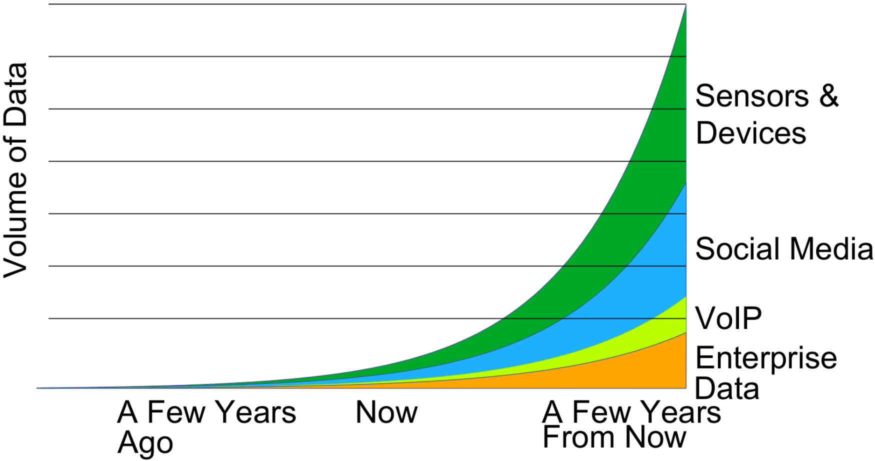
Statistics meets Optimization: Fast randomized algorithms for large data sets

Martin Wainwright

UC Berkeley
Statistics and EECS

Joint work with: Mert Pilanci
UC Berkeley & Stanford University

What is the "big data" phenomenon?



- Every day: 2.5 billion gigabytes of data created
- Last two years: creation of 90% of the world's data (source: IBM)

How can algorithms be scaled?

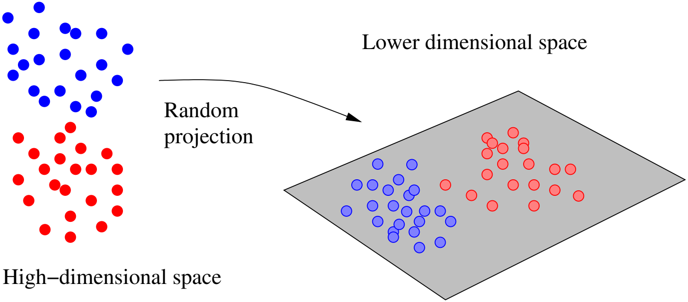
Massive data sets require fast algorithms but with rigorous guarantees.

How can algorithms be scaled?

Massive data sets require fast algorithms but with rigorous guarantees.

Randomized projection is a general purpose tool:

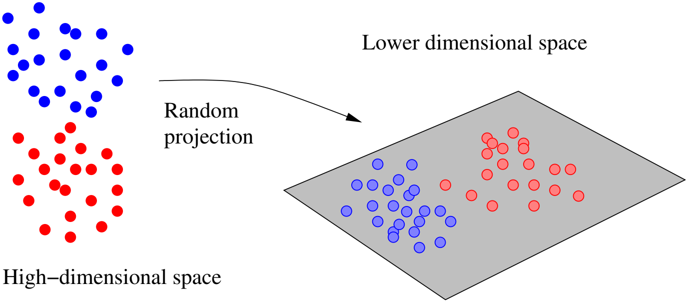
- Choose a random subspace of "low" dimension m.
- Project data into subspace, and solve reduced dimension problem.



How can algorithms be scaled?

Randomized projection is a general purpose tool:

- Choose a random subspace of "low" dimension m.
- Project data into subspace, and solve reduced dimension problem.

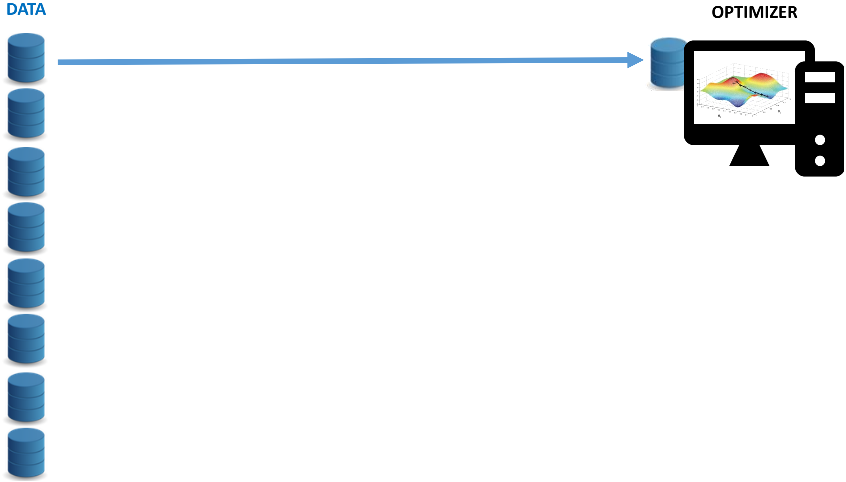


Widely studied and used:

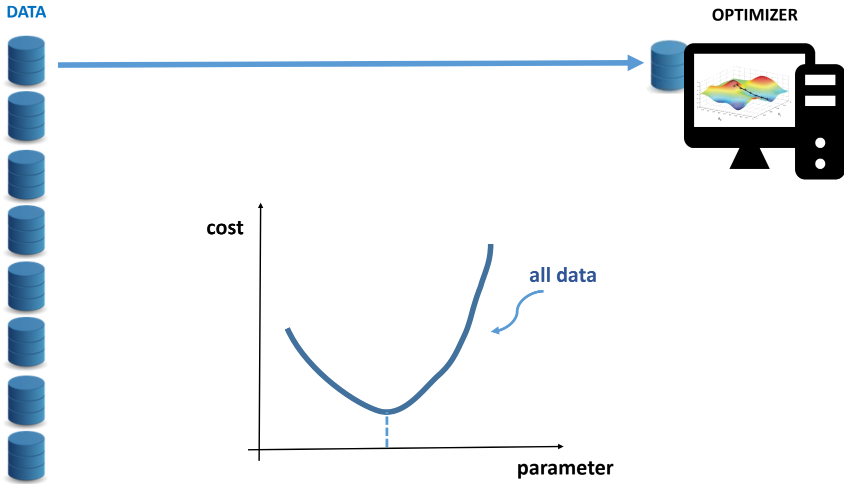
- Johnson & Lindenstrauss (1984): in Banach/Hilbert space geometry
- various surveys and books: Vempala, 2004; Mahoney et al., 2011 Cormode et al., 2012.

