Proximal Algorithms for Large-scale Convex Nonsmooth Optimization

Laurent Condat

King Abdullah University of Science and Technology (KAUST)
Thuwal, Saudi Arabia

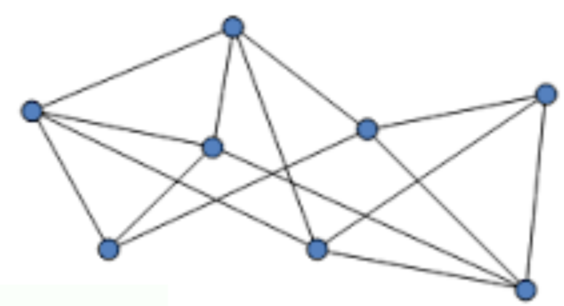
Feb. 14, 2022

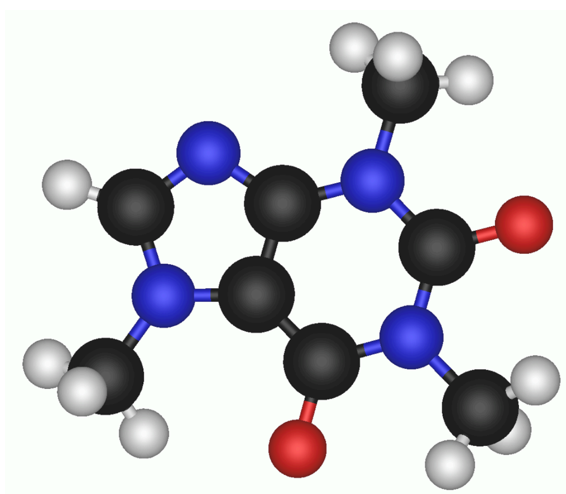
Optimization

Find $x^\star \in \underset{x\in\mathcal{X}}{\arg\min}\ \Psi(x)$

$\mathcal{X}$ is a real Hilbert space:

$$\begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \end{bmatrix}$$





Optimization

$$\text{Find } x^\star \in \underset{x\in\mathcal{X}}{\arg\min}\ \Psi(x) = \sum_{m=1}^{M} g_m(L_m x)$$

with

- linear operators $L_m : \mathcal{X} \to \mathcal{U}_m$
- real Hilbert spaces $\mathcal{X}, \mathcal{U}_m$
- functions g_m

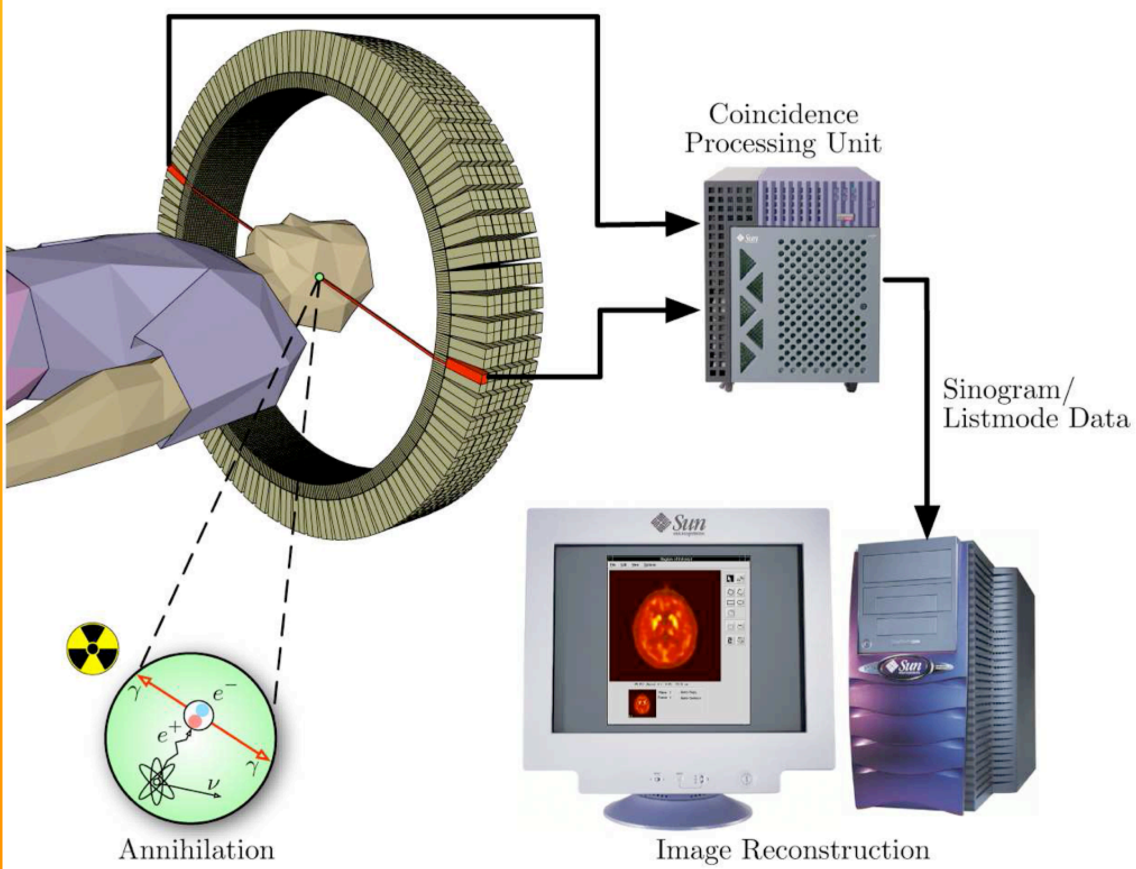
Optimization

$$\text{Find } x^\star \in \underset{x\in\mathcal{X}}{\arg\min}\ \Psi(x) = \sum_{m=1}^{M} g_m(L_m x)$$

Example: $\Psi(x) = \|Ax - y\|^2 + \|x\|_1$

Motivation: image processing

Given data $y \approx Ax^\sharp$

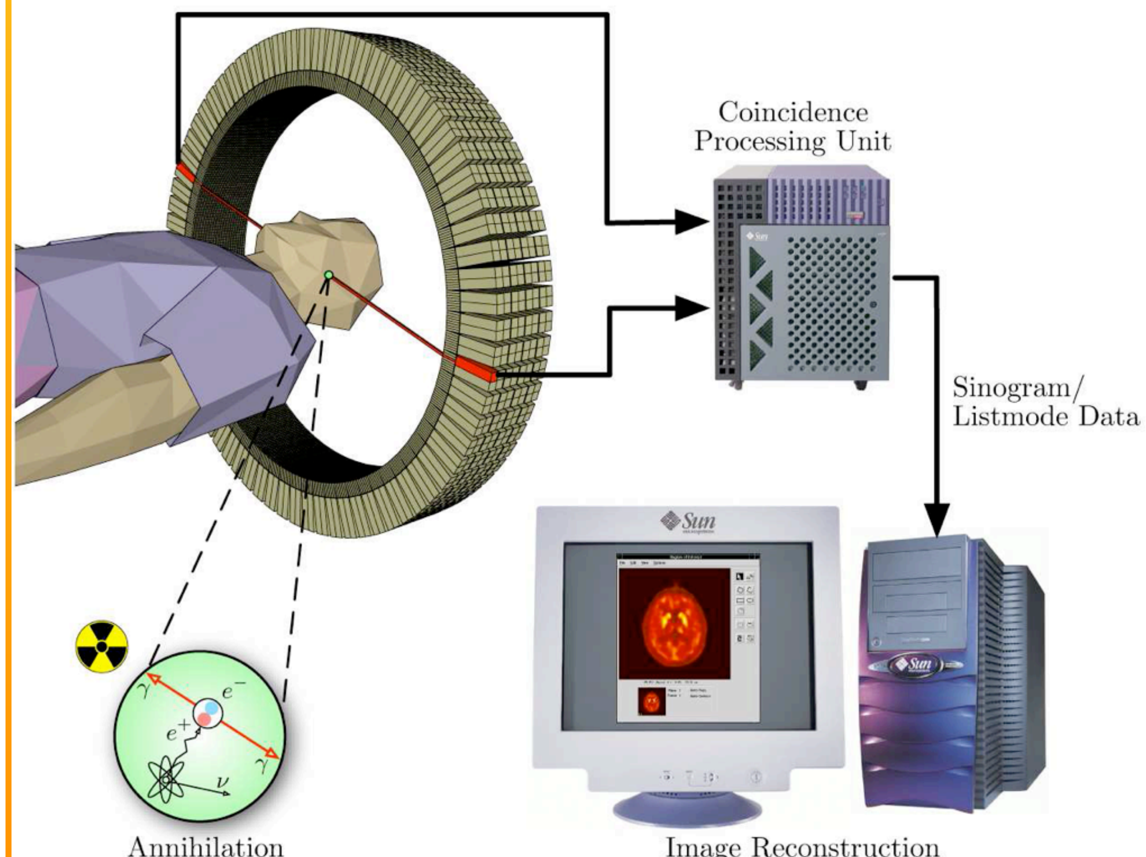


Motivation: image processing

Given data $\quad y \approx Ax^\sharp$

estimate the unknown image $x^\sharp$ by solving

$$\text{Find} \quad x^\star \in \underset{x \in \mathbb{R}^{N_1 \times N_2}}{\arg\min} \Big\{ D(Ax, y) + R(x) \Big\}$$

