

robust control perspectives on algorithm analysis and design

Laurent Lessard

Northeastern University

NCCR Automation Symposium, ETH Zürich
May 23, 2022

Unconstrained optimization:

$$\begin{array}{ll}\text{minimize} & f(x) \\ \text{subject to} & x \in \mathbb{R}^d\end{array}$$

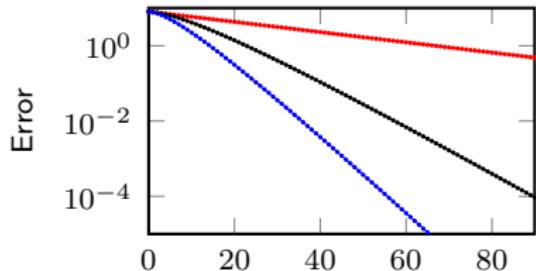
- need algorithms that are *fast* and *simple*
- currently favored family: *first-order methods*

Gradient descent

$$x_{k+1} = x_k - \alpha \nabla f(x_k)$$

Polyak acceleration (Heavy Ball)

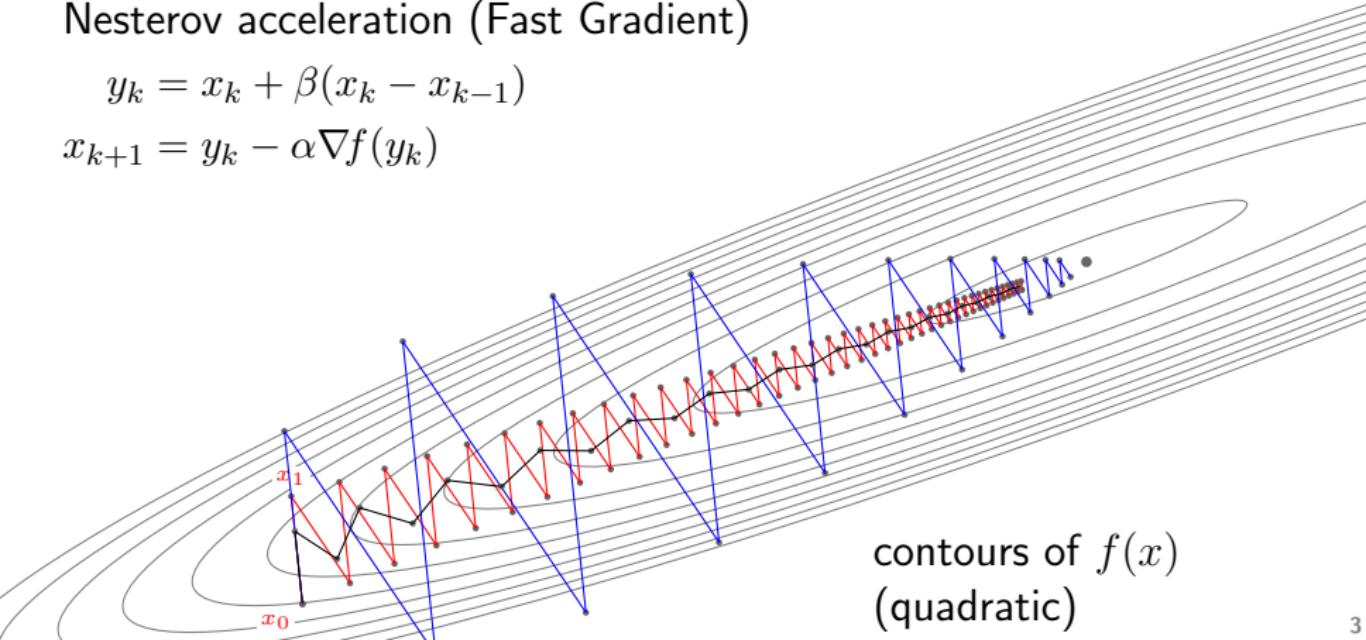
$$x_{k+1} = x_k - \alpha \nabla f(x_k) + \beta(x_k - x_{k-1})$$



Nesterov acceleration (Fast Gradient)

$$y_k = x_k + \beta(x_k - x_{k-1})$$

$$x_{k+1} = y_k - \alpha \nabla f(y_k)$$



contours of $f(x)$
(quadratic)

Algorithm selection and
tuning is an art.

Warm-up: Gradient descent

Suppose $f : \mathbb{R}^d \rightarrow \mathbb{R}$ satisfies $0 \prec mI_d \preceq \nabla^2 f(x) \preceq LI_d$ for all x and we apply gradient descent:

$$x_{k+1} = x_k - \alpha \nabla f(x_k)$$

how fast does $\|x_k - x_\star\| \rightarrow 0$?

What is the best α (in terms of m and L)?

Nesterov's bound:

First, prove this fact about f (use co-coercivity, Cauchy–Schwarz):

$$(x - y)^\top (\nabla f(x) - \nabla f(y)) \geq \frac{mL}{m+L} \|x - y\|^2 + \frac{1}{m+L} \|\nabla f(x) - \nabla f(y)\|^2$$

Then, if $\alpha \leq \frac{2}{m+L}$, we have:

$$\begin{aligned}\|x_{k+1} - x_\star\|^2 &= \|x_k - x_\star - \alpha \nabla f(x_k)\|^2 \\ &= \|x_k - x_\star\|^2 - 2\alpha (x_k - x_\star)^\top \nabla f(x_k) + \alpha^2 \|\nabla f(x_k)\|^2 \\ &\leq \left(1 - \frac{2\alpha m L}{m+L}\right) \|x_k - x_\star\|^2 + \alpha \left(\alpha - \frac{2}{m+L}\right) \|\nabla f(x_k)\|^2\end{aligned}$$

$$\|x_{k+1} - x_\star\| \leq \underbrace{\sqrt{1 - \frac{2\alpha m L}{m+L}}}_{\rho} \|x_k - x_\star\|$$

Bound is minimized when $\alpha = \frac{2}{m+L}$ and $\rho = \frac{L-m}{L+m}$.

Polyak's bound:

Use this fact about f (from the mean value theorem):

$$\nabla f(x_k) = \nabla f(x_\star) + \int_0^1 \nabla^2 f(x_\star + \tau(x_k - x_\star))(x_k - x_\star) d\tau = A_k(x_k - x_\star)$$

where $mI_d \preceq A_k \preceq LI_d$ by assumption.

$$\begin{aligned}\|x_{k+1} - x_\star\| &= \|x_k - x_\star - \alpha \nabla f(x_k)\| \\&= \|(1 - \alpha A_k)(x_k - x_\star)\| \\&\leq \|1 - \alpha A_k\| \cdot \|x_k - x_\star\| \\&\leq \underbrace{\max\{|1 - \alpha m|, |1 - \alpha L|\}}_{\rho} \cdot \|x_k - x_\star\|\end{aligned}$$

Bound is minimized when $\alpha = \frac{2}{m+L}$ and $\rho = \frac{L-m}{L+m}$.

Can we make it a
science?

$$\underset{x \in \mathbb{R}^d}{\text{minimize}} \quad f(x)$$

In this talk:

- Iterative algorithms can be viewed as robust controllers.
- Algorithm analysis can be performed **rapidly and automatically** by solving small semidefinite programs.
- Match or exceed state-of-the-art performance bounds.
- Automated analysis enables **principled design**.

SDP bound

Gradient descent with stepsize α converges with rate ρ if there exists some $\lambda \geq 0$ such that:

$$\begin{bmatrix} \rho^2 + 2mL\lambda & (m+L)\lambda & 1 \\ (m+L)\lambda & 2\lambda & \alpha \\ 1 & \alpha & 1 \end{bmatrix} \succeq 0$$

Reduces to $\rho \geq \max\{|1 - \alpha L|, |1 - \alpha m|\}$.
(same as Polyak bound)

Further optimizing yields $\alpha_{\text{opt}} = \frac{2}{m+L}$ and $\rho_{\text{opt}} = \frac{L-m}{L+m}$.

SDP bound

Linear matrix inequality can be rewritten as:

$$\begin{bmatrix} 1 & -\alpha \\ -\alpha & \alpha^2 \end{bmatrix} - \rho^2 \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} + \lambda \begin{bmatrix} -2mL & m+L \\ m+L & -2 \end{bmatrix} \preceq 0$$

Multiplying by $(x_k - x_*, \nabla f(x_k))$ on both sides:

$$\underbrace{\|x_{k+1} - x_*\|^2}_{V(x_{k+1})} - \rho^2 \underbrace{\|x_k - x_*\|^2}_{V(x_k)} + \underbrace{\lambda \begin{bmatrix} x_k - x_* \\ \nabla f(x_k) \end{bmatrix}^\top \begin{bmatrix} -2mL & m+L \\ m+L & -2 \end{bmatrix} \begin{bmatrix} x_k - x_* \\ \nabla f(x_k) \end{bmatrix}}_{\text{always nonnegative}} \preceq 0$$

$V(x) = \|x - x_*\|^2$ is a Lyapunov function that certifies geometric convergence with rate ρ .

Outline

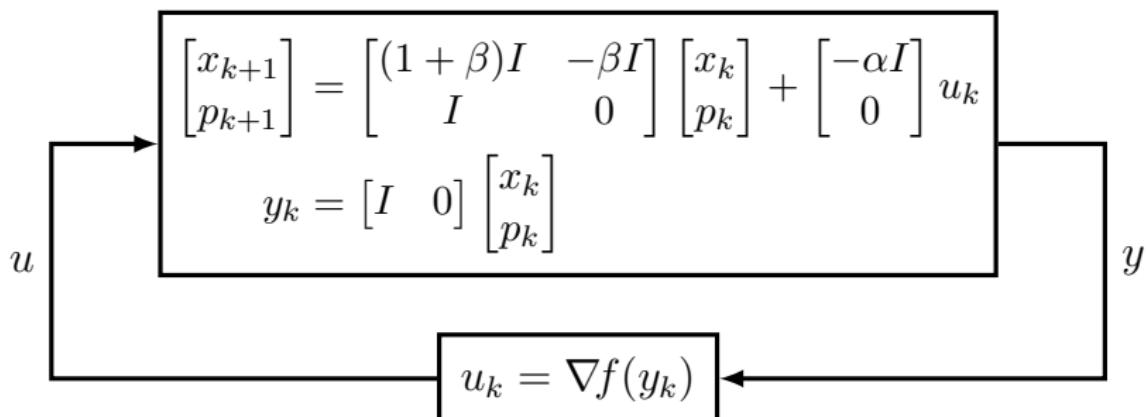
- Iterative algorithms as dynamical systems
- Lyapunov approach to stability
- Stochastic optimization
- Distributed optimization

Dynamical system interpretation

Heavy ball: $x_{k+1} = x_k - \alpha \nabla f(x_k) + \beta(x_k - x_{k-1})$

Define $u_k := \nabla f(x_k)$ and $p_k := x_{k-1}$

algorithm (linear, known, decoupled)