Randomized sketching for optimization

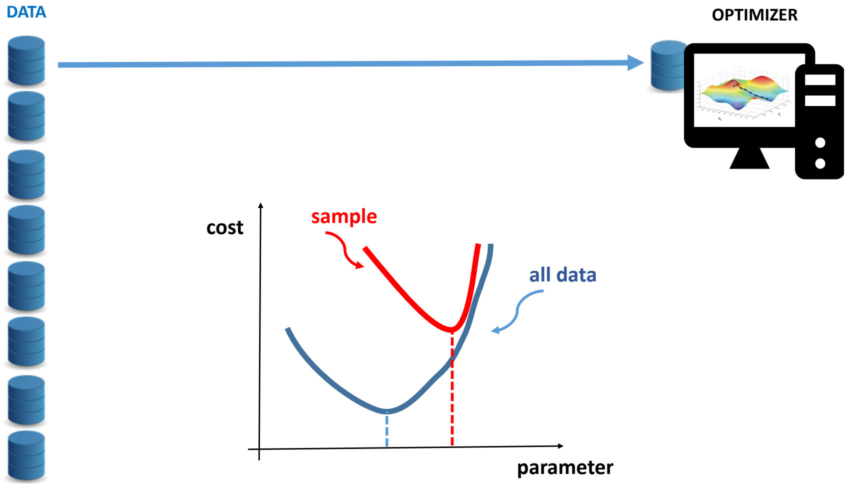
Randomized sketching for optimization



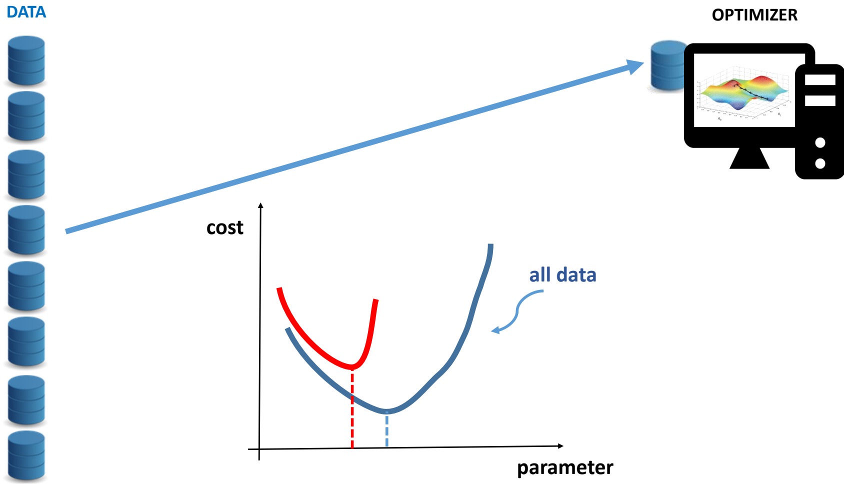
Randomized sketching for optimization



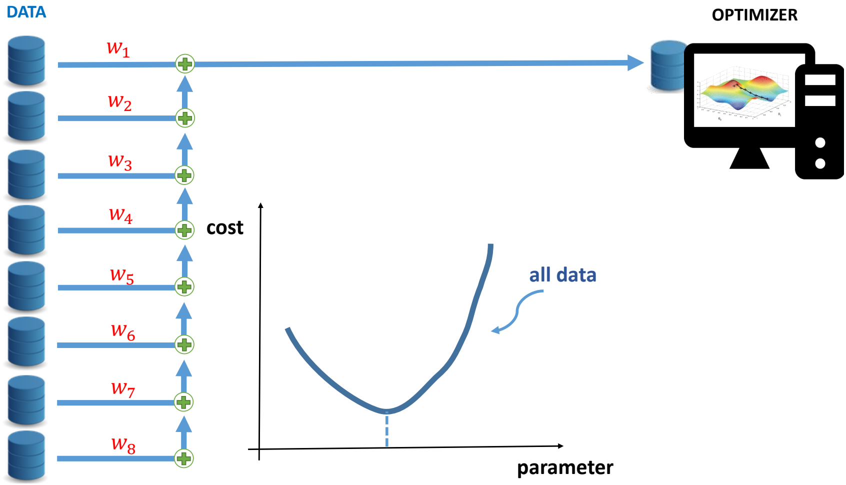
Randomized sketching for optimization



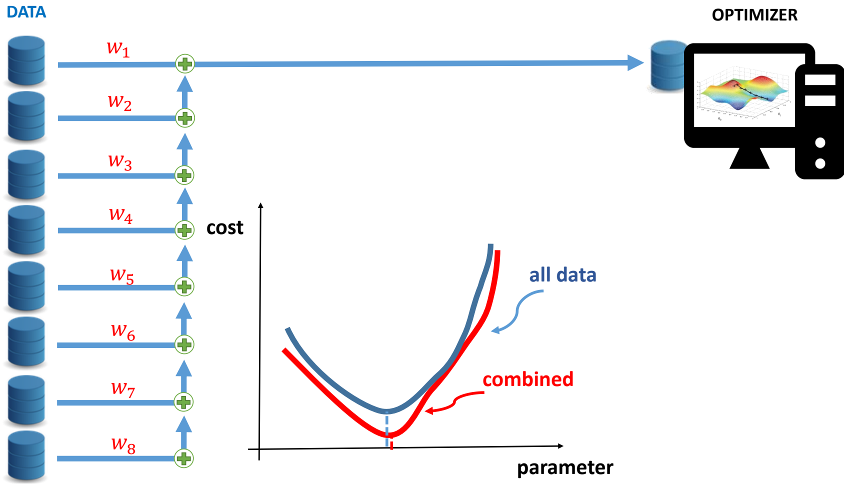
Randomized sketching for optimization



Randomized sketching for optimization



Randomized sketching for optimization



Randomized projection for constrained least-squares

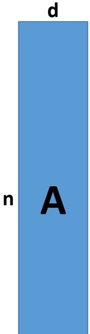
- Given data matrix $A \in \mathbb{R}^{n \times d}$, and response vector $y \in \mathbb{R}^n$
- Least-squares over convex constraint set $\mathcal{C} \subseteq \mathbb{R}^d$:

$$x_{\text{LS}} = \arg\min_{x \in \mathcal{C}} \underbrace{\|Ax - y\|_2^2}_{f(Ax)}$$

Randomized projection for constrained least-squares

- Given data matrix $A \in \mathbb{R}^{n \times d}$, and response vector $y \in \mathbb{R}^n$
- Least-squares over convex constraint set $\mathcal{C} \subseteq \mathbb{R}^d$:

$$x_{\text{LS}} = \arg\min_{x \in \mathcal{C}} \underbrace{\|Ax - y\|_2^2}_{f(Ax)}$$



Randomized projection for constrained least-squares

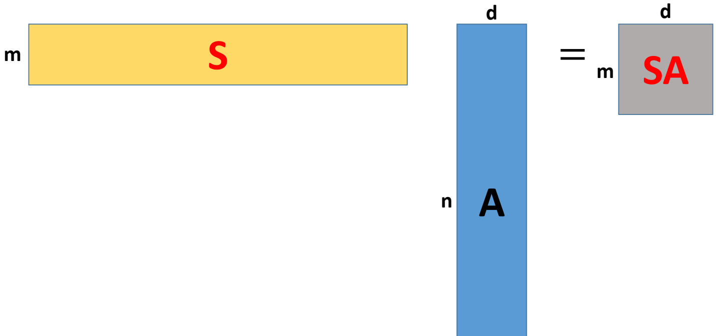
- Given data matrix $A \in \mathbb{R}^{n \times d}$, and response vector $y \in \mathbb{R}^n$
- Least-squares over convex constraint set $\mathcal{C} \subseteq \mathbb{R}^d$:

$$x_{\mathrm{LS}} = \arg\min_{x \in \mathcal{C}} \underbrace{\|Ax - y\|_2^2}_{f(Ax)}$$

- Randomized approximation: (Sarlos, 2006)

$$\widehat{x} = \arg\min_{x \in \mathcal{C}} \|S(Ax - y)\|_2^2$$

- Random projection matrix $S \in \mathbb{R}^{m \times n}$



A general approximation-theoretic bound

The randomized solution $\widehat{x} \in \mathcal{C}$ provides δ-accurate cost approximation if

$$f(Ax_{\text{LS}}) \;\le\; f(A\widehat{x}) \;\le\; (1+\delta)\, f(Ax_{\text{LS}}).$$

A general approximation-theoretic bound

The randomized solution $\widehat{x} \in \mathcal{C}$ provides δ-accurate cost approximation if

$$f(Ax_{\text{LS}}) \;\leq\; f(A\widehat{x}) \;\leq\; (1+\delta)\, f(Ax_{\text{LS}}).$$

Theorem (Pilanci & W, 2015)

For a broad class of random projection matrices, a sketch dimension

$$m \gtrsim \frac{\text{effrank}(A; \mathcal{C})}{\delta}$$

yields δ-accurate cost approximation with exp. high probability.

A general approximation-theoretic bound

The randomized solution $\widehat{x} \in \mathcal{C}$ provides δ-accurate cost approximation if

$$f(Ax_{\text{LS}}) \;\leq\; f(A\widehat{x}) \;\leq\; (1+\delta)\, f(Ax_{\text{LS}}).$$

Theorem (Pilanci & W, 2015)

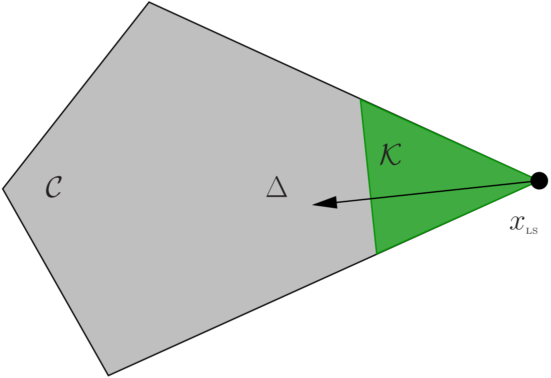
For a broad class of random projection matrices, a sketch dimension

$$m \gtrsim \frac{\text{effrank}(A;\mathcal{C})}{\delta}$$

yields δ-accurate cost approximation with exp. high probability.

