Automated tight Lyapunov analysis for first-order splitting methods

Pontus Giselsson

Collaborators

Manu Upadhyaya

Sebastian Banert

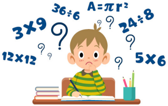
Adrien Taylor

Talk scope

- Methodology for proving algorithm convergence
- Focus on first-order methods for convex optimization that use
  - proximal operator or gradient evaluations
  - scalar multiplications and vector additions

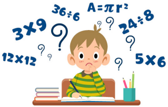
Proving convergence

- Traditional way:



Proving convergence

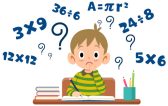
- Traditional way:



- Modern way with computer assisted PEP and IQC:

Proving convergence

- Traditional way:

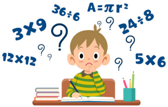


- Modern way with computer assisted PEP and IQC:

- End goal?:

Proving convergence

- Traditional way:

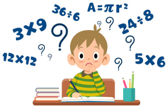


- Modern way with computer assisted PEP and IQC:

- End goal?:

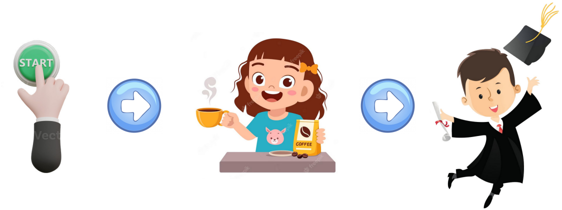
Proving convergence

- Traditional way:



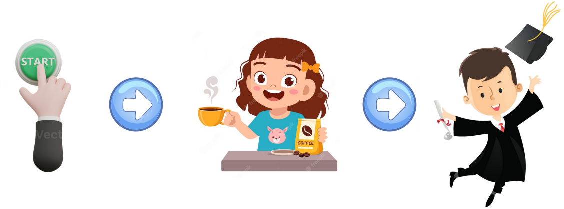
- Modern way with computer assisted PEP and IQC:

- End goal?:



Towards end goal

- End goal:



- Tried to contribute to this with automatic Lyapunov analysis

Example: What we achieved while drinking coffee

- Chambolle–Pock ("with $L = \mathrm{Id}$"): $\underset{x \in \mathcal{H}}{\mathrm{minimize}}(f_1(x) + f_2(x))$

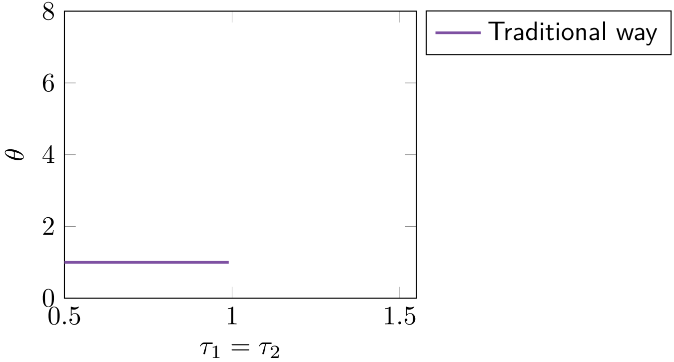
$$x_{k+1} = \mathrm{prox}_{\tau_1 f_1}(x_k - \tau y_k)$$
$$y_{k+1} = \mathrm{prox}_{\tau_2 f_2^*}(y_k + \tau_2 (x_{k+1} + \theta(x_{k+1} - x_k)))$$

Example: What we achieved while drinking coffee

- Chambolle–Pock ("with $L = \mathrm{Id}$"): $\underset{x\in\mathcal{H}}{\text{minimize}}(f_1(x) + f_2(x))$

$$x_{k+1} = \mathrm{prox}_{\tau_1 f_1}(x_k - \tau y_k)$$
$$y_{k+1} = \mathrm{prox}_{\tau_2 f_2^*}(y_k + \tau_2 (x_{k+1} + \theta(x_{k+1} - x_k)))$$

- Convergent parameter choices (primal-dual gap, f_1 and f_2 pcc)



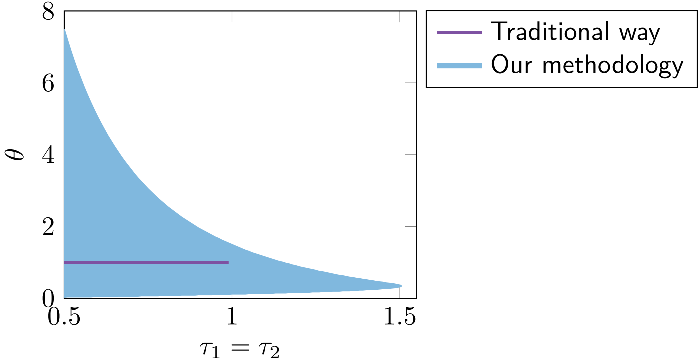
(Caveat: verified on a 0.01×0.01 grid of region)

