Optimization

Benjamin Recht
University of California, Berkeley

Stephen Wright
University of Wisconsin-Madison

optimization

cost

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in \Omega \end{array}$$

constraints

might be too much to cover in 3 hours

optimization (for big data?)

cost

$$\begin{array}{ll} \text{minimize} & \mathbb{E}_\xi[f(x,\xi)] \\ \text{subject to} & x \in \mathrm{conv}(\mathcal{A}) \end{array}$$

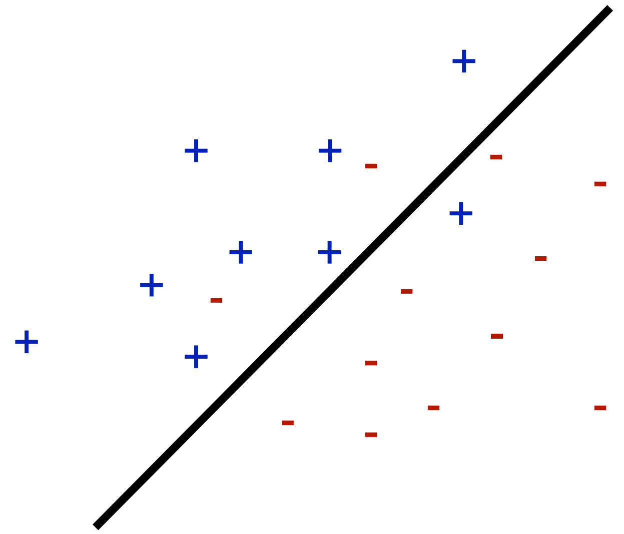
constraints

- distribution over ξ is well-behaved
- $\mathcal{A}$ is simple (low-cardinality, low-dimension, low-complexity)

$$\text{minimize} \quad \mathbb{E}_\xi[f(x,\xi)] + P(x)$$

closely related cousin where P is a simple convex function

Support Vector Machines



cancer vs other illness
fraud vs normal purchase
up-going vs down-going muons

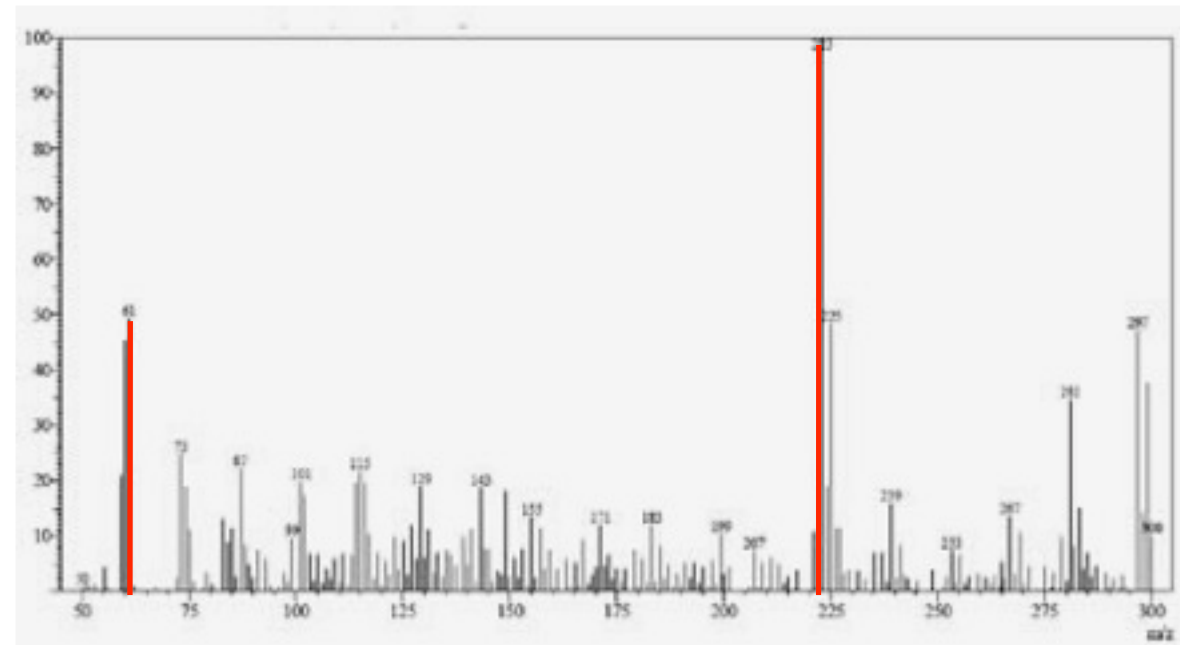
minimize $\underbrace{\sum_{i=1}^{n} \max(1 - y_i x^T z_i, 0)}_{\text{sample average over observed data and labels}} + \underbrace{\lambda \|x\|_2^2}_{\text{regularizer to select low-complexity models}}$

LASSO

Compressed Sensing

reduce number of measurements
required for signal acquisition

Sparse Modeling

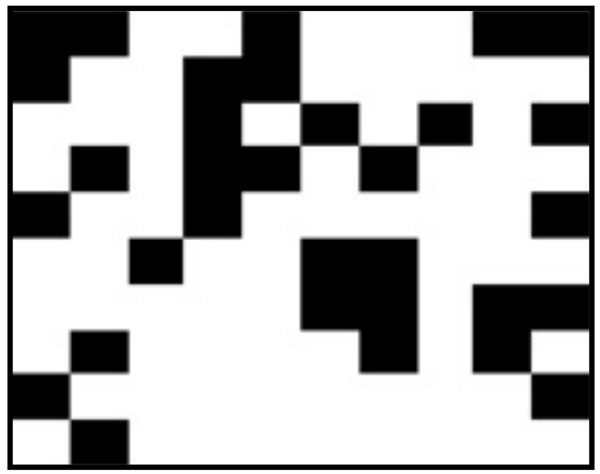


search for a sparse set of markers
for classification

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^{n} (a_i^T x - b_i)^2 \\ \text{subject to} & \|x\|_1 \leq R \end{array}$$

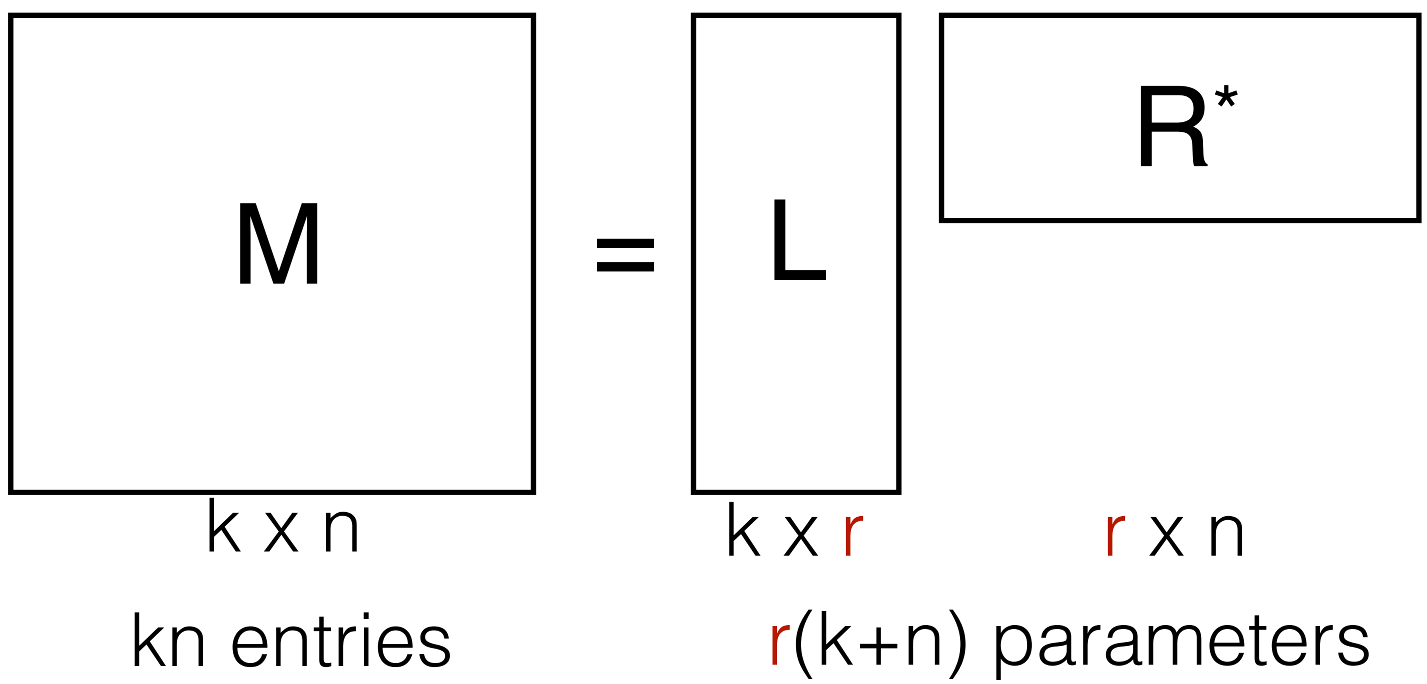
Matrix Completion

M =



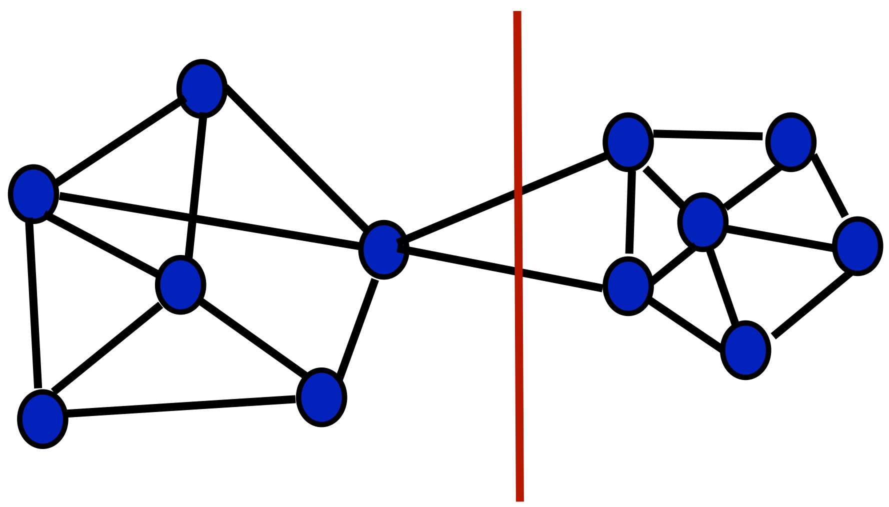
M_{ij} known for black cells
M_{ij} unknown for white cells
Rows index features
Columns index examples
Entries specified on set E