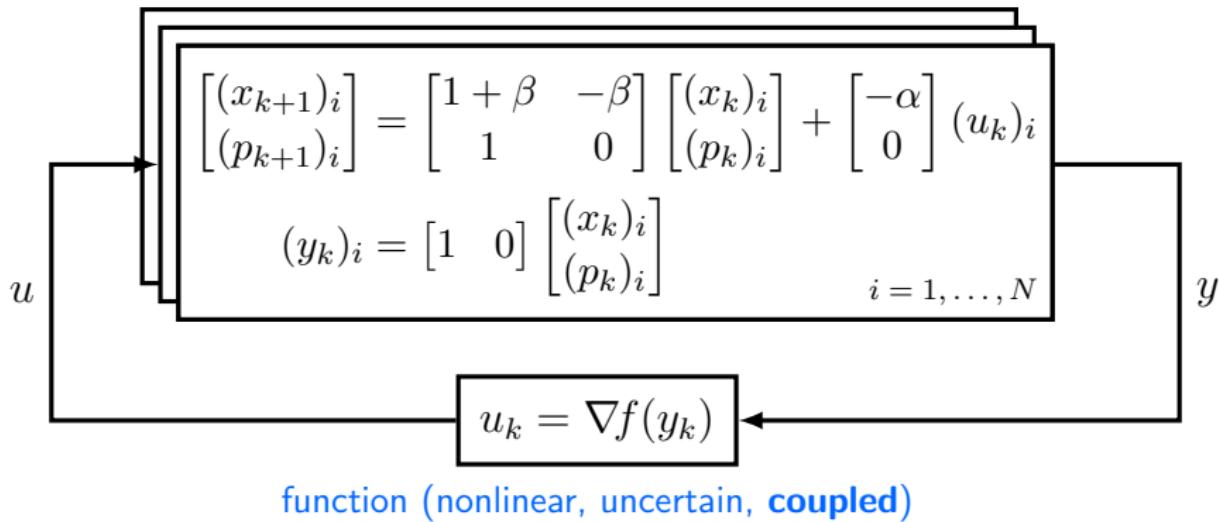
function (nonlinear, uncertain, coupled)

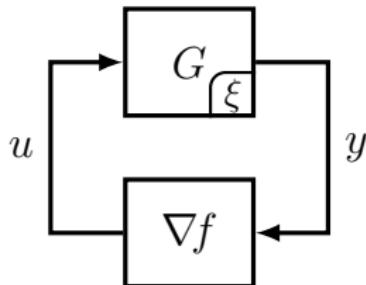
Dynamical system interpretation

Heavy ball: $x_{k+1} = x_k - \alpha \nabla f(x_k) + \beta(x_k - x_{k-1})$

Define $u_k := \nabla f(x_k)$ and $p_k := x_{k-1}$

algorithm (linear, known, **decoupled**)





$$\xi_{k+1} = A\xi_k + Bu_k$$

$$y_k = C\xi_k$$

$$u_k = \nabla f(y_k)$$

$$\left[\begin{array}{c|c} A & B \\ \hline C & 0 \end{array} \right] = \left\{ \begin{array}{l} \left[\begin{array}{c|c} 1 & -\alpha \\ \hline 1 & 0 \end{array} \right] \\ \text{Gradient} \\ \\ \left[\begin{array}{cc|c} 1+\beta & -\beta & -\alpha \\ 1 & 0 & 0 \\ \hline 1 & 0 & 0 \end{array} \right] \\ \text{Heavy ball} \\ \\ \left[\begin{array}{cc|c} 1+\beta & -\beta & -\alpha \\ 1 & 0 & 0 \\ \hline 1+\beta & -\beta & 0 \end{array} \right] \\ \text{Nesterov} \end{array} \right.$$



Functions are often defined in terms of the inequalities they must satisfy.



Convex function

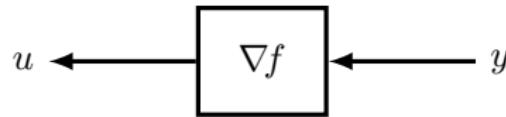
- $f(y) \geq f(x) + \nabla f(x)^T(y - x)$
- $(\nabla f(y) - \nabla f(x))^T(y - x) \geq 0$

Convex with Lipschitz gradients

- $f(y) \leq f(x) + \nabla f(x)^T(y - x) + \frac{L}{2}\|y - x\|^2$
- $(\nabla f(y) - \nabla f(x))^T(y - x) \leq L\|y - x\|^2$
- $f(y) \geq f(x) + \nabla f(x)^T(y - x) + \frac{1}{2L}\|\nabla f(y) - \nabla f(x)\|^2$

Strongly convex with Lipschitz gradients

- $f(y) - f(x) - \nabla f(x)^T(y - x) \geq \frac{1}{2(L-m)}(mL\|y - x\|^2 - 2m(\nabla f(y) - \nabla f(x))^T(y - x) + \|\nabla f(y) - \nabla f(x)\|^2)$



Nonconvex examples

- Weak strong convexity [Necoara et al. '15]:

$$f(x_\star) \geq f(x) + \nabla f(x)^\top (x_\star - x) + \frac{m}{2} \|x - x_\star\|^2$$
- Restricted secant inequality [Zhang,Yin '13]:

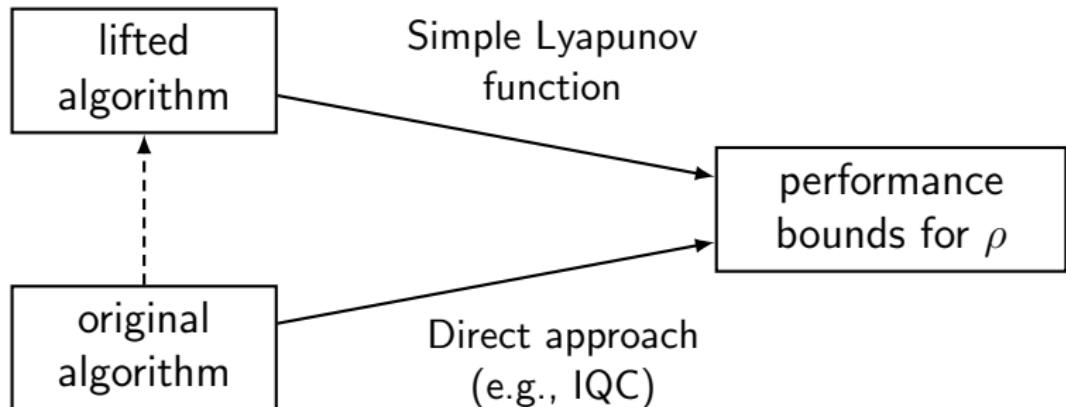
$$\nabla f(x)^\top (x - x_\star) \geq \frac{m}{2} \|x - x_\star\|^2$$
- Quadratic Growth [Anitescu '00]:

$$f(x) - f(x_\star) \geq \frac{m}{2} \|x - x_\star\|^2$$
- Error bound [Luo,Tseng '93]:

$$\frac{m}{2} \|x - x_\star\| \leq \|\nabla f(x)\|$$
- Polyak–Łojasiewicz ['63]:

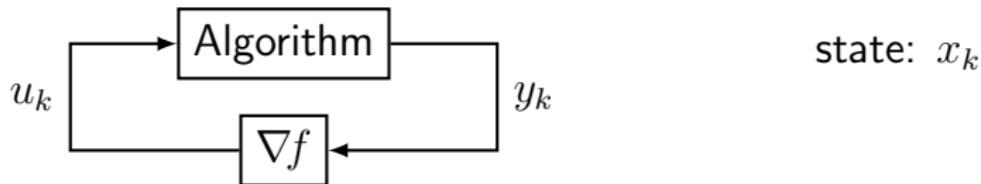
$$\frac{1}{2} \|\nabla f(x)\|^2 \geq \frac{m}{L} (f(x) - f(x_\star))$$

Lifting approach



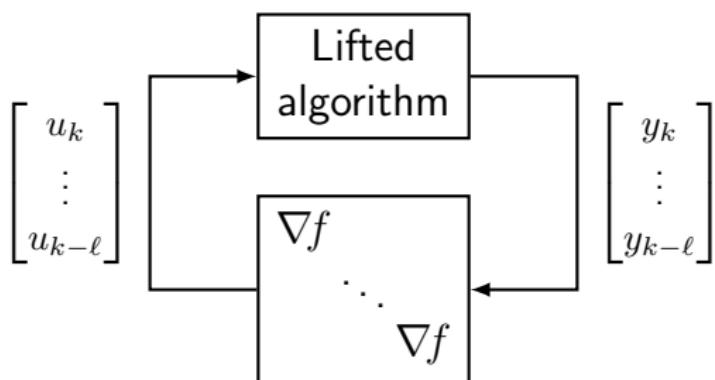
Lifting increases the number of variables but allows use of a simpler Lyapunov function.

Original system:



state: x_k

Lifted system:



state: $\bar{x}_k =$

$$\begin{bmatrix} x_k \\ y_{k-1} \\ \vdots \\ y_{k-\ell} \\ u_{k-1} \\ \vdots \\ u_{k-\ell} \end{bmatrix}$$

Outline of approach

- Use $V(\bar{x}) = \bar{x}_k^T P \bar{x}_k + p^T \bar{f}_k$; quadratic in algorithm state and linear in function values.
- Write all possible linear combinations of valid inequalities satisfied by $f(x)$: $\Pi(\Lambda) \geq 0$. Use interpolation conditions [Taylor et al. 2017] to eliminate redundant inequalities.
- Use S-procedure; Find P, p, Λ such that

$$V(\bar{x}_{k+1}) - \rho^2 V(\bar{x}_k) + \Pi(\Lambda) \leq 0 \quad \text{for all } \bar{x}_k, y_k, u_k$$

Since $\Pi(\Lambda) \geq 0$, this implies $V(x_{k+1}) \leq \rho^2 V(x)$.

Higher lifting dimension means more interpolation conditions, and potentially less conservatism.

Related work

Finite-time performance

- [Drori and Teboulle. 2014]
- PEP approach [Taylor, Hendrickx, Glineur. 2017]
- Tight bounds, but LMI size depends on horizon length

Asymptotic performance

- IQC approach [Lessard, Recht, Packard. 2016]
- More IQCs [Michalowsky, Scherer, Ebenbauer. 2020]

Proposed Lyapunov approach

- Similar in spirit to PEP
- Recovers same bounds as IQC approaches

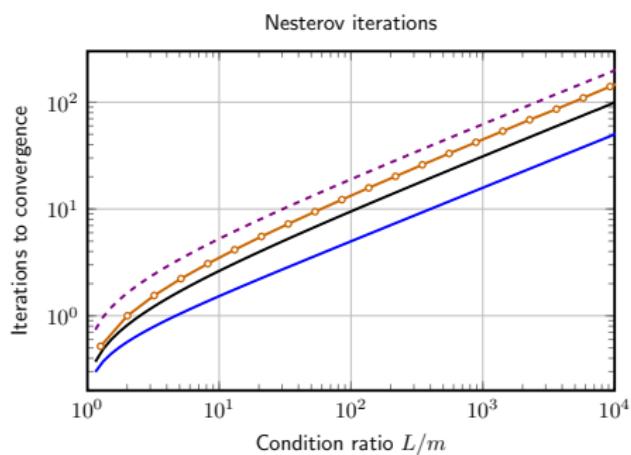
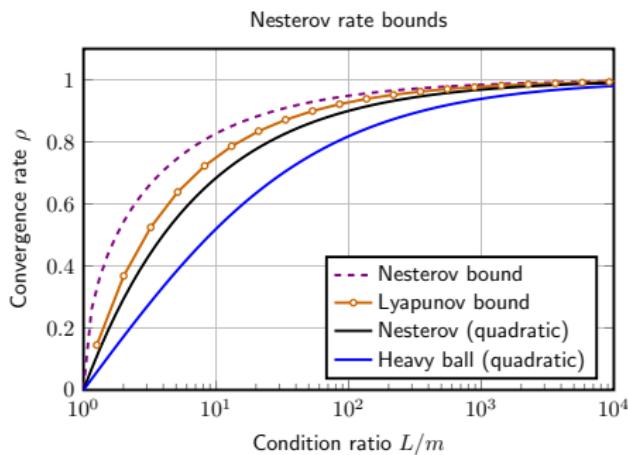
Efficiency

- Search for Lyapunov function is a linear matrix inequality.
- Size does *not* depend on function domain dimension d .
- Typically need bisection search to find smallest feasible ρ .
- $\ell = 1$ appears sufficient for best ρ bound.