Example: What we achieved while drinking coffee

- Chambolle–Pock ("with $L = \mathrm{Id}$"): $\underset{x\in\mathcal{H}}{\text{minimize}}(f_1(x) + f_2(x))$

$$x_{k+1} = \text{prox}_{\tau_1 f_1}(x_k - \tau y_k)$$
$$y_{k+1} = \text{prox}_{\tau_2 f_2^*}(y_k + \tau_2 (x_{k+1} + \theta(x_{k+1} - x_k)))$$

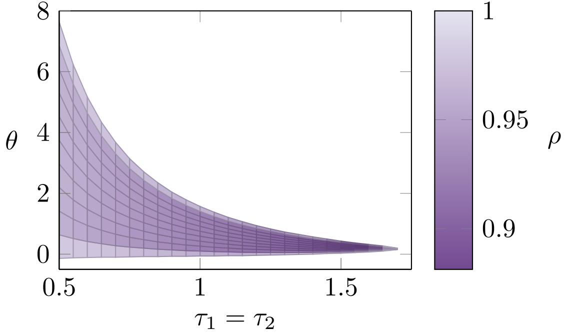
- Convergent parameter choices (primal-dual gap, f_1 and f_2 pcc)



(Caveat: verified on a 0.01×0.01 grid of region)

Chambolle–Pock linear convergence

- Tight contraction rate–both 0.05-strongly convex and 50-smooth:



- Improved rate with larger $\tau_1 = \tau_2$

Chambolle–Pock linear convergence

- Optimal convergence rate for different parameter restrictions[1]

Parameter restriction	$\tau_1 = \tau_2$	θ	ρ
All convergent	1.6	0.22	0.8812
Cvx+cxv convergent	1.5	0.35	0.8891
Traditional	0.99	1	0.9266
DR	1	1	0.9234

- Better rates outside traditional region

[1] for points evaluated on our 0.01×0.01 grid

Outline

- **Setting and main result preview**
- Algorithm representation
- Lyapunov inequality definition
- The necessary and sufficient condition
- Algorithm examples

Setting

- Let $\mathcal{F}_{\sigma_i,\beta_i}$ be class of σ_i-strongly convex and β_i-smooth functions
- Convex optimization problems

$$\underset{y\in\mathcal{H}}{\text{minimize}}\sum_{i=1}^{m} f_i(y)$$

where each $f_i \in \mathcal{F}_{\sigma_i,\beta_i}$ with $0 \leq \sigma_i < \beta_i \leq \infty$

- Associated inclusion problem

$$\text{find } y \in \mathcal{H} \text{ such that } 0 \in \sum_{i=1}^{m} \partial f_i(y)$$

where ∂f_i are subdifferential operators

- Problem class $\boldsymbol{\mathcal{F}}_{\boldsymbol{\sigma},\boldsymbol{\beta}}$: $f_i \in \mathcal{F}_{\sigma_i,\beta_i}$ and inclusion solvable

Main result statement

Given a first-order method for an inclusion problem class, we provide

- a necessary and sufficient condition for the existence of a *quadratic Lyapunov inequality*
- a quadratic Lyapunov inequality if one exists

The necessary and sufficient condition

- Condition is feasibility of (small) semi-definite program
- Derived with inspiration from
  - performance estimation (PEP) (Drori and Teboulle, Taylor et al.)
  - integral quadratic constraints (IQC) (Lessard et al.)
  - tight automated analysis framework (Taylor/Van Scoy/Lessard)
  - Lyapunov analysis (Taylor/Bach)
- Based on specific algorithm representation for wide applicability

Outline

- Setting and main result preview
- **Algorithm representation**
- Lyapunov inequality definition
- The necessary and sufficient condition
- Algorithm examples

Algorithm representation

- Algorithm representation on state space form[1]:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= (A \otimes \mathrm{Id})\boldsymbol{x}_k + (B \otimes \mathrm{Id})\boldsymbol{u}_k \\ \boldsymbol{y}_k &= (C \otimes \mathrm{Id})\boldsymbol{x}_k + (D \otimes \mathrm{Id})\boldsymbol{u}_k \\ \boldsymbol{u}_k &\in \boldsymbol{\partial f}(\boldsymbol{y}_k) \\ \boldsymbol{F}_k &= \boldsymbol{f}(\boldsymbol{y}_k) \end{aligned}$$

- Product space notation for function and subdifferentials

$$\begin{aligned} \boldsymbol{f}(\boldsymbol{y}) &= \Big(f_1\Big(y^{(1)}\Big), \ldots, f_m\Big(y^{(m)}\Big)\Big), \\ \boldsymbol{\partial f}(\boldsymbol{y}) &= \prod_{i=1}^{m} \partial f_i\Big(y^{(i)}\Big) \end{aligned}$$

where

$$\boldsymbol{y} = \Big(y^{(1)}, \ldots, y^{(m)}\Big), \quad \boldsymbol{u} = \Big(u^{(1)}, \ldots, u^{(m)}\Big), \quad \boldsymbol{x} = \Big(x^{(1)}, \ldots, x^{(n)}\Big)$$

meaning $u_k^{(i)} \in \partial f_i(y_k^{(i)})$ for all $i \in [\![1, m]\!]$

[1] Model used in control literature, Lessard et al. 2016, and similar to model in Morin/Banert/Giselsson