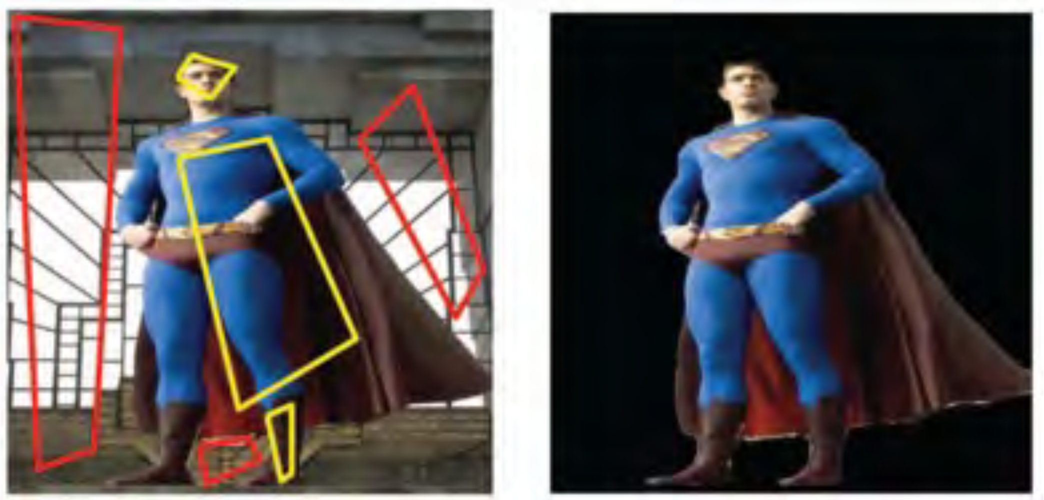
- How do you fill in the missing data?



$$\text{minimize} \quad \sum_{(u,v)\in E}(X_{uv} - M_{uv})^2 + \mu\|\mathbf{X}\|_*$$

Graph Cuts





Bhusnurmath and Taylor, 2008

- Image Segmentation
- Entity Resolution
- Topic Modeling

$$
\begin{aligned}
\text{minimize} \quad & \textstyle\sum_{(u,v)\in E} |x_u - x_v| \\
\text{subject to} \quad & x_u \in [0,1] \quad \text{if } u \in V \\
& x_a = 0 \quad \text{if } a \in A \\
& x_b = 1 \quad \text{if } b \in B
\end{aligned}
$$

optimization

cost

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in \Omega \end{array}$$

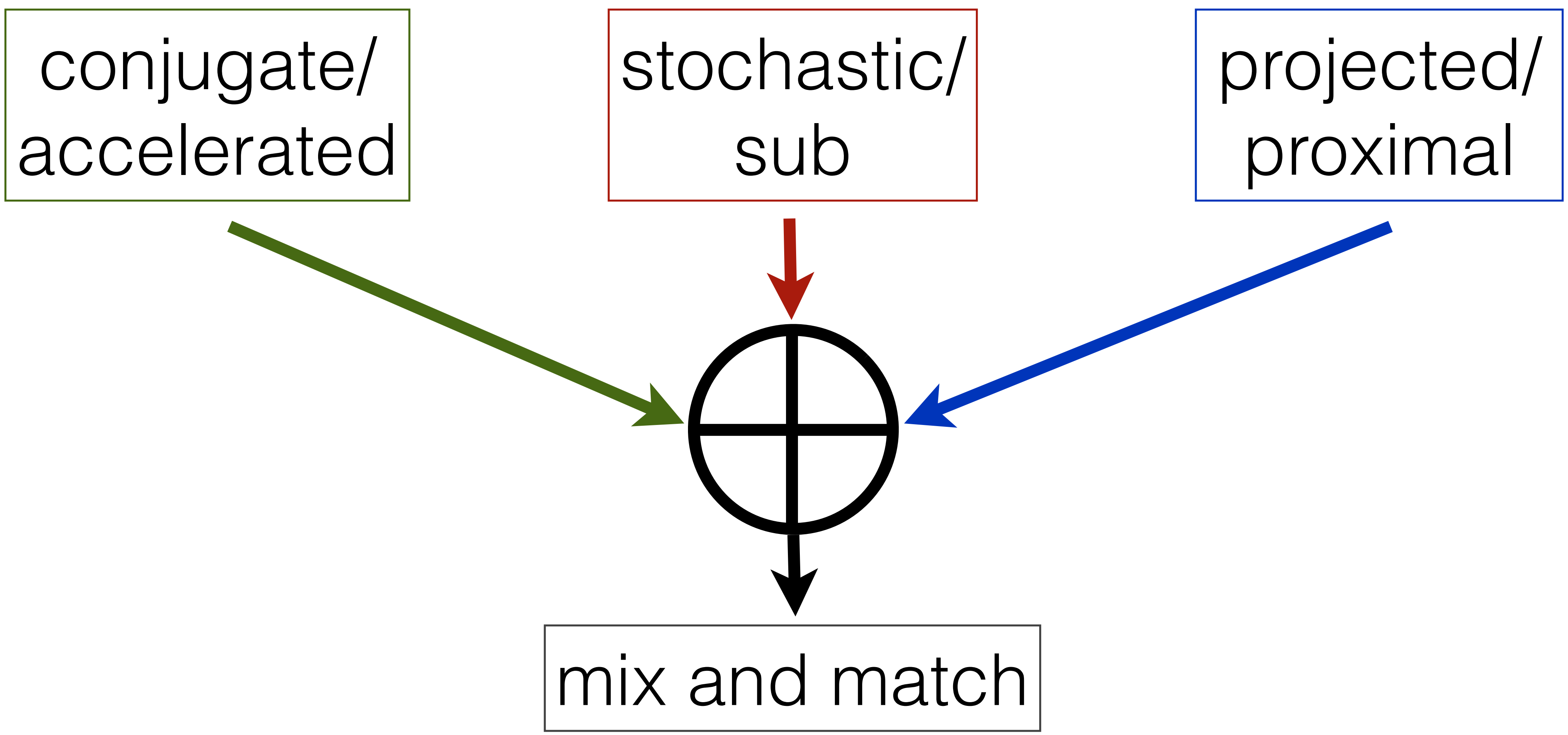
constraints

might be too much to cover in 3 hours

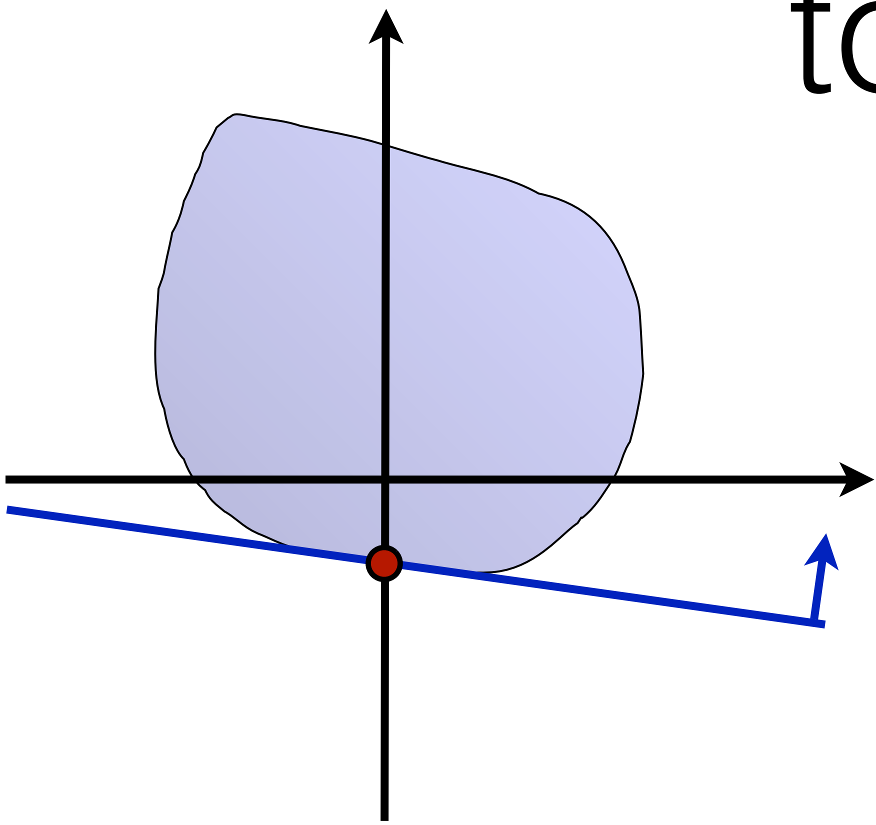
- optimization is *ubiquitous*
- optimization is *modular*
- optimization is *declarative*

$$x[k+1] \leftarrow x[k] + \alpha_k v[k]$$

Today: $\boxed{\text{gradient descent}}$



tomorrow: duality



$$\min_x f(x) = \max_z g(z)$$

$$\min_x \max_z \mathcal{L}(x, z) = \max_z \min_x \mathcal{L}(x, z)$$

- find problems that always lower bound the optimal value.
- puts problem in NP ∩ coNP
- information from one problem informs the other
- some times easier to solve one than the other
- basis of many proof techniques in data science (and tons of other areas too!)

what we'll be skipping...