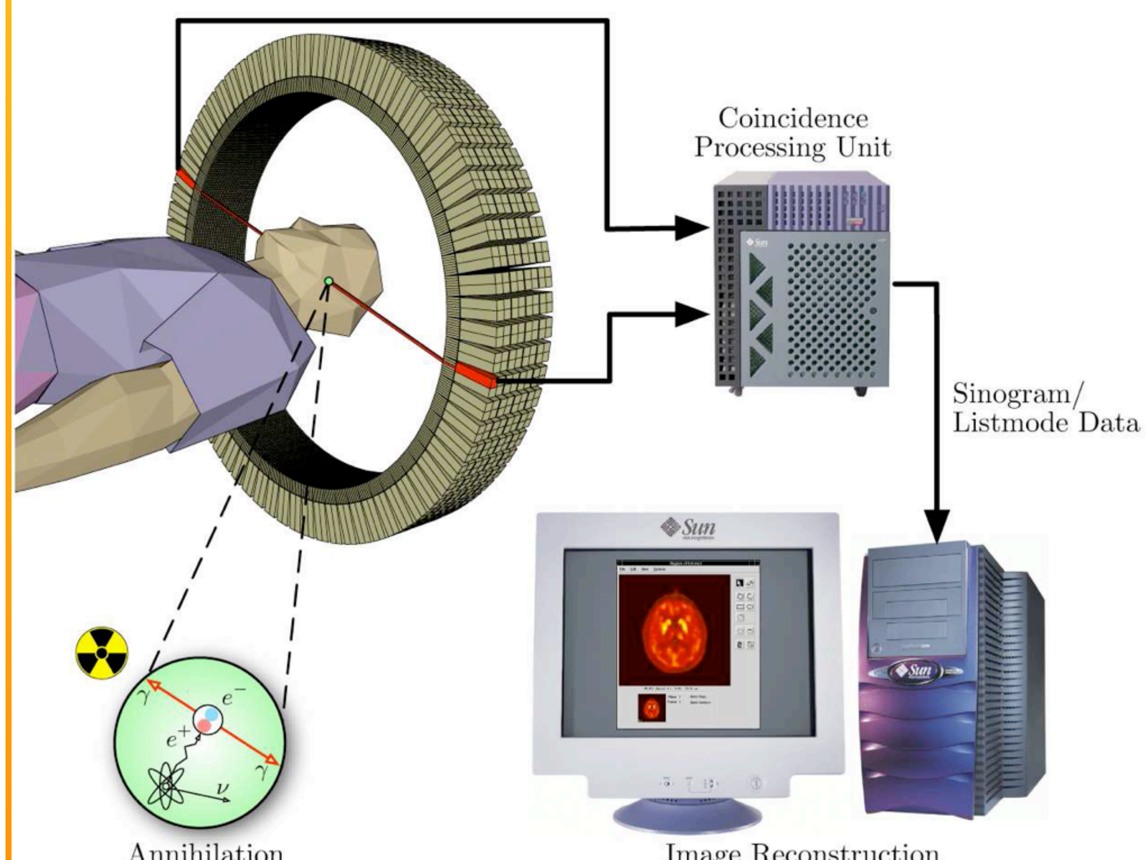


Motivation: image processing

Given data $\quad y \approx Ax^\sharp$

estimate the unknown image $x^\sharp$ by solving

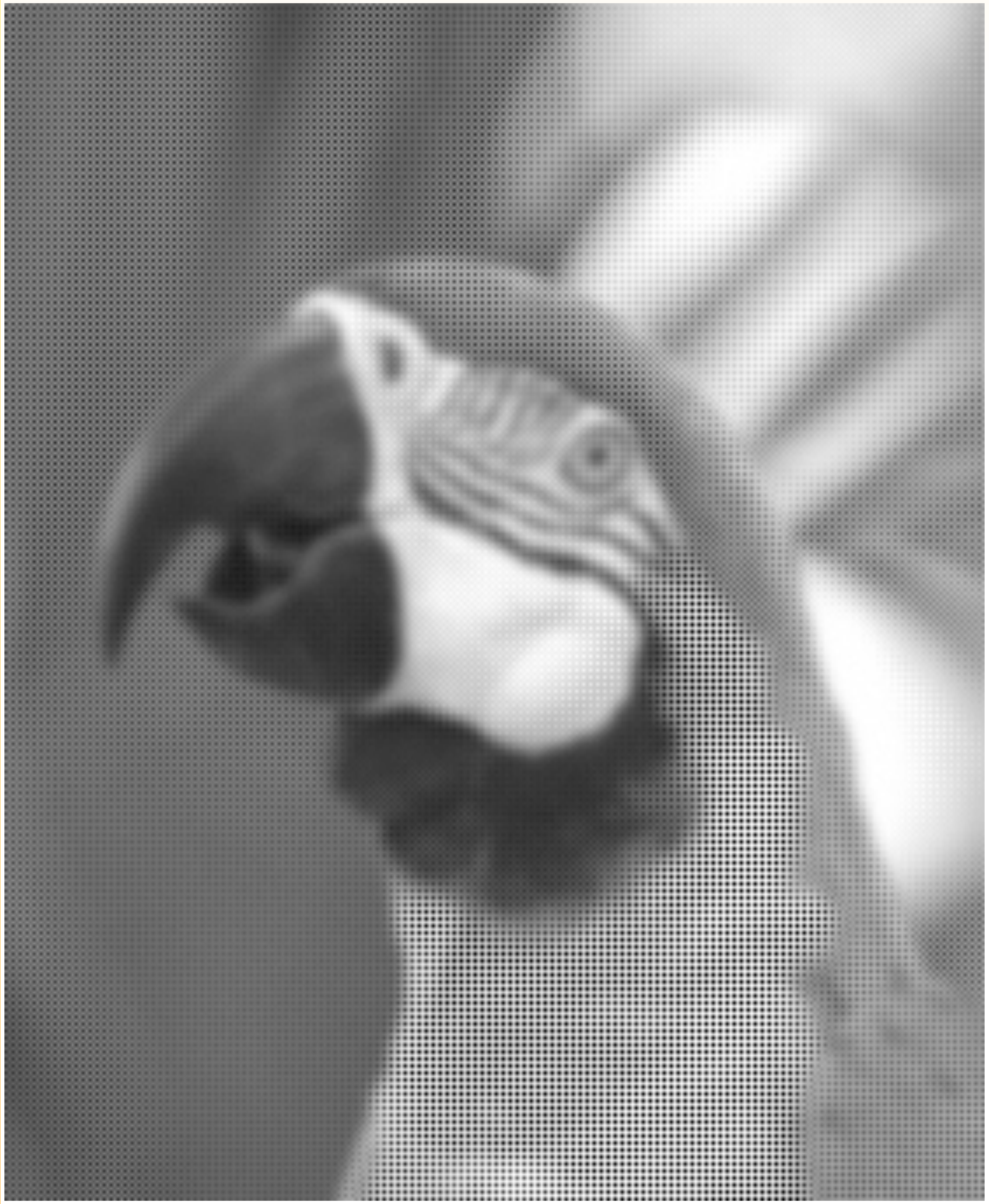
$$\text{Find } \; x^\star \in \underset{x \in \mathbb{R}^{N_1 \times N_2}}{\arg\min} \left\{ D(Ax, y) + R(x) \right\}$$

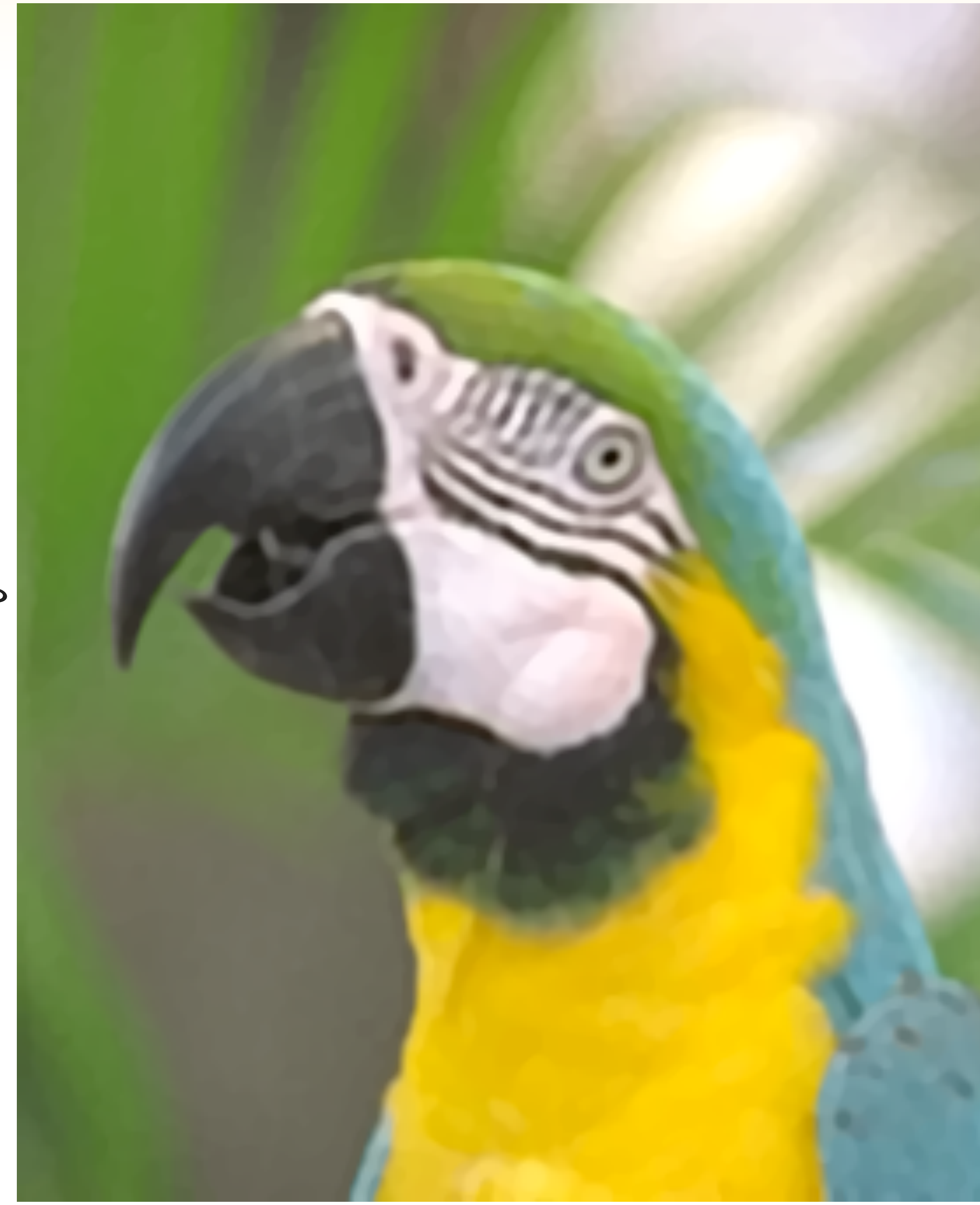


LC, "Discrete total variation: New definition and minimization", 2017

LC, "A generic proximal algorithm...", 2014

Example: joint demosaicking-deblurring



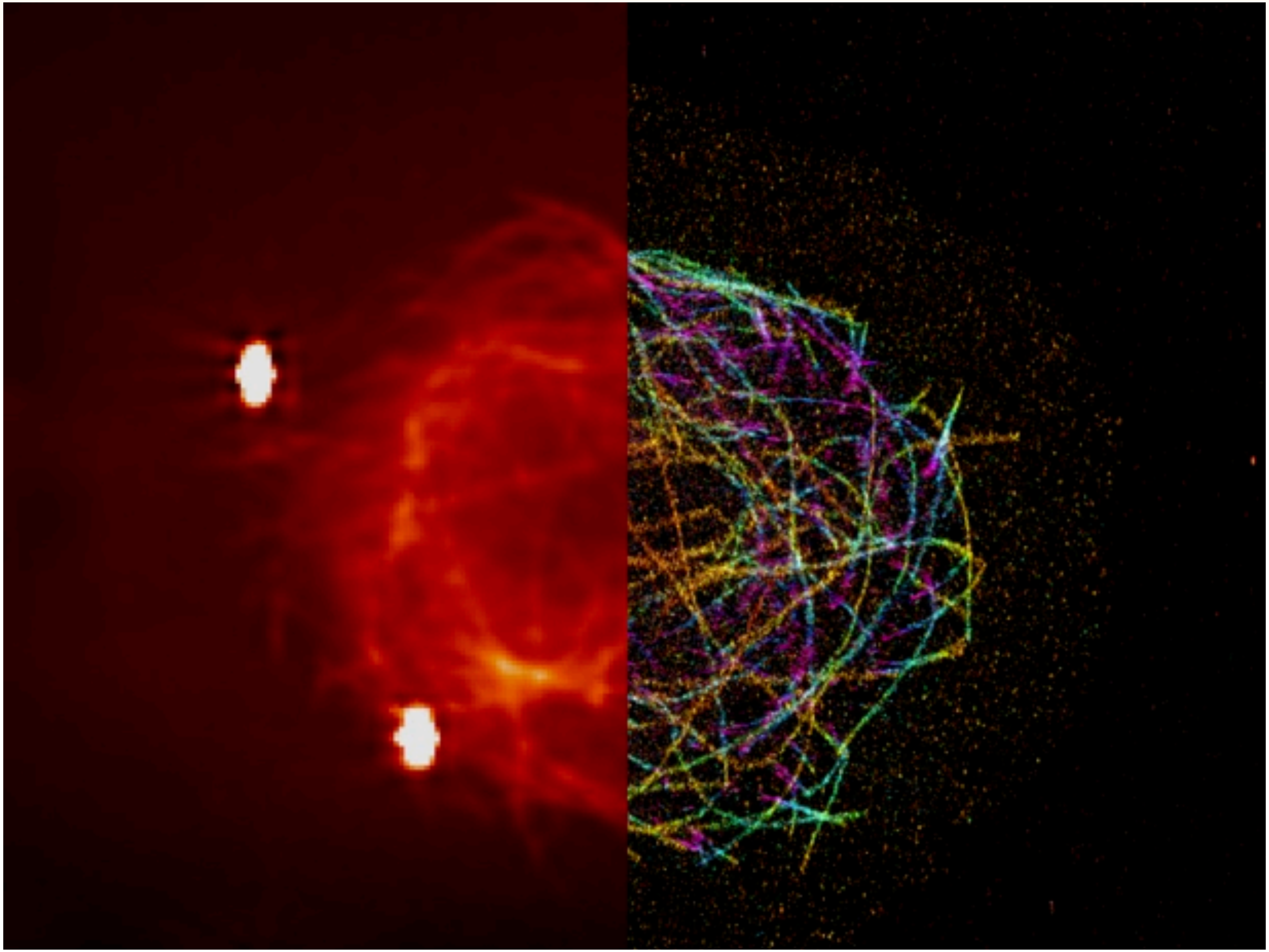


LC, "A simple, fast and efficient approach to denoisaicking...", 2010

Example: pansharpening/fusion

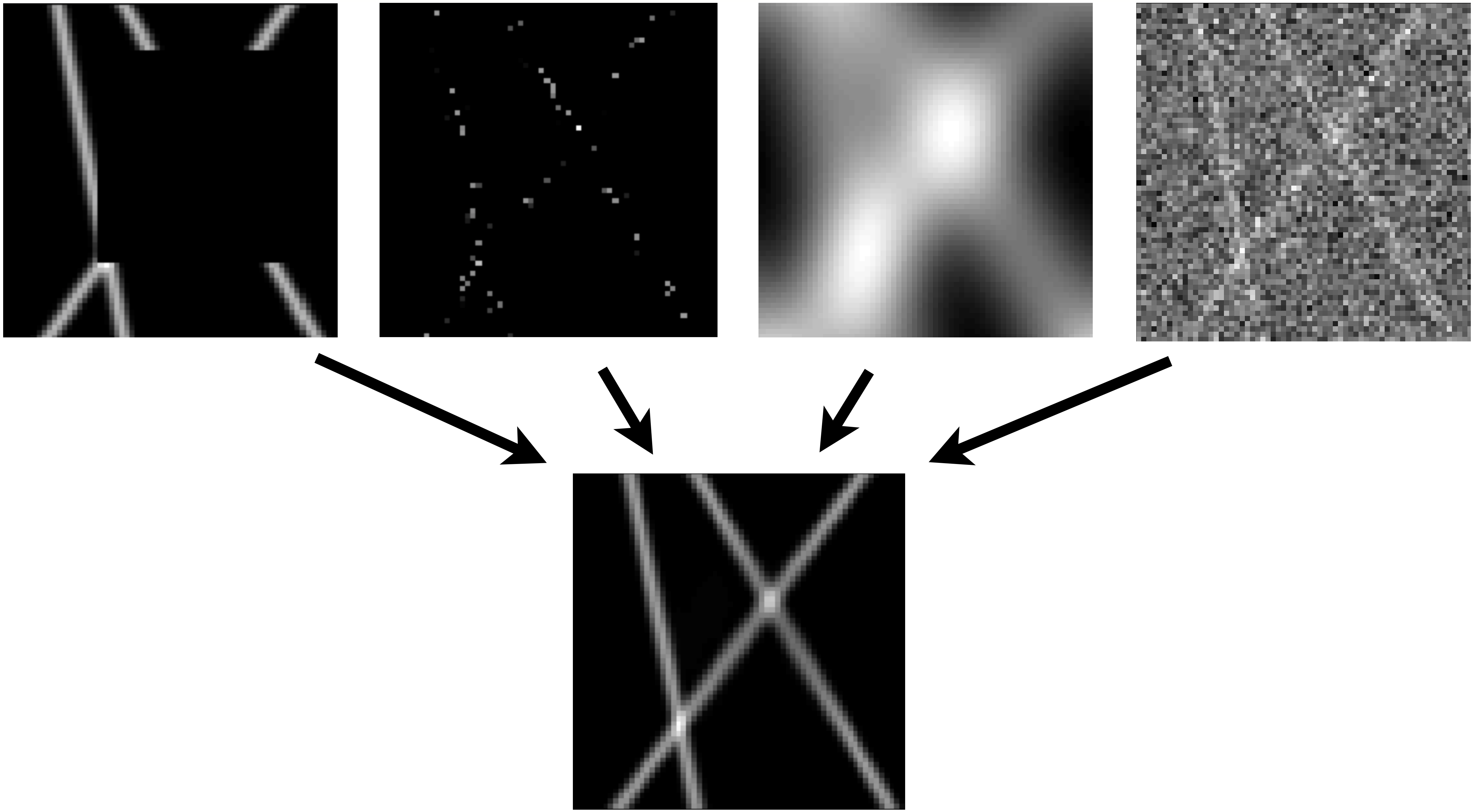
He, LC et al., "A new pansharpening method...", 2014

Super-resolution



regular PALM [Galbraith et al.]

