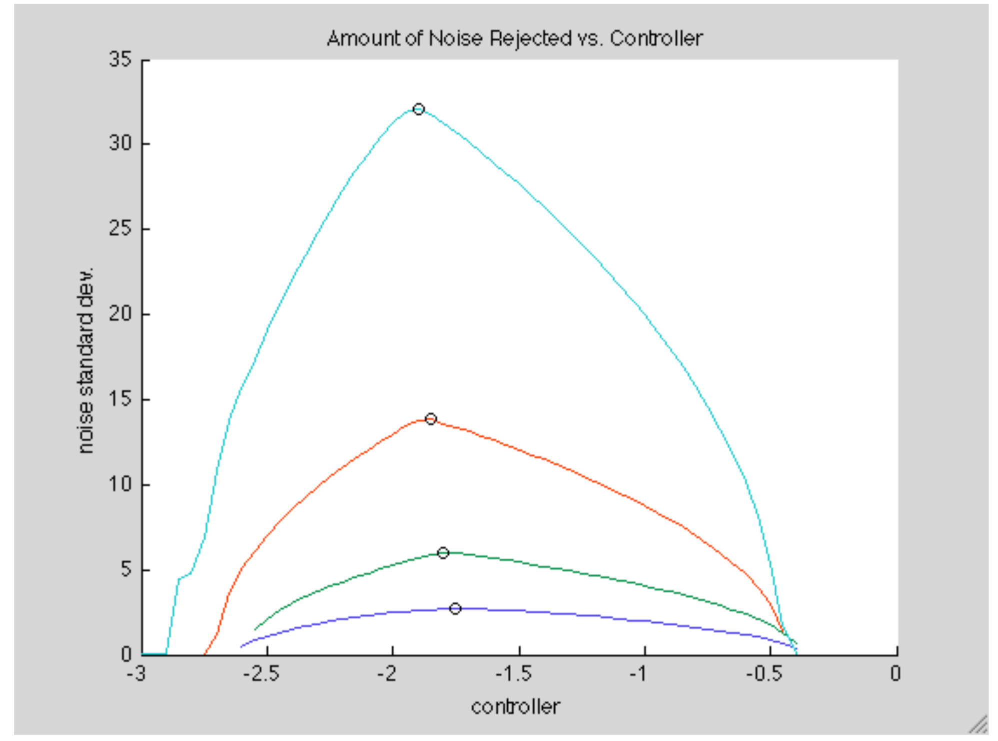
- 1965: Kushner provides Lyapunov-like techniques for obtaining probabilistic guarantees about trajectories of Markov chains; paper includes several handworked examples, but he doesn't have the computational machinery to develop general algorithms.
- 2001: Prajna et al. provide an algorithm for bounding trajectories of switching systems with Gaussian noise. They use sum-of-squares programming on Martingales, but cannot handle noise at the origin and use a basis that leads to conservative results.
- Our contribution: we combine Prajna's algorithm with Kushner's theory to handle noise at the origin. We also work in a basis that provides much tighter bounds at the expense of more difficult computations.

Martingale Theorem

- A non-negative function V of the state x is a *c-martingale* if $\mathbb{E}[\dot{V}(x)] \leq c$.
- Theorem (Kushner 1965): Suppose that V is a c-martingale in the region where $V(x) < \rho$. Then the probability that x leaves the region $\{x \mid V(x) < \rho\}$ before time T is at most $\frac{V(x(0))+cT}{\rho}$.
- Time-varying version also holds as long as V is a continuous function of time.

Use in controller synthesis

- A single martingale V will yield bounds for an entire family of controllers (see figure to right).
- We can use this bound as a proxy for controller quality and optimize our choice of controller against the provided bound.
- Repeating this process is called *DK iteration*.



Martingales for Gaussian Systems

Calculating the Expected Derivative

- Consider a system with Gaussian noise: $dx(t) = f(x)dt + g(x)dw(t)$, where $dw(t)$ is a vector of i.i.d. Wiener processes.
- Then $\mathbb{E}[\dot{V}(x,t)] = \frac{\partial V}{\partial t} + \frac{\partial V}{\partial x} f(x) + \frac{1}{2}\operatorname{Tr}\left(g(x)^T \frac{\partial^2 V}{\partial x^2} g(x)\right)$.
- We will consider functions of the form $V(x) = e^{x^T S x}$.
- In this case, we have

$$\mathbb{E}[\dot{V}(x,t)] = e^{x^T S x}\left(x^T \dot{S} x + 2x^T S f(x) + \frac{1}{2}\operatorname{Tr}\left(g^T S g\right) + \frac{1}{2} x^T S g g^T S x\right)$$

Relaxing to a Polynomial Condition

- We want to check if $p(x)e^{q(x)} \leq c$ for polynomials p and q.
- Re-arrange: $p(x) \leq ce^{-q(x)}$.
- Note that $1 - q(x) \leq e^{-q(x)}$ by convexity.
- Sufficient condition: $p(x) \leq c(1 - q(x))$.
- This is a polynomial condition, and Schur complements can be used to make it bilinear in the decision variables. We then obtain the end result:
- If $\begin{bmatrix} I & g^T S x \\ x^T S g & c(1 - x^T S x) - \operatorname{Tr}(g^T S g) - 4x^T S f(x) - 2x^T \dot{S} x + \lambda(x,t)(x^T S x - \rho) \end{bmatrix} \succeq 0,$ then the probability of a trajectory leaving the region $x^T S x < \rho$ is at most $\frac{e^{x(0)^T S x(0)} + cT}{e^{\rho}}$.