- 2nd order/newton/BFGS
- interior point methods/ellipsoid methods
- active set methods, manifold identification
- branch and bound
- *integrating combinatorial thinking*
- derivative-free optimization
- soup of heuristics (simulated annealing, genetic algorithms, ...)
- *modeling*

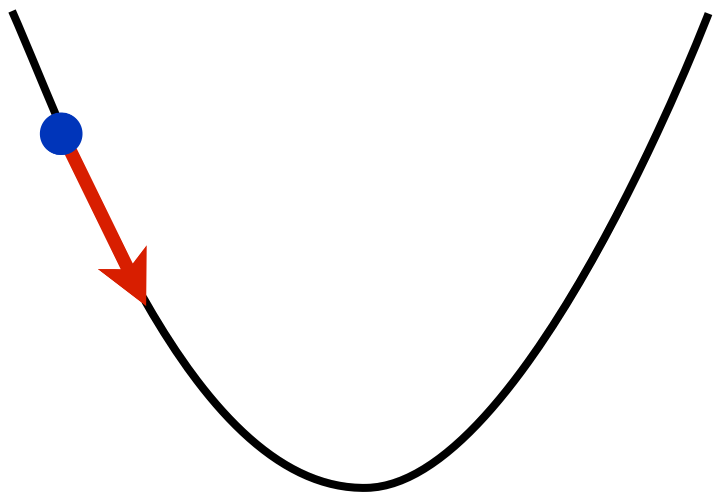
optimality conditions

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

$$\boxed{\text{Search for } \nabla f(x) = 0}$$

- Turns a geometric problem into an algebraic problem: solve for the point where the gradient vanishes.
- Is necessary for optimality (sufficient for convex, smooth f)

$$\boxed{\begin{array}{c} x[k+1] \leftarrow x[k] + \alpha_k v[k] \\ \text{gradient descent} \end{array}}$$



Assume there exits an $x_\star \in \mathcal{D}$ where $\nabla f(x_\star) = 0$

Suppose the map $\psi(x) = x - \alpha \nabla f(x)$ is contractive on $\mathcal{D}$

$\|\psi(x) - \psi(z)\| \leq \beta \|x - z\|$ for some $0 \leq \beta < 1$

run gradient descent starting at $x[0] \in \mathcal{D}$

$$\begin{aligned} \|x[k+1] - x_\star\| &= \|x[k] - \alpha \nabla f(x[k]) - x_\star\| \\ &= \|\psi(x[k]) - \psi(x_\star)\| && \psi(x_\star) = x_\star \\ &\leq \beta \|x[k] - x_\star\| && \text{contractivity} \\ &\vdots \\ &\leq \beta^{k+1} \|x[0] - x_\star\| && \textit{linear rate} \end{aligned}$$

- If f is 2x differentiable, contractivity means f is convex on $\mathcal{D}$

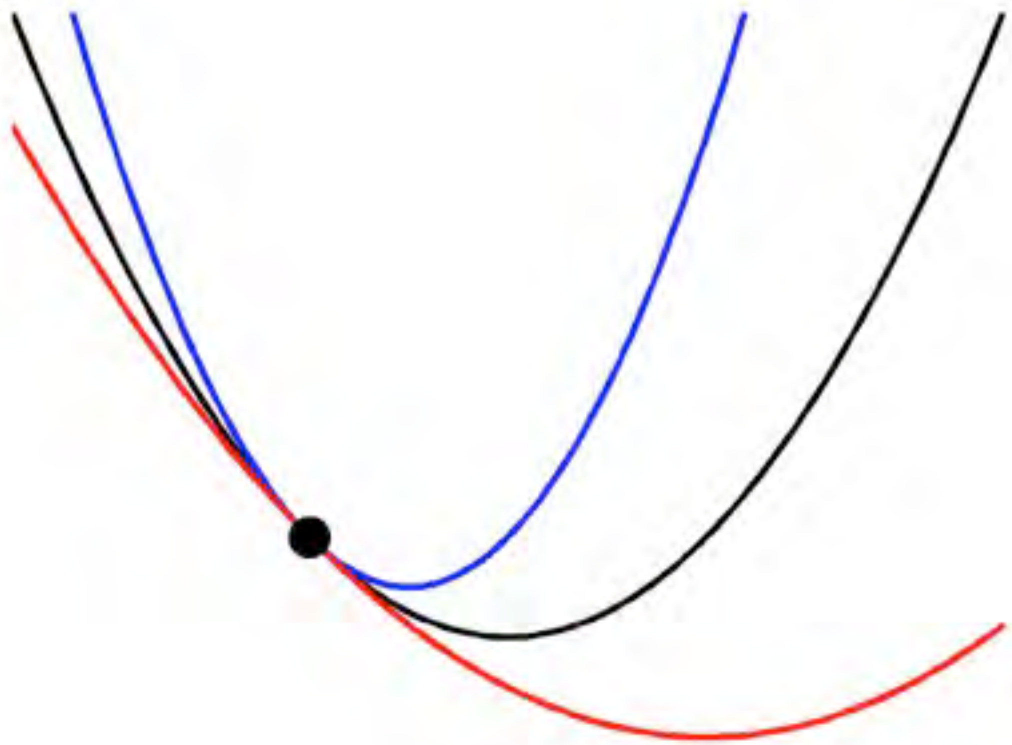
$$\tfrac{1}{t}\|\psi(x+t\Delta x)-\psi(x)\| \le \beta\|\Delta x\| \quad \text{for all } t>0$$

$$\lim_{t\to 0^+} \tfrac{1}{t}\|\psi(x+t\Delta x)-\psi(x)\| = \lim_{t\to 0^+} \|\Delta x - \tfrac{\alpha}{t}(\nabla f(x+t\Delta x)-\nabla f(x))\|$$
$$= \|\Delta x - \alpha\nabla^2 f(x)\Delta x\| \le \beta\|\Delta x\|$$

$$\Downarrow$$

$$\|I-\alpha\nabla^2 f(x)\| \le \beta$$

$$\underbrace{\tfrac{1-\beta}{\alpha} I \preceq}_{\text{convex}} \nabla^2 f(x) \underbrace{\preceq \tfrac{1+\beta}{\alpha} I}_{\text{Lipschitz gradients}}$$



convexity

$$f(tx + (1-t)z) \leq tf(x) + (1-t)f(x)$$

$$f(z) \geq f(x) + \nabla f(x)^T(z-x) + \underline{\tfrac{\ell}{2}\|z-x\|^2}$$

strong convexity

$$\boxed{\kappa = \frac{L}{\ell} \geq 1}$$

condition number of Hessian

Lipschitz gradients

$$\|\nabla f(x) - \nabla f(z)\| \leq L\|x - z\|$$

$$f(z) \leq f(x) + \nabla f(x)^T(z-x) + \tfrac{L}{2}\|z-x\|^2$$

follows from Taylor's theorem

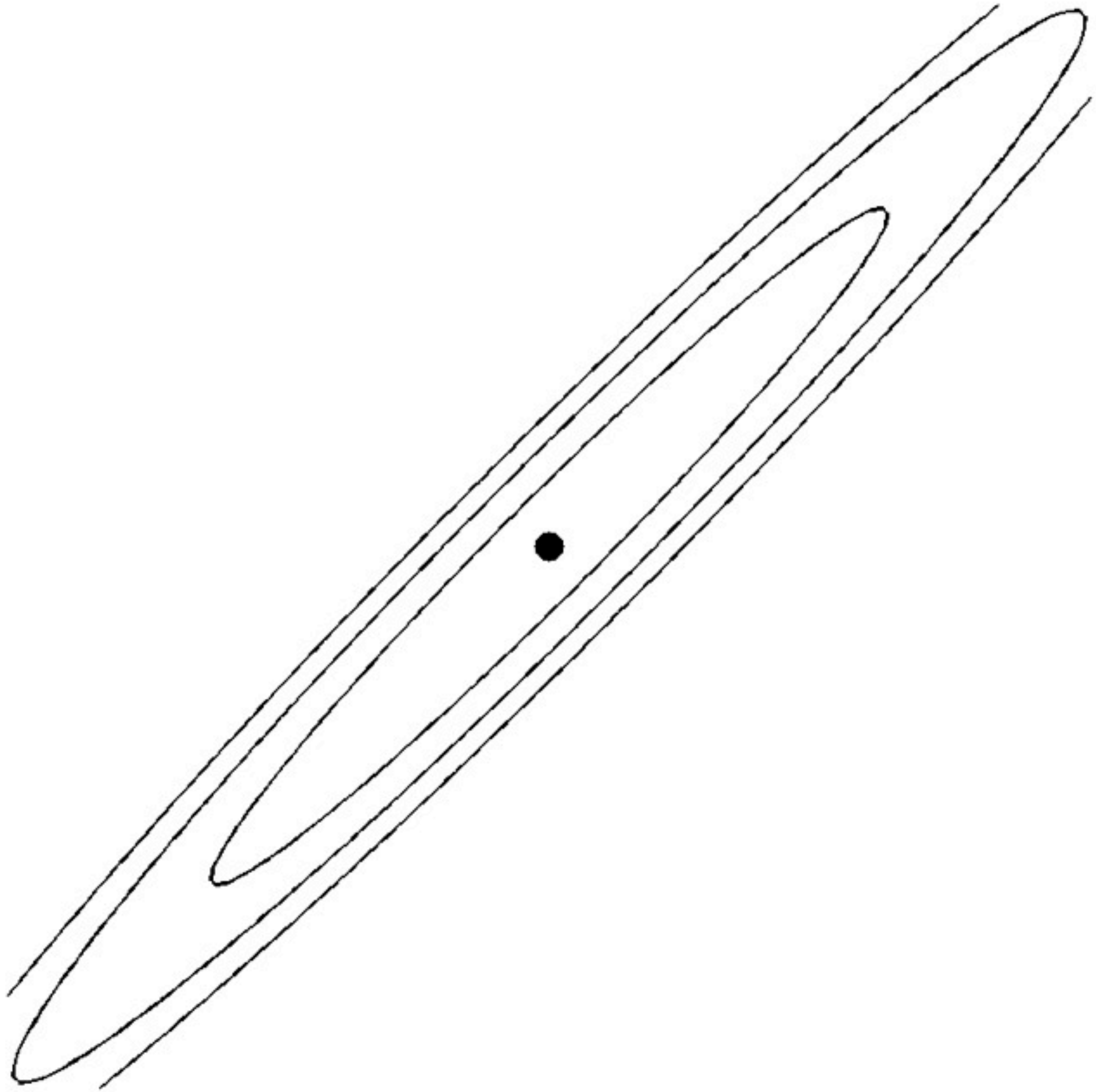
With step size $\alpha = \frac{2}{\ell+L}$, $\|x[k] - x_\star\| \leq \left(1 - \dfrac{2}{\kappa+1}\right)^k \|x[0] - x_\star\|.$