Example: super-resolution of lines



Polisano, LC et al., "A convex approach to superresolution...", 2019

Convex optimization

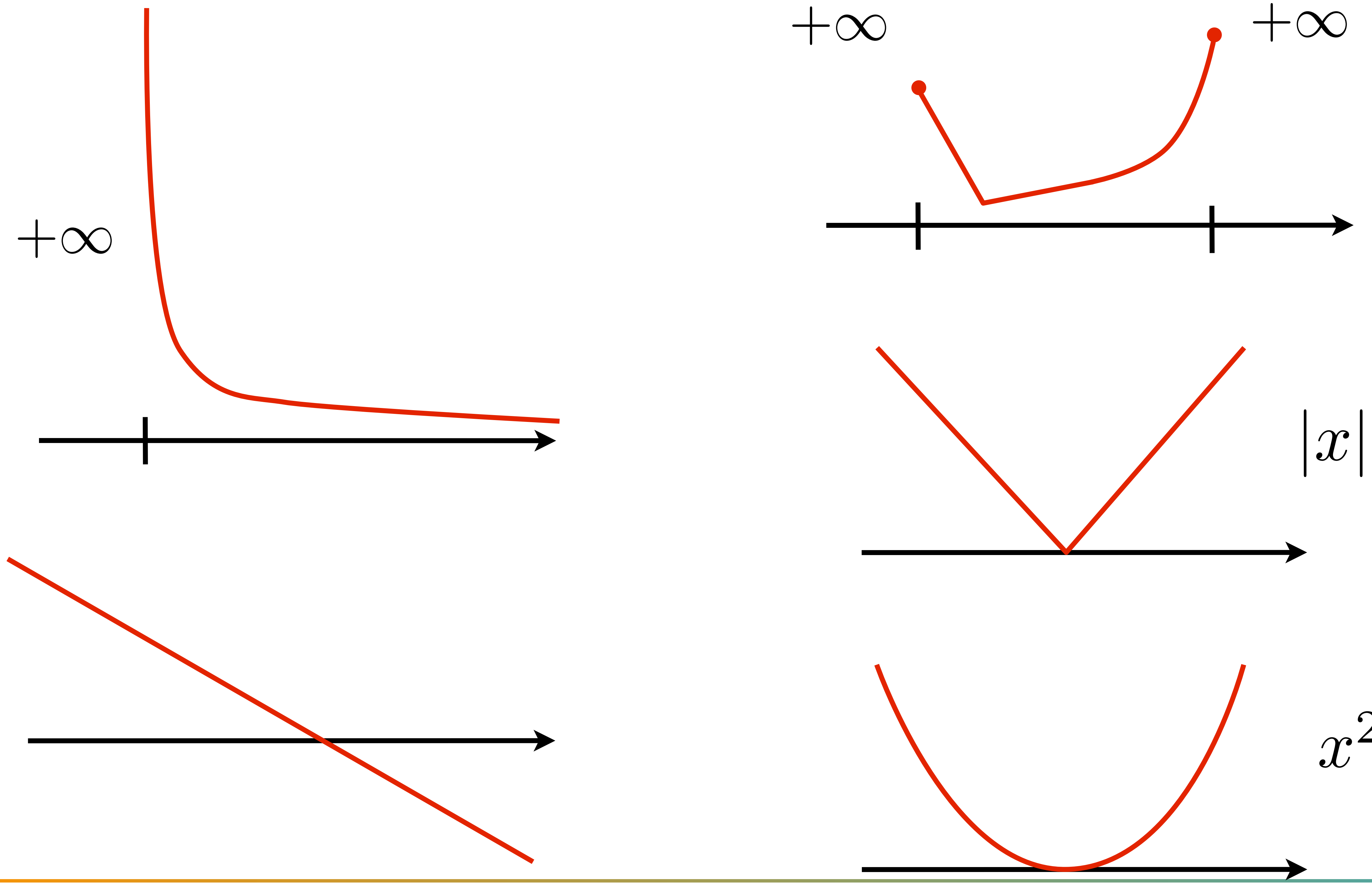
$$\text{Find } x^\star \in \arg\min_{x\in\mathcal{X}} \; \Psi(x) = \sum_{m=1}^{M} g_m(L_m x)$$

with

- linear operators $L_m : \mathcal{X} \to \mathcal{U}_m$
- real Hilbert spaces $\mathcal{X}, \mathcal{U}_m$
- **convex** functions $g_m : \mathcal{U}_m \to \mathbb{R} \cup \{+\infty\}$

Convex functions

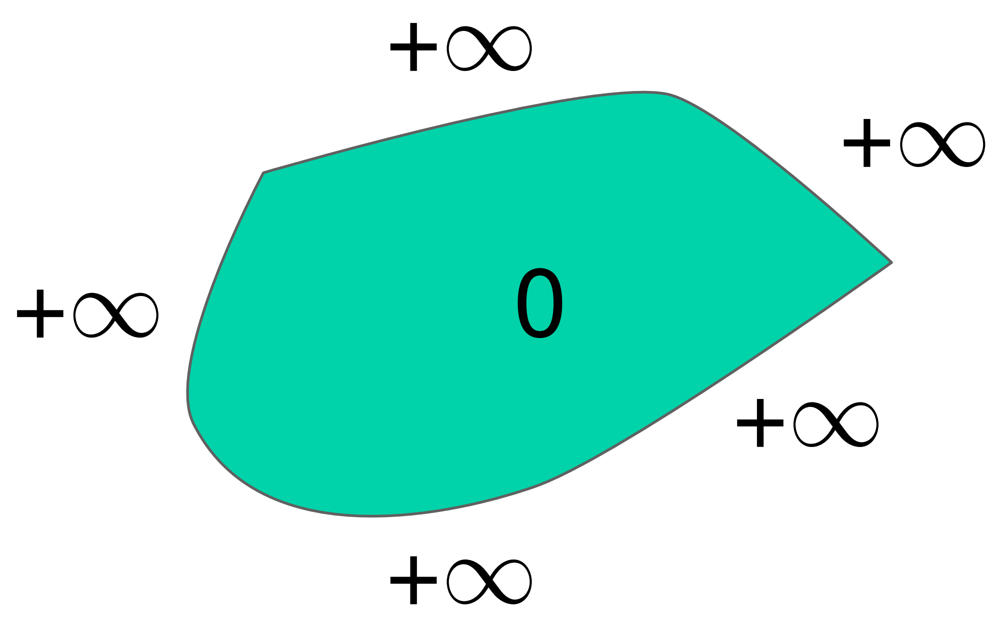
Some convex functions: $\mathbb{R} \to \mathbb{R} \cup \{+\infty\}$



Indicator functions

For a closed convex set $\Omega \subset \mathcal{X}$, its **indicator function** is

$$\imath_\Omega(x) = \begin{cases} 0 & \text{if } x \in \Omega, \\ +\infty & \text{else.} \end{cases}$$



$\imath_\Omega$ is convex.

Constrained optimization

$$\text{Find } x^\star \in \underset{x\in\mathcal{X}}{\arg\min}\ \Psi(x) = \sum_{m=1}^{M} g_m(L_m x)$$

encompasses the presence of constraints:

$$\underset{x\in\Omega}{\text{minimize}}\ f(x) \quad\equiv\quad \underset{x\in\mathcal{X}}{\text{minimize}}\ f(x)\ \text{ s.t. } x \in \Omega$$

$$\equiv\quad \underset{x\in\mathcal{X}}{\text{minimize}}\ f(x) + \imath_\Omega(x)$$

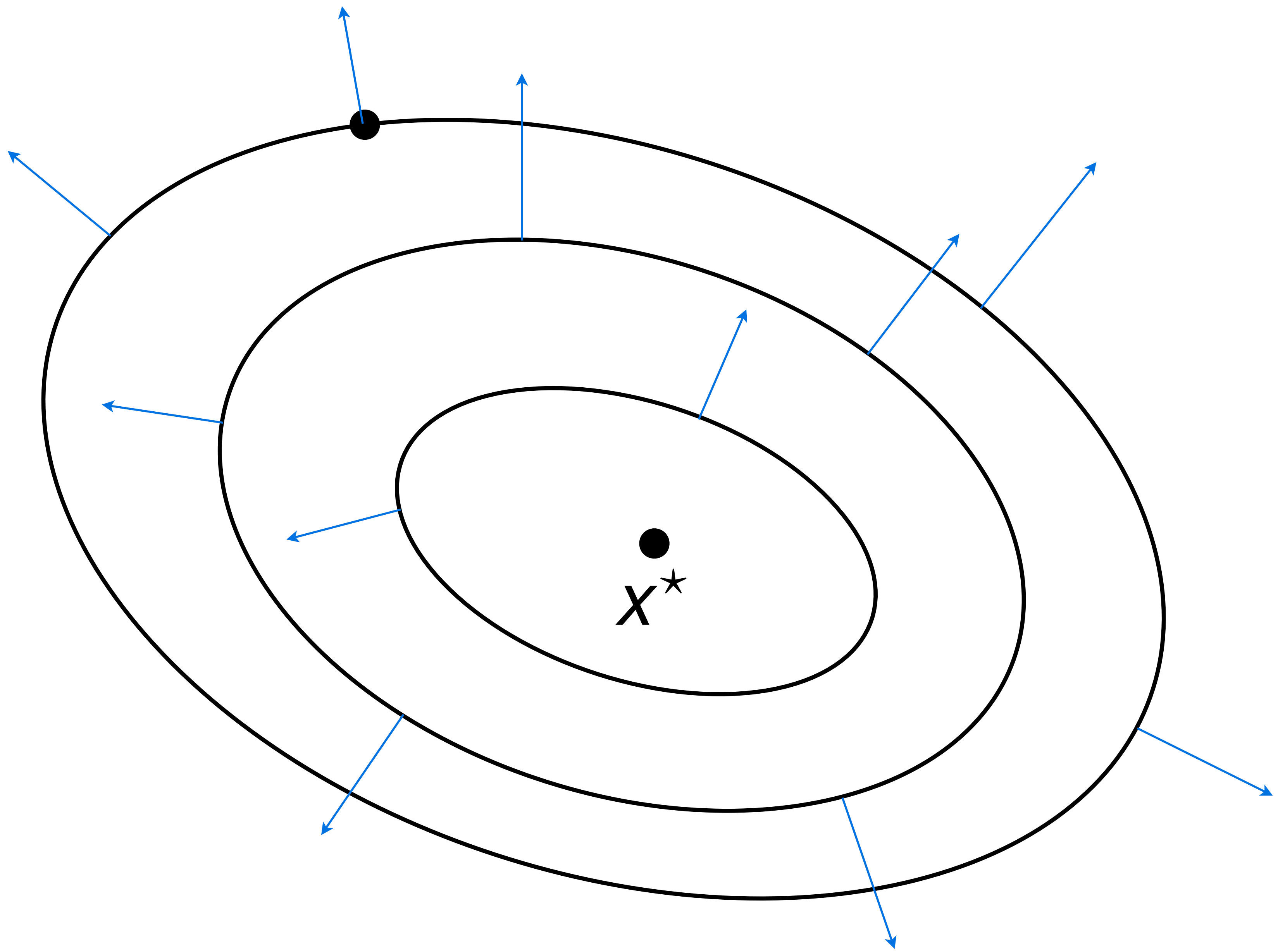
Optimization algorithms

$$\text{Find } x^\star \in \underset{x\in\mathcal{X}}{\arg\min}\ \ \Psi(x) = \sum_{m=1}^{M} g_m(L_m x)$$

☞ iterative algorithm computing
$x^{k+1} = T(x^k)$
with x^k converging to $x^\star = T(x^\star)$