Optimization Techniques for Choosing a Martingale

- Since our bound on the failure probability has an e^{ρ} term in the dominator, we will try to make ρ as large as possible.
- Note that if we fix c and λ, the constraint is linear in S and ρ. Likewise, if we fix S and ρ, the constraint is linear in c and λ.
- **Optimization strategy:** First fix c and λ and optimize S and ρ, then fix S and ρ and optimize c and λ. But, there are a few issues to resolve.

Issues

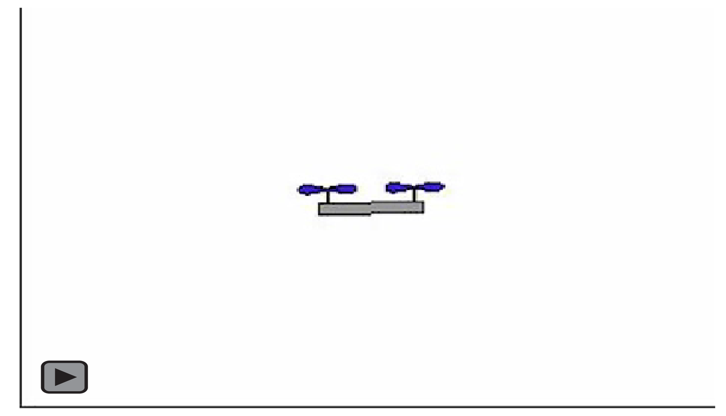
- We need to find an initial feasible point.
- In the step when we fix ρ, we need a different objective function (maximizing ρ does not make since if ρ is fixed).
- Modern numerical optimizers typically give solutions slightly outside the feasible region. Iterative maximization can cause these errors to accumulate and lead to numerical instability.

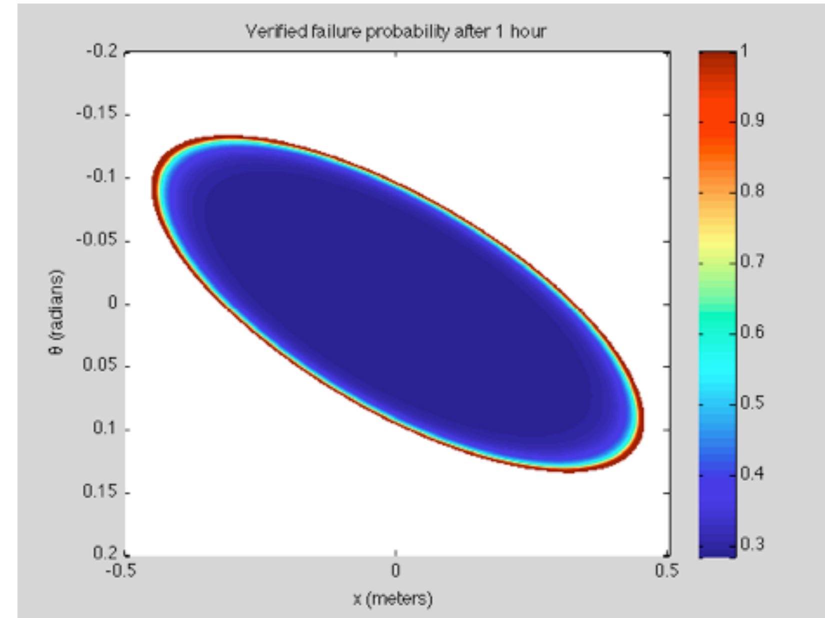
Solutions

- Initialize with an S matrix for the linearized system and a small value of ρ.
- First minimize c, then maximize c, and take the average of the two solutions (this attempts to find a point close to the middle of the feasible region).
- After each maximization step, find a feasible point whose objective value is only slightly less than the value obtained from the maximization. For instance, if the maximization returns $\rho = 2.3$, then find a feasible point with the added constraint $\rho > 2.29$.