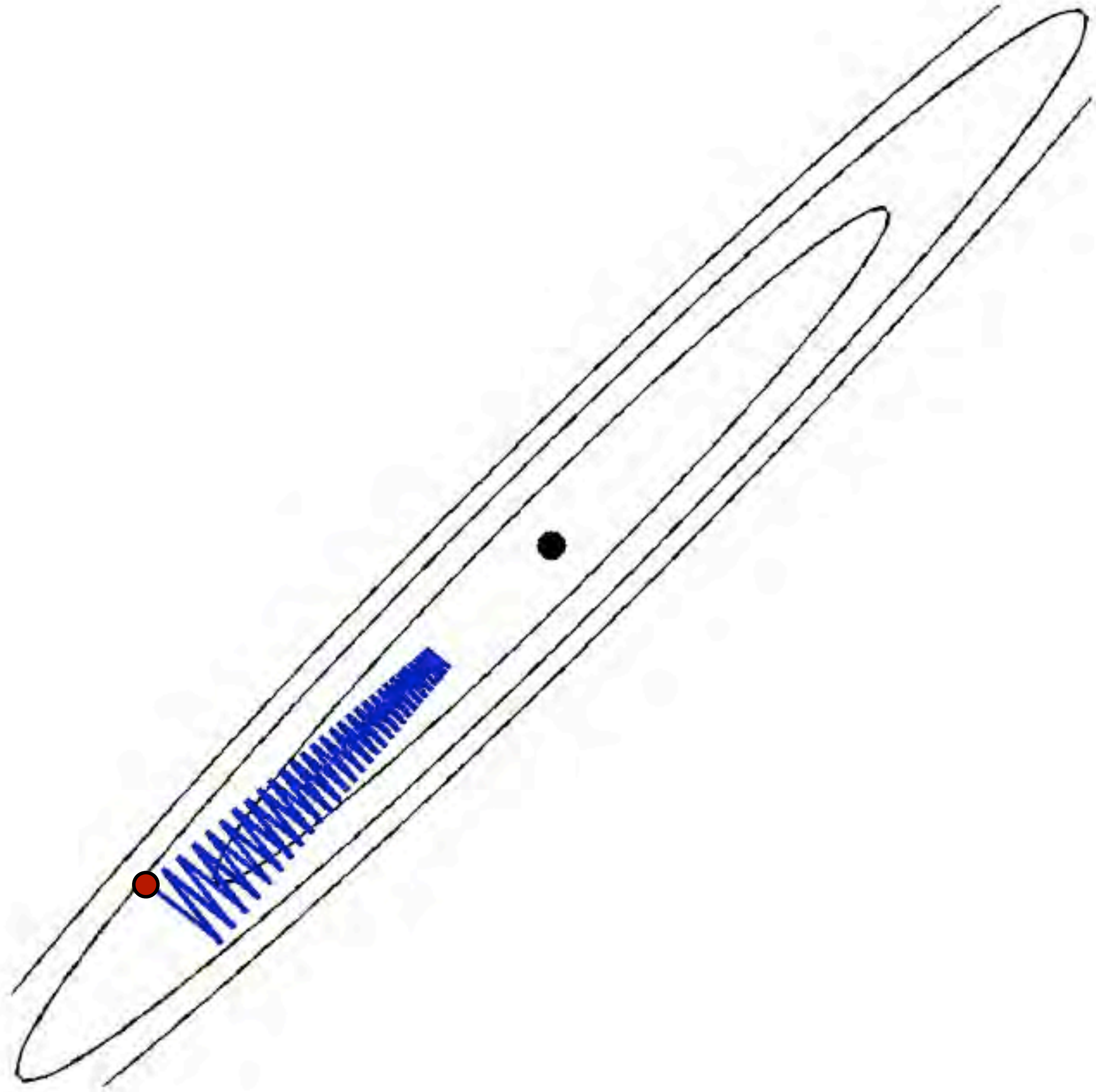
$$f(x[k]) - f_\star \leq L\left(1 - \tfrac{2}{\kappa+1}\right)^{2k} \|x[0] - x_\star\|^2$$

Note on convergence rate

With step size $\alpha = \frac{2}{\ell+L}$, $\|x[k] - x_\star\| \leq \left(1 - \frac{2}{\kappa + 1}\right)^k \|x[0] - x_\star\|$.

- If you don't know the exact stepsize, can we achieve the rate?
  - *Exact line search:* at each iteration, find the α that minimizes f(x+αd).
  - *Backtracking line search:* Reduce α by constant multiple until the function value sufficiently decreases.
- Both achieve linear rate of convergence.
- More sophisticated line searches often used in practice, but none improve over this rate in the worst case.





acceleration/multistep

gradient method akin to an ODE

$$x[k+1] = x[k] - \alpha \nabla f(x[k])$$
$$\dot{x} = -\nabla f(x)$$

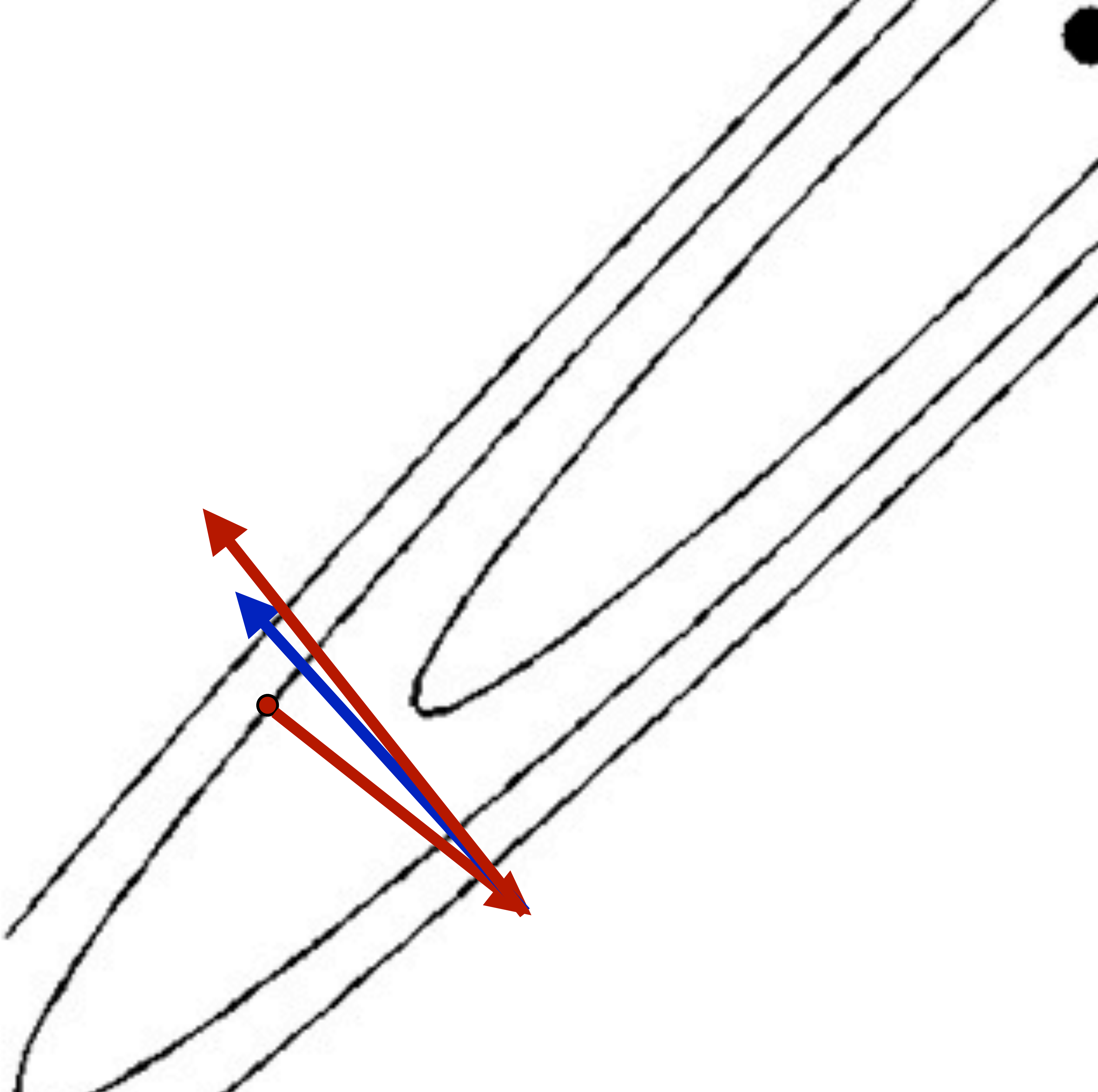
to prevent oscillation, add a second order term

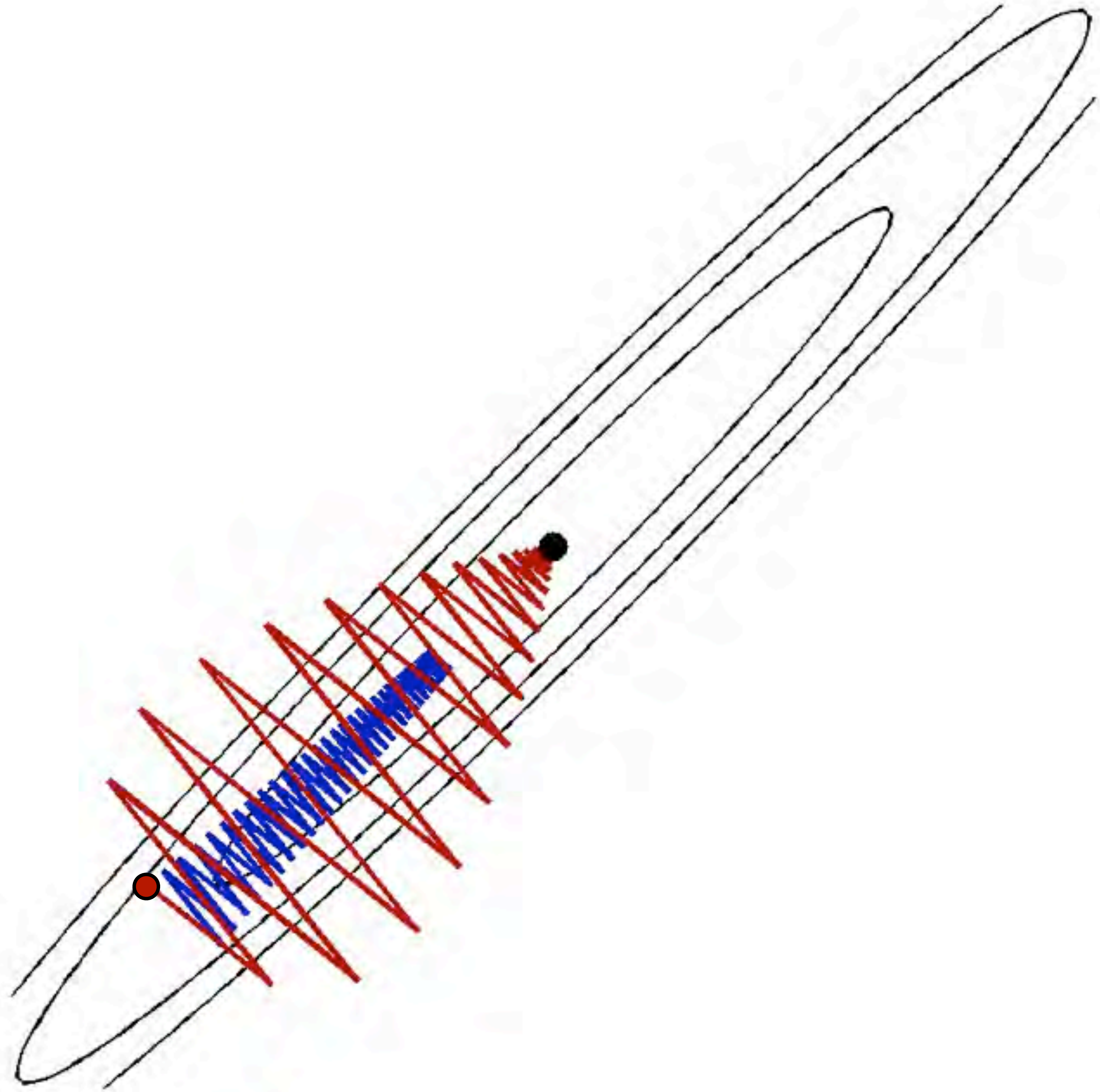
$$\ddot{x} = -b\dot{x} - \nabla f(x)$$
$$x[k+1] = x[k] - \alpha \nabla f(x[k]) + \beta (x[k] - x[k-1])$$

