120 Years of Lyapunov's Methods

Stephen Boyd

Springer Colloquium, UC Berkeley, 4/22/15

Outline

Conserved quantities

- dynamical system $\dot{x} = f(x)$
- scalar valued function $V : \mathbf{R}^n \to \mathbf{R}$
- V is a **conserved quantity** (or **integral of the motion** or **invariant**) if along every trajectory x, $V(x(t))$ is constant:

$$\dot{V}(x) = \frac{d}{dt} \left. V(x(t)) \right|_{\dot{x}=f(x)} = \nabla V(x)^T f(x) = 0$$

for all x

Conserved quantities

- dynamical system $\dot{x} = f(x)$
- scalar valued function $V : \mathbf{R}^n \to \mathbf{R}$
- V is a **conserved quantity** (or **integral of the motion** or **invariant**) if along every trajectory x, $V(x(t))$ is constant:

$$\dot{V}(x) = \frac{d}{dt}\, V(x(t))|_{\dot{x}=f(x)} = \nabla V(x)^T f(x) = 0$$

  for all x
- classical examples:
  - total energy of a lossless mechanical system
  - total angular momentum about an axis of an isolated system
  - total fluid in a closed system
- trajectories stay in *level sets* of V, $\{z \in \mathbf{R}^n \mid V(z) = a\}$

Dissipated quantities

- V is **dissipated quantity** if $V(x(t))$ is nonincreasing:

$$\dot{V}(x) = \nabla V(x)^T f(x) \leq 0$$

for all x

Dissipated quantities

- V is **dissipated quantity** if $V(x(t))$ is nonincreasing:

  $$\dot{V}(x) = \nabla V(x)^T f(x) \leq 0$$

  for all x
- examples:
  - energy of system with loss
  - total fluid in leaky system
- trajectories stay in *sublevel sets* of V, $\{z \in \mathbf{R}^n \mid V(z) \leq a\}$
- if these are bounded, then trajectories are bounded

Outline

Lyapunov's brilliant idea

- V **doesn't have to come from physics**

Lyapunov's brilliant idea

- V **doesn't have to come from physics**
- let's **search** for V that establishes some property we'd like to know

Lyapunov's brilliant idea

- V **doesn't have to come from physics**
- let's **search** for V that establishes some property we'd like to know
- use V to establish properties of trajectories, even (especially) when we cannot explicitly write down trajectories
- classic example: if we find V with bounded sublevel sets, $\dot{V} \leq 0$, then all trajectories are bounded

The breadth of Lyapunov's idea

can be used for a wide variety of problems, way beyond stability

- performance indices
- decay/growth rate, Lyapunov exponent
- uncertain dynamics, stochastic systems
- time delay systems
- reachability
- input/output analysis (passivity, gain)
- state feedback synthesis
- stochastic control

in each case, need to find a V that satisfies some properties, or optimizes some bound

The big question: How do you find V (and verify its properties)?

for linear dynamics, quadratic costs, there are analytical methods

- Lyapunov functions typically quadratic
- can be found by solving linear equations
- are typically sharp

The big question: How do you find V (and verify its properties)?

for linear dynamics, quadratic costs, there are analytical methods

- Lyapunov functions typically quadratic
- can be found by solving linear equations
- are typically sharp

traditional sources for finding a suitable Lyapunov function V

- physics (say, kinetic plus potential energy)
- exact Lyapunov function for a related linear system
- graphical methods (circle, Popov criterion)

How do you find V?

Lyapunov's approach (1890s, 00s)

- choose form of V (*e.g.*, quadratic), called **Lyapunov function candidate**
- find values of parameters for which required properties hold (typically 'by hand')

How do you find V?

Lyapunov's approach (1890s, 00s)

- choose form of V (*e.g.*, quadratic), called **Lyapunov function candidate**
- find values of parameters for which required properties hold (typically 'by hand')

the modern approach (1990s, 00s)

- choose linearly parametrized Lyapunov function candidate
- find values of parameters for which required properties hold using (numerical) **convex optimization**
- first proposed by Pyatnitskii, Skorodinskii

How do you find V?

Lyapunov's approach (1890s, 00s)

- choose form of V (*e.g.*, quadratic), called **Lyapunov function candidate**
- find values of parameters for which required properties hold (typically 'by hand')

the modern approach (1990s, 00s)

- choose linearly parametrized Lyapunov function candidate
- find values of parameters for which required properties hold using (numerical) **convex optimization**
- first proposed by Pyatnitskii, Skorodinskii

Lyapunov would have understood (and approved)

Finding V via convex optimization

- for quadratic $V = x^T Px$, many properties can be certified by **matrix inequalities** involving P
- for example: bounded sublevel sets $\Longleftrightarrow P > 0$
- these matrix inequalities are **convex** in the parameter P
- so searching over P is a **convex optimization problem**
- hence, readily **solved** (numerically)

More sophisticated methods

S-procedure (1940s) (Lur'e, . . .)