- past work on unconstrained case $\mathcal{C} = \mathbb{R}^d$: effective rank equivalent to rank(A) (Sarlos, 2006; Mahoney et al. 2011)
- effective rank can be much smaller than standard rank

Favorable dependence on optimum x_{LS}

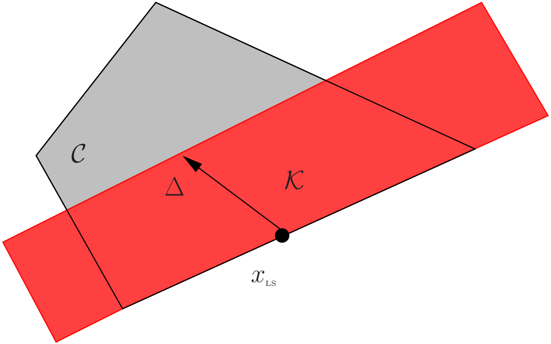


Tangent cone $\mathcal{K}$ at x_{LS}

Set of feasible directions at the optimum x_{LS}

$$\mathcal{K} = \left\{ \Delta \in \mathbb{R}^d \mid \Delta = t\,(x - x_{\text{LS}}) \quad \text{for some } x \in \mathcal{C}. \right\}.$$

Unfavorable dependence on optimum x_{LS}



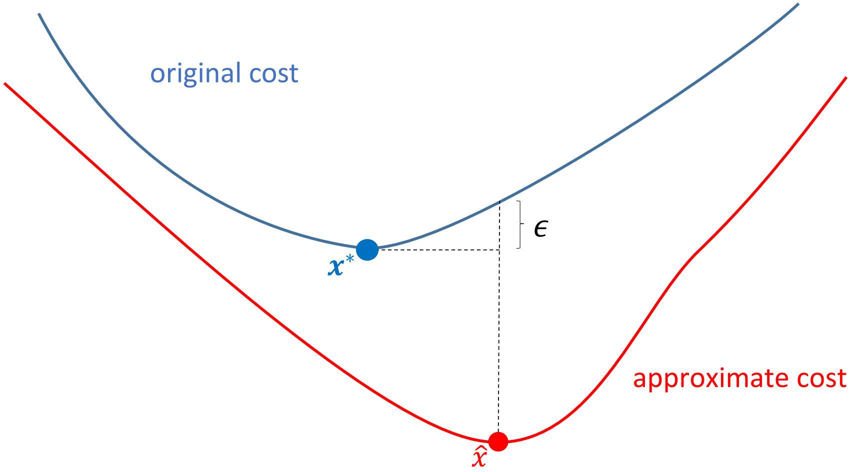
Tangent cone $\mathcal{K}$ at x_{LS}

Set of feasible directions at the optimum x_{LS}

$$\mathcal{K} = \left\{ \Delta \in \mathbb{R}^d \mid \Delta = t\,(x - x_{LS}) \quad \text{for some } x \in \mathcal{C}. \right\}.$$

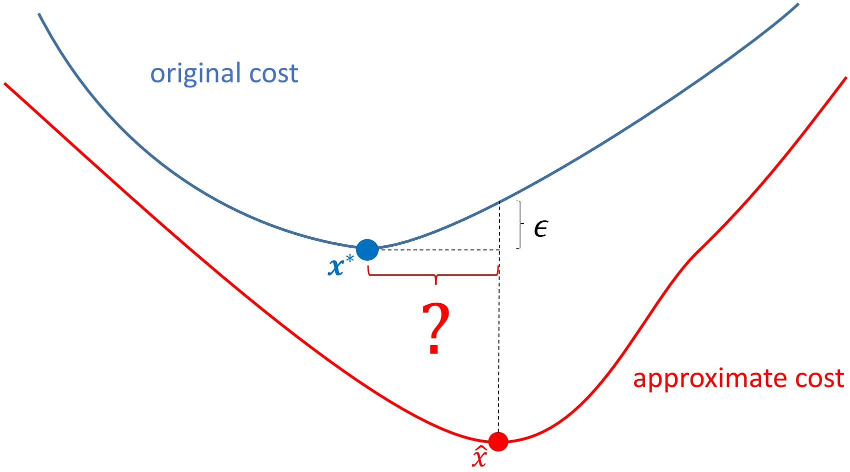
But what about solution approximation?

$$x^* = \arg\min_{x \in \mathcal{C}} \|Ax - y\|_2^2 \qquad \text{and} \qquad \widehat{x} \in \arg\min_{x \in \mathcal{C}} \|S(Ax - y)\|_2^2$$



But what about solution approximation?

$$x^* = \arg\min_{x \in \mathcal{C}} \|Ax - y\|_2^2 \qquad \text{and} \qquad \widehat{x} \in \arg\min_{x \in \mathcal{C}} \|S(Ax - y)\|_2^2$$

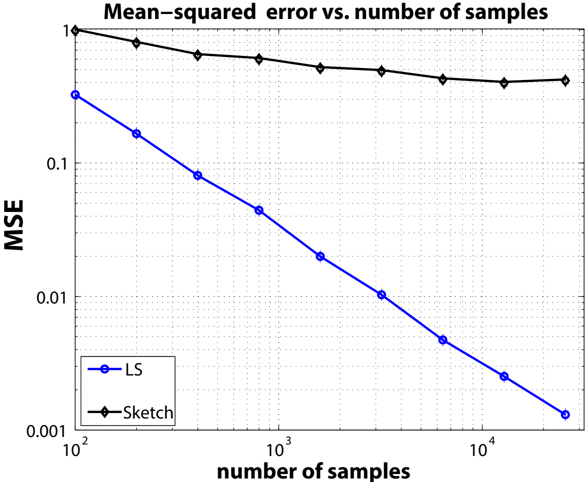


Failure of standard random projection

- Noisy observation model: $y = Ax^* + w$ where $w \sim N(0, \sigma^2 I_n)$.

Failure of standard random projection

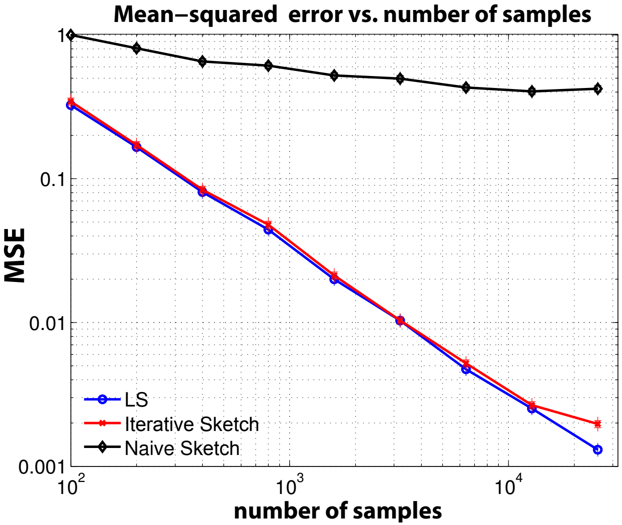
- Noisy observation model: $y = Ax^* + w$ where $w \sim N(0, \sigma^2 I_n)$.