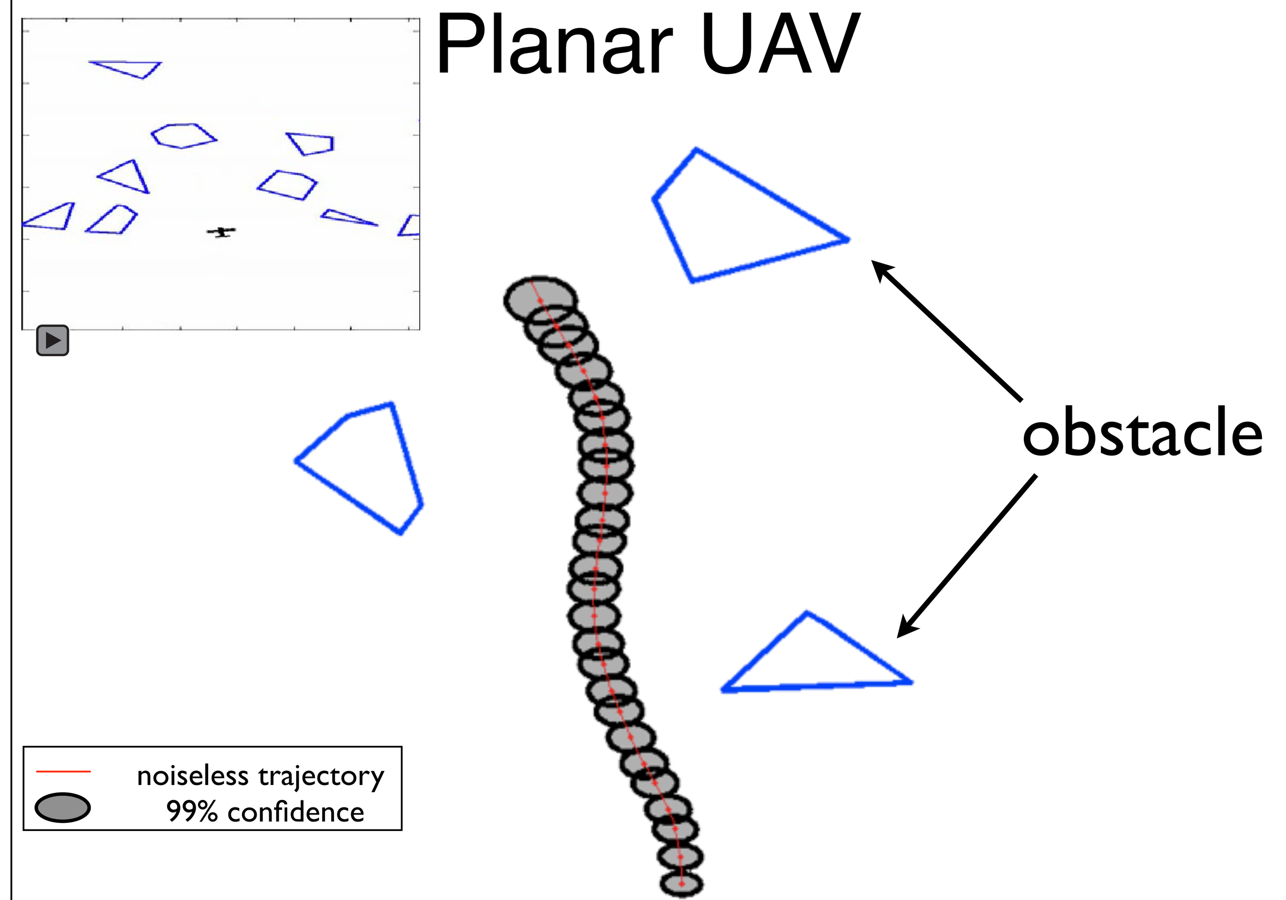
Results

Planar Quadrotor





Planar UAV

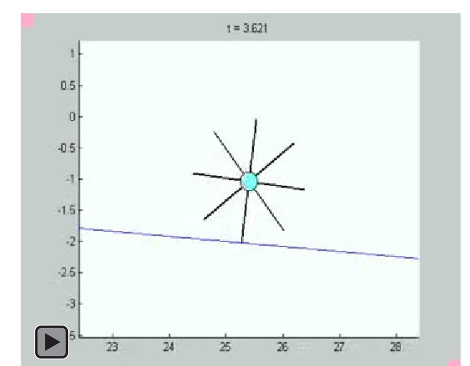


Comparison to Other Methods

Quality of Bound

- We compared our approach to a "worst-case" method and to the true answer for the rimless wheel system (see below right).
- This system has non-Gaussian noise, so we tried both linearizing the noise and adding a state variable to filter the noise through a nonlinear transformation.
- Results:

Method	Verified # of Ground Impacts
worst-case (non-linear noise)	313
worst-case (linear noise)	428
our method (non-linear noise)	50
our method (linear noise)	12647
exact computation	643600



Scalability

- We compared the scalability of our approach to state discretization for verifying stability of a multi-room heating system (dimension of state space grows with number of rooms).
- State discretization only scales to 7 rooms (taking about 6 hours). Our approach solves the 7-room case in under 15 minutes.
- Our approach can handle at least the 10-room case, and scales polynomially with dimension.