$$x[k+1] = x[k] + \alpha p[k]$$
$$p[k] = -\nabla f(x[k]) + \beta p[k-1]$$

heavy ball method (constant α,β)

when f is quadratic, this is *Chebyshev's iterative method*





analysis

$$x[k+1] = x[k] + \alpha p[k]$$
$$p[k] = -\nabla f(x[k]) + \beta p[k-1]$$

heavy ball method (constant **α**,**β**)

Analyze by defining a composite error vector: $w_k := \begin{bmatrix} x[k] - x_\star \\ x[k-1] - x_\star \end{bmatrix}$

Then $w[k+1] = Bw[k] + o(\|w[k]\|)$

where $B := \begin{bmatrix} -\alpha\nabla^2 f(x_\star) + (1+\beta)I & -\beta I \\ I & 0 \end{bmatrix}$

analysis (cont.)

$$w[k+1] = Bw[k] + o(\|w[k]\|)$$

B has the same eigenvalues as $\begin{bmatrix} -\alpha\Lambda + (1+\beta)I & -\beta I \\ I & 0 \end{bmatrix}$

$$\Lambda = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$$

where λ_i are the eigenvalues of $\nabla^2 f(x_\star)$

Choose α,β to explicitly minimize the max eigenvalue of B to obtain

$$\alpha = \frac{4}{L}\frac{1}{(1+1/\sqrt{\kappa})^2} \qquad \beta = \left(1 - \frac{2}{\sqrt{\kappa}+1}\right)^2.$$

Leads to linear convergence for $\|x[k] - x_\star\|_2$ with rate approximately

$$\left(1 - \frac{2}{\sqrt{\kappa}+1}\right)$$

about those rates...

- Best steepest descent: Linear rate approx $\left(1 - \frac{2}{\kappa + 1}\right)$
- Heavy-ball: Linear rate approx $\left(1 - \frac{2}{\sqrt{\kappa} + 1}\right)$

- Big difference! To yield $\|x[k] - x_\star\|_2 < \epsilon \|x[0] - x_\star\|_2$

$$k \geq \frac{\kappa}{2} \log(1/\epsilon)$$ gradient descent

$$k \geq \frac{\sqrt{\kappa}}{2} \log(1/\epsilon)$$ heavy ball

- A factor of $\kappa^{1/2}$ difference. e.g. if κ=100, need 10 times fewer steps.

conjugate gradients

$$x[k+1] = x[k] + \alpha_k p[k]$$
$$p[k] = -\nabla f(x[k]) + \beta_k p[k-1]$$

Choose α_k by line search (to reduce f)

Choose β_k such that *p[k]* is approximately conjugate to *p[1], ..., p[k-1]* (*really only makes sense for quadratics, but whatever...*)

- Does not achieve a better rate than heavy ball
- Gets around having to know parameters
- Convergence proofs very sketchy (except when *f* is quadratic) and need elaborate line search to guarantee local convergence.

optimal method

$$\boxed{\alpha_k = \frac{1}{L}}$$

Nesterov's optimal method (1983,2004)

$$x[k+1] = x[k] + \alpha_k p[k]$$

$$p[k] = -\nabla f(x[k] + \beta_k(x[k] - x[k-1])) + \beta_k p[k-1]$$

heavy ball with *extragradient step*

$$\lambda_{k+1}^2 = (1 - \lambda_{k+1})\lambda_k^2 + \kappa^{-1}\lambda_{k+1}$$

$$\beta_k = \frac{\lambda_k(1-\lambda_k)}{\lambda_k^2 + \lambda_{k+1}}$$

$$t_k = \tfrac{1}{2}\left(1 + \sqrt{1 + 4t_k^2}\right)$$

$$\beta_k = \frac{t_k - 1}{t_{k+1}}$$

FISTA (Beck and Teboulle 2007)

$$\beta_k = \frac{k-1}{k+2}$$

- Recent fixes use line search to find parameters and still achieve optimal rate (modulo log factors)
- Analysis based on *estimate sequences*, using simple quadratic approximations to f

why “optimal?”

you can’t beat the heavy ball convergence rate using only gradients and function evaluations.

$$f(x) = x_1^2 + \sum_{i=1}^{n-1} (x_i - x_{i+1})^2 + x_n^2 - 2x_1 + \mu \|x\|_2^2$$

$$\boxed{\mu I \preceq \nabla^2 f(x) \preceq (4+\mu) I}$$

$$\boxed{\kappa \approx 1 + \frac{4}{\mu}}$$

- start at x[0] = e_1.
- after k steps, x[j] = 0 for j>k+1

- norm of the optimal solution on the unseen coordinates tends to $\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^{2k}$

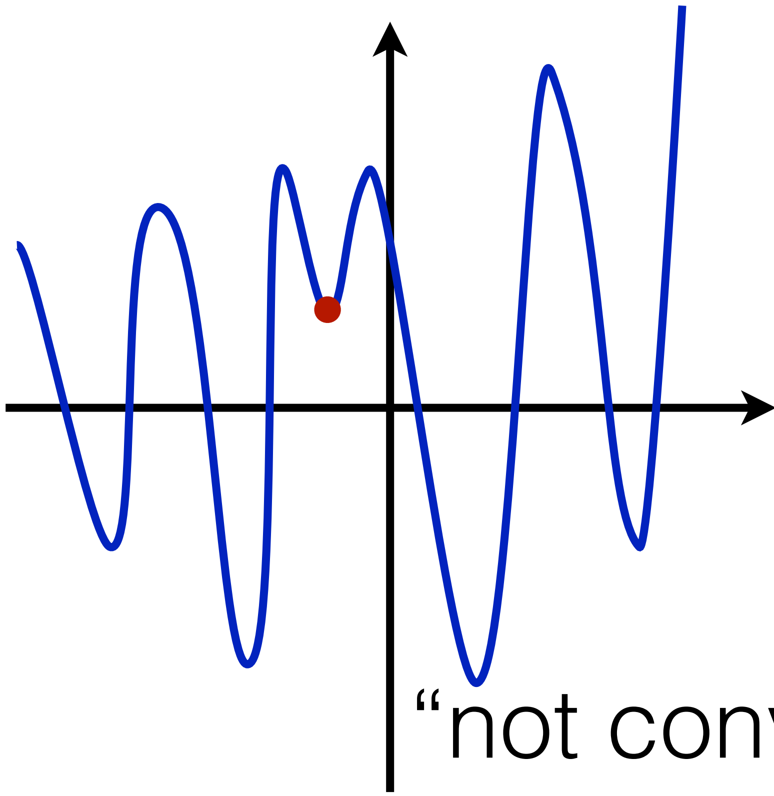
not strongly convex (ℓ=0)

- gradient descent: $$f(x[k]) - f_\star \le \frac{2L\|x[0] - x_\star\|_2^2}{k+4}$$
- optimal method: $$f(x[k]) - f_\star \le \frac{4L\|x[0] - x_\star\|_2^2}{(k+2)^2}$$
- Big difference! To yield $f(x[k]) - f_\star < \epsilon$

  gradient descent $$k \ge \frac{2L\|x[0] - x_\star\|_2^2}{\epsilon} - 4$$

  optimal method $$k \ge \frac{2L\|x[0] - x_\star\|_2}{\sqrt{\epsilon}} - 2$$

- A factor of $\epsilon^{1/2}$ difference. e.g. if ϵ=0.0001, need 100 times fewer steps.



nonconvexity

can still efficiently find a point where $\|\nabla f(x)\| \le \epsilon$ in time $O(1/\epsilon^2)$

"not convex"

n.b. nonconvexity really lets you model *anything*

$$f(x) = \sum_{i,j=1}^{d} Q_{ij} x_i^2 x_j^2$$

$\nabla f(0) = 0$ for all Q

checking if 0 is a local minimum in NP-hard

stochastic gradient

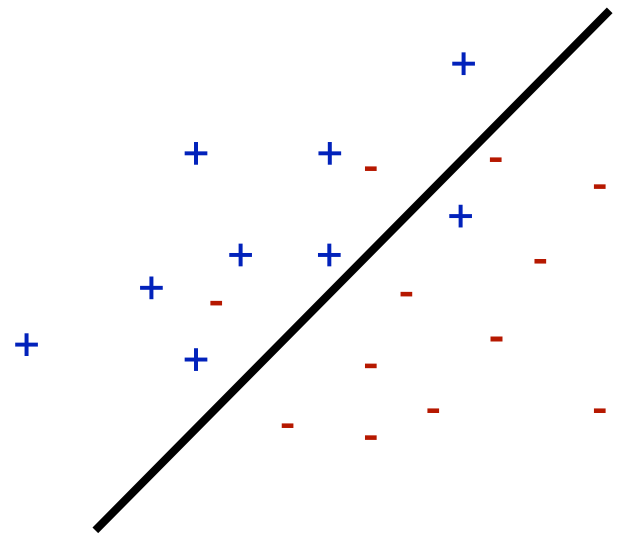
$$\text{minimize} \quad \mathbb{E}_{\xi}[f(x,\xi)]$$