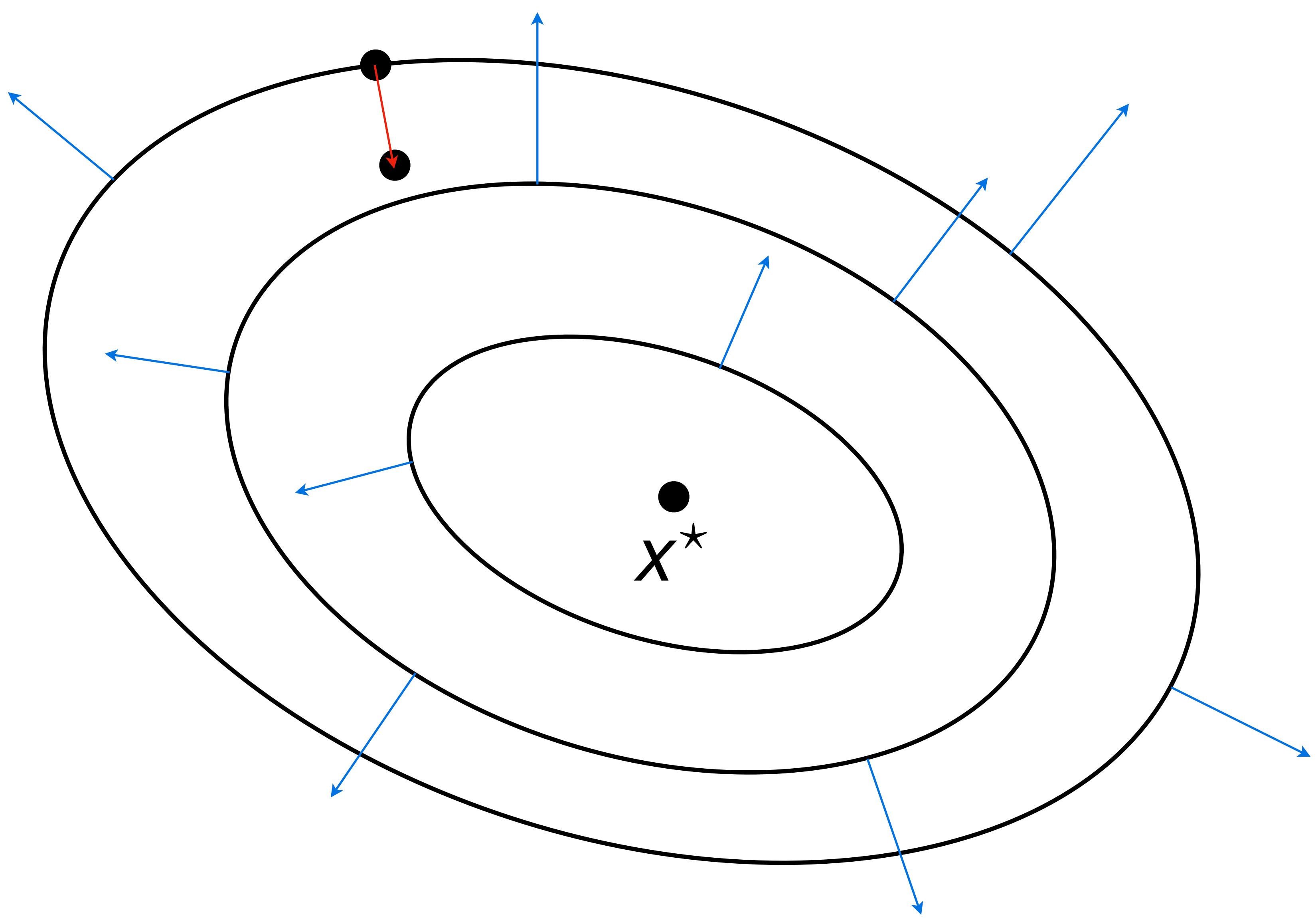
Smooth minimization: gradient descent

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



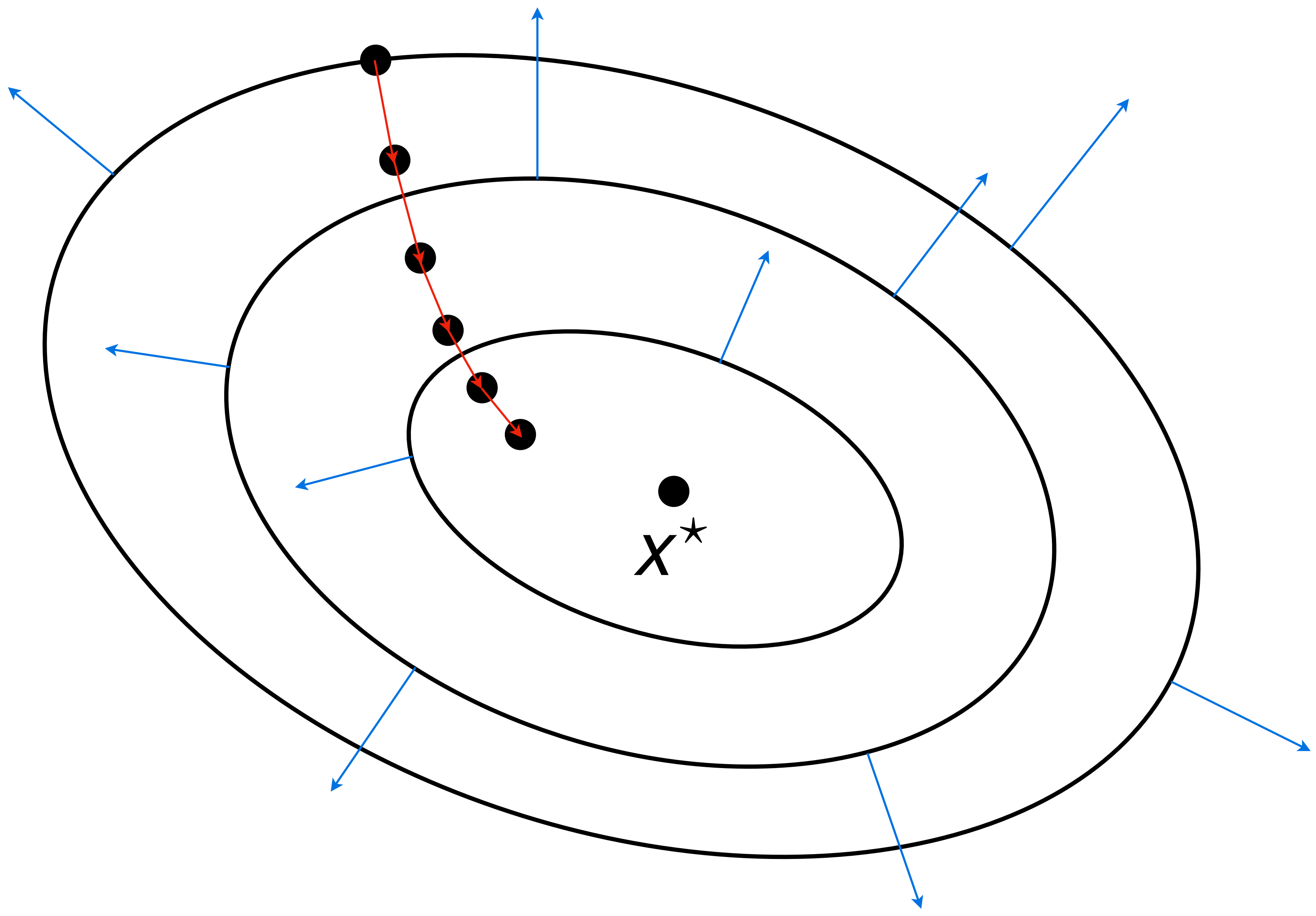
Smooth minimization: gradient descent

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

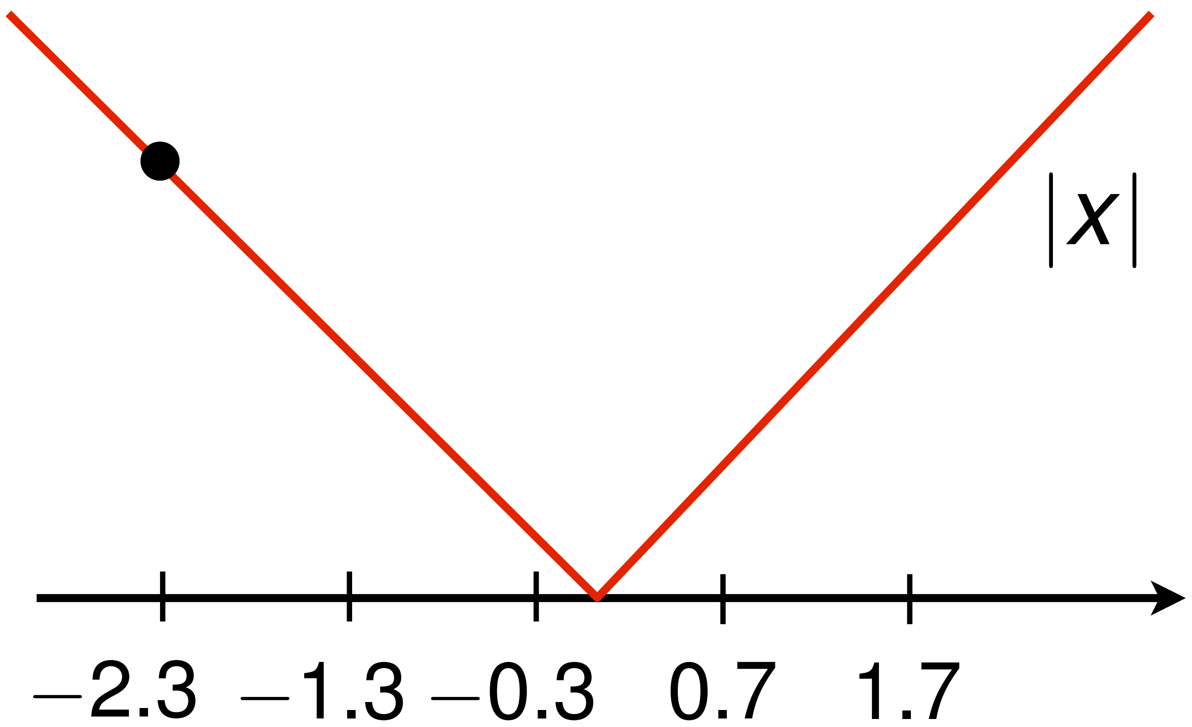


Smooth minimization: gradient descent

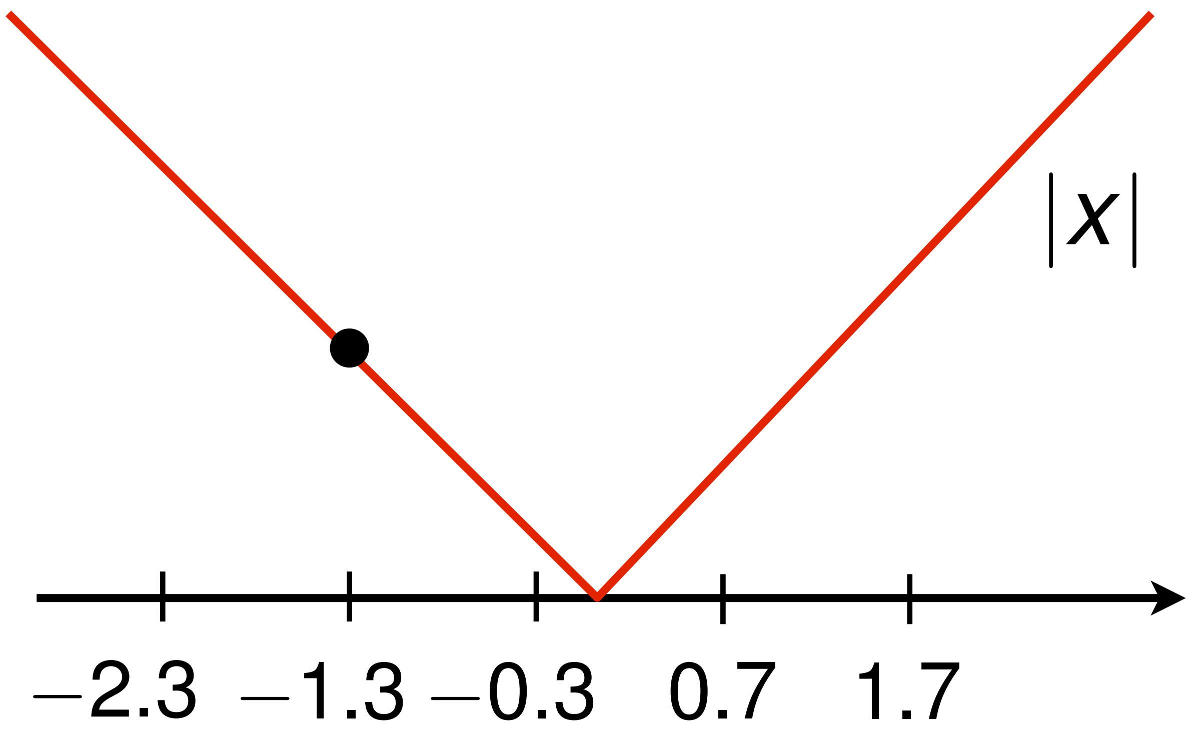
$$x^{k+1} = x^k - \gamma \nabla f(x^k)$$



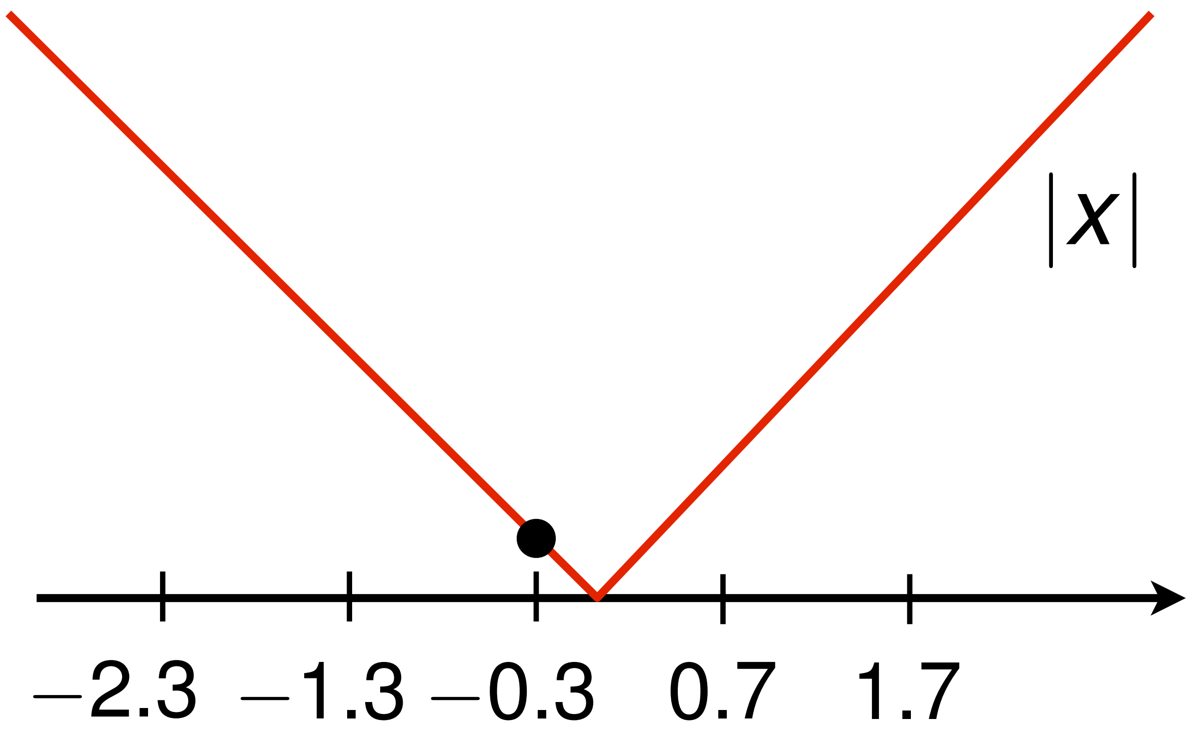
Nonsmooth minimization?