Example: Smooth strongly convex functions. Given m, L and algorithm parameters (e.g., α, β), computing tightest ρ bound takes < 50 ms on a laptop.

Nesterov's method

$$x_{k+1} = x_k - \alpha \nabla f(x_k + \beta(x_k - x_{k-1})) + \beta(x_k - x_{k-1})$$



- Lyapunov bound **improves** upon Nesterov's own bound.
- Number of iterations is still asymptotically the same.

More results

$$x_{k+1} = x_k - \alpha \nabla f(x_k + \eta(x_k - x_{k-1})) + \beta(x_k - x_{k-1})$$

Nesterov's tuning uses $\alpha = \frac{1}{L}$ and $\beta = \eta = \frac{\sqrt{L}-\sqrt{m}}{\sqrt{L}+\sqrt{m}}$.

- Shown in [Safavi et al. 2018] that using $\beta = \eta = \frac{2L-m-\sqrt{2Lm-m^2}}{2L+\sqrt{2Lm-m^2}}$ instead yields an improved rate.
- Triple Momentum Method [Van Scy et al. 2017] allowed (α, β, η) to be different; yields fastest known algorithm on strongly convex functions.

These results leveraged a dynamical systems perspective.

Algorithm design for noise robustness

(with B. Van Scoy)

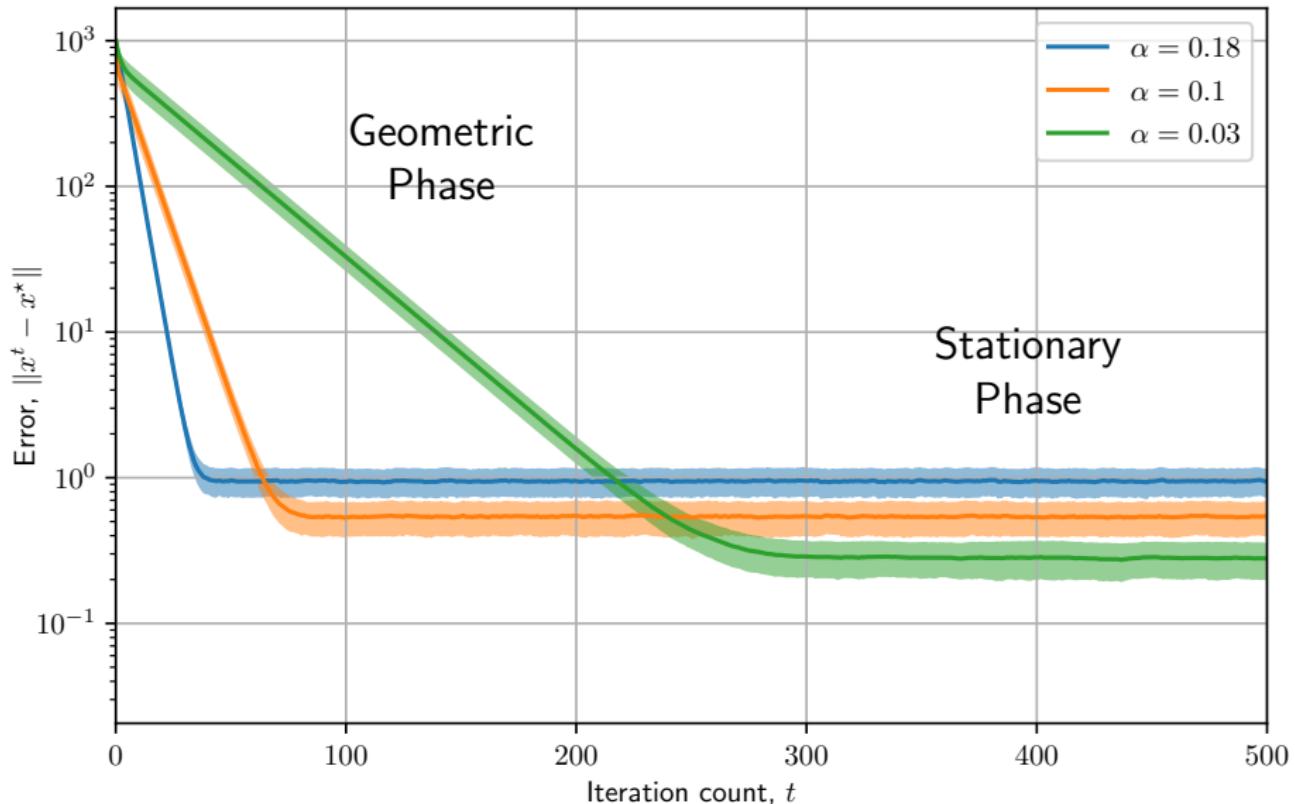
Gradient descent (GD)

$$x_{t+1} = x_t - \alpha g(x_k)$$

- α is the stepsize
- $g(x) = \nabla f(x) + w$ is the noisy gradient

Random quadratic function: $f(x) = x^\top Qx$ with $x \in \mathbb{R}^{10}$.

Eigenvalues satisfy $1 \leq \lambda(Q) \leq 10$ and $w_t \sim \mathcal{N}(0, I)$.



Performance metrics

Rate of convergence (ρ)

$$\|x_k - x^*\| \leq (\text{const}) \cdot \rho^k$$

Smaller ρ = faster convergence (geometric phase).

Sensitivity to noise (γ)

$$\limsup_{N \rightarrow \infty} \frac{1}{N} \mathbf{E} \sum_{k=0}^{N-1} \|x_k - x^*\|^2 = \gamma^2$$

Smaller γ = more noise robustness (stationary phase).

Question

How can we mediate the trade-off between speed and robustness for accelerated algorithms?

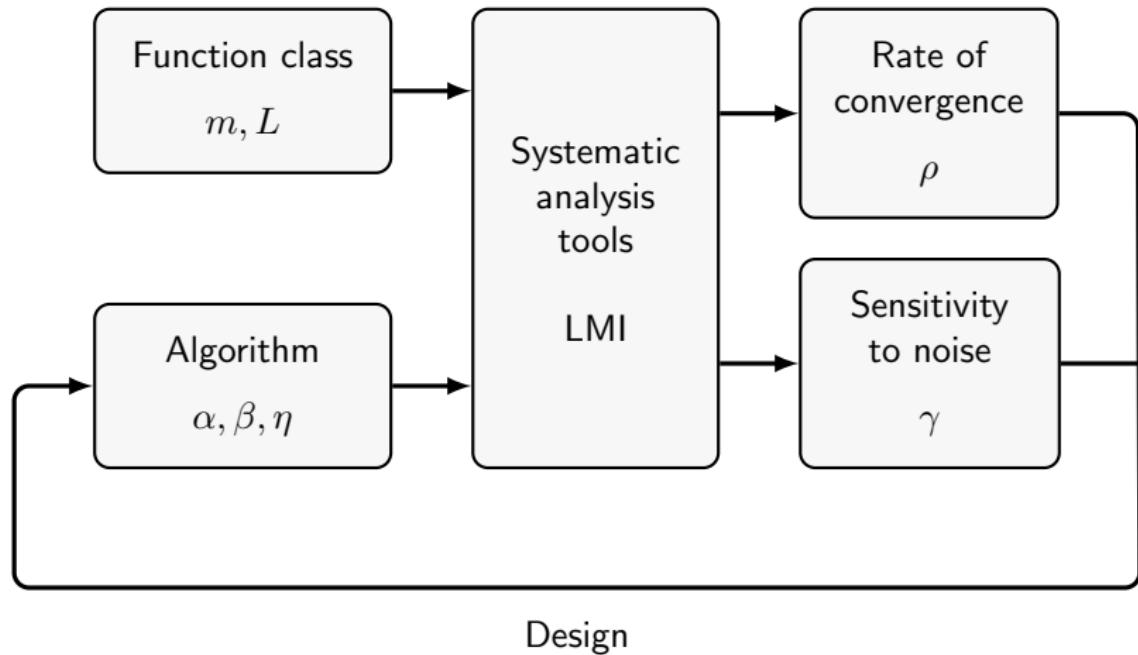
Algorithm: 3-parameter family

$$x_{k+1} = x_k - \alpha g(x_k + \eta(x_k - x_{k-1})) + \beta(x_k - x_{k-1})$$

Generalizes Polyak and Nesterov acceleration:

- Same template as Triple Momentum Method.
- Recovers Gradient descent when $\beta = \eta = 0$.
- Recovers Polyak acceleration when $\eta = 0$.
- Recovers Nesterov acceleration when $\eta = \beta$.

Outline



Bounding ρ

If $x_{k+1} = F(x_k)$ and we can find a function $V(x)$ satisfying

$$V(x) \geq \|x\|^2 \quad (\text{positivity})$$

$$V(F(x)) \leq \rho^2 V(x) \quad (\text{decrease condition})$$

Then we have geometric decrease:

$$\|x_k\|^2 \leq V(x_k) \leq \rho^2 V(x_{k-1}) \leq \cdots \leq \rho^{2k} V(x_0)$$

Just add valid inequalities to both conditions:

$$V(x) \geq \|x\|^2 + \Pi(\Lambda_1) \quad (\text{positivity})$$

$$V(F(x)) + \Pi(\Lambda_2) \leq \rho^2 V(x) \quad (\text{decrease condition})$$

Bounding γ

If $x_{k+1} = F(x_k, w_k)$ and we can find a function $V(x)$ satisfying

$$\begin{aligned}\mathbf{E} V(x) &\geq 0 \quad (\text{positivity}) \\ \mathbf{E} V(F(x, w)) - \mathbf{E} V(x) + \mathbf{E} \|x\|^2 &\leq \gamma^2 \quad (\text{decrease})\end{aligned}$$

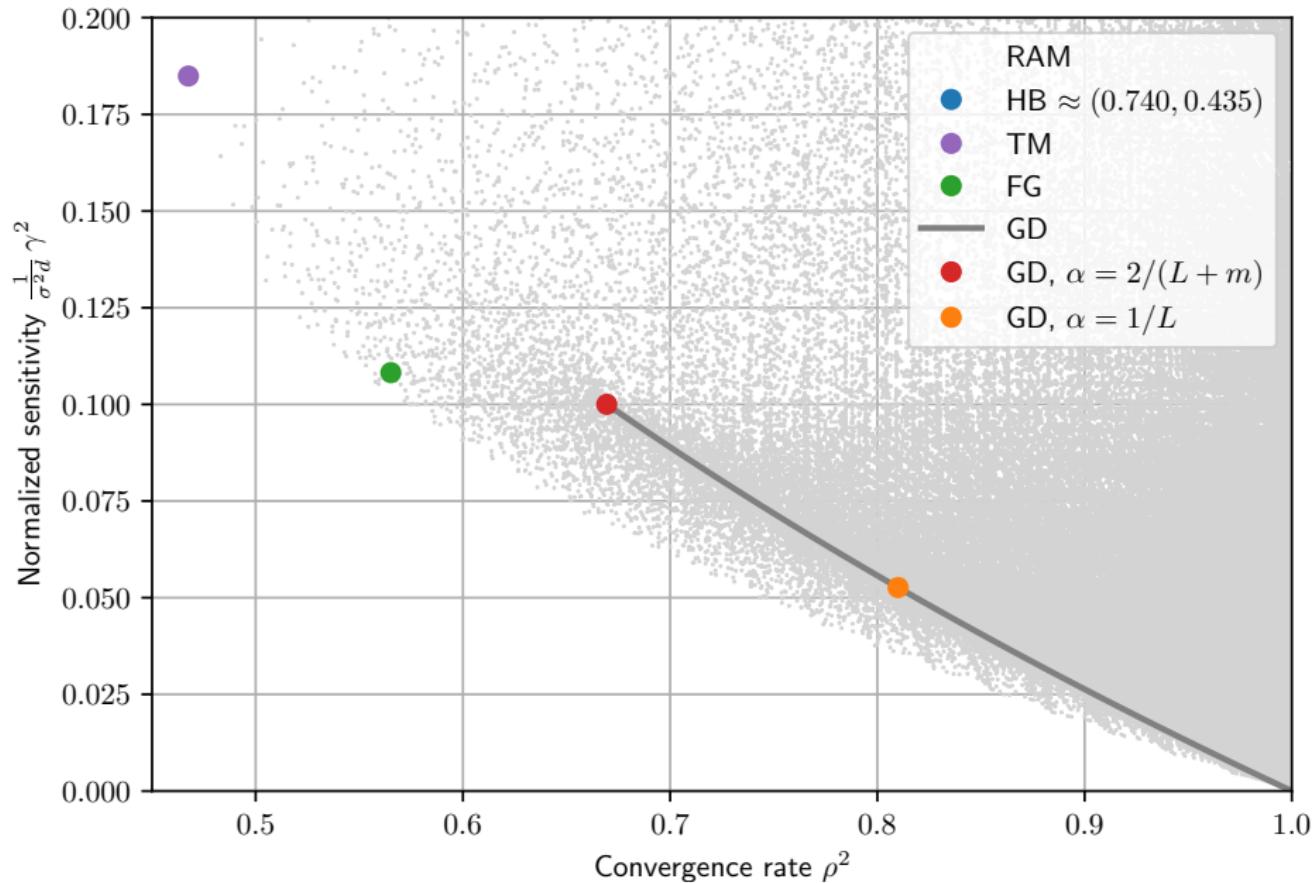
Then we have bounded steady-state covariance:

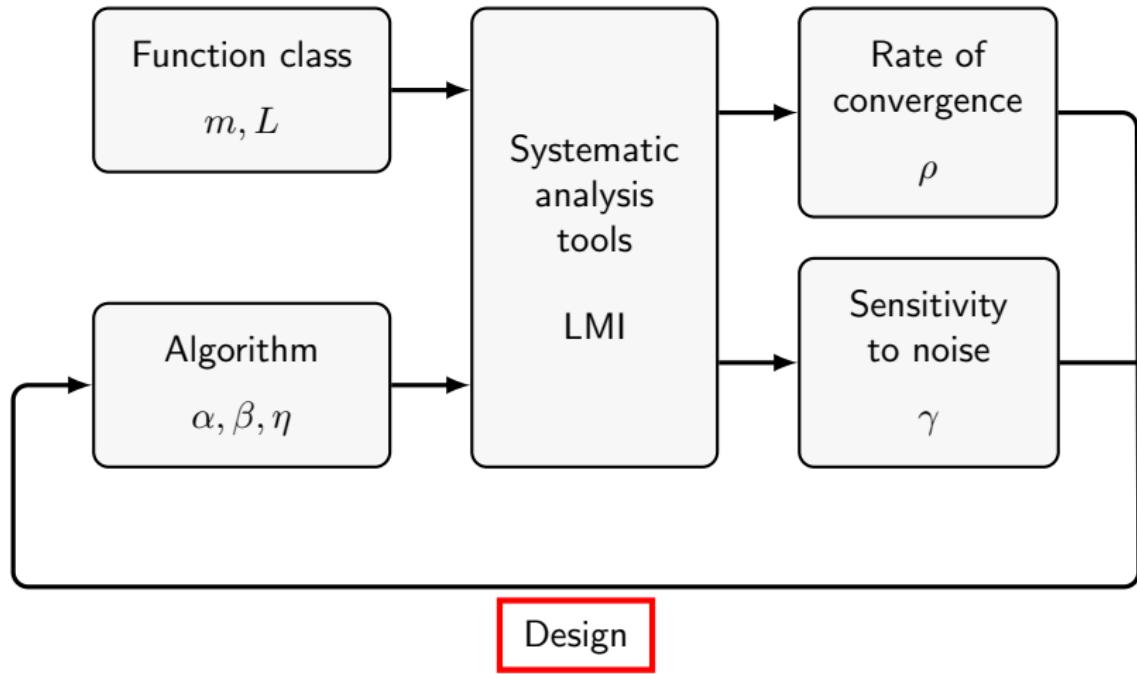
$$\begin{aligned}\mathbf{E} V(x_N) - \mathbf{E} V(x_0) + \mathbf{E} \sum_{k=0}^{N-1} \|x_k\|^2 &\leq N\gamma^2 \\ \implies \limsup_{N \rightarrow \infty} \frac{1}{N} \mathbf{E} \sum_{k=0}^{N-1} \|x_k\|^2 &\leq \gamma^2\end{aligned}$$

Efficiency

- Size does *not* depend on function domain dimension d .
- Typically need bisection search to find smallest feasible ρ . and $\ell = 1$ appears sufficient for best ρ bound.
- $\ell = 4$ appears sufficient for best γ bound, but no bisection is needed.
- For any choice of $(\alpha, \beta, \eta, L, m)$, it takes < 100 ms to compute the optimal (ρ, γ) .

(ρ, γ) tradeoff ($m = 1, L = 10$)





Design

Challenges

- In principle, solution is a *semialgebraic set*.
- Optimality conditions yield polynomials of degree > 200 that do not factor nicely.

Find algorithms that:

- Have simple and explicit algebraic expressions.
- Are near-optimal.

Numerically guided search

1. Use numerical solver (e.g. Nelder–Mead) to find locally optimal (α, β, η) , e.g. fix ρ and minimize γ .
2. Write LMI as polynomial optimization problem.
3. Substitute numerical solution to find active constraints.
At optimality, matrices in LMI will drop rank.
4. Look for analytic solution to system of active constraints.

Robust Accelerated Method (RAM)

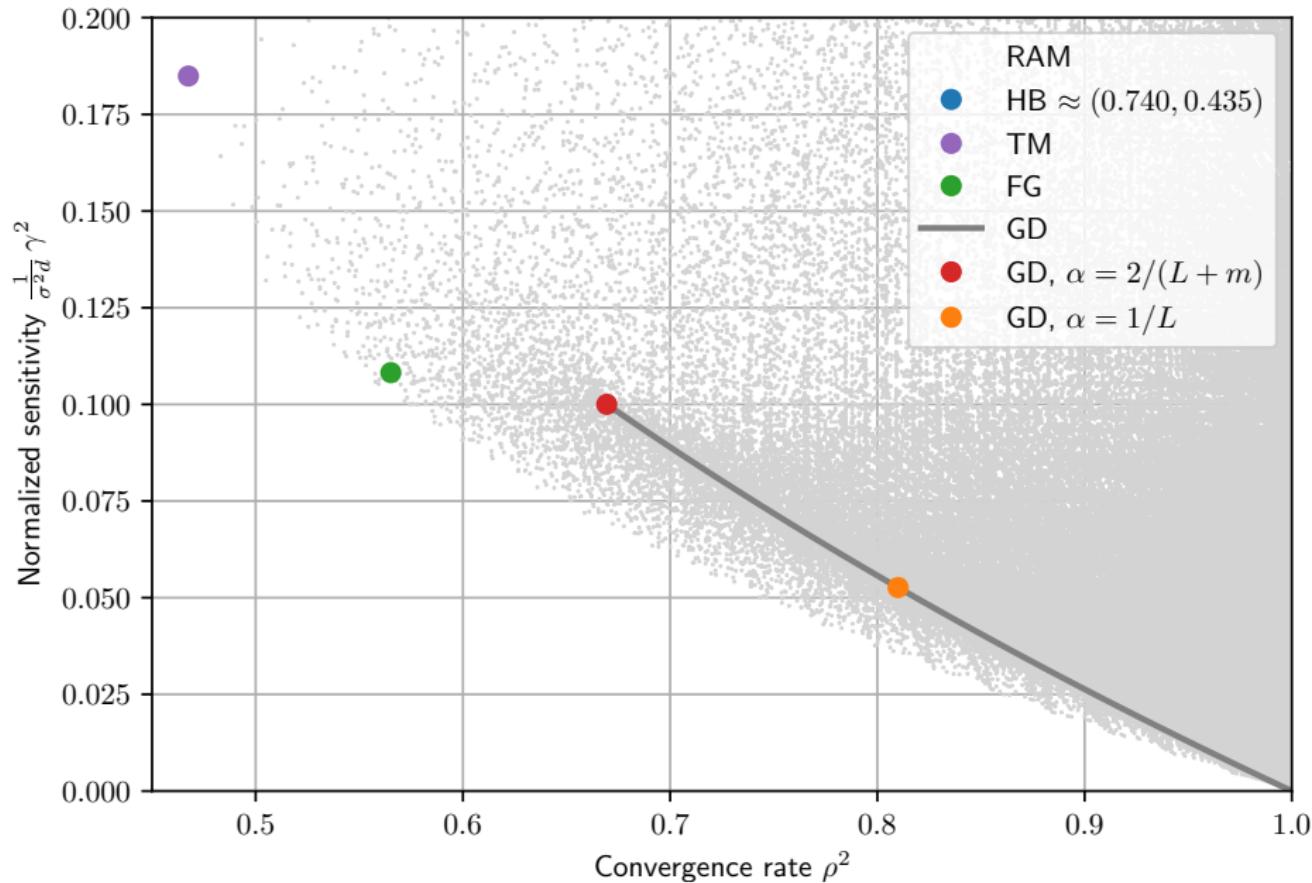
Let $\rho \in [1 - \sqrt{\frac{m}{L}}, 1)$. RAM is the 3-parameter algorithm

$$\alpha = \frac{(1+\rho)(1-\rho)^2}{m}, \quad \beta = \rho \frac{L(1-\rho+2\rho^2)-m(1+\rho)}{(L-m)(3-\rho)},$$
$$\eta = \rho \frac{L(1-\rho^2)-m(1+2\rho-\rho^2)}{(L-m)(3-\rho)(1-\rho^2)}.$$