Stochastic Gradient Descent:

For each k, sample ξ_k and compute $\boxed{x[k+1] = x[k] - \alpha_k \nabla_x f(x[k], \xi_k)}$

- Robbins and Monro (1950)
- Adaptive Filtering (1960s-1990s)
- Back Propagation in Neural Networks (1980s)
- Online Learning, Stochastic Approximation (2000s)

Support Vector Machines



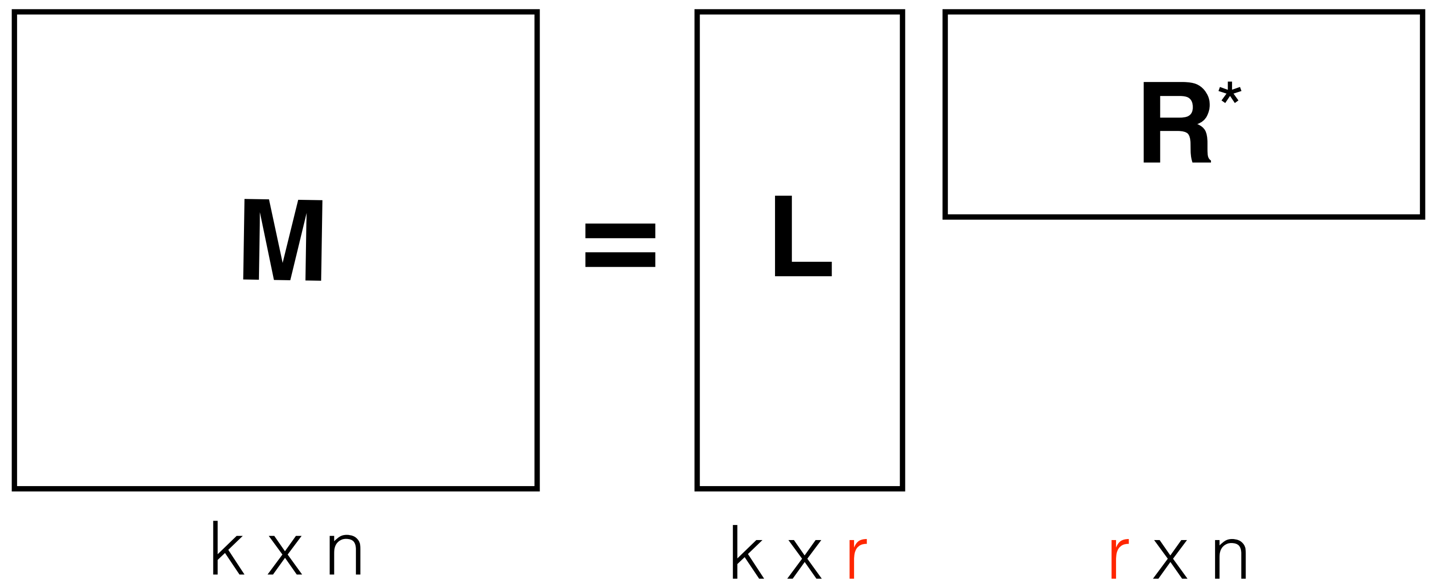
cancer vs other illness
fraud vs normal purchase
up-going vs down-going muons

minimize $\sum_{i=1}^{n} \max(1 - y_i x^T z_i, 0) + \lambda \|x\|_2^2$

$$\text{minimize}_x \sum_{i=1}^{n} \left(\max(1 - y_i x^T z_i, 0) + \frac{\lambda}{n} \|x\|^2 \right)$$

- Step 1: Pick i and compute the sign of the assignment: $\hat{y}_i = \operatorname{sign}(x^T z_i)$
- Step 2: If $\hat{y}_i \neq y_i$, $x \leftarrow (1 - \frac{\alpha\lambda}{n}) x + \alpha y_i z_i$

matrix completion



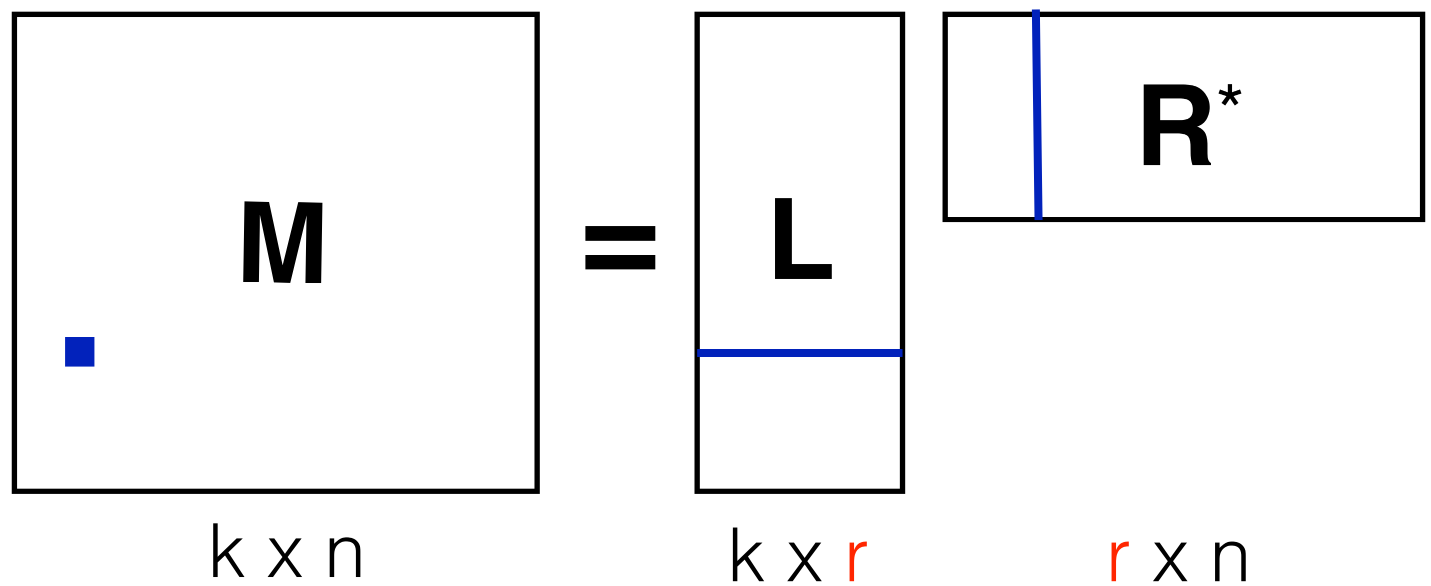
Entries Specified on set E

minimize $\sum_{(u,v)\in E}(X_{uv} - M_{uv})^2 + \mu\|\mathbf{X}\|_*$

Idea: approximate $\mathbf{X} \approx \mathbf{L}\mathbf{R}^T$

$$\text{minimize}_{(\mathbf{L},\mathbf{R})} \sum_{(u,v)\in E} \left\{ (\mathbf{L}_u\mathbf{R}_v^T - M_{uv})^2 + \mu_u\|\mathbf{L}_u\|_F^2 + \mu_v\|\mathbf{R}_v\|_F^2 \right\}$$

SGD code for matrix completion



$$\text{minimize}_{(\mathbf{L},\mathbf{R})} \sum_{(u,v)\in E} \left\{ (\mathbf{L}_u \mathbf{R}_v^T - M_{uv})^2 + \mu_u \|\mathbf{L}_u\|_F^2 + \mu_v \|\mathbf{R}_v\|_F^2 \right\}$$

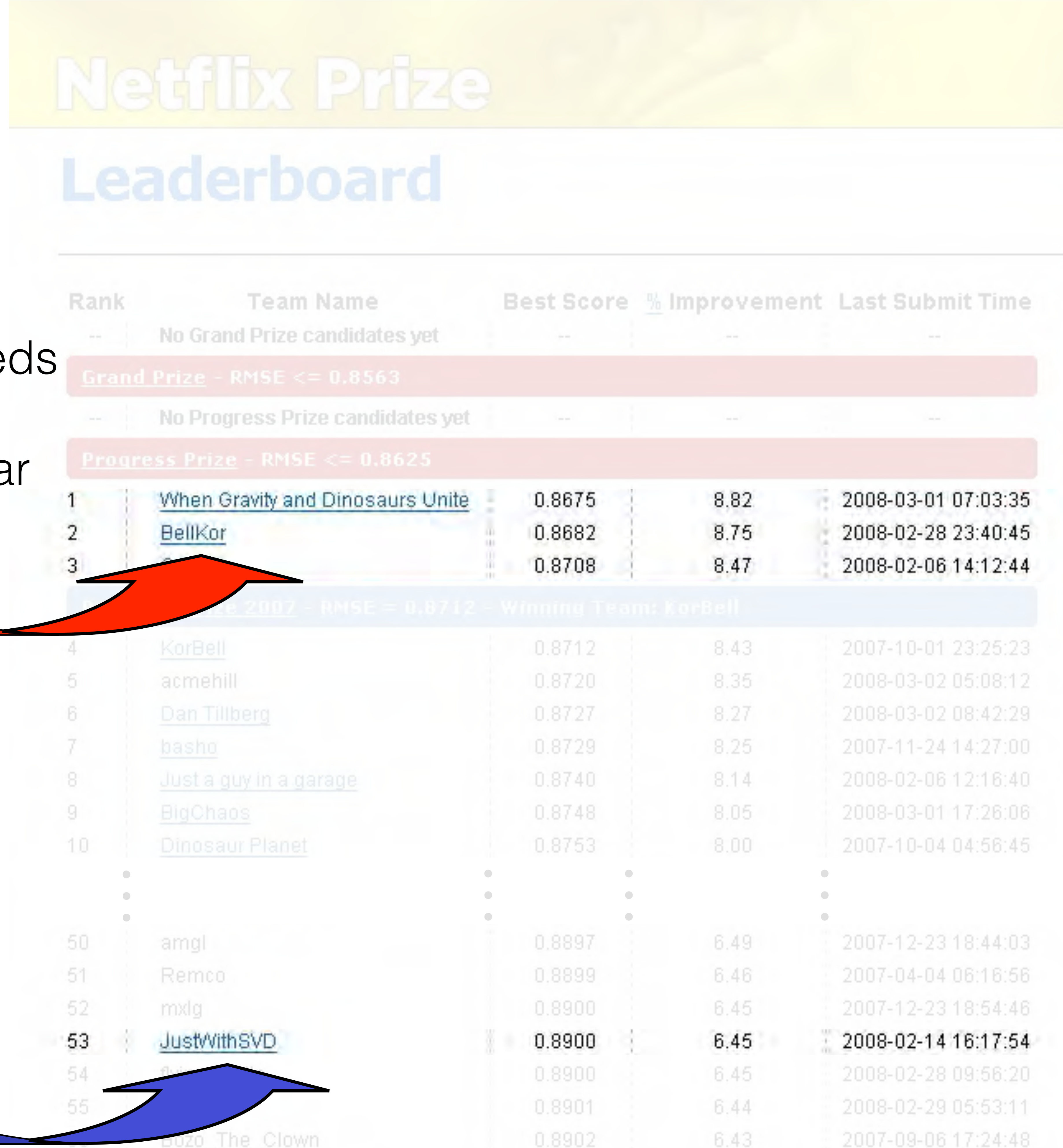
- Step 1: Pick (u,v) and compute residual:

$$e = (\mathbf{L}_u \mathbf{R}_v^T - M_{uv})$$

- Step 2: Take a mixture of current model and corrected model:

$$\begin{bmatrix} \mathbf{L}_u \\ \mathbf{R}_v \end{bmatrix} \leftarrow \begin{bmatrix} (1-\gamma\mu_u)\mathbf{L}_u - \gamma e \mathbf{R}_v \\ (1-\gamma\mu_v)\mathbf{R}_v - \gamma e \mathbf{L}_u \end{bmatrix}$$