- verify quadratic inequality on set defined by quadratics
- S-procedure is simple but powerful sufficient condition

sum-of-squares (2000s) (Parrilo, Lall, . . .)

- use higher-order polynomial Lyapunov function candidates
- certify inequalities by expressing as sum of squares of polynomials

More sophisticated methods

S-procedure (1940s) (Lur'e, ...)

- verify quadratic inequality on set defined by quadratics
- S-procedure is simple but powerful sufficient condition

sum-of-squares (2000s) (Parrilo, Lall, ...)

- use higher-order polynomial Lyapunov function candidates
- certify inequalities by expressing as sum of squares of polynomials

... **these too reduce to convex optimization problems**

Outline

Convex optimization — Classical form

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, m \\ & Ax = b \end{array}$$

- variable $x \in \mathbf{R}^n$
- $f_0, \ldots, f_m$ are **convex**: for $\theta \in [0, 1]$,

$$f_i(\theta x + (1-\theta)y) \leq \theta f_i(x) + (1-\theta) f_i(y)$$

i.e., f_i have nonnegative (upward) curvature

Convex optimization — Cone form

$$
\begin{array}{ll}
\text{minimize} & c^T x \\
\text{subject to} & x \in K \\
& Ax = b
\end{array}
$$

- variable $x \in \mathbf{R}^n$
- $K \subset \mathbf{R}^n$ is a proper cone
  - K nonnegative orthant $\longrightarrow$ LP
  - K Lorentz cone $\longrightarrow$ SOCP
  - K positive semidefinite matrices $\longrightarrow$ SDP
- the ‘modern’ canonical form

Why

- beautiful, nearly complete theory
  - duality, optimality conditions, ...

Why

- beautiful, nearly complete theory
  - duality, optimality conditions, ...
- effective algorithms, methods (in theory and practice)
  - get **global solution** (and optimality certificate)
  - polynomial complexity

Why

- beautiful, nearly complete theory
  - duality, optimality conditions, ...
- effective algorithms, methods (in theory and practice)
  - get **global solution** (and optimality certificate)
  - polynomial complexity
- conceptual unification of many methods

Why

- beautiful, nearly complete theory
  - duality, optimality conditions, ...
- effective algorithms, methods (in theory and practice)
  - get **global solution** (and optimality certificate)
  - polynomial complexity
- conceptual unification of many methods
- **lots of applications** (many more than previously thought)

Application areas

- machine learning, statistics
- finance
- supply chain, revenue management, advertising
- signal and image processing, vision
- networking
- circuit design
- combinatorial optimization
- quantum mechanics

...and control (especially, searching for Lyapunov functions)

History

- mathematical basis: convex analysis (1900–)
- simplex method for LP (1948)   (Kantorovich, Dantzig, . . .)
- subgradient methods (1960s)   (Shor, . . .)
- interior-point methods (1988–)   (Dikin, Nemirovski, Nesterov, . . .)
- high level languages for convex optimization (2005–)
  (Grant, Boyd, Jalden, . . .)

Modeling languages

- high level language support for convex optimization
  - describe problem in high level language
  - description is automatically transformed to cone problem
  - solved by standard solver, transformed back to original form

Modeling languages

- high level language support for convex optimization
  - describe problem in high level language
  - description is automatically transformed to cone problem
  - solved by standard solver, transformed back to original form

- enables rapid prototyping
- **ideal for teaching** (can do a lot with short scripts)

CVX

parser/solver written in Matlab (M. Grant, 2005)

example: a regularized, constrained approximation problem

$$\begin{array}{ll} \text{minimize} & \|Ax - b\|_2 + \lambda \|x\|_1 \\ \text{subject to} & x \geq -1 \end{array}$$

its CVX specification:

```
cvx_begin
    variable x(n)
    minimize norm(A*x-b)+lambda*norm(x,1)
    subject to x >= -1
cvx_end
```

Outline

Worst-case performance for time-varying system

- $x_{t+1} = A_t x_t$, $A_t \in \mathcal{A} = \{A^{(1)}, \ldots, A^{(K)}\}$,
- quadratic sum performance index: $J = \sum_{t=0}^{\infty} x_t^T Q x_t$, $Q \geq 0$
- given x_0, find $J^{\mathrm{wc}} = \sup_{A_0, A_1, \ldots} J$
- exact answer when $K = 1$: solve Lyapunov equation

$$A^T P A + Q = P$$

  for P; if $P \geq 0$, then $J = x_0^T P x_0$

Lyapunov performance bound

- suppose $V \geq 0$ and satisfies Lyapunov inequalities

$$V(A^{(i)}x) + x^T Q x \leq V(x) \qquad i = 1, \ldots, K$$

- this implies $V(x_{t+1}) + x_t^T Q x_t \leq V(x_t)$, so

$$\sum_{t=0}^{T} x_t^T Q x_t \leq V(x_0) - V(x_{T+1})$$

- so $J^{\mathrm{wc}} \leq V(x_0)$
- optimize upper bound: minimize $V(x_0)$ over candidate V
- when done over all functions (in principle), bound is tight