- Least-squares accuracy: $\mathbb{E}\|x_{\rm LS} - x^*\|_A^2 = \sigma^2 \frac{\text{rank}(A)}{n}$

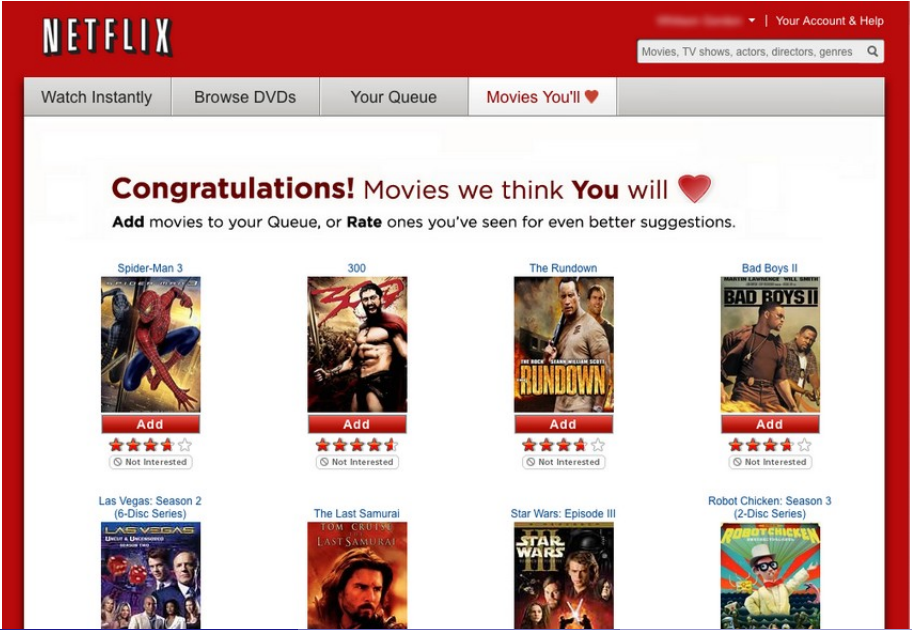
Overcoming this barrier?

Overcoming this barrier? Sequential scheme....



Iterative projection scheme yields accurate tracking of original least-squares solution.

Application to Netflix data



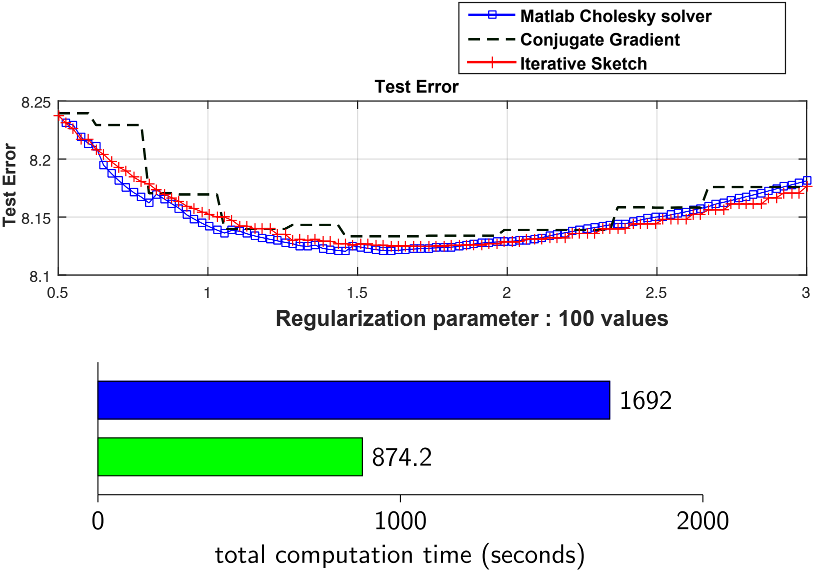
Netflix data set

- 2 million $\times$ 17000 matrix A of ratings (users $\times$ movies)
- Predict the ratings of a particular movie
- Least-squares regression with ℓ_2 regularization

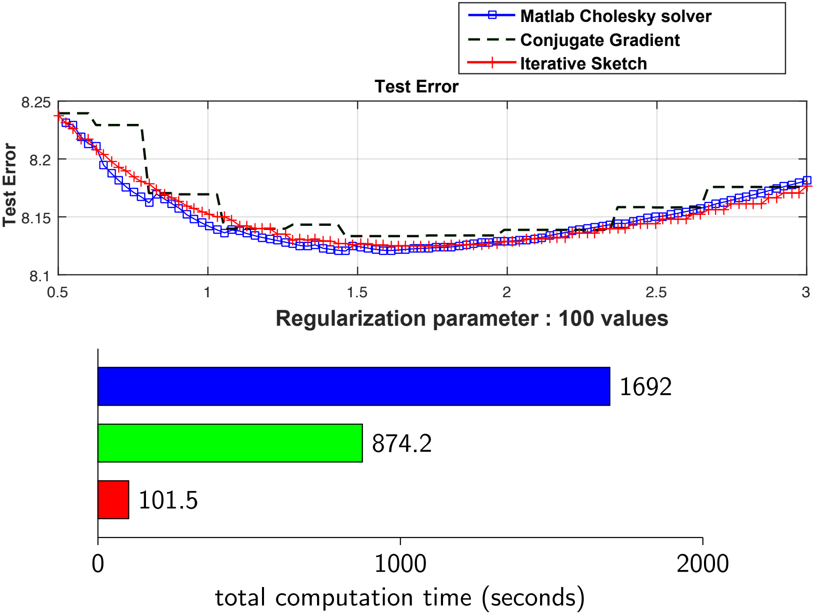
$$\min_x \|Ax - y\|_2^2 + \lambda \|x\|_2^2$$

- Partition into test and training sets, solve for all values of $\lambda \in \{1, 2, ..., 100\}$.