Mixture of hundreds of models, including nuclear norm



Netflix Prize

Leaderboard

Rank	Team Name	Best Score	% Improvement	Last Submit Time
--	No Grand Prize candidates yet	--	--	--
Grand Prize - RMSE <= 0.8563				
--	No Progress Prize candidates yet	--	--	--
Progress Prize - RMSE <= 0.8625				
1	When Gravity and Dinosaurs Unite	0.8675	8.82	2008-03-01 07:03:35
2	BellKor	0.8682	8.75	2008-02-28 23:40:45
3	[illegible]	0.8708	8.47	2008-02-06 14:12:44
Progress Prize 2007 - RMSE = 0.8712 - Winning Team: KorBell				
4	KorBell	0.8712	8.43	2007-10-01 23:25:23
5	acmehill	0.8720	8.35	2008-03-02 05:08:12
6	Dan Tillberg	0.8727	8.27	2008-03-02 08:42:29
7	basho	0.8729	8.25	2007-11-24 14:27:00
8	Just a guy in a garage	0.8740	8.14	2008-02-06 12:16:40
9	BigChaos	0.8748	8.05	2008-03-01 17:26:06
10	Dinosaur Planet	0.8753	8.00	2007-10-04 04:56:45
⋮	⋮	⋮	⋮	⋮
50	amgl	0.8897	6.49	2007-12-23 18:44:03
51	Remco	0.8899	6.46	2007-04-04 06:16:56
52	mxlg	0.8900	6.45	2007-12-23 18:54:46
53	JustWithSVD	0.8900	6.45	2008-02-14 16:17:54
54	[illegible]	0.8900	6.45	2008-02-28 09:56:20
55	[illegible]	0.8901	6.44	2008-02-29 05:53:11
[illegible]	Bozo_The_Clown	0.8902	6.43	2007-09-06 17:24:48

nuclear norm (a.k.a. SVD)

SGD and BIG Data

minimize $\mathbb{E}_{\xi}[f(x,\xi)]$

For each k, sample ξ_k and compute $x[k+1] = x[k] - \alpha_k \nabla_x f(x[k], \xi_k)$

Ideal for big data analysis:

- small, predictable memory footprint
- robustness against noise in data
- rapid learning rates
- one algorithm!

Why should this work?

Example: Computing the mean

$$\text{minimize} \quad \sum_{k=1}^{4} (x-k)^2$$

Stepsize = 1/2k

$$x_0 = 0$$

$$x_1 = x_0 - (x_0 - 1) = 1$$

$$x_2 = x_1 - (x_1 - 2)/2 = 1.5$$

$$x_3 = x_2 - (x_2 - 3)/3 = 2$$

$$x_4 = x_3 - (x_3 - 4)/4 = 2.5$$

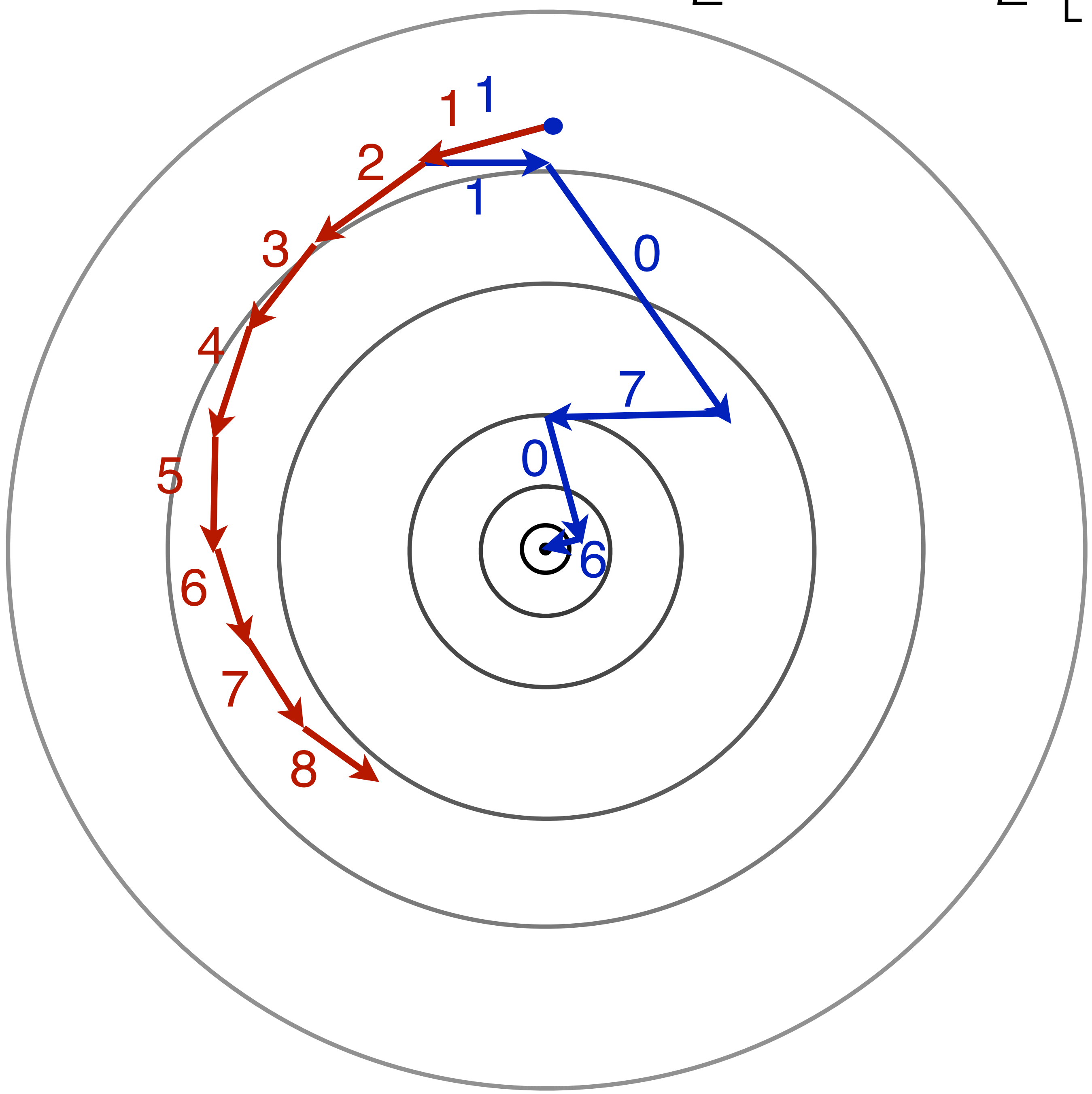
In general, if we minimize $\sum_{k=1}^{N} (x - z_k)^2$

SGD returns: $x_N = \frac{1}{N} \sum_{k=1}^{N} z_k$

$$\text{minimize} \quad \sum_{k=0}^{9} \left(\cos\left(\tfrac{\pi k}{10}\right) x_1 + \sin\left(\tfrac{\pi k}{10}\right) x_2 \right)^2 = 5x_1^2 + 5x_2^2$$

Stepsize = 1/2

$$x - \frac{1}{2}\nabla f_j(x) = \frac{1}{2} \begin{bmatrix} 1 - c_j & -s_j \\ -s_j & 1 + c_j \end{bmatrix} x$$



Choose directions in order

Choose a direction uniformly with replacement

convergence of sgd

minimize $\quad f(x)$

Assume f is strongly convex with parameter ℓ and has Lipschitz gradients with parameter L

Assume at each iteration we sample $G(x)$, an unbiased estimate of $\nabla f(x)$, independent of x and the past iterates

Assume $\|G(x)\| \leq M$ almost surely.

$$x[k+1] = x[k] - \alpha_k G_k(x[k])$$

$$\begin{aligned}
&\|x[k+1] - x_\star\|_2^2 \\
&\qquad\qquad = \|x[k] - \alpha_k G_k(x[k]) - x_\star\|^2 \\
&\qquad\qquad = \|x[k] - x_\star\|_2^2 - 2\alpha_k (x[k] - x^*)^T G_k(x[k]) + \alpha_k^2 \|G_k(x[k])\|^2.
\end{aligned}$$

$$\boxed{\text{Define} \quad a_k = \mathbb{E}\left[\|x[k] - x_\star\|_2^2\right]}$$

$$a_{k+1} \leq a_k - 2\alpha_k \mathbb{E}[(x[k] - x^*)^T G_k(x[k])] + \alpha_k^2 M^2.$$

By iterating expectation:

$$\begin{aligned}
\mathbb{E}[(x[k] - x_\star)^T G_k(x[k])] &= \mathbb{E}_{G_{[k-1]}} \mathbb{E}_{G_k}[(x[k] - x_\star)^T G_k(x[k]) | G_{[k-1]}] \\
&= \mathbb{E}[(x[k] - x_\star)^T \nabla f(x[k])]
\end{aligned}$$