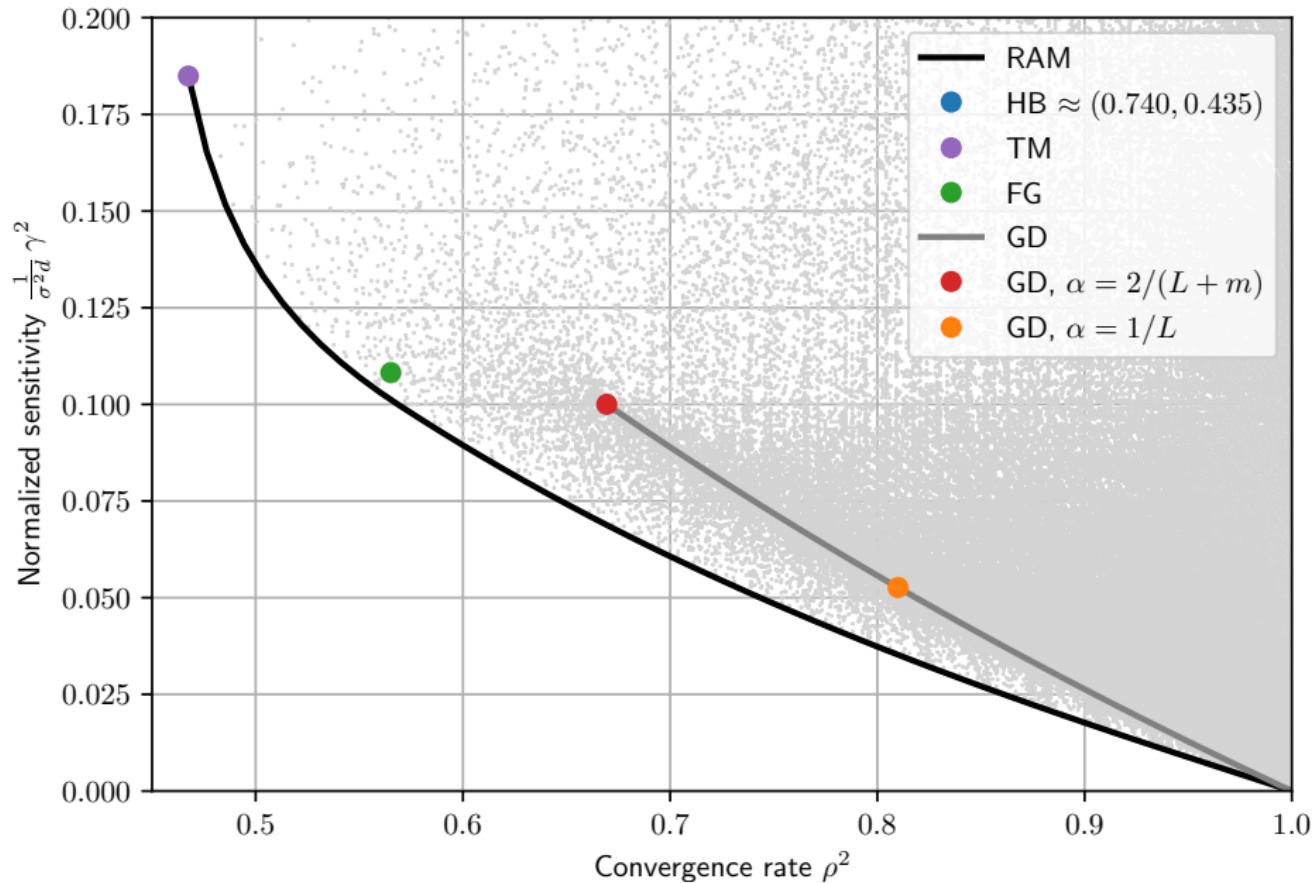
On the class $F_{m,L}$, RAM achieves $\rho_{\text{RAM}} = \rho$.

For larger ρ , RAM is *near-Pareto optimal*

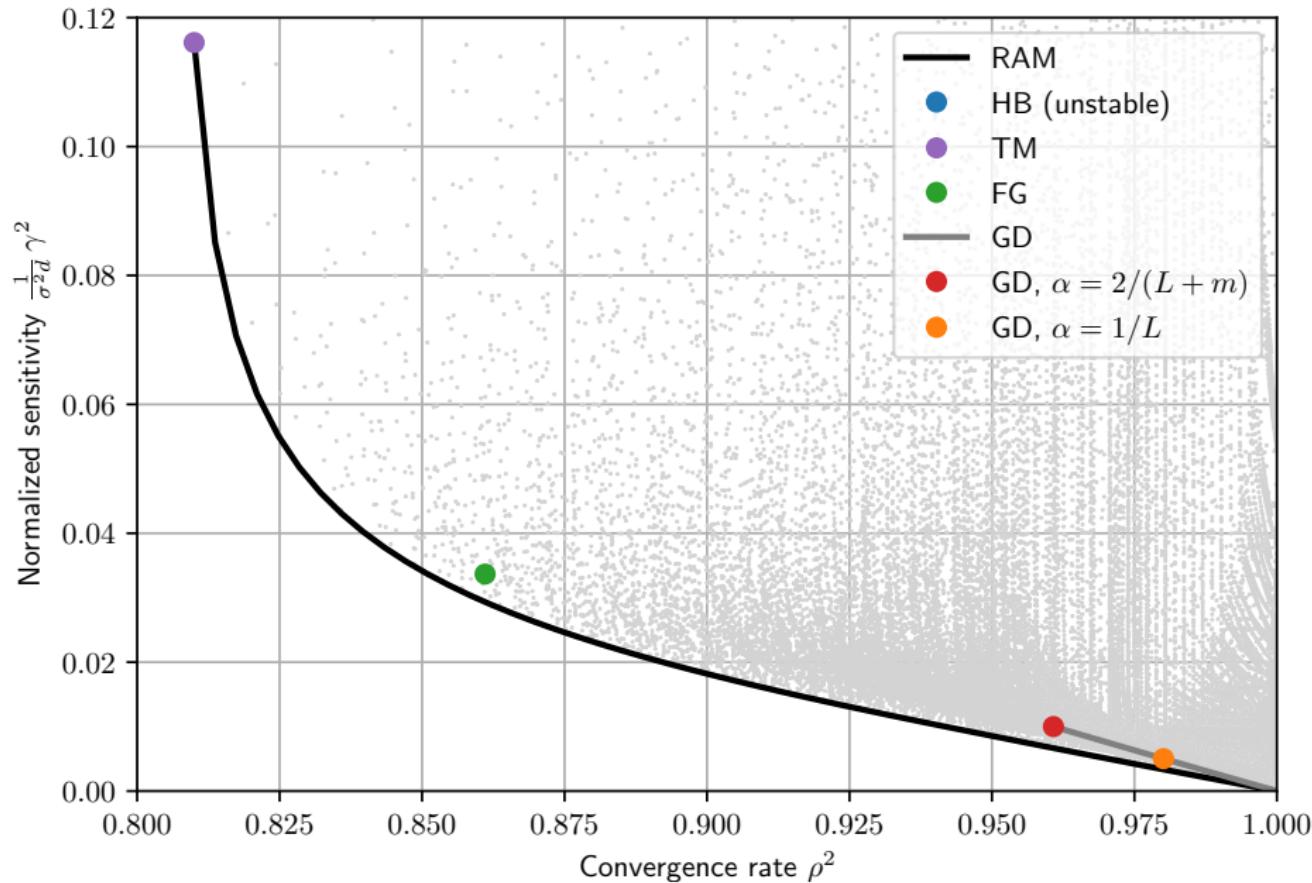
(ρ, γ) tradeoff ($m = 1, L = 10$)



(ρ, γ) tradeoff ($m = 1, L = 10$)

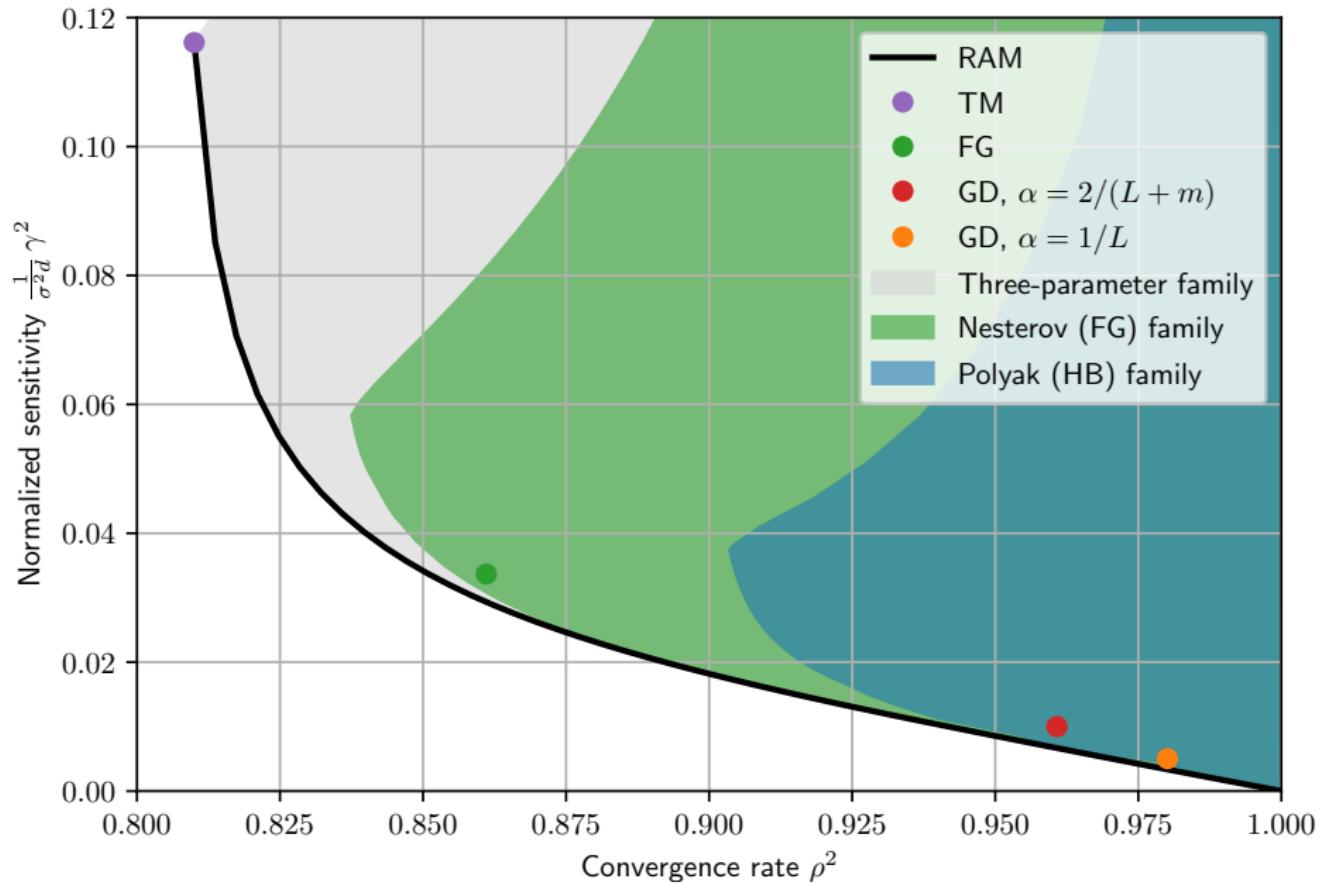


(ρ, γ) tradeoff ($m = 1, L = 100$)



RAM uses (α, β, η) . What if we use only Polyak or only Nesterov acceleration?