Quadratic candidate

- now take quadratic candidate: $V(x) = x^T P x$
- reduces to $P \geq 0$,

$$A^{(i)T} P A^{(i)} + Q \leq P, \quad i = 1, \ldots, K$$

- convex constraints on P (LMIs)
- we minimize $x_0^T P x_0$ (a convex problem; an SDP)

CVX source

```
cvx_begin sdp
    variable P(n,n) symmetric
    P >= 0
    for i=1:k
      A{i}'*P*A{i}+Q <= P
    end
    minimize x0'*P*x0
cvx_end
```

Approximate worst-case simulation

- to get sequence $A_0, A_1, \ldots$ that yields large J, choose

$$A_t = \underset{A \in \mathcal{A}}{\operatorname{argmax}} \; V(Ax_t)$$

- greedily maximizes V
- gives lower bound on J^{wc}, so we get **gap** (difference between upper and lower bounds)

Numerical example

- $n = 10$ states, $K = 10$ dynamics matrices
- random data
- bound gives $J^{\text{ub}} = 24.7$
- approximate worst-case simulation gives $J^{\text{lb}} = 16.2$

Numerical example

- $n = 10$ states, $K = 10$ dynamics matrices
- random data
- bound gives $J^{\text{ub}} = 24.7$
- approximate worst-case simulation gives $J^{\text{lb}} = 16.2$
- gap could be improved by, *e.g.*,
  - considering all pairs $A^{(i)}A^{(j)}$
  - using quartic or higher order Lyapunov function

Outline

Stochastic control

- $x_{t+1} = f(x_t, u_t, w_t)$
- w_t IID, independent of x_0
- state feedback policy: $u_t = \mu(x_t)$
- stage cost function $g(x, u)$
- average stage cost

$$J^\mu = \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T} \mathbf{E}\, g(x_t, u_t)$$

- stochastic control problem: find policy μ that minimizes J^μ

Dynamic programming 'solution'

- find (value function) V, α that satisfy Bellman equation

$$V(x) + \alpha = \min_{u \in \mathcal{U}} \left(g(x, u) + \mathbf{E}\, V(f(x, u, w_t)) \right)$$

  (V defined up to constant)

- then optimal policy is

$$\mu^\star(x) = \operatorname*{argmin}_{u \in \mathcal{U}} \left(g(x, u) + \mathbf{E}\, V(f(x, u, w_t)) \right)$$

  with associated average stage cost $J^\star = \alpha$

Dynamic programming 'solution'

- find (value function) V, α that satisfy Bellman equation

$$V(x) + \alpha = \min_{u \in \mathcal{U}} \left(g(x, u) + \mathbf{E}\, V(f(x, u, w_t)) \right)$$

  (V defined up to constant)

- then optimal policy is

$$\mu^\star(x) = \operatorname*{argmin}_{u \in \mathcal{U}} \left(g(x, u) + \mathbf{E}\, V(f(x, u, w_t)) \right)$$

  with associated average stage cost $J^\star = \alpha$

- a solution in principle only, except for a few special cases (*e.g.*, f affine, g convex quadratic)

Approximate dynamic programming

- ADP policy:

$$\mu^{\mathrm{adp}}(x) = \underset{u \in \mathcal{U}}{\operatorname{argmin}} \left(g(x,u) + \mathbf{E}\, V^{\mathrm{adp}}(f(x,u,w_t)) \right)$$

- V^{adp} is **approximate value function**, chosen so that
  - minimization required to evaluate μ^{adp} is tractable
  - average cost J^{adp} attained is near optimal, or at least small

Approximate dynamic programming

- ADP policy:

$$\mu^{\mathrm{adp}}(x) = \underset{u \in \mathcal{U}}{\operatorname{argmin}} \left(g(x,u) + \mathbf{E}\, V^{\mathrm{adp}}(f(x,u,w_t)) \right)$$

- V^{adp} is **approximate value function**, chosen so that
  - minimization required to evaluate μ^{adp} is tractable
  - average cost J^{adp} attained is near optimal, or at least small
- with well chosen V^{adp}, often works well (judged by simulation)

Approximate dynamic programming

- ADP policy:

$$\mu^{\mathrm{adp}}(x) = \underset{u \in \mathcal{U}}{\operatorname{argmin}} \left(g(x,u) + \mathbf{E}\, V^{\mathrm{adp}}(f(x,u,w_t)) \right)$$

- V^{adp} is **approximate value function**, chosen so that
  - minimization required to evaluate μ^{adp} is tractable
  - average cost J^{adp} attained is near optimal, or at least small
- with well chosen V^{adp}, often works well (judged by simulation)
- but how suboptimal is J^{adp}?