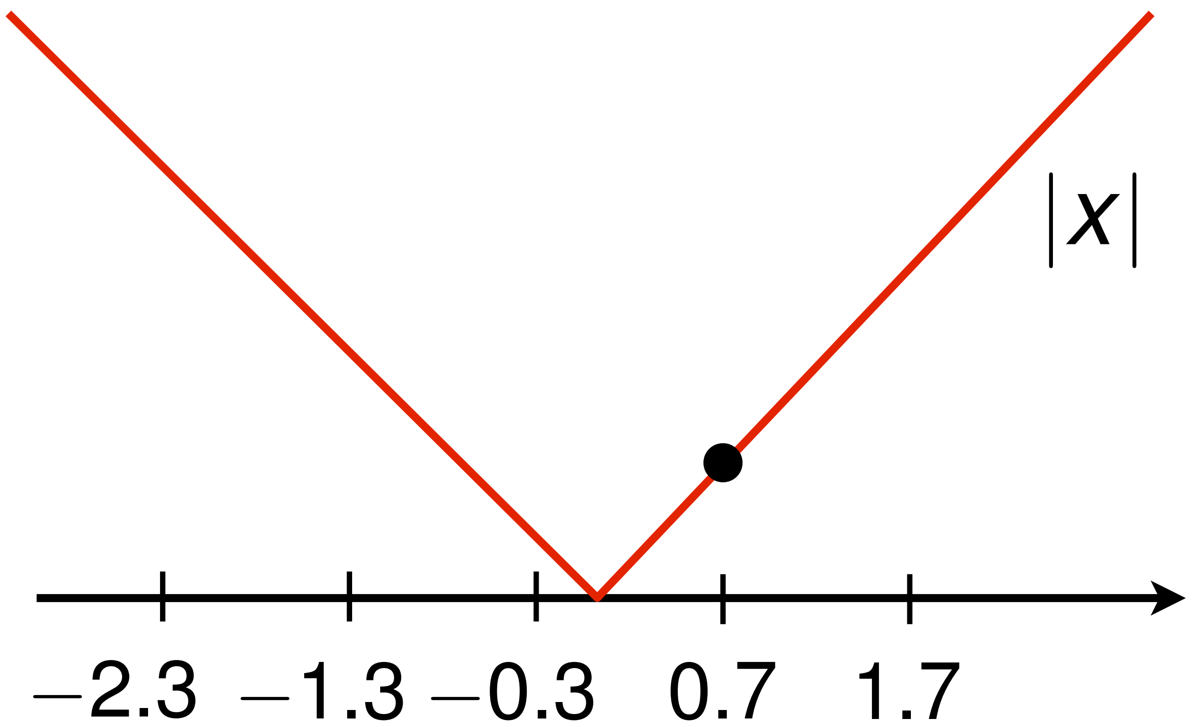
Nonsmooth minimization?



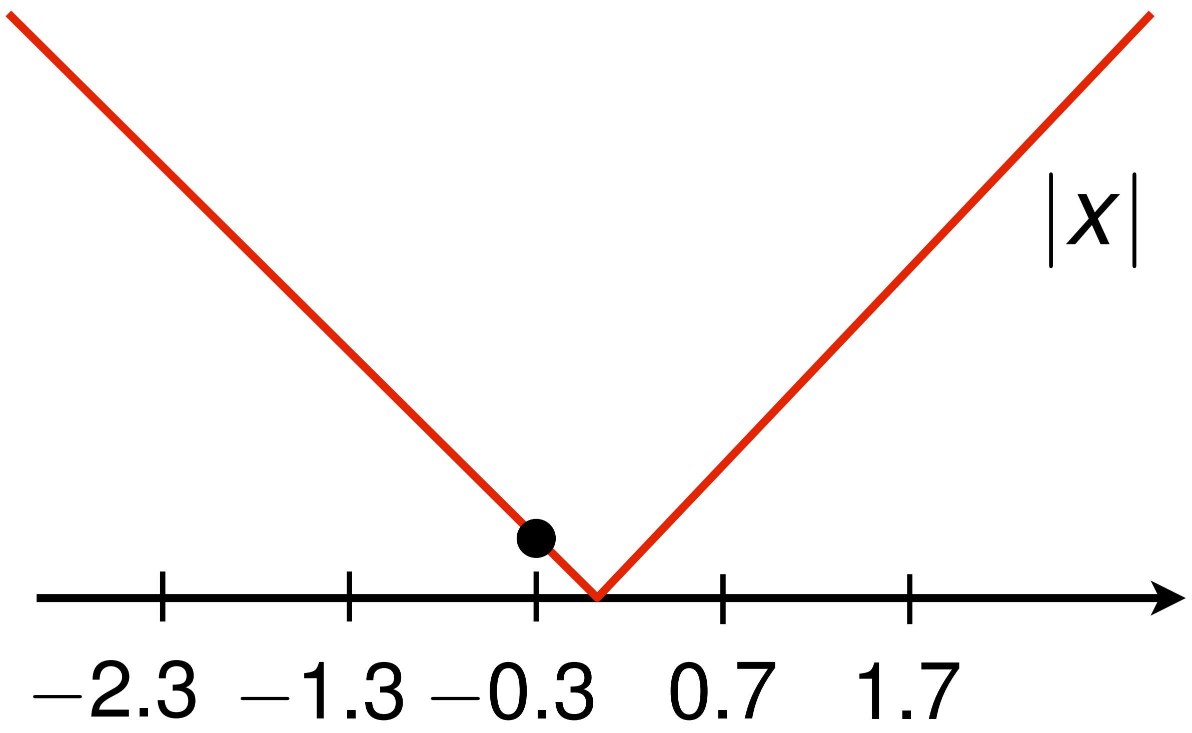
Nonsmooth minimization?



Nonsmooth minimization?

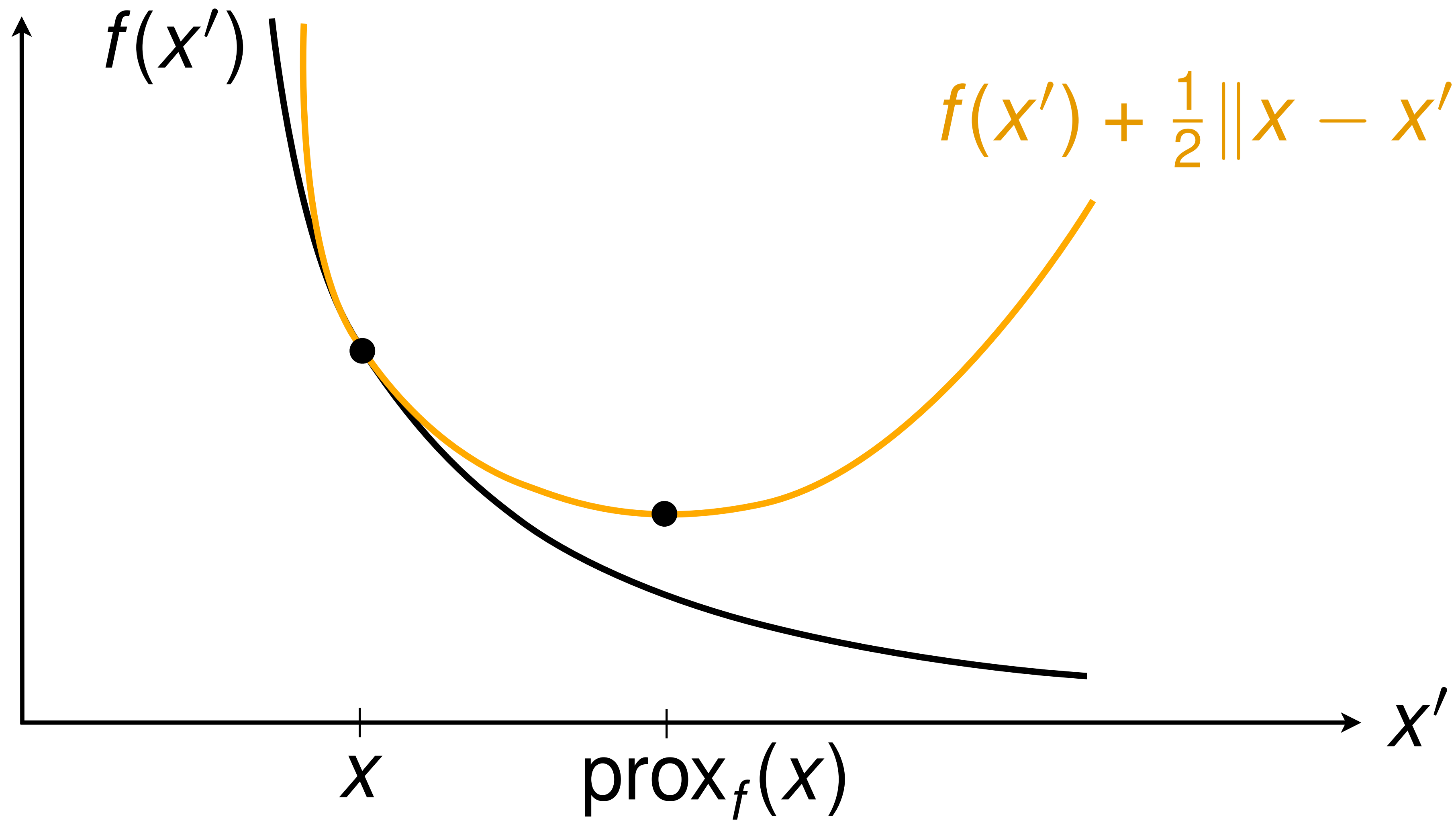


Nonsmooth minimization?



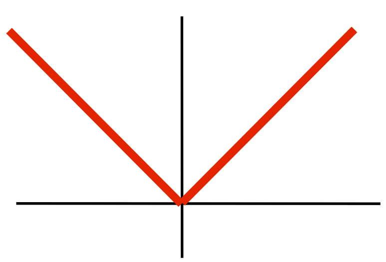
The proximity operator

$$\mathrm{prox}_f : \mathcal{H} \to \mathcal{H} : x \mapsto \operatorname*{arg\,min}_{x' \in \mathcal{H}} f(x') + \tfrac{1}{2}\|x - x'\|^2$$



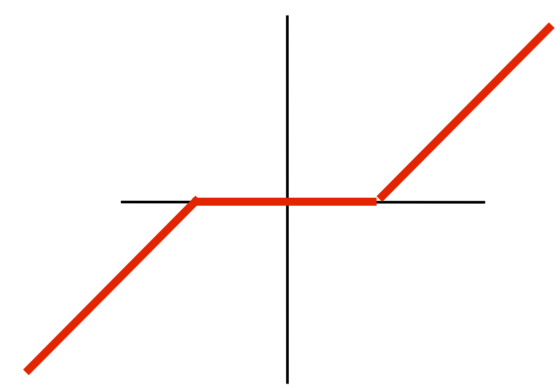
The proximity operator

$$\mathrm{prox}_f : \mathcal{H} \to \mathcal{H} : x \mapsto \underset{x' \in \mathcal{H}}{\arg\min}\, f(x') + \tfrac{1}{2}\|x - x'\|^2$$



$f(x) = |x|$

☞

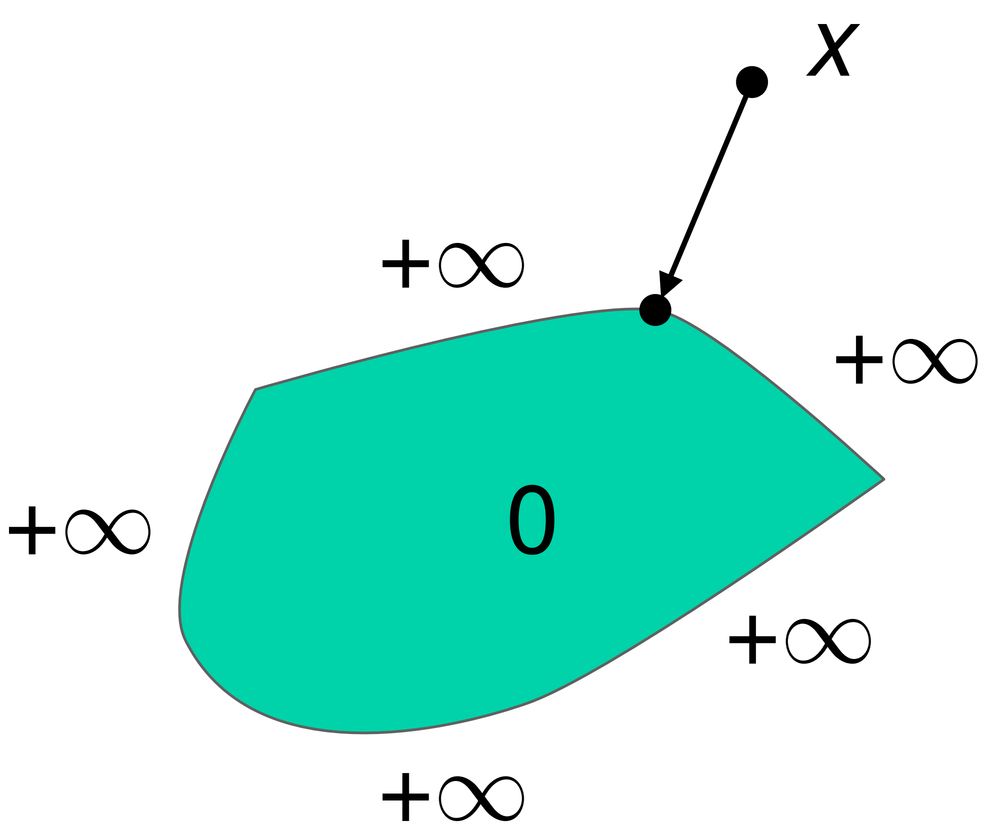


$\mathrm{prox}_f(x) = \mathrm{sgn}(x) \max(|x| - 1, 0)$

The proximity operator

$$\mathrm{prox}_f : \mathcal{H} \rightarrow \mathcal{H} : x \mapsto \underset{x' \in \mathcal{H}}{\arg\min}\, f(x') + \tfrac{1}{2}\|x - x'\|^2$$

$$\mathrm{prox}_{\imath_\Omega} = \mathrm{proj}_\Omega$$



The proximity operator

Exact, finite time, algorithms are available to compute the proximity operator of:

- $\|X\|_* \to$ SVD, $O(N^3)$
- 1-D TV $\to$ taut-string alg., $O(N)$
- 2-D anisotropic TV $\to$ graph cuts
- proj. onto the simplex $\to$ $O(N)$

...