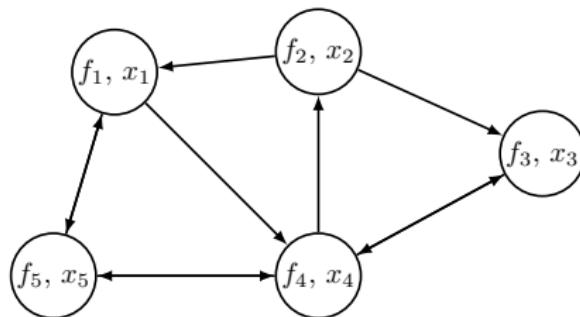
Nesterov and Polyak limitation ($m = 1, L = 100$)



Distributed optimization

(with A. Sundararajan and B. Van Scoy)

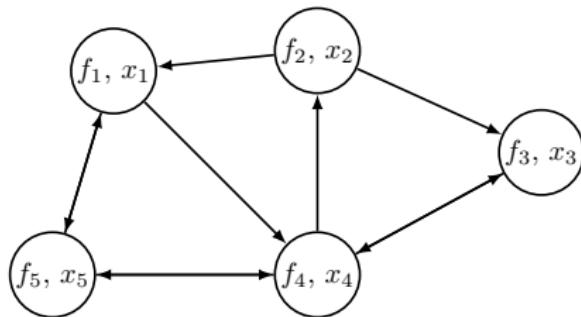
$$\begin{array}{ll}\text{minimize}_{x_1, \dots, x_n} & \sum_{i=1}^n f_i(x_i) \\ \text{subject to} & x_1 = x_2 = \dots = x_n\end{array}$$



At each timestep, agent i can:

1. Evaluate its local gradient ∇f_i
2. Transmit information to its immediate neighbors
3. Perform local computations

Gossip matrix



- W is doubly stochastic
- The iteration $x_i^{k+1} = \sum_{j=1}^n W_{ij}x_j^k$ converges to average

$$W = \begin{bmatrix} \frac{1}{3} & \frac{1}{3} & 0 & 0 & \frac{1}{3} \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 \\ \frac{1}{3} & 0 & \frac{1}{3} & 0 & \frac{1}{3} \\ \frac{1}{3} & 0 & 0 & \frac{1}{3} & \frac{1}{3} \end{bmatrix}$$

Distributed algorithms

$$x^{k+1} = Wx^k - \alpha \nabla f(x^k)$$

DGD'09

$$x^{k+1} = (I + W)x^k - \frac{I+W}{2}x^{k-1} - \alpha \nabla f(x^k) + \alpha \nabla f(x^{k-1})$$

EXTRA'15

$$x^{k+1} = 2Wx^k - W^2x^{k-1} - \alpha W^2(\nabla f(x^k) - \nabla f(x^{k-1}))$$

AugDGM'15

$$x^{k+1} = 2Wx^k - W^2x^{k-1} - \alpha \nabla f(x^k) + \alpha \nabla f(x^{k-1})$$

DIGing'17

$$x^{k+1} = (I + W)(x^k - \frac{1}{2}x^{k-1}) - \alpha \nabla f(\frac{I+W}{2}x^k) + \alpha \nabla f(\frac{I+W}{2}x^{k+1})$$

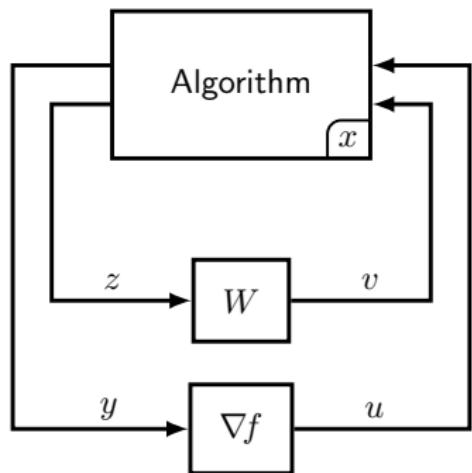
ExDiff'17

$$x^{k+1} = (I + W)x^k - \frac{I+W}{2}(x^{k-1} + \alpha \nabla f(x^k) - \alpha \nabla f(x^{k-1}))$$

NIDS'19

and more...

General algorithm form



$$\begin{bmatrix} x_i^{k+1} \\ y_i^k \\ z_i^k \end{bmatrix} = \begin{bmatrix} A & B_u & B_v \\ C_y & D_{yu} & D_{yv} \\ C_z & D_{zu} & D_{zv} \end{bmatrix} \begin{bmatrix} x_i^k \\ u_i^k \\ v_i^k \end{bmatrix}$$

$$v_i^k = \sum_{j=1}^n W_{ij}^k z_j^k$$

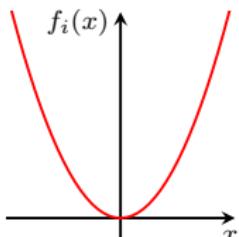
$$u_i^k = \nabla f_i(y_i^k)$$

Many known algorithms fit in this form!

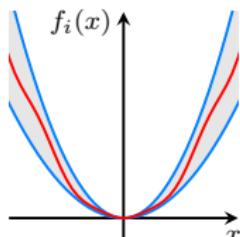
	2 state variables	3 state variables
1 communicated variable	<p>General Form</p> $\left[\begin{array}{c cc c} A & B_u & B_v \\ \hline C_y & D_{yu} & D_{yv} \\ C_z & D_{zu} & D_{zv} \end{array} \right]$ <p>Exact Diffusion (ExDIFF)</p> $\left[\begin{array}{c cc c} 1 & -1 & -\alpha & 1 \\ \hline \frac{1}{2} & 0 & -\alpha & \frac{1}{2} \\ 1 & 0 & -\frac{1}{2} & 0 \\ 1 & 0 & 0 & 0 \end{array} \right]$	<p>EXTRA</p> $\left[\begin{array}{cc cc c} 1 & -\frac{1}{2} & \alpha & -\alpha & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ \hline 1 & 0 & 0 & 0 & 0 \\ 1 & -\frac{1}{2} & 0 & 0 & 0 \end{array} \right]$ <p>NIDS</p> $\left[\begin{array}{cc cc c} 1 & -\frac{1}{2} & \frac{\alpha}{2} & -\frac{\alpha}{2} & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ \hline 1 & 0 & 0 & 0 & 0 \\ 1 & -\frac{1}{2} & \frac{\alpha}{2} & -\frac{\alpha}{2} & 0 \end{array} \right]$
2 communicated variables	<p>Unified DIGing (uDIG)</p> $\left[\begin{array}{cc cc cc} 0 & -\alpha & -\alpha & 1 & 0 \\ \hline \frac{L+m}{2} & 0 & -1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ \hline 1 & 0 & 0 & 0 & 0 \\ -\frac{L+m}{2} & 1 & 1 & 0 & 0 \end{array} \right]$	<p>DIGing</p> $\left[\begin{array}{cc cc cc} 0 & -\alpha & 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ \hline 0 & -\alpha & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{array} \right]$
2 communicated variables	<p>Unified EXTRA (uEXTRA)</p> $\left[\begin{array}{cc cc cc} 0 & -\alpha & -\alpha & 1 & 0 \\ \hline 0 & 0 & -1 & L & 1 \\ 1 & 0 & 0 & 0 & 0 \\ \hline 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & -L & 0 \end{array} \right]$	<p>AugDGM</p> $\left[\begin{array}{cc cc cc} 0 & 0 & 0 & 0 & 1 & -\alpha \\ 0 & 0 & -1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 1 & -\alpha \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{array} \right]$

Local functions: Each f_i satisfies a quadratic bound:

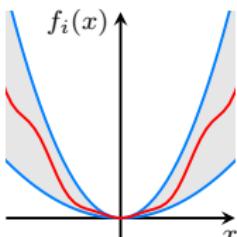
$$(\nabla f_i(x) - \nabla f_i(x_\star) - m(x - x_\star))^\top (\nabla f_i(x) - \nabla f_i(x_\star) - L(x - x_\star)) \leq 0$$



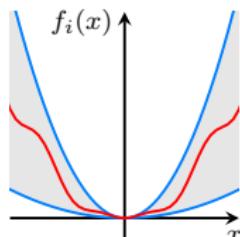
$$L/m = 1$$



$$L/m = 2$$

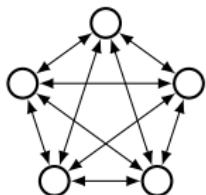


$$L/m = 4$$

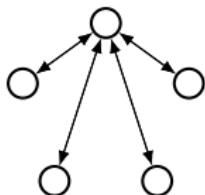


$$L/m = 8$$

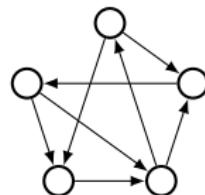
Gossip matrix: balanced, doubly stochastic, $\|\frac{1}{n}\mathbf{1}\mathbf{1}^\top - W^k\| \leq \sigma$



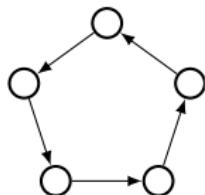
$$\sigma = 0$$



$$\sigma = 0.667$$



$$\sigma = 0.707$$



$$\sigma = 0.809$$

Define $M_0 := \begin{bmatrix} -2mL & L+m \\ L+m & -2 \end{bmatrix}$ and $M_1 := \begin{bmatrix} \sigma^2 - 1 & 1 \\ 1 & -1 \end{bmatrix}$.

If there exist matrices $P \succ 0$, $Q \succ 0$, and $R \succeq 0$ such that

$$\left[\begin{array}{cc} A & B_u \\ I & 0 \\ \hline C_y & D_{yu} \\ 0 & I \end{array} \right]^T \left[\begin{array}{ccc|c} P & 0 & 0 & 0 \\ 0 & -\rho^2 P & 0 & 0 \\ \hline 0 & 0 & M_0 & 0 \end{array} \right] \left[\begin{array}{cc} A & B_u \\ I & 0 \\ \hline C_y & D_{yu} \\ 0 & I \end{array} \right] \preceq 0$$

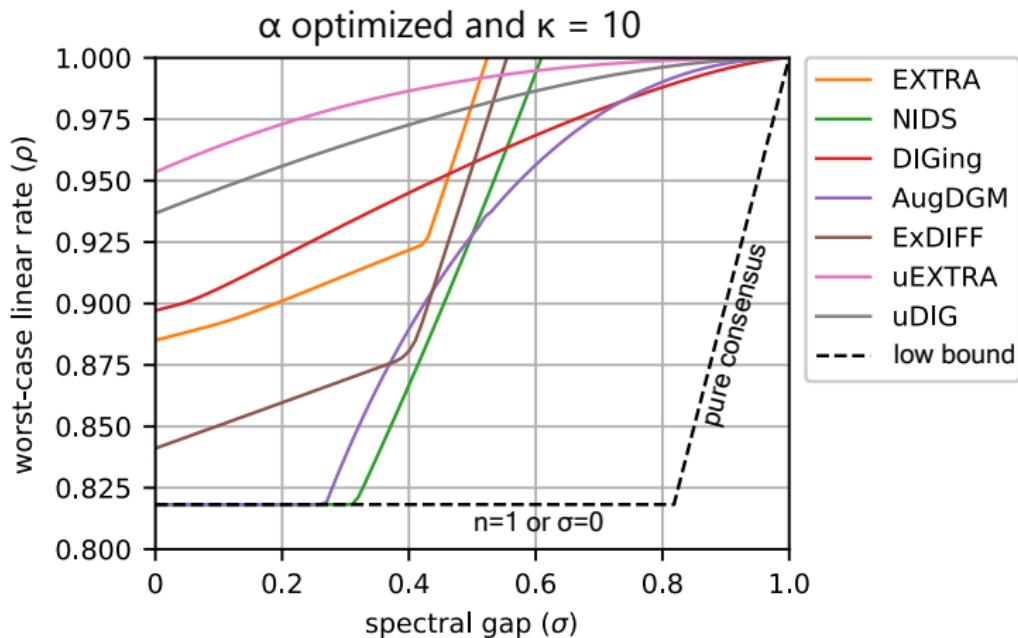
$$\left[\begin{array}{ccc} A & B_u & B_v \\ I & 0 & 0 \\ \hline C_y & D_{yu} & D_{yv} \\ 0 & I & 0 \\ \hline C_z & D_{zu} & D_{zv} \\ 0 & 0 & I \end{array} \right]^T \left[\begin{array}{ccc|c} Q & 0 & 0 & 0 \\ 0 & -\rho^2 Q & 0 & 0 \\ \hline 0 & 0 & M_0 & 0 \\ 0 & 0 & 0 & M_1 \otimes R \end{array} \right] \left[\begin{array}{ccc} A & B_u & B_v \\ I & 0 & 0 \\ \hline C_y & D_{yu} & D_{yv} \\ 0 & I & 0 \\ \hline C_z & D_{zu} & D_{zv} \\ 0 & 0 & I \end{array} \right] \preceq 0$$

Then there exists a constant $c > 0$ such that

$$\|x_i^k - x_\star\| \leq c \rho^k$$

for all agents $i \in \{1, \dots, n\}$ and all iterations $k \geq 0$.

Algorithm comparison



- α is gradient stepsize (optimized by grid search).

All algorithms analyzed using the same theorem!