Chambolle–Pock

- Algorithm:

$$
\begin{aligned}
x_{k+1} &= \mathrm{prox}_{\tau_1 f_1}(x_k - \tau y_k), \\
y_{k+1} &= \mathrm{prox}_{\tau_2 f_2^*}(y_k + \tau_2 \left(x_{k+1} + \theta(x_{k+1} - x_k)\right))
\end{aligned}
$$

- Algorithm in our state-space representation:

$$
\begin{aligned}
\boldsymbol{x}_{k+1} &= \left(\begin{bmatrix} 1 & -\tau_1 \\ 0 & 0 \end{bmatrix}_{\mathrm{Id}} \right) \boldsymbol{x}_k + \left(\begin{bmatrix} -\tau_1 & 0 \\ 0 & 1 \end{bmatrix}_{\mathrm{Id}} \right) \boldsymbol{u}_k, \\
\boldsymbol{y}_k &= \left(\begin{bmatrix} 1 & -\tau_1 \\ 1 & \frac{1}{\tau_2} - \tau_1(1+\theta) \end{bmatrix}_{\mathrm{Id}} \right) \boldsymbol{x}_k + \left(\begin{bmatrix} -\tau_1 & 0 \\ -\tau_1(1+\theta) & -\frac{1}{\tau_2} \end{bmatrix}_{\mathrm{Id}} \right) \boldsymbol{u}_k, \\
\boldsymbol{u}_k &\in \boldsymbol{\partial f}(\boldsymbol{y}_k),
\end{aligned}
$$

- Algorithm parameters appear in (A, B, C, D)

Proximal gradient method with heavy-ball momentum

- Algorithm:

$$x_{k+1} = \mathrm{prox}_{\gamma f_2}(x_k - \gamma \nabla f_1(x_k) + \delta_1(x_k - x_{k-1})) + \delta_2(x_k - x_{k-1})$$

- Algorithm in our state-space representation:

$$\begin{aligned}
\boldsymbol{x}_{k+1} &= \left(\begin{bmatrix} 1+\delta_1+\delta_2 & -\delta_1-\delta_2 \\ 1 & 0 \end{bmatrix} \otimes \mathrm{Id}\right) \boldsymbol{x}_k + \left(\begin{bmatrix} -\gamma & -\gamma \\ 0 & 0 \end{bmatrix} \otimes \mathrm{Id}\right) \boldsymbol{u}_k \\
\boldsymbol{y}_k &= \left(\begin{bmatrix} 1 & 0 \\ 1+\delta_1 & -\delta_1 \end{bmatrix} \otimes \mathrm{Id}\right) \boldsymbol{x}_k + \left(\begin{bmatrix} 0 & 0 \\ -\gamma & -\gamma \end{bmatrix} \otimes \mathrm{Id}\right) \boldsymbol{u}_k, \\
\boldsymbol{u}_k &\in \boldsymbol{\partial f}(\boldsymbol{y}_k),
\end{aligned}$$

- Algorithm parameters appear in (A, B, C, D)
- Same structure as previous algorithm, just new (A, B, C, D)

Algorithm fixed points

- Algorithm fixed points $\boldsymbol{\xi}_\star = (\boldsymbol{x}_\star, \boldsymbol{u}_\star, \boldsymbol{y}_\star, \boldsymbol{F}_\star)$ satisfy

$$\begin{aligned}
\boldsymbol{x}_\star &= (A \otimes \mathrm{Id})\boldsymbol{x}_\star + (B \otimes \mathrm{Id})\boldsymbol{u}_\star \\
\boldsymbol{y}_\star &= (C \otimes \mathrm{Id})\boldsymbol{x}_\star + (D \otimes \mathrm{Id})\boldsymbol{u}_\star \\
\boldsymbol{u}_\star &\in \boldsymbol{\partial f}(\boldsymbol{y}_\star) \\
\boldsymbol{F}_\star &= \boldsymbol{f}(\boldsymbol{y}_\star)
\end{aligned}$$

- Algorithm objective: find fixed point $\boldsymbol{\xi}_\star$, extract solution from $\boldsymbol{\xi}_\star$

Fixed-point encoding property

- We are only interested in algorithms such that

  finding a fixed point $\quad\Longleftrightarrow\quad$ solving inclusion problem

- More specifically:
  - from each solution, it should be possible to construct fixed point
  - from each fixed point, it should be possible to extract solution
- Such algorithms have the *fixed-point encoding property* (FPEP)

Restrictions on (A, B, C, D)

- Let

$$N = \begin{bmatrix} I \\ -\mathbf{1}^\top \end{bmatrix} \in \mathbb{R}^{m\times(m-1)}$$

- Result:

The algorithm has the fixed-point encoding property

$$\Longleftrightarrow$$

The matrices (A, B, C, D) *satisfy*

$$\operatorname{ran} \begin{bmatrix} BN & 0 \\ DN & -\mathbf{1} \end{bmatrix} \subseteq \operatorname{ran} \begin{bmatrix} I - A \\ -C \end{bmatrix}$$

$$\operatorname{null} \begin{bmatrix} I - A & -B \end{bmatrix} \subseteq \operatorname{null} \begin{bmatrix} N^\top C & N^\top D \\ 0 & \mathbf{1}^\top \end{bmatrix},$$