LC, "A direct algorithm for 1-D total variation...", 2013

LC, "Fast projection onto the simplex...", 2016

Pustelnik, LC, "Proximity operator of a sum...", 2017

Proximal splitting algorithms

$$\text{Find } x^\star \in \operatorname*{arg\,min}_{x \in \mathcal{X}} \sum_{m=1}^{M} g_m(L_m x)$$

No easy form of $\mathrm{prox}_{g_1+g_2}$ or $\mathrm{prox}_{g \circ L}$

Proximal splitting algorithms

$$\text{Find } x^\star \in \arg\min_{x\in\mathcal{X}} \sum_{m=1}^{M} g_m(L_m x)$$

☞ We want full splitting, with individual activation of L_m, L_m^*, the gradient or proximity operator of g_m.

only fast operations in $\widetilde{\mathcal{O}}(N)$, with $N =$ dimension

⚠ typically, $N \sim 10^6$–10^9

Proximal splitting algorithms

$$\text{Find } x^\star \in \arg\min_{x \in \mathcal{X}} \left(f(x) + \sum_{m=1}^{M} g_m(L_m x) + h(x) \right)$$

with:

- h smooth with β-Lipschitz continuous gradient
  $\rightarrow$ calls to ∇h
- simple functions f and g_m $\rightarrow$ calls to $\text{prox}_{\gamma f}$ and $\text{prox}_{\sigma_m g_m}$
- calls to L_m, L_m^*

Product space trick

$$\text{Find } x^\star \in \operatorname*{arg\,min}_{x \in \mathcal{X}} \Big(f(x) + g(Lx) + h(x) \Big)$$

$$\boldsymbol{g}(\boldsymbol{u}) = \sum_{m=1}^{M} g_m(u_m)$$

$$\boldsymbol{L}x = (L_1 x, \dots, L_m x)$$

☞ $$\boldsymbol{g}(\boldsymbol{L}x) = \sum_{m=1}^{M} g_m(L_m x)$$

Minimization of 3 functions

$$\text{Find } x^\star \in \operatorname*{arg\,min}_{x \in \mathcal{X}} \Big(f(x) + g(Lx) + h(x) \Big)$$

with:

- h smooth with β-Lipschitz continuous gradient
  $\rightarrow$ calls to ∇h
- simple functions f and g $\rightarrow$ calls to $\mathrm{prox}_{\gamma f}$ and $\mathrm{prox}_{\sigma g}$
- calls to L, L^*

Minimization of 3 functions

minimize $f + g \circ L + h$

	$f + h$	☞ forward-backward alg.
1979	$f + g$	☞ Douglas-Rachford alg. = ADMM
2011	$f + g \circ L$	☞ Chambolle-Pock alg.
2013	$f + g \circ L + h$	☞ LC, "A primal-dual splitting method for convex optimization...", 2013 Vu, "A splitting algorithm for dual monotone inclusions...", 2013

Minimization of 3 functions

minimize $f + g \circ L + h$

	$f + h$	forward-backward alg.
1979	$f + g$	Douglas-Rachford alg. = ADMM
2011	$f + g \circ L$	Chambolle-Pock alg.
2011	$g \circ L + h$	Loris-Verhoeven alg.

Combettes, LC, Pesquet, Vu, "A forward-backward view of some primal-dual optimization...", 2014

Minimization of 3 functions

minimize $f + g \circ L + h$

	$f + h$	☞	forward-backward alg.
1979	$f + g$	☞	Douglas-Rachford alg. = ADMM
2011	$f + g \circ L$	☞	Chambolle-Pock alg.
2011	$g \circ L + h$	☞	Loris-Verhoeven alg.
2013	$f + g \circ L + h$	☞	Condat-Vu alg.
2018		☞	PD3O alg. (Yan)
2020		☞	PDDY alg.

Salim, LC et al., "Dualize, split, randomize...", 2020

4 Primal-dual algorithms

minimize

$$f + g \circ L + h$$

Condat–Vu algorithm form I

$$\left\lfloor \begin{array}{l} x^{k+1} = \mathrm{prox}_{\gamma f}\big(x^k - \gamma \nabla h(x^k) - \gamma L^* u^k\big) \\ u^{k+1} = \mathrm{prox}_{\sigma g^*}\big(u^k + \sigma L(2x^{k+1} - x^k)\big) \end{array} \right.$$

Condat–Vu algorithm form II

$$\left\lfloor \begin{array}{l} u^{k+1} = \mathrm{prox}_{\sigma g^*}\big(u^k + \sigma L x^k\big) \\ x^{k+1} = \mathrm{prox}_{\gamma f}\big(x^k - \gamma \nabla h(x^k) - \gamma L^*(2u^{k+1} - u^k)\big) \end{array} \right.$$

PD3O algorithm

$$\left\lfloor \begin{array}{l} x^{k+1} = \mathrm{prox}_{\gamma f}\big(x^k - \gamma \nabla h(x^k) - \gamma L^* u^k\big) \\ u^{k+1} = \mathrm{prox}_{\sigma g^*}\big(u^k + \sigma L(2x^{k+1} - x^k - \gamma \nabla h(x^{k+1}) + \gamma \nabla h(x^k))\big) \end{array} \right.$$

PDDY algorithm

$$\left\lfloor \begin{array}{l} u^{k+1} = \mathrm{prox}_{\sigma g^*}\big(u^k + \sigma L x^k\big) \\ x^{k+1} = \mathrm{prox}_{\gamma f}\big(x^k - \gamma \nabla h(x^k - \gamma L^*(u^{k+1} - u^k)) - \gamma L^*(2u^{k+1} - u^k)\big) \end{array} \right.$$

4 Primal-dual algorithms

Condat–Vu algorithm form I

$$\left\lfloor \begin{array}{l} x^{k+1} = \mathrm{prox}_{\gamma f}\big(x^k - \gamma \nabla h(x^k) - \gamma L^* u^k\big) \\ u^{k+1} = \mathrm{prox}_{\sigma g^*}\big(u^k + \sigma L(2x^{k+1} - x^k)\big) \end{array} \right.$$

Salim, LC et al., "Dualize, split, randomize: Fast nonsmooth optimization algorithms", 2020

Condat–Vu algorithm form II

$$\left\lfloor \begin{array}{l} u^{k+1} = \mathrm{prox}_{\sigma g^*}\big(u^k + \sigma L x^k\big) \\ x^{k+1} = \mathrm{prox}_{\gamma f}\big(x^k - \gamma \nabla h(x^k) - \gamma L^*(2u^{k+1} - u^k)\big) \end{array} \right.$$

PD3O algorithm

$$\left\lfloor \begin{array}{l} x^{k+1} = \mathrm{prox}_{\gamma f}\big(x^k - \gamma \nabla h(x^k) - \gamma L^* u^k\big) \\ u^{k+1} = \mathrm{prox}_{\sigma g^*}\big(u^k + \sigma L(2x^{k+1} - x^k - \gamma \nabla h(x^{k+1}) + \gamma \nabla h(x^k))\big) \end{array} \right.$$

PDDY algorithm

$$\left\lfloor \begin{array}{l} u^{k+1} = \mathrm{prox}_{\sigma g^*}\big(u^k + \sigma L x^k\big) \\ x^{k+1} = \mathrm{prox}_{\gamma f}\big(x^k - \gamma \nabla h(x^k - \gamma L^*(u^{k+1} - u^k)) - \gamma L^*(2u^{k+1} - u^k)\big) \end{array} \right.$$

Primal-dual algorithms

Condat-Vu I & II PD3O PDDY $f + g \circ L + h$

Chambolle-Pock

$f + g \circ L$

Loris-Verhoeven

$g \circ L + h$

Douglas-Rachford (ADMM)

$f + g$

forward-backward

$f + h$

Primal-dual algorithms

Condat-Vu I & II PD3O PDDY $f + g \circ L + h$

Chambolle-Pock

$f + g \circ L$

Loris-Verhoeven

$g \circ L + h$

Douglas-Rachford (ADMM)

$f + g$

LC et al. "Proximal Splitting Algorithms for Convex Optimization: A Tour of Recent Advances, with New Twists", 2019

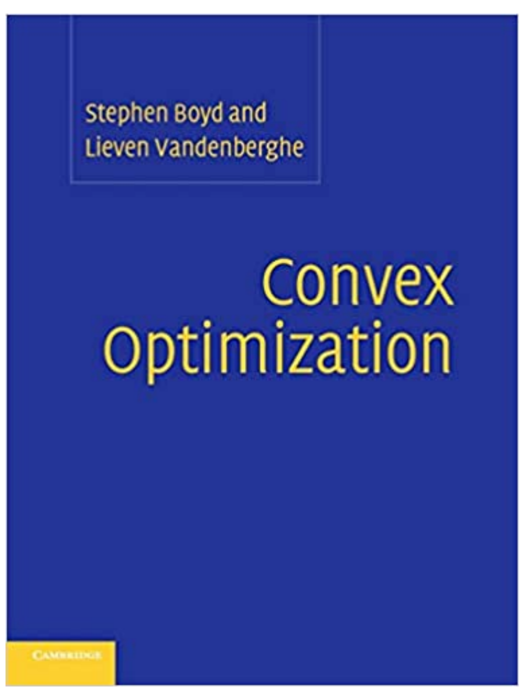
Principle

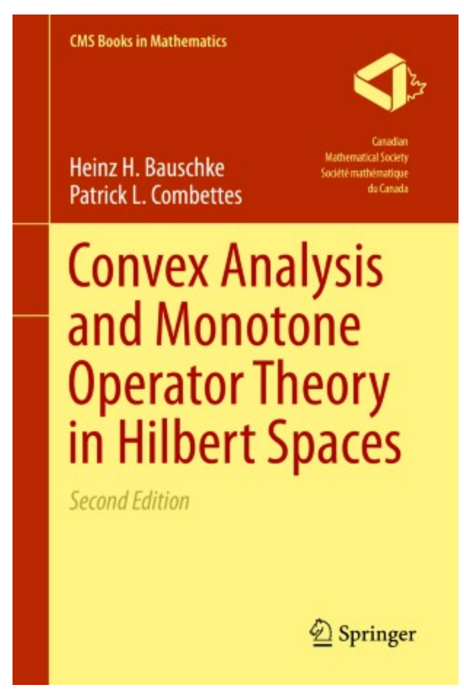
convex analysis
monotone and fixed-point
operator theory

allow us
to design

can be
analyzed
with

algorithms for large-scale
nonsmooth optimization





LC et al. "Proximal Splitting Algorithms for Convex Optimization: A Tour of Recent Advances, with New Twists", 2019

Summary

⚠ nonsmooth functions

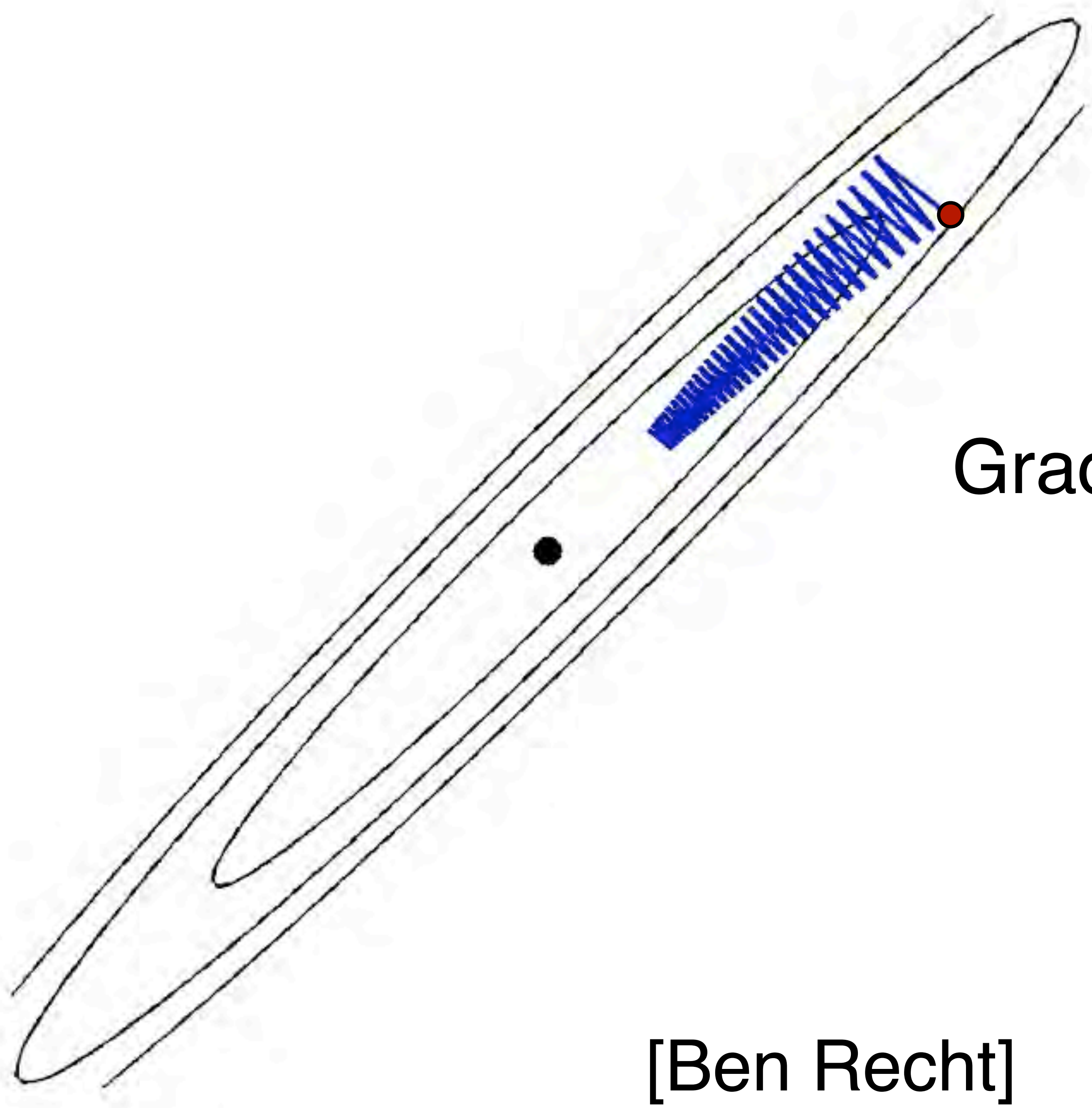
⚠ large scale

☞ proximal splitting algorithms ☺

Condat-Vu, PD3O, PDDY: versatile primal-dual algorithms

LC et al. "Proximal Splitting Algorithms for Convex Optimization: A Tour of Recent Advances, with New Twists", 2019

⚠ Speed?