Bellman inequality

- suppose V, α satisfy Bellman inequality

$$V(x) + \alpha \leq \min_{u \in \mathcal{U}} \left(g(x,u) + \mathbf{E}\, V(f(x,u,w_t)) \right)$$

- then $\alpha \leq J^\star$, *i.e.*, α is lower bound on optimal control performance
- optimize performance bound: maximize α subject to Bellman inequality

Bellman inequality

- suppose V, α satisfy Bellman inequality

$$V(x) + \alpha \leq \min_{u \in \mathcal{U}} \left(g(x, u) + \mathbf{E}\, V(f(x, u, w_t)) \right)$$

- then $\alpha \leq J^\star$, *i.e.*, α is lower bound on optimal control performance
- optimize performance bound: maximize α subject to Bellman inequality
- solution is natural choice for V^{adp}
- yields a **performance bound**, and a (typically good) **suboptimal policy**

Linear-quadratic finite input stochastic control

- linear dynamics $x_{t+1} = Ax_t + Bu_t + w_t$
- input set $\mathcal{U} = \{u^{(1)}, \ldots, u^{(K)}\}$ is **finite**
- $\mathbf{E}\, w_t = 0$, $\mathbf{E}\, w_t w_t^T = W$
- convex quadratic stage cost $g(x, u) = x^T Qx + u^T Ru, \ Q, \ R \geq 0$

Performance bound via Bellman inequality

- quadratic Lyapunov function candidate $V(x) = x^T P x$
- Bellman inequality is

$$\begin{array}{rcl} x^T P x + \alpha & \leq & x^T Q x + u^T R u + \mathbf{E}(Ax + Bu + w_t)^T P (Ax + Bu + w_t) \\ & = & x^T Q x + u^T R u + (Ax + Bu)^T P (Ax + Bu) + \mathbf{Tr}(PW) \end{array}$$

  for all x, $u \in \mathcal{U}$

- can express as convex constraints (LMIs)

$$\left[\begin{array}{cc} A^T P A + Q - P & A^T P B u \\ u^T B^T P A & u^T (R + B^T P B) u + \mathbf{Tr}(PW) - \alpha \end{array} \right] \geq 0, \qquad u \in \mathcal{U}$$

- maximize α (gives SDP)

CVX source

```
cvx_begin sdp
    variable P(n,n) symmetric
    variable alpha
    P >= 0
    for i=1:k
      [A'*P*A+Q-P      A'*P*B*u{i};
       u{i}'*B'*P*A    u{i}'*(R+B'*P*B)*u{i}+trace(P*W)-alpha] >= 0
    end
    maximize alpha
cvx_end
```

Numerical example

- $n = 10$ states, $m = 3$ inputs, $K = 10$ input values
- random data

Numerical example

- $n = 10$ states, $m = 3$ inputs, $K = 10$ input values
- random data
- LQR cost ($u \in \mathbf{R}^3$): 42.5

Numerical example

- $n = 10$ states, $m = 3$ inputs, $K = 10$ input values
- random data
- LQR cost ($u \in \mathbf{R}^3$): 42.5
- lower bound on optimal cost: 68.1

Numerical example

- $n = 10$ states, $m = 3$ inputs, $K = 10$ input values
- random data
- LQR cost ($u \in \mathbf{R}^3$): 42.5
- lower bound on optimal cost: 68.1
- performance achieved by ADP: 78.2

Outline

Ideas

- Lyapunov's methods apply to far more than just stability theory

Ideas

- Lyapunov's methods apply to far more than just stability theory
- when **Lyapunov's methods** are coupled to **convex optimization**
  - the combination is very powerful and expressive
  - it's also very concrete — you get numerical answers

Ideas

- Lyapunov's methods apply to far more than just stability theory
- when **Lyapunov's methods** are coupled to **convex optimization**
  - the combination is very powerful and expressive
  - it's also very concrete — you get numerical answers
- modern convex optimization tools make it easy to do

Ideas

- Lyapunov's methods apply to far more than just stability theory
- when **Lyapunov's methods** are coupled to **convex optimization**
  - the combination is very powerful and expressive
  - it's also very concrete — you get numerical answers
- modern convex optimization tools make it easy to do
- should be universally taught

References

- *Linear Matrix Inequalities in System and Control Theory* (Boyd, El Ghaoui, Feron, Balakrishnan)
- *Convex Optimization* (Boyd, Vandenberghe)
- CVX (Grant, Boyd)

- all (freely) available on the web; use google to find them
- books contain many additional references