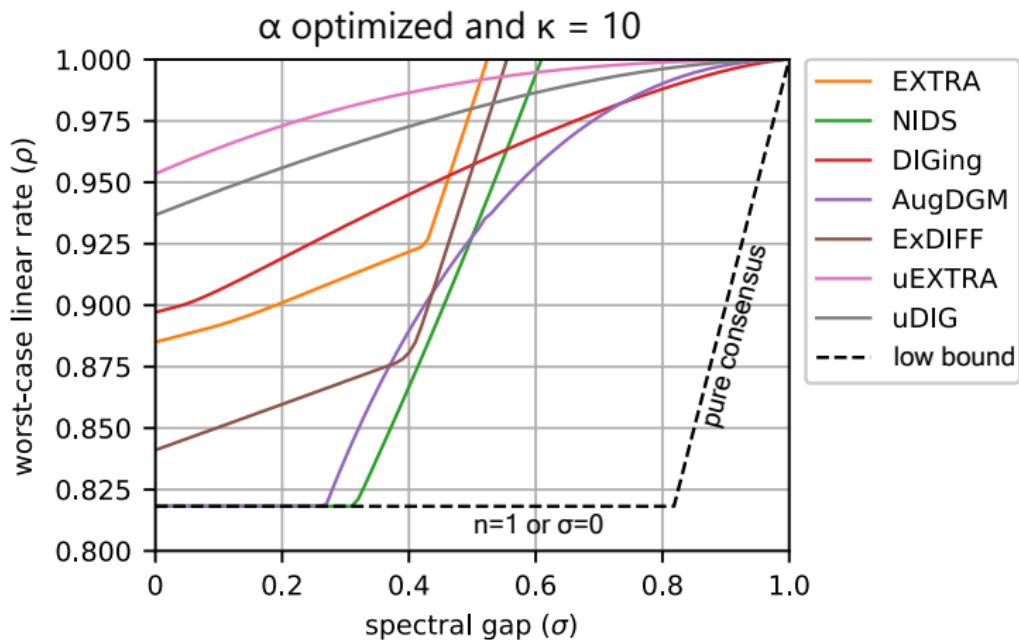
Algorithm design

- Analysis is now easy. Design is still hard:
(nonlinear, nonconvex, constrained)
- Look for *simple* algorithms.
- Result:

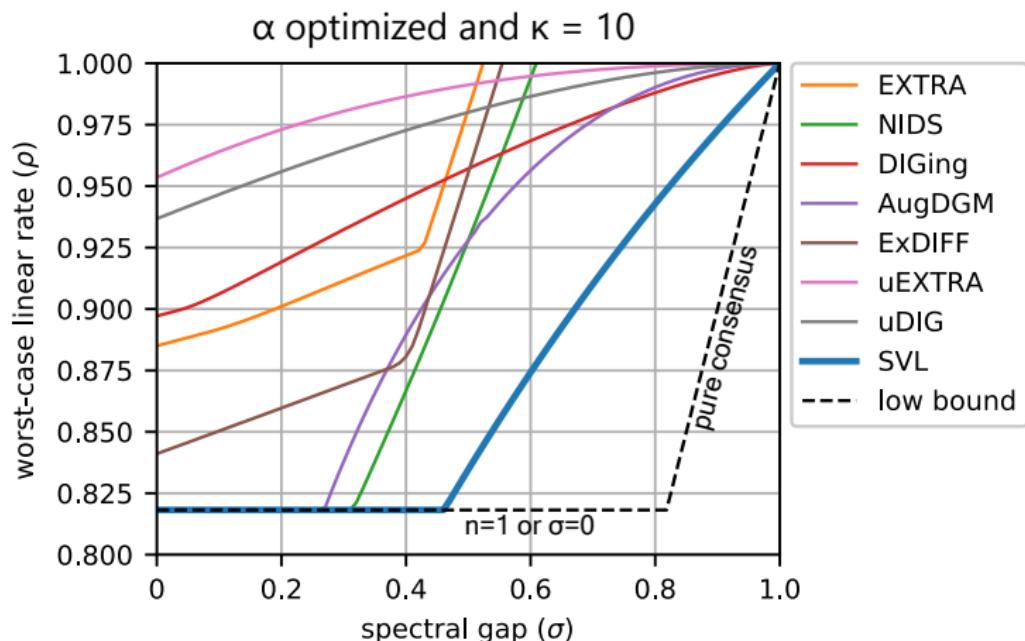
$$\begin{bmatrix} A & B_u & B_v \\ C_y & D_{yu} & D_{yv} \\ C_z & D_{zu} & D_{zv} \end{bmatrix} = \left[\begin{array}{ccc|c|c} 1-\gamma & \beta & -\alpha & \gamma \\ -1 & 1 & 0 & 1 \\ \hline 1-\delta & 0 & 0 & \delta \\ \hline 1 & 0 & 0 & 0 \end{array} \right]$$

- Closed-form expressions for optimal $(\alpha, \beta, \gamma, \delta)$ and ρ as functions of (L, m, σ) .

Algorithm comparison with SVL



Algorithm comparison with SVL



Best worst-case performance for *any* known algorithm!

A simple interpretation of SVL

ADMM:
$$\begin{cases} x_i^{k+1} = \arg \min_x f_i(x) + (x - y_i^k)^T z_i^k + \frac{\beta}{2} \|x - y_i^k\|^2 \\ y_i^{k+1} = \frac{1}{n} \sum_{j=1}^n x_j^{k+1} \\ z_i^{k+1} = z_i^k + \beta (x_i^{k+1} - y_i^{k+1}) \end{cases}$$

Inexact ADMM:
$$\begin{cases} x_i^{k+1} = y_i^k - \alpha (\nabla f(y_i^k) + z_i^k) \\ y_i^{k+1} = \sum_{j=1}^n w_{ij}^{k+1} x_j^{k+1} \\ z_i^{k+1} = z_i^k + \beta (x_i^{k+1} - y_i^{k+1}) \end{cases}$$

SVL is equivalent to inexact ADMM!

Summary

- Iterative algorithms as dynamical systems.
- Algorithm analysis can be performed **rapidly and automatically** by solving small semidefinite programs.
- Obtain simple and interpretable Lyapunov certificates that match or exceed state-of-the-art performance bounds.
- Algorithm design for near-Pareto-optimal rate–robustness trade-off.
- Algorithm design for distributed optimization.

Thank you!

- Robustness trade-off (with more function classes and designs!): <https://arxiv.org/abs/2109.05059>
- Distributed optimization:
[IEEE Trans. Contr. Network Sys.](https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9186300) 7(4), 1597–1608, 2020.
- All papers available on my website:
<https://laurentlessard.com>
- Funding acknowledgement:
NSF 1750162, 1936648

Backup Slides

Numerically guided search

1. Use numerical solver (e.g. Nelder–Mead) to find locally optimal (α, β, η) , e.g. fix ρ and minimize γ .
2. Write LMI as polynomial optimization problem.
3. Substitute numerical solution to find active constraints.
At optimality, matrices in LMI will drop rank.
4. Look for analytic solution to system of active constraints.

Simulation

Nesterov's worst-case quadratic:

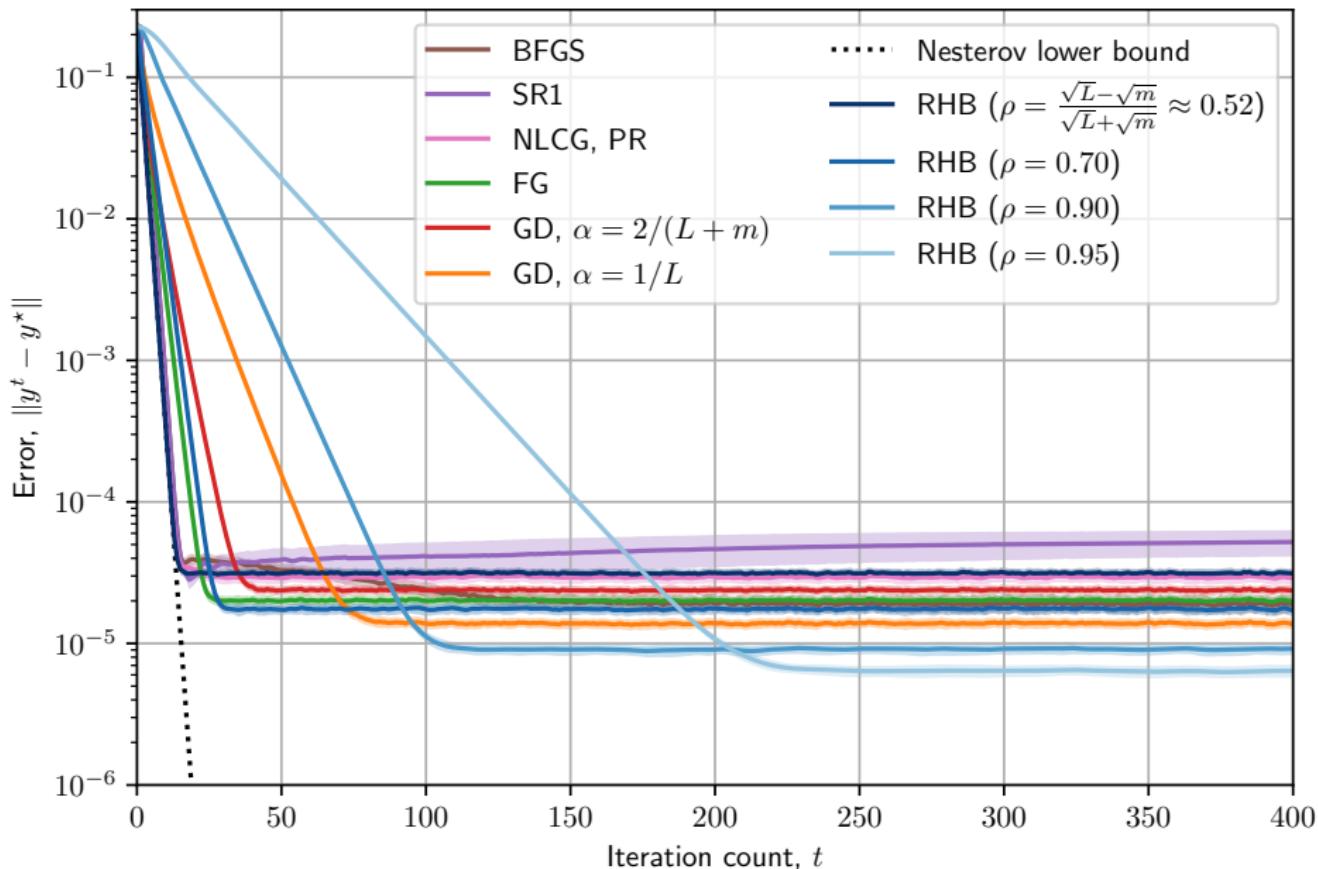
$$\nabla^2 f(x) = \begin{bmatrix} \frac{L+m}{2} & \frac{L-m}{4} & 0 & \dots \\ \frac{L-m}{4} & \frac{L+m}{2} & \frac{L-m}{4} & \dots \\ 0 & \frac{L-m}{4} & \frac{L+m}{2} & \dots \\ \ddots & \ddots & \ddots & \ddots \end{bmatrix}$$

Lower bound (any algorithm):