(block row/column containing $N^\top/N$ *removed when* $m = 1$*)*

- (A, B, C, D) of algorithms that “work” satisfy FPEP conditions

Examples without FPEP

- $(A, B, C, D) = (0, 0, 0, 0)$ does not satisfy FPEP
- Backward–backward splitting is given by

$$x_{k+1} = \mathrm{prox}_{\gamma f_2}(\mathrm{prox}_{\gamma f_1}(x_k))$$

  does not solve inclusion problem
- Backward–backward fits in framework with matrices

$$A = 1, \quad B = \begin{bmatrix} -\gamma & -\gamma \end{bmatrix}, \quad C = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad D = \begin{bmatrix} -\gamma & 0 \\ -\gamma & -\gamma \end{bmatrix}$$

  that do not satisfy the FPEP conditions

Extract solution from fixed point

- Fixed points of algorithms with FPEP satisfy for some $y_\star$:

$$\sum_{i=1}^{m} u_\star^{(i)} = 0 \qquad \text{and} \qquad y_\star^{(1)} = \ldots = y_\star^{(m)} = y_\star$$

- Then $y_\star$ solves the inclusion problem since

$$0 = \sum_{i=1}^{m} u_\star^{(i)} \in \sum_{i=1}^{m} \partial f_i(y_\star^{(i)}) = \sum_{i=1}^{m} \partial f_i(y_\star)$$

Causal implementation

- Assume D lower triangular with nonpositive diagonal and let

$$I_{\text{differentiable}} = \{i \in [\![1, m]\!] : \beta_i < +\infty\}$$
$$I_D = \{i \in [\![1, m]\!] : [D]_{i,i} \neq 0]\}$$

  satisfy $I_{\text{differentiable}} \cup I_D = [\![1, m]\!]$
- Then the algorithm can be implemented using only
  - proximal or gradient evaluations of each f_i
  - scalar multiplications and vector additions

Explicit causal implementation

- The algorithm is:

for $k = 0, 1, \ldots$

for $i = 1, \ldots, m$

$$v_k^{(i)} = \sum_{j=1}^{n} [C]_{i,j} x_k^{(j)} + \sum_{j=1}^{i-1} [D]_{i,j} u_k^{(j)},$$

$$y_k^{(i)} = \begin{cases} \operatorname{prox}_{-[D]_{i,i} f_i}\left(v_k^{(i)}\right) & \text{if } i \in I_D, \\ v_k^{(i)} & \text{if } i \notin I_D, \end{cases}$$

$$u_k^{(i)} = \begin{cases} (-[D]_{i,i})^{-1}\left(v_k^{(i)} - y_k^{(i)}\right) & \text{if } i \in I_D, \\ \nabla f_i\left(y_k^{(i)}\right) & \text{if } i \notin I_D, \end{cases}$$

$$F_k^{(i)} = f_i\left(y_k^{(i)}\right),$$

$$\boldsymbol{x}_{k+1} = \left(x_{k+1}^{(1)}, \ldots, x_{k+1}^{(n)}\right) = (A \otimes \mathrm{Id})\boldsymbol{x}_k + (B \otimes \mathrm{Id})\boldsymbol{u}_k,$$

- Many fixed-parameter first-order methods on this form!

Outline

- Setting and main result preview
- Algorithm representation
- **Lyapunov inequality definition**
- The necessary and sufficient condition
- Algorithm examples

Lyapunov analysis

- Let $\boldsymbol{\xi}_k = (\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{y}_k, \boldsymbol{F}_k)$ and $\boldsymbol{\xi}_\star = (\boldsymbol{x}_\star, \boldsymbol{u}_\star, \boldsymbol{y}_\star, \boldsymbol{F}_\star)$
- Many first-order methods analyzed using *Lyapunov inequalities*

$$V(\boldsymbol{\xi}_{k+1}, \boldsymbol{\xi}_\star) \leq \rho V(\boldsymbol{\xi}_k, \boldsymbol{\xi}_\star) - R(\boldsymbol{\xi}_k, \boldsymbol{\xi}_\star)$$

  where $\rho \in [0, 1]$ and
  - $V : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$ is a *Lyapunov function*
  - $R : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$ is a *residual function*

  and $\mathcal{S} = \mathcal{H}^n \times \mathcal{H}^m \times \mathcal{H}^m \times \mathbb{R}^m$

Lyapunov and residual function ansatz

- We consider quadratic ansatzes of the functions V and R given by

$$V(\boldsymbol{\xi}, \boldsymbol{\xi}_\star) = \mathcal{Q}(Q, (\boldsymbol{x} - \boldsymbol{x}_\star, \boldsymbol{u}, \boldsymbol{u}_\star)) + q^\top (\boldsymbol{F} - \boldsymbol{F}_\star),$$
$$R(\boldsymbol{\xi}, \boldsymbol{\xi}_\star) = \mathcal{Q}(S, (\boldsymbol{x} - \boldsymbol{x}_\star, \boldsymbol{u}, \boldsymbol{u}_\star)) + s^\top (\boldsymbol{F} - \boldsymbol{F}_\star)$$

  where $Q, S \in \mathbb{S}^{n+2m}$, $q, s \in \mathbb{R}^m$ parameterize the functions and

$$\mathcal{Q}(Q, (\boldsymbol{x} - \boldsymbol{x}_\star, \boldsymbol{u}, \boldsymbol{u}_\star)) = \langle (\boldsymbol{x} - \boldsymbol{x}_\star, \boldsymbol{u}, \boldsymbol{u}_\star), Q(\boldsymbol{x} - \boldsymbol{x}_\star, \boldsymbol{u}, \boldsymbol{u}_\star) \rangle$$