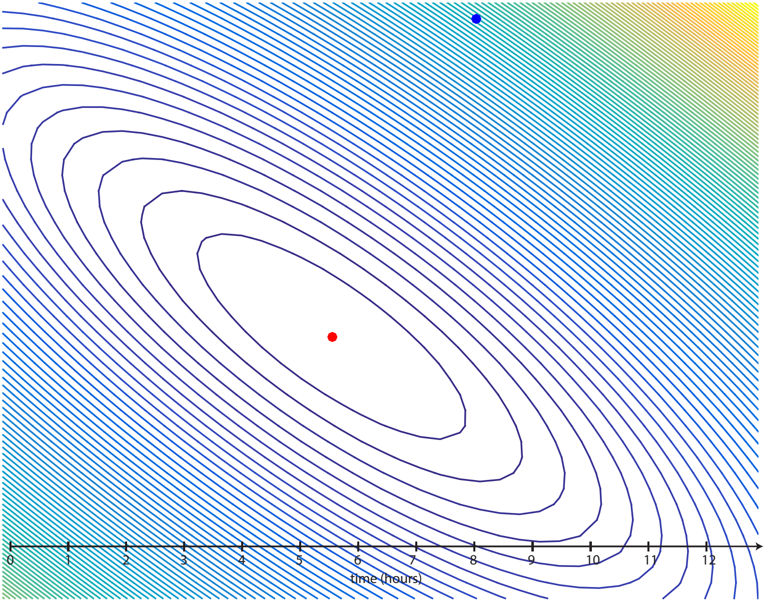
Fitting the full regularization path



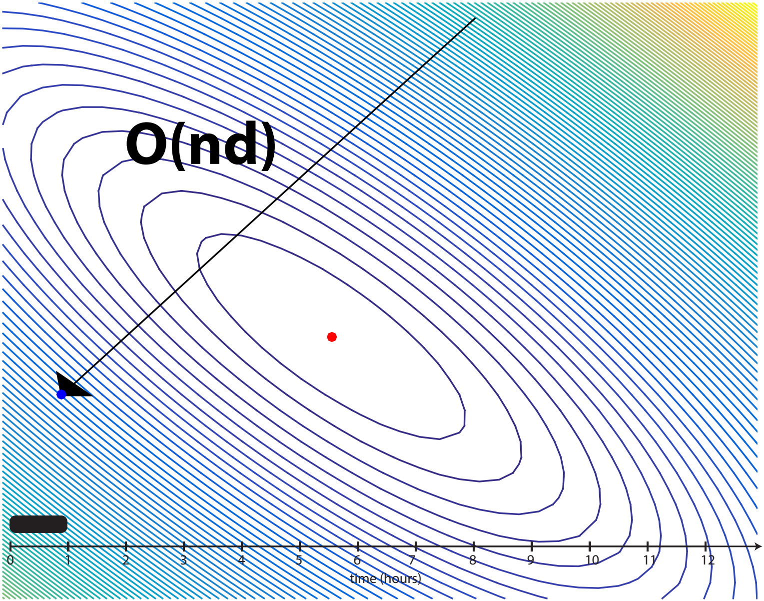
Fitting the full regularization path



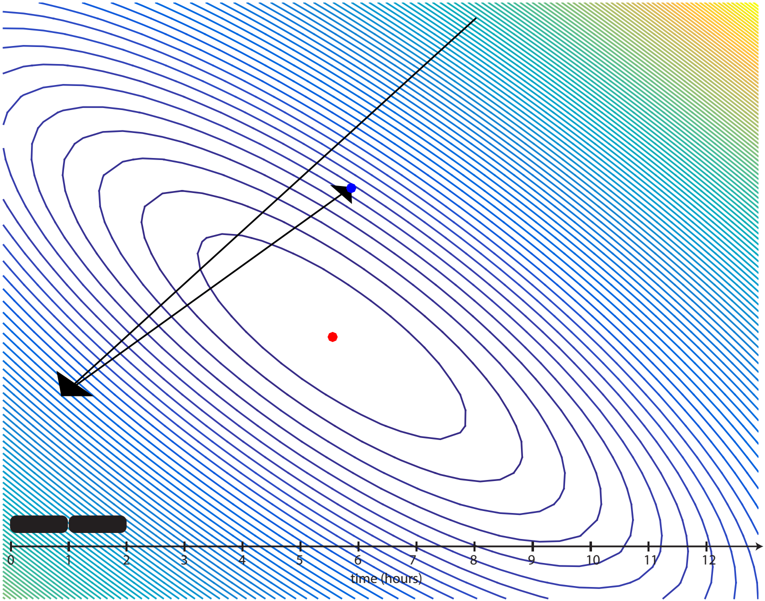
Gradient Descent



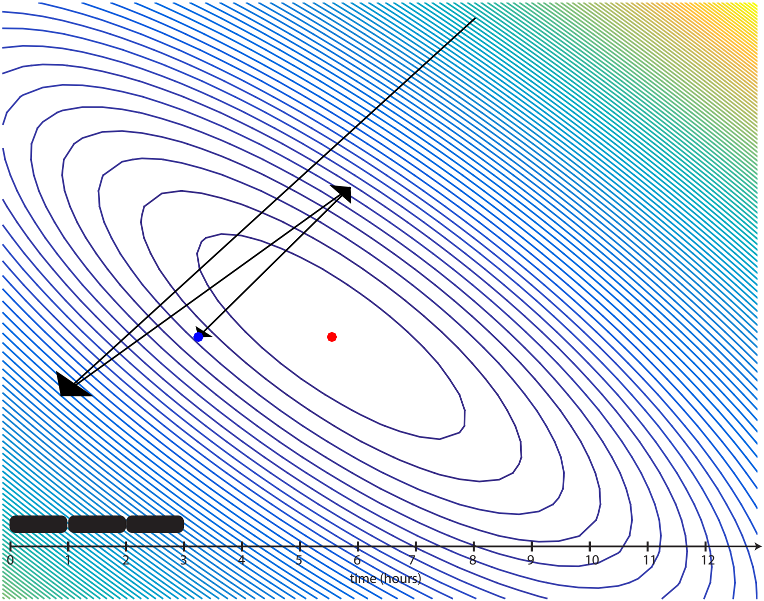
Gradient Descent



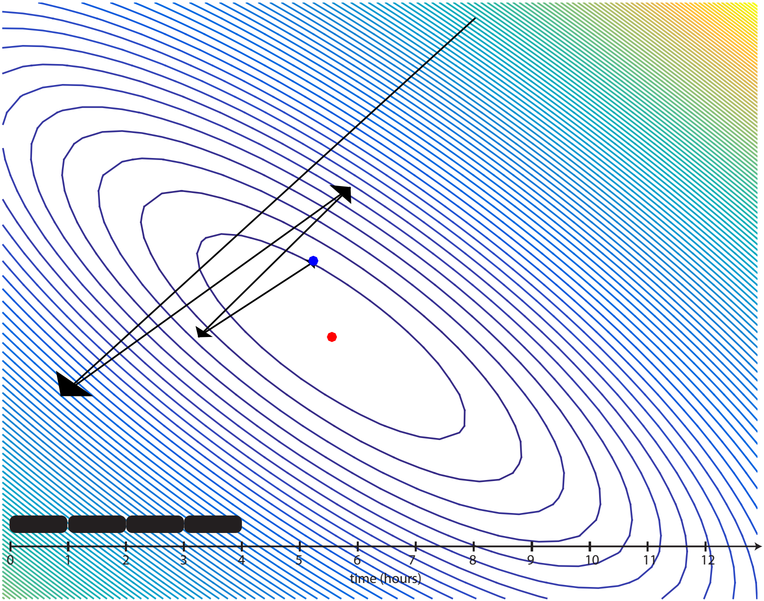
Gradient Descent



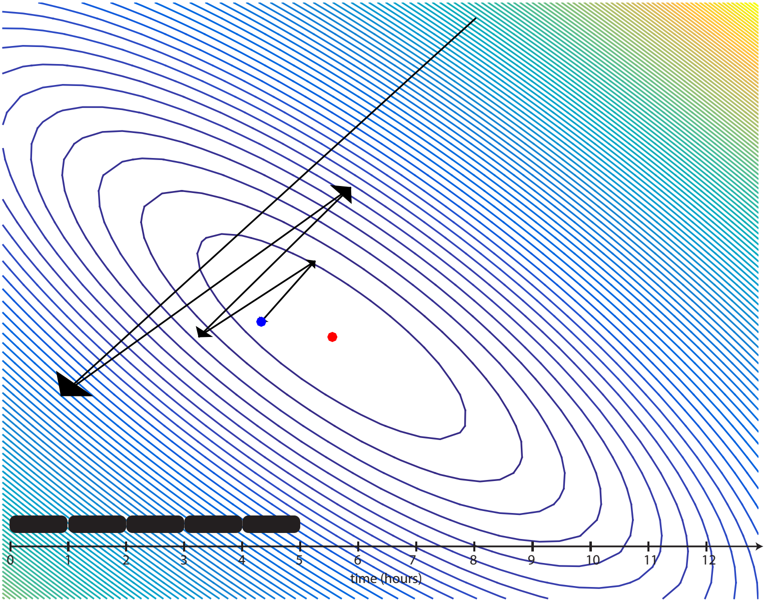
Gradient Descent



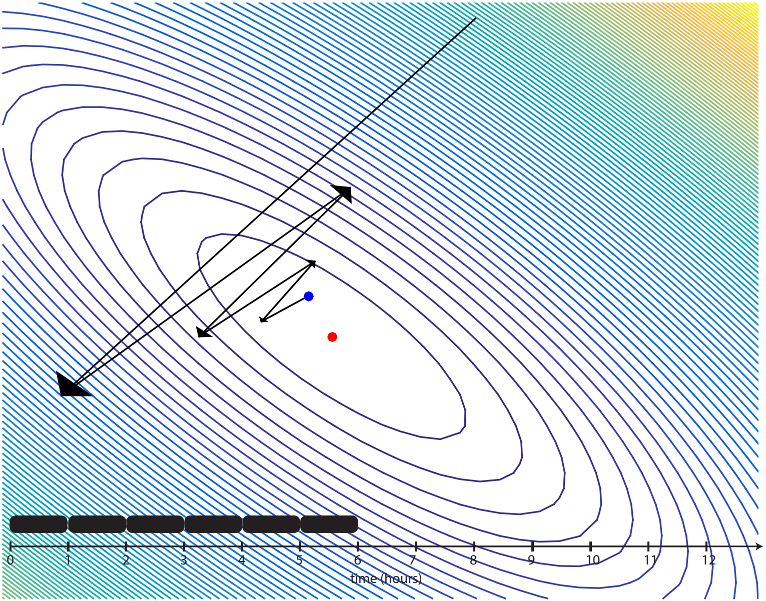
Gradient Descent



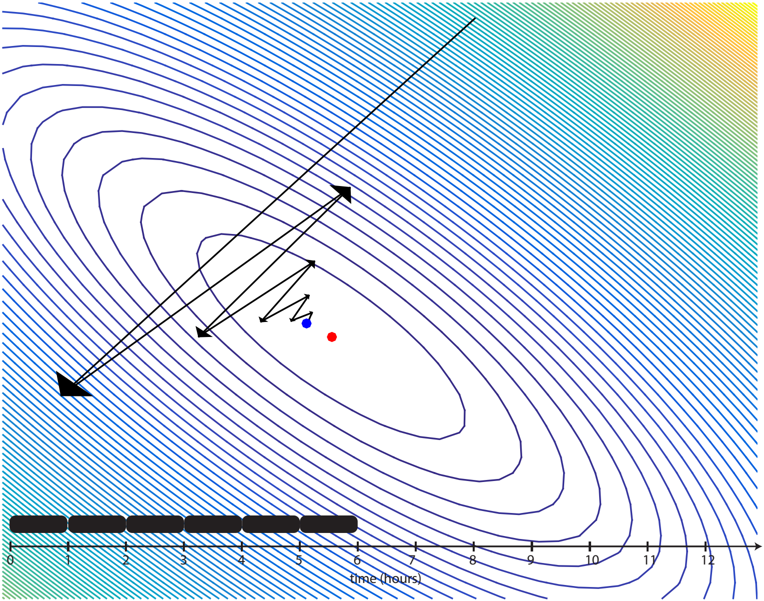
Gradient Descent



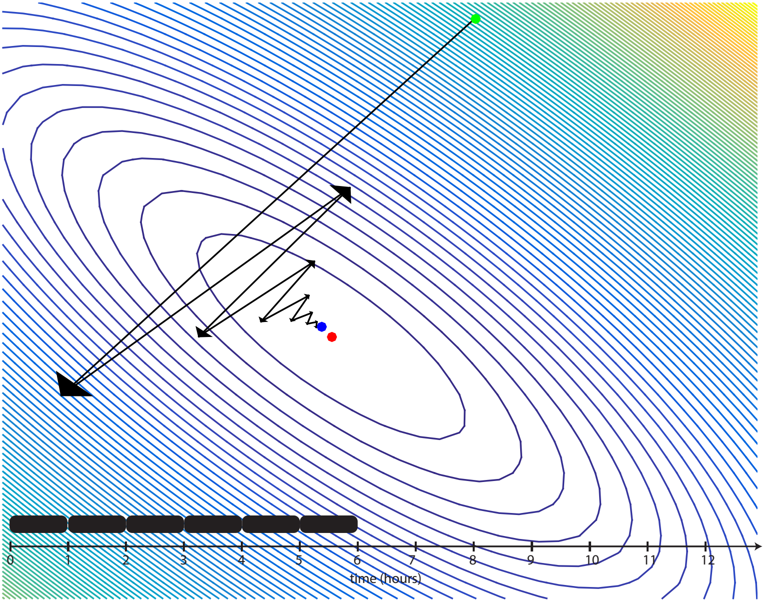
Gradient Descent



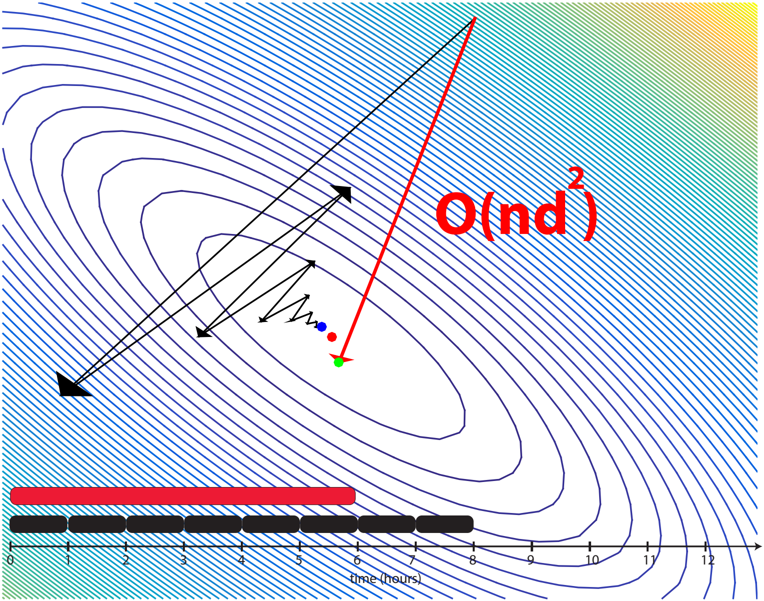
Gradient Descent vs



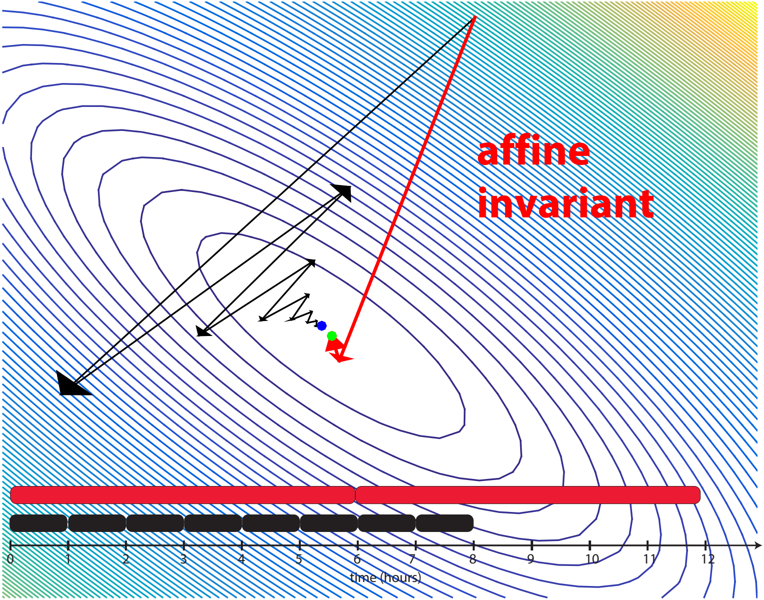
Gradient Descent vs Newton's Method



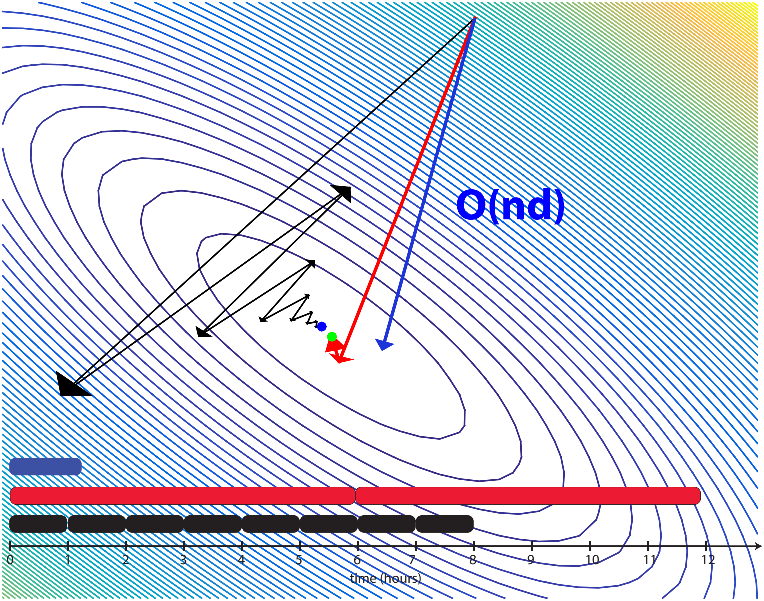
Gradient Descent vs Newton's Method



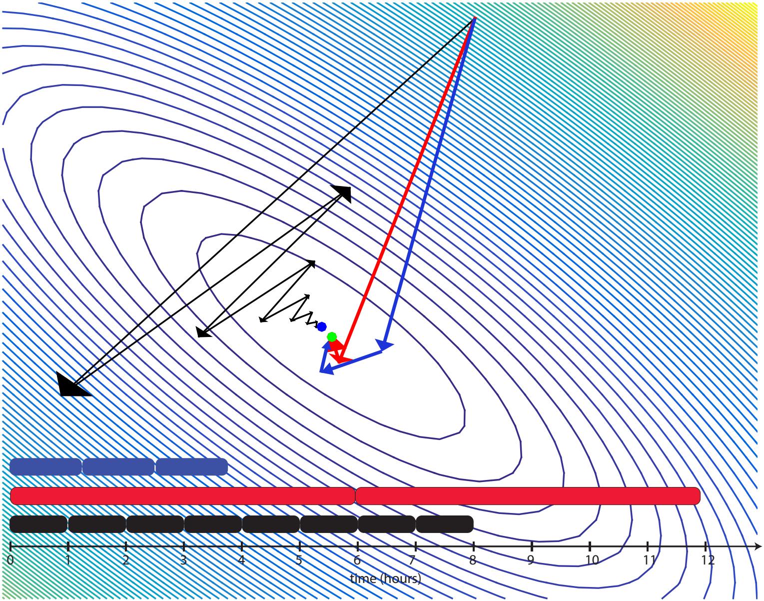
Gradient Descent vs Newton's Method



Gradient Descent vs Newton's Method



Gradient Descent vs Newton's Method



Exact and approximate forms of Newton's method

Minimize $g(x) = f(Ax)$ over convex set $\mathcal{C} \subseteq \mathbb{R}^d$:

$$x_{\text{opt}} = \arg\min_{x \in \mathcal{C}} g(x), \quad \text{where } g : \mathbb{R}^d \to \mathbb{R} \text{ is twice-differentiable.}$$

Exact and approximate forms of Newton's method

Minimize $g(x) = f(Ax)$ over convex set $\mathcal{C} \subseteq \mathbb{R}^d$:

$$x_{\text{opt}} = \arg\min_{x \in \mathcal{C}} g(x), \quad \text{where } g : \mathbb{R}^d \to \mathbb{R} \text{ is twice-differentiable.}$$

Ordinary Newton steps:

$$x^{t+1} = \arg\min_{x \in \mathcal{C}} \Big\{ \frac{1}{2}(x - x^t)^T \nabla^2 g(x^t)(x - x^t)\|_2^2 + \langle \nabla g(x^t),\, x - x^t \rangle \Big\},$$

where $\nabla^2 g(x^t)$ is Hessian at x^t.

Exact and approximate forms of Newton's method

Minimize $g(x) = f(Ax)$ over convex set $\mathcal{C} \subseteq \mathbb{R}^d$:

$$x_{\text{opt}} = \arg\min_{x \in \mathcal{C}} g(x), \quad \text{where } g : \mathbb{R}^d \to \mathbb{R} \text{ is twice-differentiable.}$$

Ordinary Newton steps:

$$x^{t+1} = \arg\min_{x \in \mathcal{C}} \Big\{ \frac{1}{2}(x - x^t)^T \nabla^2 g(x^t)(x - x^t)\|_2^2 + \langle \nabla g(x^t), x - x^t \rangle \Big\},$$

where $\nabla^2 g(x^t)$ is Hessian at x^t.

Approximate Newton steps:

- various types of quasi-Newton updates: Nocedal & Wright book: Chap. 6
- BFGS method; SR1 method etc.
- stochastic gradient + stochastic quasi-Newton (e.g., Byrd, Hansen, Nocedal & Singer 2014)

Iterative sketching for general convex functions

$$x_{\text{opt}} = \arg\min_{x \in \mathcal{C}} g(x), \quad \text{where } g : \mathbb{R}^d \to \mathbb{R} \text{ is twice-differentiable.}$$

Iterative sketching for general convex functions

$$x_{\text{opt}} = \arg\min_{x \in \mathcal{C}} g(x), \quad \text{where } g : \mathbb{R}^d \to \mathbb{R} \text{ is twice-differentiable.}$$

Ordinary Newton steps:

$$x^{t+1} = \arg\min_{x \in \mathcal{C}} \Big\{ \frac{1}{2} \|\nabla^2 g(x^t)^{1/2}(x - x^t)\|_2^2 + \langle \nabla g(x^t),\, x - x^t \rangle \Big\},$$

where $\nabla^2 g(x^t)^{1/2}$ is matrix square root Hessian at x^t.
Cost per step: $\mathcal{O}(nd^2)$ in unconstrained case.

Iterative sketching for general convex functions

$$x_{\text{opt}} = \arg\min_{x \in \mathcal{C}} g(x), \quad \text{where } g : \mathbb{R}^d \to \mathbb{R} \text{ is twice-differentiable.}$$

Ordinary Newton steps:

$$x^{t+1} = \arg\min_{x \in \mathcal{C}} \Big\{ \frac{1}{2} \|\nabla^2 g(x^t)^{1/2}(x - x^t)\|_2^2 + \langle \nabla g(x^t),\, x - x^t \rangle \Big\},$$

where $\nabla^2 g(x^t)^{1/2}$ is matrix square root Hessian at x^t.
Cost per step: $\mathcal{O}(nd^2)$ in unconstrained case.

Sketched Newton steps: Using random sketch matrix S^t:

$$\tilde{x}^{t+1} = \arg\min_{x \in \mathcal{C}} \Big\{ \frac{1}{2} \|S^t \nabla^2 g(x^t)^{1/2}(x - \tilde{x}^t)\|_2^2 + \langle \nabla g(\tilde{x}^t),\, x - \tilde{x}^t \rangle \Big\}.$$

Cost per step: $\widetilde{\mathcal{O}}(nd)$ in unconstrained case.

Convergence of Newton sketch

Run algorithm with sketch dimension $m \asymp d$ on a self-concordant function $g(x) = f(Ax)$, and data matrix $A \in \mathbb{R}^{n \times d}$ with $n \gg d$.

Convergence of Newton sketch

Run algorithm with sketch dimension $m \asymp d$ on a self-concordant function $g(x) = f(Ax)$, and data matrix $A \in \mathbb{R}^{n \times d}$ with $n \gg d$.

Theorem (Pilanci & W, 2015)

With probability at least $1 - c_0 e^{-c_1 m}$, number of iterations required for ϵ accuracy is less than

$$c_2 \log(1/\epsilon)$$

where (c_0, c_1, c_2) are universal (problem-independent) constants.

Convergence of Newton sketch

Run algorithm with sketch dimension $m \asymp d$ on a self-concordant function $g(x) = f(Ax)$, and data matrix $A \in \mathbb{R}^{n \times d}$ with $n \gg d$.

Theorem (Pilanci & W, 2015)

With probability at least $1 - c_0 e^{-c_1 m}$, number of iterations required for ϵ accuracy is less than

$$c_2 \log(1/\epsilon)$$

where (c_0, c_1, c_2) are universal (problem-independent) constants.

Dependence on sample size n, dimension d; conditioning κ; and tolerance ϵ

Algorithm	Computational cost
Gradient Descent	$\mathcal{O}(\kappa\, n\, d\, \log(1/\epsilon))$
Acc. gradient Descent	$\mathcal{O}(\sqrt{\kappa}\, nd \log(1/\epsilon))$
Newton's Method	$\mathcal{O}(nd^2 \log\log(1/\epsilon))$
Newton Sketch	$\widetilde{\mathcal{O}}(nd \log(1/\epsilon))$

Convergence of Newton sketch

Run algorithm with sketch dimension $m \asymp d$ on a self-concordant function $g(x) = f(Ax)$, and data matrix $A \in \mathbb{R}^{n \times d}$ with $n \gg d$.