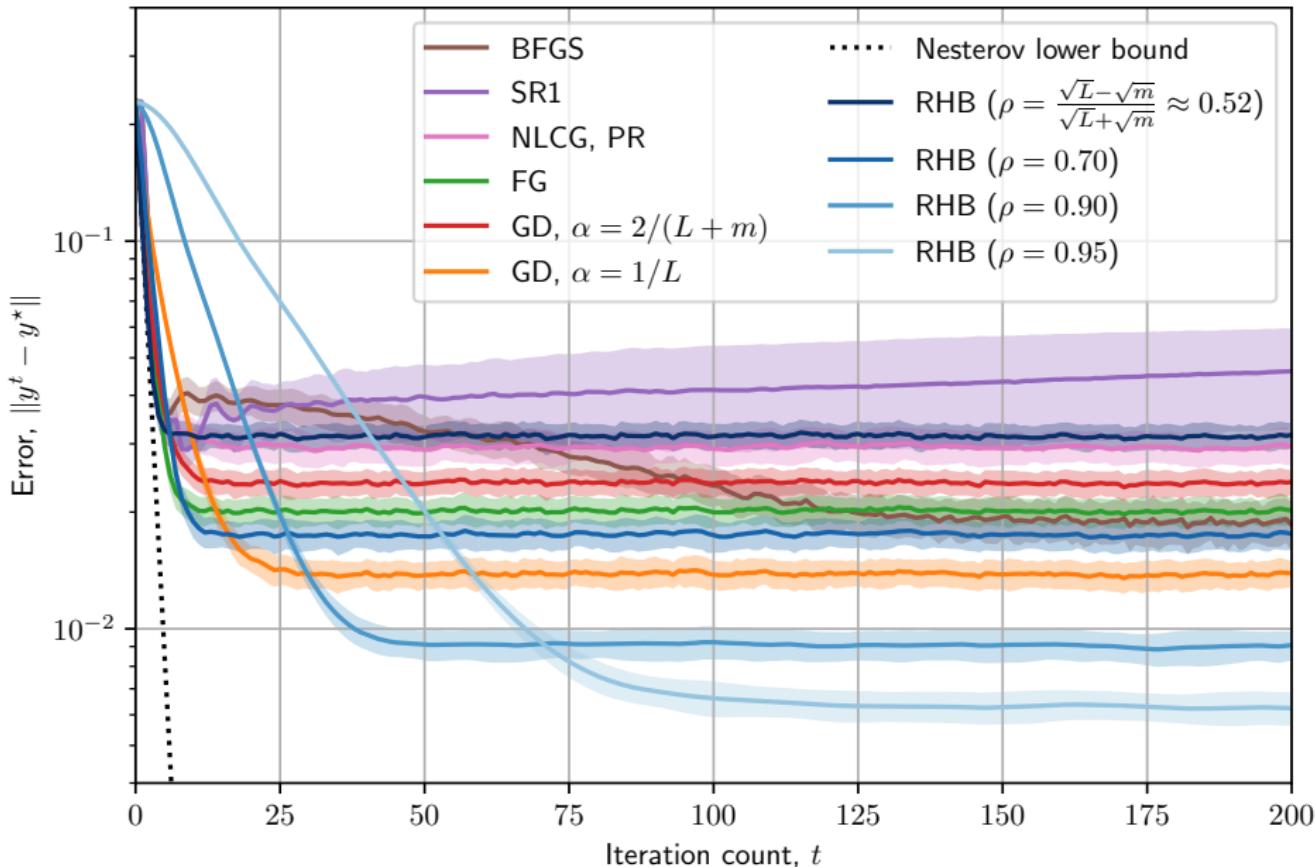
$$\|x_k - x^*\| \geq \left(\frac{\sqrt{L} - \sqrt{m}}{\sqrt{L} + \sqrt{m}} \right)^k \|x_0 - x^*\|$$

- Quasi-Newton methods (BFGS, SR1)
- Nonlinear conjugate gradient
- Fast Gradient, Heavy Ball, Gradient descent

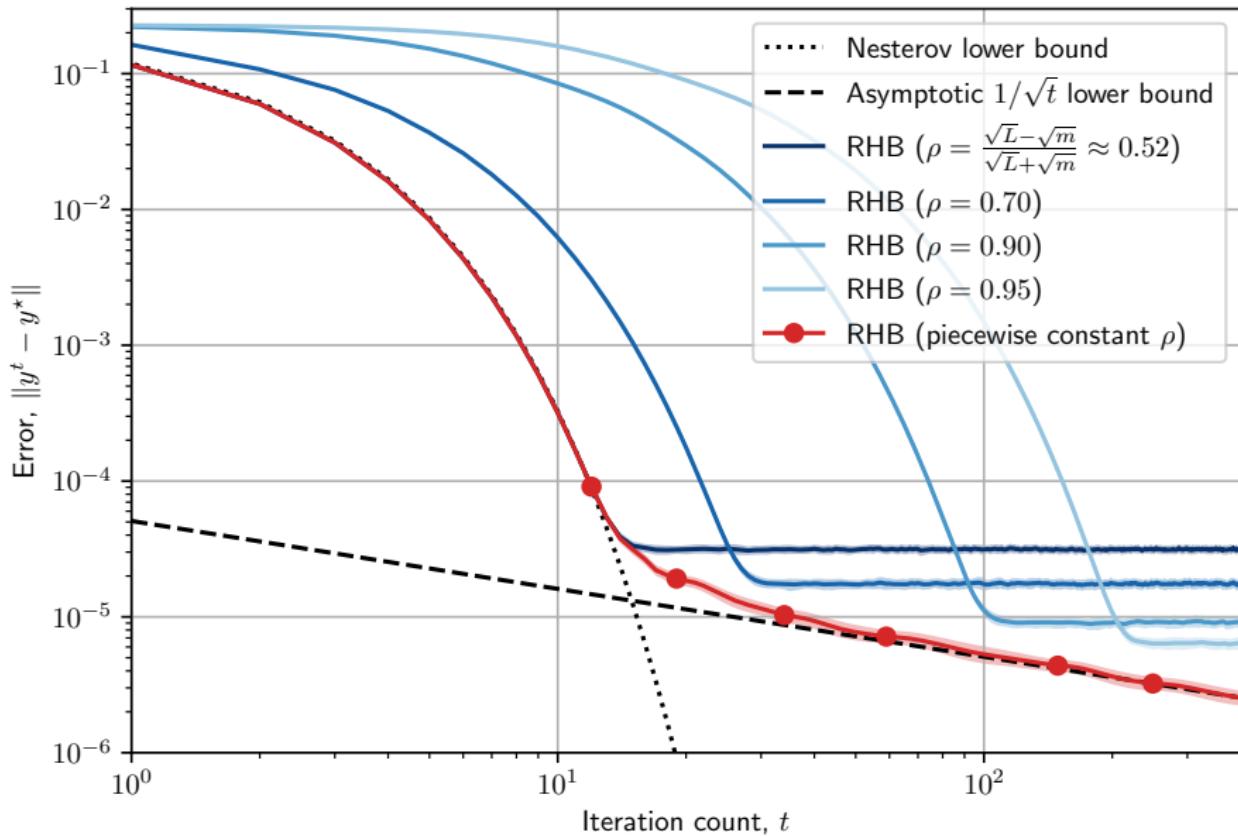
Nesterov worst-case, $d = 100$, $m = 1$, $L = 10$, $\sigma = 10^{-5}$.



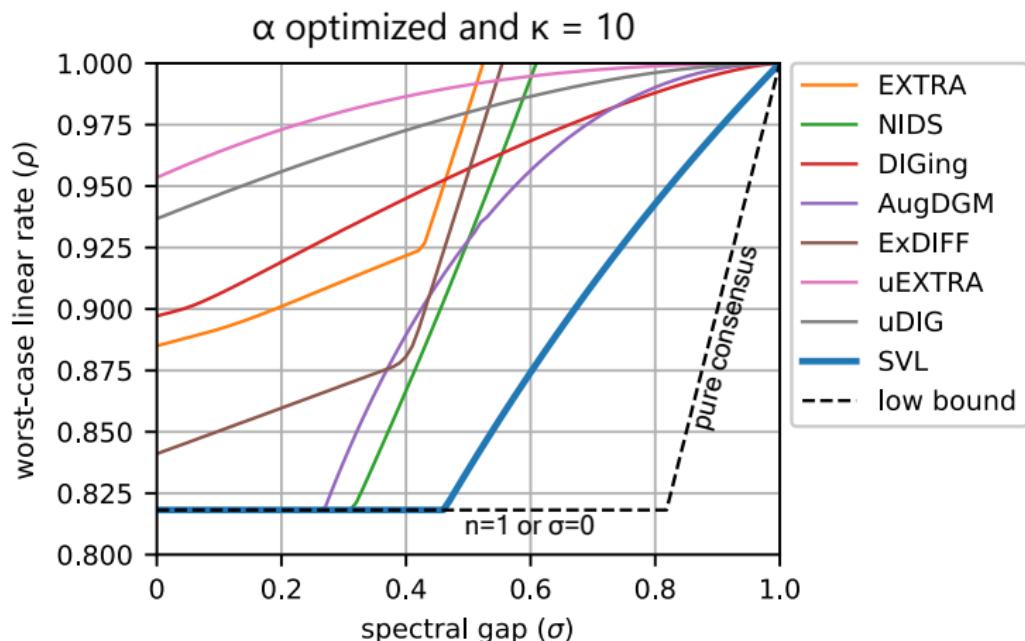
Nesterov worst-case, $d = 100$, $m = 1$, $L = 10$, $\sigma = 10^{-2}$.



Nesterov worst-case, $d = 100$, $m = 1$, $L = 10$, $\sigma = 10^{-5}$.
 Using piecewise constant ρ schedule.

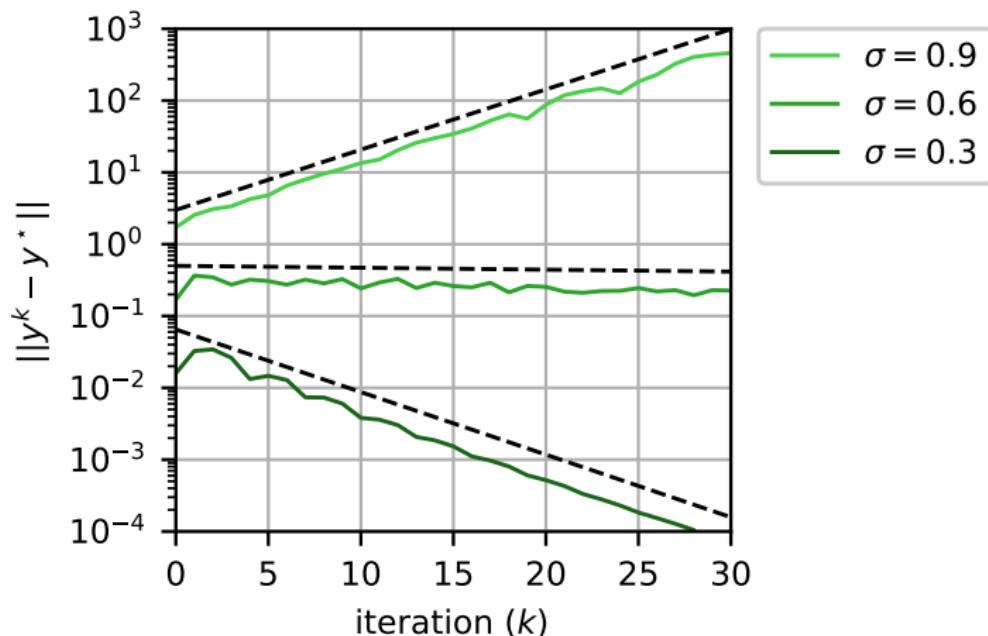


Algorithm comparison with SVL



Best worst-case performance for *any* known algorithm!

Worst-case trajectories: Tight bounds?



- NIDS with $n = 15$ agents, $d = 1$ dimensions, and $\kappa = 10$.
- Greedy heuristic: choose functions and graph at each iteration to maximize error (Requires solving a nonconvex QCQP).