- These quadratic ansatzes are quite general (some examples later)

Lyapunov analysis conclusions

- Purpose of Lyapunov analysis is to draw convergence conclusion
- Will not know (Q, q, S, s) in advance $\Rightarrow$ lower bound V and R
- Let $P, T \in \mathbb{S}^{n+2m}$, $p, t \in \mathbb{R}^m$ and

$$\underline{V}(\boldsymbol{\xi}, \boldsymbol{\xi}_\star) = \mathcal{Q}(P, (\boldsymbol{x} - \boldsymbol{x}_\star, \boldsymbol{u}, \boldsymbol{u}_\star)) + p^\top(\boldsymbol{F} - \boldsymbol{F}_\star)$$
$$\underline{R}(\boldsymbol{\xi}, \boldsymbol{\xi}_\star) = \mathcal{Q}(T, (\boldsymbol{x} - \boldsymbol{x}_\star, \boldsymbol{u}, \boldsymbol{u}_\star)) + t^\top(\boldsymbol{F} - \boldsymbol{F}_\star)$$

- Control conclusion by enforcing nonnegative lower bounds

$$V(\boldsymbol{\xi}, \boldsymbol{\xi}_\star) \geq \underline{V}(\boldsymbol{\xi}, \boldsymbol{\xi}_\star) \geq 0$$
$$R(\boldsymbol{\xi}, \boldsymbol{\xi}_\star) \geq \underline{R}(\boldsymbol{\xi}, \boldsymbol{\xi}_\star) \geq 0$$

(P, p, T, t, ρ)-quadratic Lyapunov inequality

(P, p, T, t, ρ)-*Lyapunov inequality* for algorithm over $\boldsymbol{\mathcal{F}}_{\boldsymbol{\sigma},\boldsymbol{\beta}}$:

C1. $V(\boldsymbol{\xi}_+, \boldsymbol{\xi}_\star) \leq \rho V(\boldsymbol{\xi}, \boldsymbol{\xi}_\star) - R(\boldsymbol{\xi}, \boldsymbol{\xi}_\star)$

C2. $V(\boldsymbol{\xi}, \boldsymbol{\xi}_\star) \geq \mathcal{Q}(P, (\boldsymbol{x} - \boldsymbol{x}_\star, \boldsymbol{u}, \boldsymbol{u}_\star)) + p^\top(\boldsymbol{F} - \boldsymbol{F}_\star) \geq 0$

C3. $R(\boldsymbol{\xi}, \boldsymbol{\xi}_\star) \geq \mathcal{Q}(T, (\boldsymbol{x} - \boldsymbol{x}_\star, \boldsymbol{u}, \boldsymbol{u}_\star)) + t^\top(\boldsymbol{F} - \boldsymbol{F}_\star) \geq 0$

Convergence conclusions

- For $\rho \in [0, 1[$:

$$0 \le \underline{V}(\boldsymbol{\xi}_k, \boldsymbol{\xi}_\star) \le V(\boldsymbol{\xi}_k, \boldsymbol{\xi}_\star) \le \rho^k V(\boldsymbol{\xi}_0, \boldsymbol{\xi}_\star) \to 0$$

  i.e., lower bound converges ρ-linearly to 0

- For $\rho = 1$, a telescoping summation gives

$$0 \le \sum_{k=0}^{\infty} \underline{R}(\boldsymbol{\xi}_k, \boldsymbol{\xi}_\star) \le \sum_{k=0}^{\infty} R(\boldsymbol{\xi}_k, \boldsymbol{\xi}_\star) \le V(\boldsymbol{\xi}_0, \boldsymbol{\xi}_\star)$$

- The choice of $P, T \in \mathbb{S}^{n+2m}$, $p, t \in \mathbb{R}^m$ decides conclusion

Some choices of (P, p, T, t)

- Suppose $\rho \in [0, 1[$ and let e_i be ith basis vector and

$$(P, p, T, t) = \left(\begin{bmatrix} C & D & -D \end{bmatrix}^\top e_i e_i^\top \begin{bmatrix} C & D & -D \end{bmatrix}, 0, 0, 0 \right)$$

  then $\underline{V}(\boldsymbol{\xi}_k, \boldsymbol{\xi}_\star) = \left\| y_k^{(i)} - y_\star \right\|^2 \geq 0 \Rightarrow \rho$-linear convergence