By strong convexity:

$$\nabla f(x[k])^T (x[k] - x_\star) \geq f(x[k]) - f(x_\star) + \frac{\ell}{2}\|x_k - x^*\|^2 \geq \ell \|x_k - x^*\|^2.$$

$$\boxed{a_{k+1} \leq (1 - 2\ell\alpha_k) a_k + \alpha_k^2 M^2}$$

$$a_{k+1} \leq (1 - 2\ell\alpha_k)a_k + \alpha_k^2 M^2$$

Large steps: $\theta > \frac{1}{2\ell}$, $\alpha_k = \frac{\theta}{k}$

$$\mathbb{E}[\|x[k] - x_\star\|_2^2] \leq \frac{1}{k} \cdot \max\left\{\frac{\theta^2 M^2}{2\ell\theta - 1}, \|x[0] - x_\star\|^2\right\}$$

Small steps: $\alpha < \frac{1}{2\ell}$, constant stepsize

$$\mathbb{E}[\|x[k] - x_\star\|_2^2] \leq (1 - 2\ell\alpha)^k \left(\|x[0] - x_\star\|^2 - \frac{\alpha M^2}{2\ell}\right) + \frac{\alpha M^2}{2\ell}$$

Achieves 1/k rate if run in *epochs* of diminishing stepsize

$$\text{minimize}_{x \in \mathbb{R}^d} \quad f(x) = \sum_{j=1}^{N} f_j(x)$$

Algorithm	Time per iteration	Error after T iterations	**Error after N items**
Newton	$O(d^2N+d^3)$	$C_I^{2^T}$	$C_I^{\,2}$
Gradient	$O(dN)$	$C_G^{\,T}$	C_G
SGD	$O(d)$ (or constant)	$\frac{C_S}{T}$	$\frac{C_S}{N}$

extensions

- non-smooth, non-strongly convex ($1/\sqrt{k}$)
- non-convex (converges asymptotically)
- stochastic coordinate descent (special decomposition of *f*)
- parallelization

projected gradient

smooth

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in \Omega \end{array}$$

convex

Suppose it is easy to solve

$$\Pi_\Omega(y) \leftarrow \begin{array}{ll} \text{minimize} & \|x - y\| \\ \text{subject to} & x \in \Omega \end{array}$$

unique solution

projected gradient method:

$$\boxed{x[k+1] \leftarrow \Pi_\Omega\left(x[k] + \alpha_k v[k]\right)}$$

$$x[k+1] \leftarrow \Pi_\Omega\left(x[k] + \alpha_k v[k]\right)$$

$$\boxed{\text{Key Lemma: } \|\Pi_\Omega(x) - \Pi_\Omega(z)\| \leq \|x - z\|}$$

Assume minimizer of $f \in \mathbf{\Omega}$

Assume f is strongly convex

$$\begin{aligned}
\|x[k+1] - x_\star\| &= \|\Pi_\Omega(x[k] - \alpha \nabla f(x([k])) - \Pi_\Omega(x_\star)\| && \\
&\leq \|x[k] - \alpha \nabla f(x[k]) - x_\star\| && \text{non-expansive} \\
&= \|\psi(x[k]) - \psi(x_\star)\| && \psi(x_\star) = x_\star \\
&\leq \beta \|x[k] - x_\star\| && \text{contractivity} \\
&\vdots && \\
&\leq \beta^{k+1} \|x[0] - x_\star\| && \textit{linear rate}
\end{aligned}$$

$$\text{minimize} \quad f(x) + P(x)$$

$$f(x) + P(x) \approx f(x[k]) + \nabla f(x[k])^T (x - x[k]) + \frac{1}{2\alpha}\|x - x[k]\|^2 + P(x)$$

Define $\quad \mathrm{prox}_P(x) = \arg\min_z \frac{1}{2}\|x - z\|^2 + P(z)$

Solving the approximation yields

$$x[k+1] = \mathrm{prox}_{\alpha_k P}(x[k] - \alpha_k \nabla f(x[k]))$$

proximal mapping

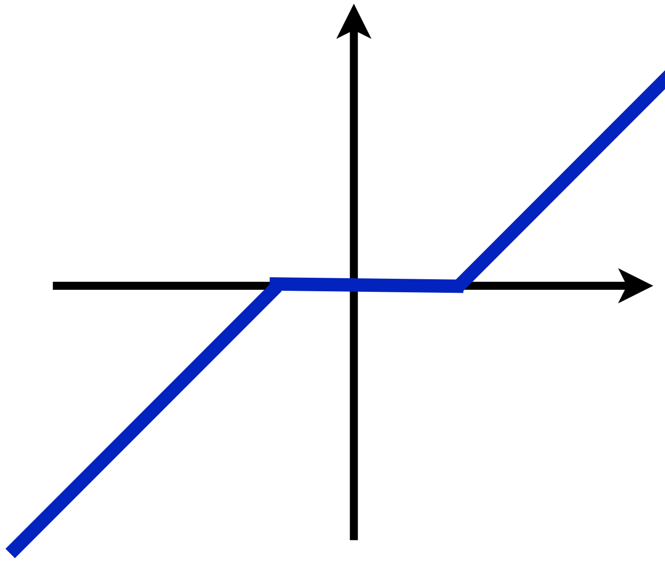
$$\mathrm{prox}_P(x) = \arg\min_z \tfrac{1}{2}\|x - z\|^2 + P(z)$$

$$P(x) = \begin{cases} 0 & x \in \Omega \\ \infty & x \notin \Omega \end{cases}$$

$$\mathrm{prox}_P(x) = \Pi_\Omega(x)$$

$$P(x) = \mu\|x\|_1$$

$$\mathrm{prox}_P(x)_i = \begin{cases} x_i + \mu & x_i < -\mu \\ 0 & -\mu \le x_i \le \mu \\ x_i - \mu & x_i > \mu \end{cases}$$



$$\text{minimize} \quad f(x) + P(x)$$

$$f(x) + P(x) \approx f(x[k]) + \nabla f(x[k])^T (x - x[k]) + \frac{1}{2\alpha}\|x - x[k]\|^2 + P(x)$$

Define $\quad \mathrm{prox}_P(x) = \arg\min_z \frac{1}{2}\|x - z\|^2 + P(z)$

Solving the approximation yields $\quad x[k+1] = \mathrm{prox}_{\alpha_k P}(x[k] - \alpha_k \nabla f(x[k]))$

Key Lemma: $\|\mathrm{prox}_P(x) - \mathrm{prox}_P(y)\| \leq \|x - y\|$

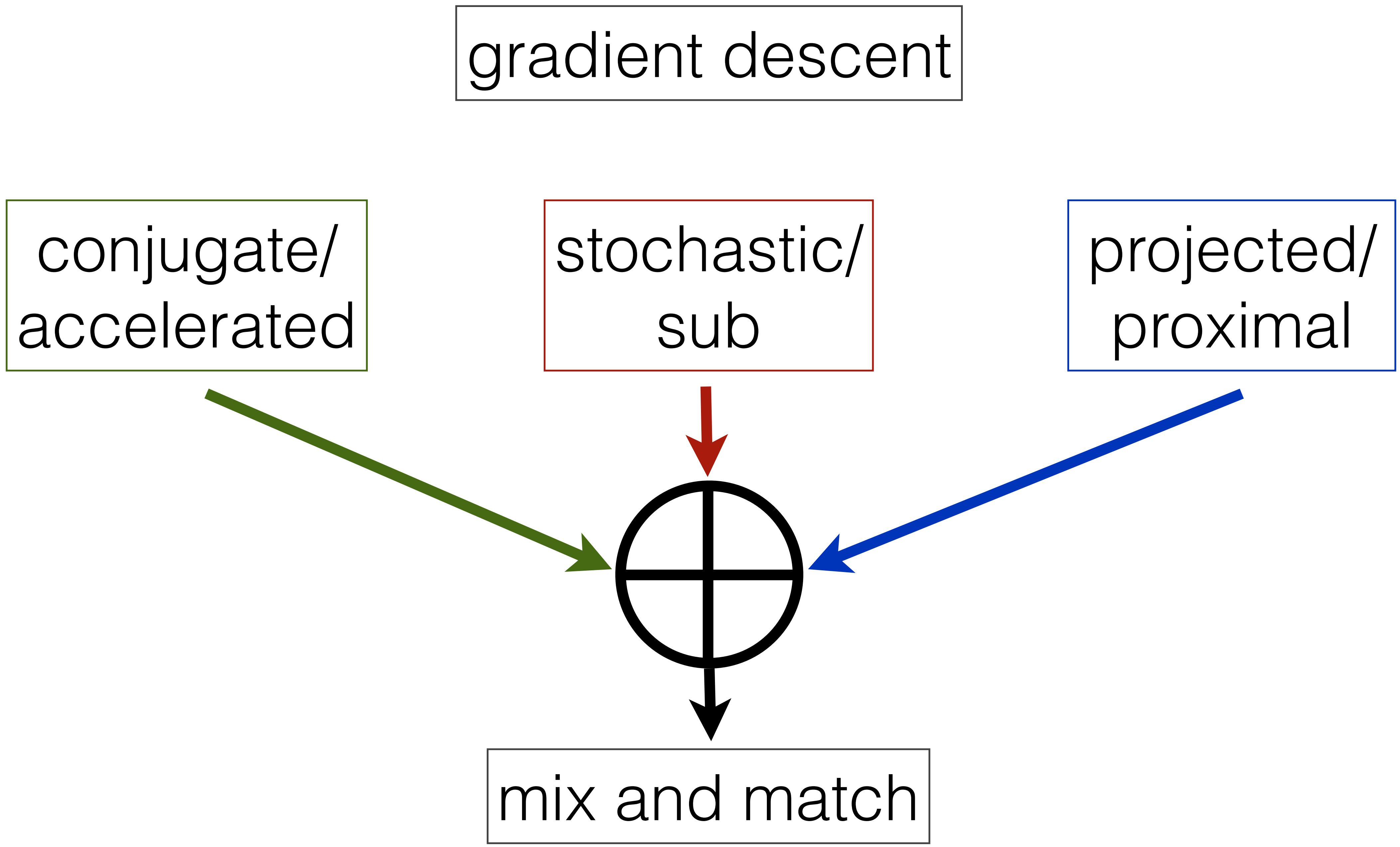
- immediately implies earlier analysis works for proximal gradient.
- projected gradient is a special case
- inherits rates of convergence from f (i.e., P=0)

More variants

- mirror descent: use a general distance
$$f(x) \approx f(x_0) + \langle \nabla f(x_0), x - x_0 \rangle + \frac{1}{2\alpha}\mathcal{D}(x, x_0)$$

- ADMM: combine multiple prox operators for complicated constraints.

$$x[k+1] \leftarrow x[k] + \alpha_k v[k]$$



Everything here combines, and you get the expected rates out.