Gradient descent

☹

[Ben Recht]

Convergence rates

Theorem – PD3O, with g continuous around $Lx^\star$:

$$\Psi(x^k) - \Psi(x^\star) = o(1/\sqrt{k})$$

Theorem – accelerated PD3O and PDDY
when h or f strongly convex, with varying stepsizes:

$$\|x^k - x^\star\|^2 = O\big(1/k^2\big)$$

Theorem – linear convergence of PD3O and PDDY
when h or f strongly convex and g smooth:

$$\|x^k - x^\star\|^2 \leq (1-\rho)^k c_0$$

LC et al. "Distributed Proximal Splitting Algorithms with Rates and Acceleration", 2022

Minimization with an affine constraint

$$\text{Find } x^\star \in \operatorname*{arg\,min}_{x\in\mathcal{X}} \; h(x) \quad \text{s.t. } Lx = y$$

h strongly convex

☞ linear convergence

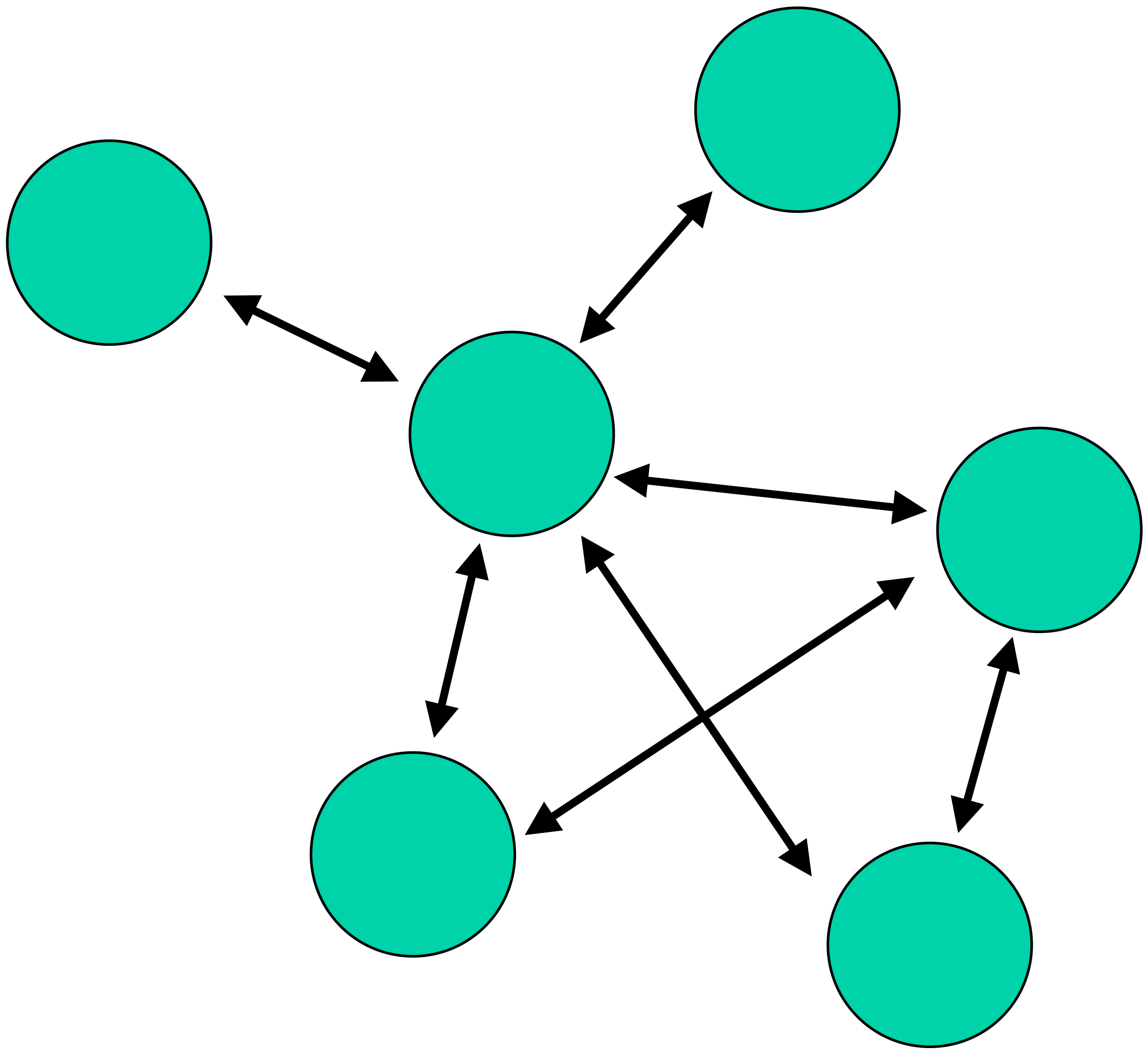
Salim, LC et al., "Dualize, split, randomize: Fast nonsmooth optimization algorithms", 2020

optimal acceleration:

Salim, LC et al., "An Optimal Algorithm for Strongly Convex Minimization under Affine Constraints", AISTATS 2022

Decentralized setting

$$\text{Find } x^\star \in \arg\min_{x\in\mathcal{X}} \sum_{m=1}^{M} h_m(x_m) \quad \text{s.t. } x_1 = \cdots = x_M$$



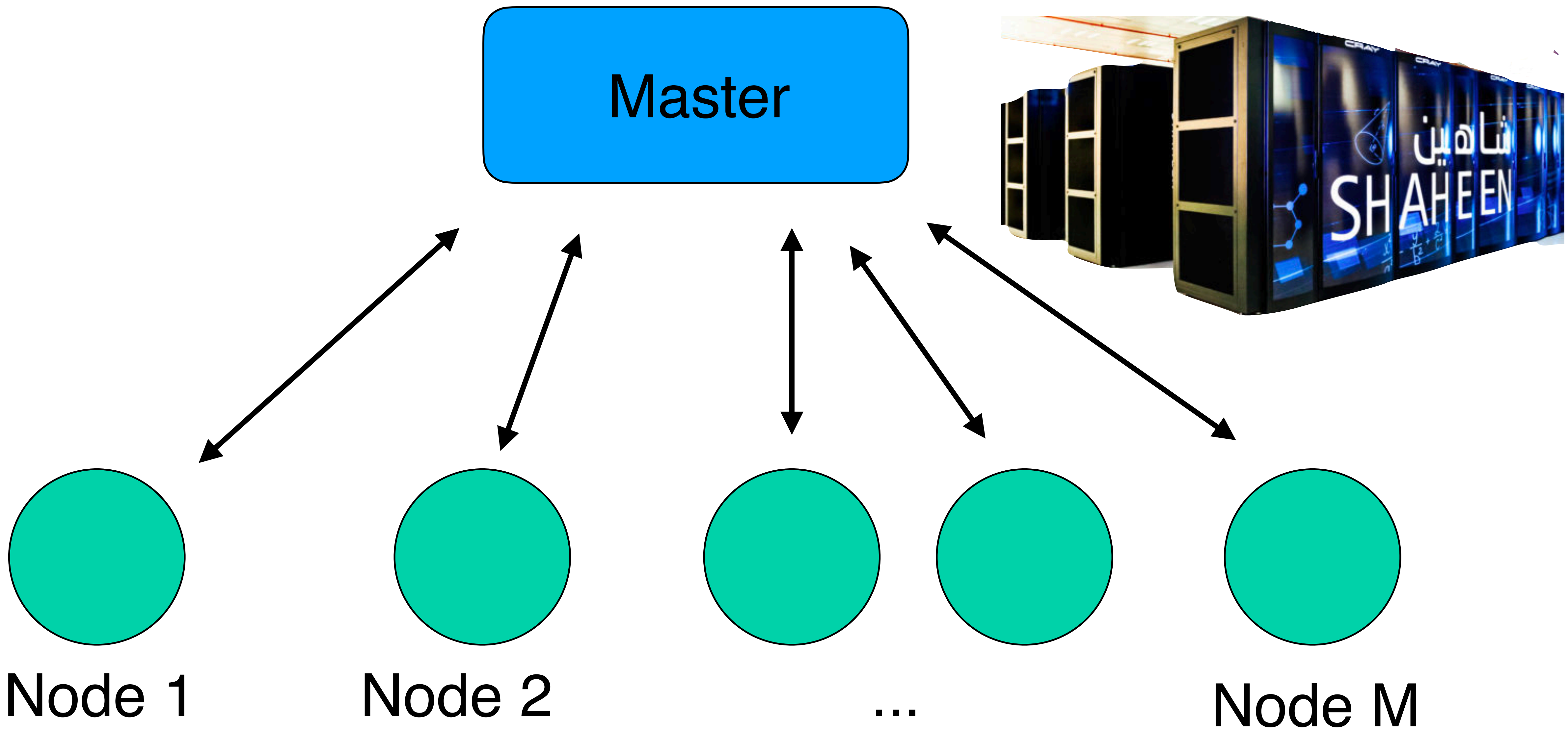
DESTROY algorithm:

Salim, LC et al., "Dualize, split, randomize: Fast nonsmooth optimization algorithms", 2020

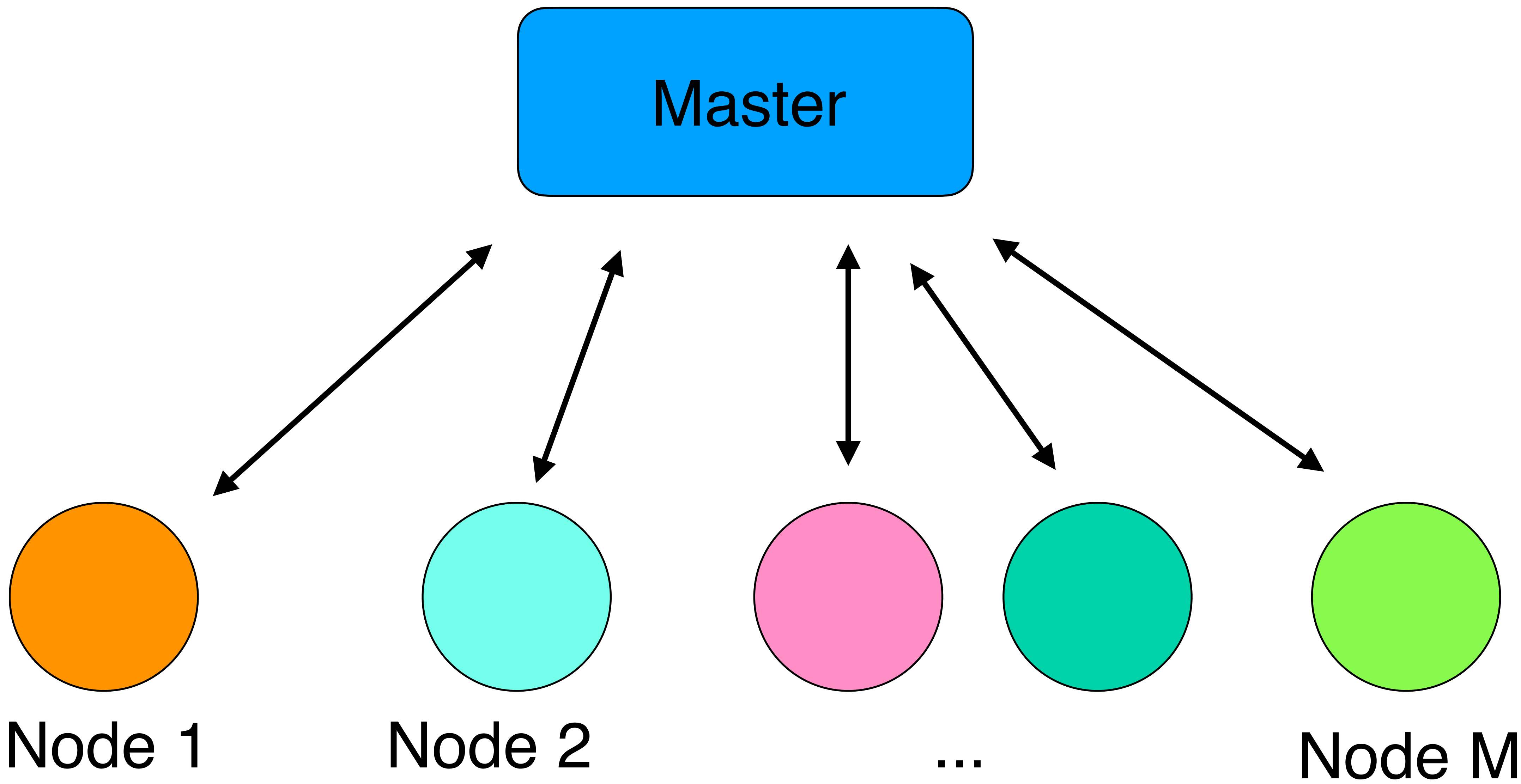
optimal acceleration:

Salim, LC et al., "An Optimal Algorithm for Strongly Convex Minimization under Affine Constraints", AISTATS 2022

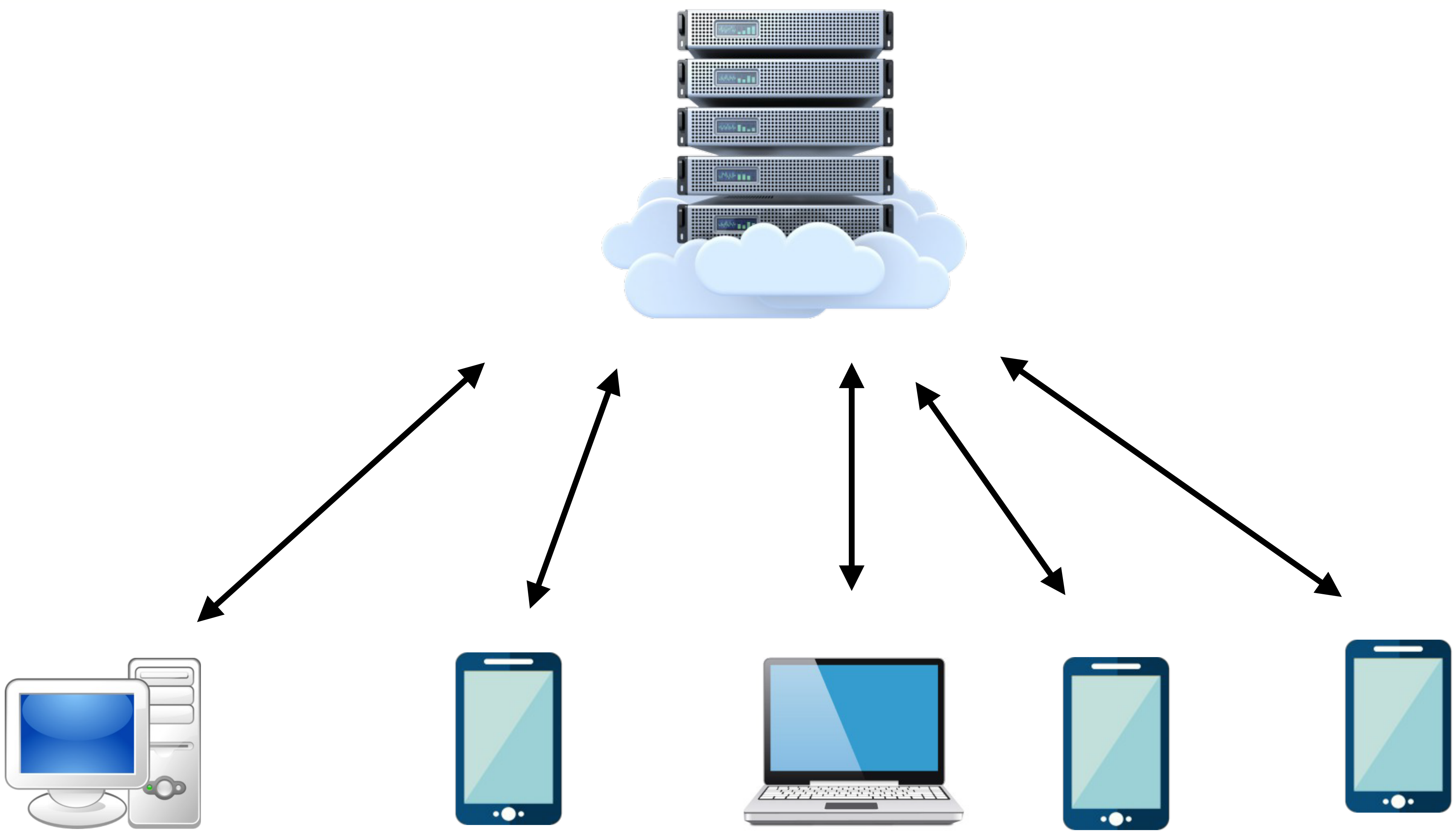
Distributed optimization



Distributed optimization



Federated learning



Distributed algorithms

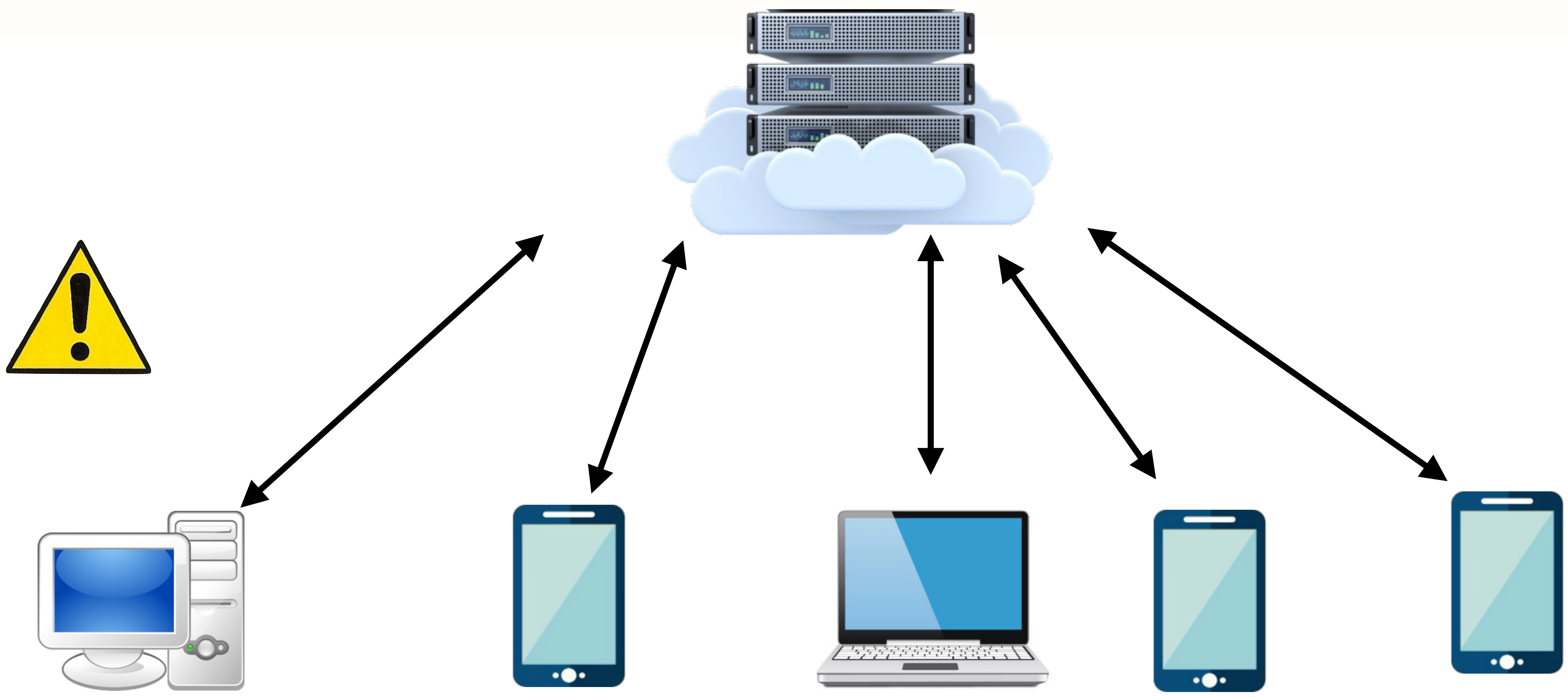
$$\underset{x\in\mathcal{X}}{\text{minimize}}\ f(x) + \frac{1}{M}\sum_{m=1}^{M} g_m(L_m x) + \frac{1}{M}\sum_{m=1}^{M} h_m(x),$$

Distributed PDDY Algorithm

input: $(\gamma_k)_{k\in\mathbb{N}}$, $\eta \geq \|\widehat{L}\|^2$, $(\omega_m)_{m=1}^M$, $x_f^0 \in \mathcal{X}$, $(u_m^0)_{m=1}^M \in \widehat{\mathcal{U}}$
initialize: $p_m^0 := L_m^* u_m^0$, $m = 1, \ldots, M$
for $k = 0, 1, \ldots$ **do**
 at all nodes, for $m = 1, \ldots, M$, **do**
  $u_m^{k+1} := \mathrm{prox}_{M\omega_m g_m^*/(\gamma_k \eta)}\big(u_m^k + \frac{M\omega_m}{\gamma_k \eta} L_m x_f^k\big)$
  $p_m^{k+1} := L_m^* u_m^{k+1}$
  $x_m^{k+1} := x_f^k - \frac{\gamma_k}{M\omega_m}(p_m^{k+1} - p_m^k)$
  $a_m^k := M\omega_m x_m^{k+1} - \gamma_{k+1}\nabla h_m(x_m^{k+1}) - \gamma_{k+1} p_m^{k+1}$
  transmit a_m^k to master
 at master, **do**
  $x_f^{k+1} := \mathrm{prox}_{\gamma_{k+1} f}\big(\frac{1}{M}\sum_{m=1}^M a_m^k\big)$
  broadcast x_f^{k+1} to all nodes
end for

LC et al. "Distributed Proximal Splitting Algorithms with Rates and Acceleration", 2022

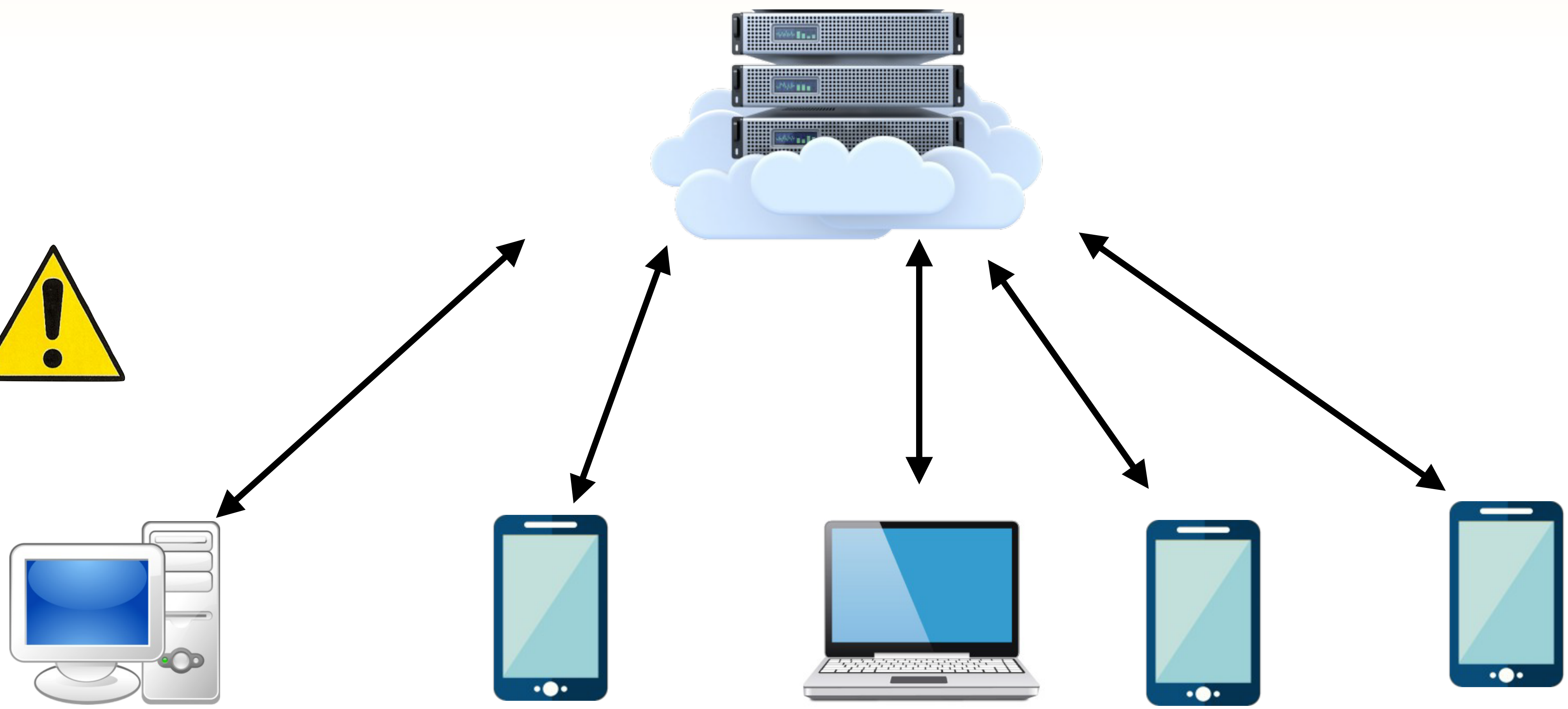
Communication bottleneck



☞ less communication with **local steps**

Malinovky et al., "From local SGD to local fixed point methods for federated learning", ICML 2020

Communication bottleneck



compression

Albasyoni et al. "Optimal Gradient Compression for Distributed and Federated Learning", 2020

LC and Richtárik, "MURANA: A Generic Framework for Stochastic Variance-Reduced Optimization", 2021

LC et al., "EF-BV: A unified theory of error feedback...", 2022

Summary

⚠ nonsmooth large-scale optimization

☞ proximal splitting algorithms ☺

⚠ speed ☞

- (math.) acceleration
- cheaper iterations (computations, communication)

Perspectives

Beyond SGD: randomized fixed-point algorithms: random/inexact/cheap estimates of operators to call or variables/coordinates to update

Acceleration of fixed-point algorithms

Applications to communication networks, data processing on graphs, federated learning