- Suppose $\rho = 1$ and $m = 1$ and let

$$(P, p, T, t) = (0, 0, 0, 1)$$

  then $\underline{R}(\boldsymbol{\xi}_k, \boldsymbol{\xi}_\star) = f_1(y_k^{(1)}) - f_1(y_\star) \geq 0$ which gives
  - function suboptimality convergence
  - ergodic $\mathcal{O}(1/k)$ function suboptimality convergence

(P, p, T, t) for duality gap convergence

- Suppose $\rho = 1$ and $m > 1$ and let

$$(P, p, T, t) = \left(0, 0, \begin{bmatrix} C & D & -D \\ 0 & 0 & I \end{bmatrix}^\top \begin{bmatrix} 0 & -\frac{1}{2}I \\ -\frac{1}{2}I & 0 \end{bmatrix} \begin{bmatrix} C & D & -D \\ 0 & 0 & I \end{bmatrix}, \mathbf{1}\right)$$

  then

$$\begin{aligned} \underline{R}(\boldsymbol{\xi}_k, \boldsymbol{\xi}_\star) &= \sum_{i=1}^{m} \left(f_i\left(y_k^{(i)}\right) - f_i\left(y_\star^{(i)}\right) - \left\langle u_\star^{(i)}, y_k^{(i)} - y_\star^{(i)} \right\rangle \right) \\ &= \mathcal{L}(\boldsymbol{y}, \boldsymbol{u}_\star) - \mathcal{L}(\boldsymbol{y}_\star, \boldsymbol{u}) \geq 0 \end{aligned}$$

  where $\mathcal{L} : \mathcal{H}^m \times \mathcal{H}^m \to \mathbb{R}$ is a *Lagrangian function* giving
  - duality gap convergence
  - ergodic $\mathcal{O}(1/k)$ duality gap convergence
- Generalization to function value suboptimality to $m > 1$

(P, p, T, t, ρ)-**quadratic Lyapunov inequality**

- (P, p, T, t, ρ)-*Lyapunov inequality* for algorithm over $\boldsymbol{\mathcal{F}}_{\boldsymbol{\sigma},\boldsymbol{\beta}}$:
  C1. $V(\boldsymbol{\xi}_+, \boldsymbol{\xi}_\star) \leq \rho V(\boldsymbol{\xi}, \boldsymbol{\xi}_\star) - R(\boldsymbol{\xi}, \boldsymbol{\xi}_\star)$
  C2. $V(\boldsymbol{\xi}, \boldsymbol{\xi}_\star) \geq \mathcal{Q}(P, (\boldsymbol{x} - \boldsymbol{x}_\star, \boldsymbol{u}, \boldsymbol{u}_\star)) + p^\top(\boldsymbol{F} - \boldsymbol{F}_\star) \geq 0$
  C3. $R(\boldsymbol{\xi}, \boldsymbol{\xi}_\star) \geq \mathcal{Q}(T, (\boldsymbol{x} - \boldsymbol{x}_\star, \boldsymbol{u}, \boldsymbol{u}_\star)) + t^\top(\boldsymbol{F} - \boldsymbol{F}_\star) \geq 0$
- Conditions should hold for points reachable by algorithm:
  - each $\boldsymbol{\xi} \in \mathcal{S}$ that is *algorithm-consistent* for $\boldsymbol{f}$
  - each *successor* $\boldsymbol{\xi}_+ \in \mathcal{S}$ of $\boldsymbol{\xi}$
  - each fixed point $\boldsymbol{\xi}_\star \in \mathcal{S}$
  - each $\boldsymbol{f} = (f_1, \ldots, f_m) \in \boldsymbol{\mathcal{F}}_{\boldsymbol{\sigma},\boldsymbol{\beta}}$

  which adds complication compared to if $\boldsymbol{\xi}, \boldsymbol{\xi}_+, \boldsymbol{\xi}_\star \in \mathcal{S}^3$

Traditional way to find Lyapunov inequality

- Use inequalities for function class that algorithm solves
- Combine with algorithm updates
- Manipulate to arrive at Lyapunov inequality

Outline

- Setting and main result preview
- Algorithm representation
- Lyapunov inequality definition
- **The necessary and sufficient condition**
- Algorithm examples

Main result

Given:

- a first-order method on state-space representation form
- convergence deciding data (P, p, T, t) and ρ

We provide:

- a necessary and sufficient condition for the existence of a (P, p, T, t, ρ)-*quadratic Lyapunov inequality*
- a quadratic Lyapunov inequality (Q, q, S, s) if one exists

Necessary and sufficient condition

There exists a Lyapunov inequality satisfying C1-C3

$$\Longleftrightarrow^{(1)}$$

A particular SDP involving (Q, q, S, s) is feasible

$^{(1)}$ Assuming dimension independence and Slater condition

Necessary and sufficient condition

There exists a Lyapunov inequality satisfying C1-C3

$$\Longleftrightarrow^{(1)}$$

A particular SDP involving (Q, q, S, s) is feasible

$$\text{C1}\begin{cases}\lambda^{\text{C1}}_{(l,i,j)} \geq 0 \text{ for each } l \in [\![1,m]\!] \text{ and distinct } i,j \in \{\varnothing,+,\star\},\\ \Sigma_\varnothing^\top(\rho Q - S)\Sigma_\varnothing - \Sigma_+^\top Q \Sigma_+ + \sum_{l=1}^{m} \sum_{\substack{i,j\in\{\varnothing,+,\star\}\\ i\neq j}} \lambda^{\text{C1}}_{(l,i,j)} \boldsymbol{M}_{(l,i,j)} \succeq 0,\\ \begin{bmatrix}\rho q - s\\ -q\end{bmatrix} + \sum_{l=1}^{m} \sum_{\substack{i,j\in\{\varnothing,+,\star\}\\ i\neq j}} \lambda^{\text{C1}}_{(l,i,j)} \boldsymbol{a}_{(l,i,j)} = 0,\end{cases}$$

$$\text{C2}\begin{cases}\lambda^{\text{C2}}_{(l,i,j)} \geq 0 \text{ for each } l \in [\![1,m]\!] \text{ and distinct } i,j \in \{\varnothing,\star\},\\ \Sigma_\varnothing^\top(Q - P)\Sigma_\varnothing + \sum_{l=1}^{m} \sum_{\substack{i,j\in\{\varnothing,\star\}\\ i\neq j}} \lambda^{\text{C2}}_{(l,i,j)} \boldsymbol{M}_{(l,i,j)} \succeq 0,\\ \begin{bmatrix}q - p\\ 0\end{bmatrix} + \sum_{l=1}^{m} \sum_{\substack{i,j\in\{\varnothing,\star\}\\ i\neq j}} \lambda^{\text{C2}}_{(l,i,j)} \boldsymbol{a}_{(l,i,j)} = 0,\end{cases}$$

$$\text{C3}\begin{cases}\lambda^{\text{C3}}_{(l,i,j)} \geq 0 \text{ for each } l \in [\![1,m]\!] \text{ and distinct } i,j \in \{\varnothing,\star\},\\ \Sigma_\varnothing^\top(S - T)\Sigma_\varnothing + \sum_{l=1}^{m} \sum_{\substack{i,j\in\{\varnothing,\star\}\\ i\neq j}} \lambda^{\text{C3}}_{(l,i,j)} \boldsymbol{M}_{(l,i,j)} \succeq 0,\\ \begin{bmatrix}s - t\\ 0\end{bmatrix} + \sum_{l=1}^{m} \sum_{\substack{i,j\in\{\varnothing,\star\}\\ i\neq j}} \lambda^{\text{C3}}_{(l,i,j)} \boldsymbol{a}_{(l,i,j)} = 0,\end{cases}$$

$^{(1)}$ Assuming dimension independence and Slater condition

How to arrive at condition?

- C1-C3 equivalent to that optimal value of

$$
\begin{array}{ll}
\text{maximize} & \Phi(\boldsymbol{\xi}, \boldsymbol{\xi}_+, \boldsymbol{\xi}_\star) \\
\text{subject to} & \boldsymbol{x}_+ = (A \otimes \mathrm{Id})\boldsymbol{x} + (B \otimes \mathrm{Id})\boldsymbol{u}, \\
& \boldsymbol{y} = (C \otimes \mathrm{Id})\boldsymbol{x} + (D \otimes \mathrm{Id})\boldsymbol{u}, \\
& \boldsymbol{u} \in \boldsymbol{\partial f}(\boldsymbol{y}), \\
& \boldsymbol{F} = \boldsymbol{f}(\boldsymbol{y}), \\
& \boldsymbol{y}_+ = (C \otimes \mathrm{Id})\boldsymbol{x}_+ + (D \otimes \mathrm{Id})\boldsymbol{u}_+, \\
& \boldsymbol{u}_+ \in \boldsymbol{\partial f}(\boldsymbol{y}_+), \\
& \boldsymbol{F}_+ = \boldsymbol{f}(\boldsymbol{y}_+), \\
& \boldsymbol{x}_\star = (A \otimes \mathrm{Id})\boldsymbol{x}_\star + (B \otimes \mathrm{Id})\boldsymbol{u}_\star, \\
& \boldsymbol{y}_\star = (C \otimes \mathrm{Id})\boldsymbol{x}_\star + (D \otimes \mathrm{Id})\boldsymbol{u}_\star, \\
& \boldsymbol{u}_\star \in \boldsymbol{\partial f}(\boldsymbol{y}_\star), \\
& \boldsymbol{F}_\star = \boldsymbol{f}(\boldsymbol{y}_\star), \\
& \boldsymbol{f} \in \boldsymbol{\mathcal{F}}_{\boldsymbol{\sigma},\boldsymbol{\beta}},
\end{array}
\tag{PEP}
$$

  is non-positive with different quadratic Φ for C1-C3
- Solved using PEP ideas

Outline

- Setting and main result preview
- Algorithm representation
- Lyapunov inequality definition
- The necessary and sufficient condition
- **Algorithm examples**

Using the methodology

We apply our methodology in two different ways:

B1. Find the smallest possible $\rho \in [0, 1[$ via bisection search

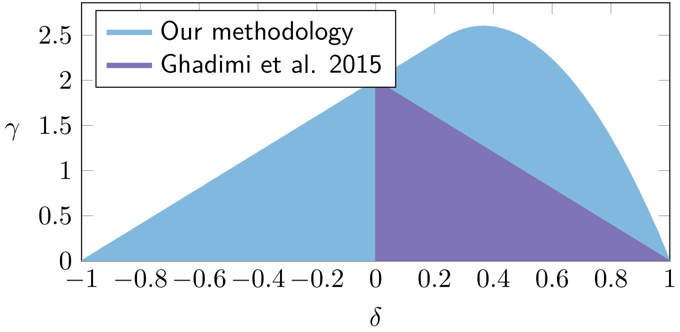
B2. Fix $\rho = 1$ and find range of algorithm parameters for which there exists a (P, p, T, t, ρ)-Lyapunov inequality on pre-specified grid

Gradient method with heavy-ball momentum

- Algorithm

$$x_{k+1} = x_k - \gamma \nabla f_1(x_k) + \delta(x_k - x_{k-1})$$

- Function suboptimality convergence region for $f_1 \in \mathcal{F}_{0,1}$



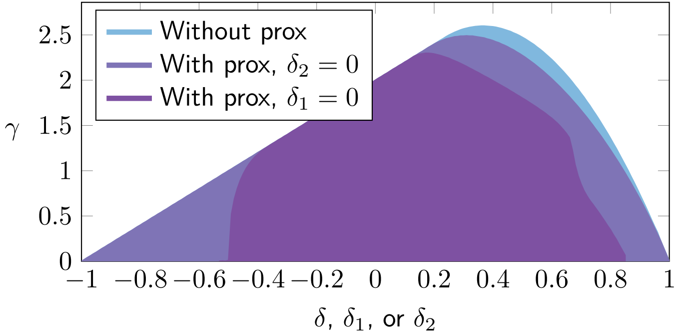
- Larger parameter region with function suboptimality convergence

Proximal gradient method with heavy-ball momentum

- Algorithm

$$x_{k+1} = \mathrm{prox}_{\gamma f_2}(x_k - \gamma\nabla f_1(x_k) + \delta_1(x_k - x_{k-1})) + \delta_2(x_k - x_{k-1})$$

  reduces to grad heavy-ball method if $\delta_1 = 0$ or $\delta_2 = 0$
- Duality gap convergence region $f_1 \in \mathcal{F}_{0,1}$ and $f_2 \in \mathcal{F}_{0,\infty}$



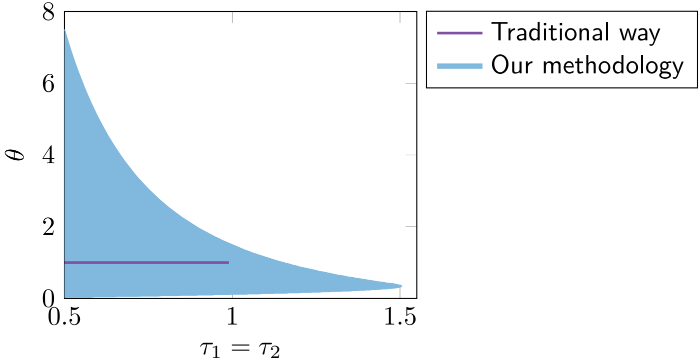
- Convergent parameter region smaller with prox
- Larger region if momentum inside prox

Chambolle–Pock

- Chambolle–Pock ("with $L = \mathrm{Id}$"): $\underset{x \in \mathcal{H}}{\text{minimize}}(f_1(x) + f_2(x))$

$$x_{k+1} = \text{prox}_{\tau_1 f_1}(x_k - \tau y_k)$$
$$y_{k+1} = \text{prox}_{\tau_2 f_2^*}(y_k + \tau_2 (x_{k+1} + \theta(x_{k+1} - x_k)))$$

- Convergent parameter choices (primal-dual gap, f_1 and f_2 pcc)



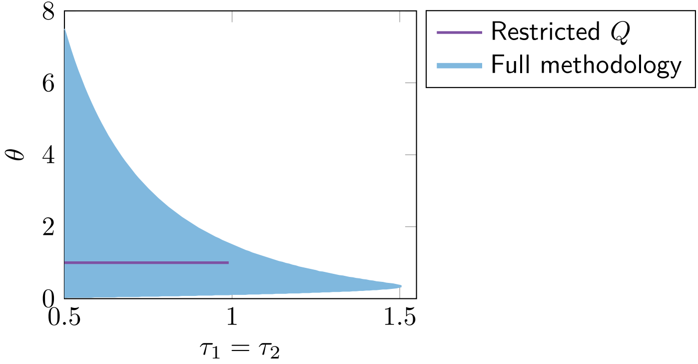
(Caveat: verified on a 0.01×0.01 grid of region)

Chambolle–Pock—Restricted Lyapunov

- Restrict the Lyapunov search space by imposing

$$Q = \begin{bmatrix} Q_{xx} & 0 \\ 0 & 0 \end{bmatrix}, \qquad (P, p) = \left(\begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix}, 0 \right)$$

- Convergent parameter choices (primal-dual gap, f_1 and f_2 pcc)



- Restriction in Lyapunov ansatz gives traditional parameter region

Thank you

arXiv:tomorrow