Theorem (Pilanci & W, 2015)

With probability at least $1 - c_0 e^{-c_1 m}$, number of iterations required for ϵ accuracy is less than

$$c_2 \log(1/\epsilon)$$

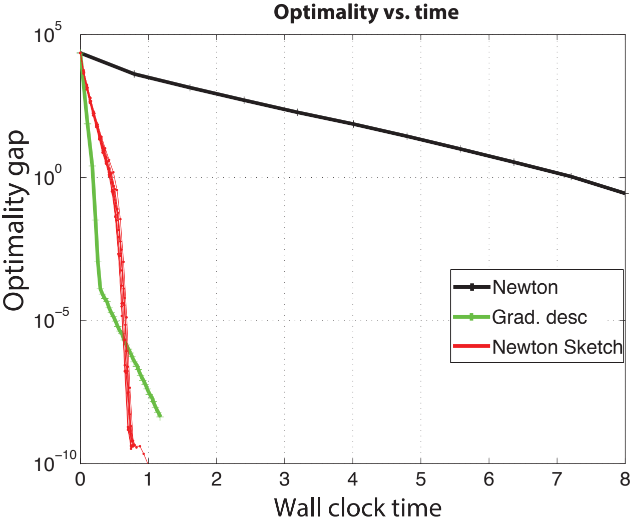
where (c_0, c_1, c_2) are universal (problem-independent) constants.

Dependence on sample size n, dimension d; conditioning κ; and tolerance ϵ

Algorithm	Computational cost
Gradient Descent	$\mathcal{O}(\kappa\, n\, d\, \log(1/\epsilon))$
Acc. gradient Descent	$\mathcal{O}(\sqrt{\kappa}\, nd \log(1/\epsilon))$
Newton's Method	$\mathcal{O}(nd^2 \log\log(1/\epsilon))$
Newton Sketch	$\widetilde{\mathcal{O}}(nd \log(1/\epsilon))$

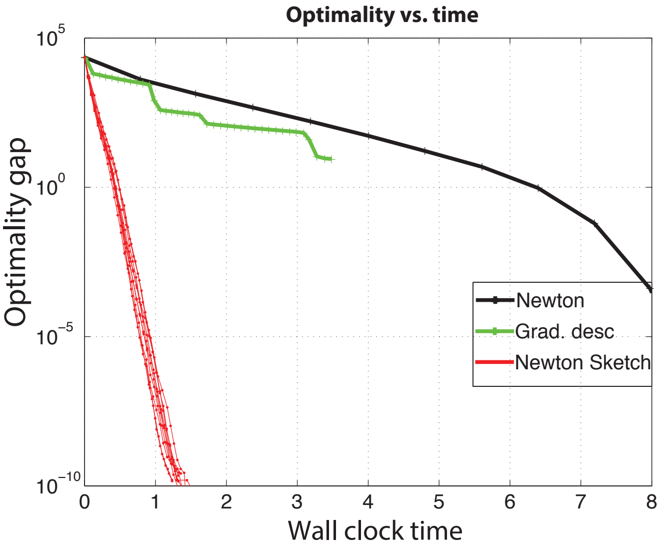
Note: Dependence on condition number κ unavoidable among 1st-order methods (Nesterov, 2004)

Logistic regression: uncorrelated features



Sample size $n = 500,000$ with $d = 5,000$ features

Logistic regression: correlated features



Sample size $n = 500,000$ with $d = 5,000$ features

Consequences for linear programming

- LP in standard form:

$$\min_{Ax \leq b} c^T x \qquad \text{where } A \in \mathbb{R}^{n \times d},\ b \in \mathbb{R}^n \text{ and } c \in \mathbb{R}^d.$$

Consequences for linear programming

- LP in standard form:

$$\min_{Ax \leq b} c^T x \qquad \text{where } A \in \mathbb{R}^{n \times d},\ b \in \mathbb{R}^n \text{ and } c \in \mathbb{R}^d.$$

- interior point methods for LP solving: based on unconstrained sequence

$$x_\mu := \arg\min_{x \in \mathbb{R}^d} \Big\{ \mu c^T x - \sum_{i=1}^n \log(b_i - a_i^T x) \Big\}.$$

Consequences for linear programming

- LP in standard form:

$$\min_{Ax \leq b} c^T x \qquad \text{where } A \in \mathbb{R}^{n \times d},\ b \in \mathbb{R}^n \text{ and } c \in \mathbb{R}^d.$$

- interior point methods for LP solving: based on unconstrained sequence

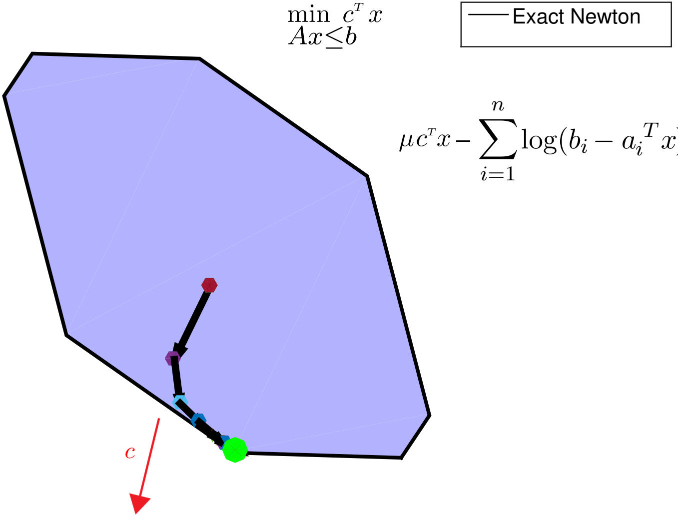
$$x_\mu := \arg\min_{x \in \mathbb{R}^d} \Big\{ \mu c^T x - \sum_{i=1}^n \log(b_i - a_i^T x) \Big\}.$$

- as parameter $\mu \to +\infty$, the path x_μ approaches an optimal solution x^* from the interior

Standard central path

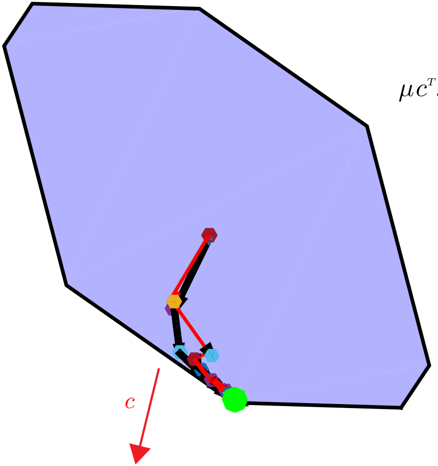
$$\min \; c^T x \quad Ax \leq b$$

$$\mu c^T x - \sum_{i=1}^{n} \log(b_i - a_i{}^T x)$$



Newton sketch follows central path

$$\min_{Ax \leq b} \; c^T x$$



$$\mu c^T x - \sum_{i=1}^{n} \log(b_i - a_i^T x)$$

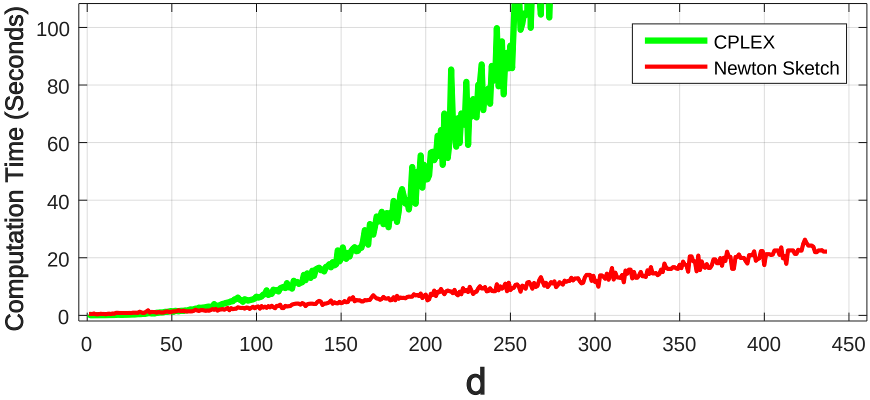
Linear Programs

Consequence: An LP with n constraints and d variables can be solved in $\approx O(nd)$ time when $n \gg d$.

Performance compared to CPLEX

Random ensembles of linear programs

Sample size $n = 10,000$
Dimensions $d = 1, 2, ..., 500$



CODE: `eecs.berkeley.edu/~mert/LP.zip`

Summary

- high-dimensional data: challenges and opportunities
- optimization at large scales:
  - Need fast methods...
  - But approximate answers are OK
  - randomized algorithms (with strong control) are useful
- this talk:
  - the power of random projection
  - information-theoretic analysis reveals deficiency of classical sketch
  - Newton sketch: a fast and randomized Newton-type method

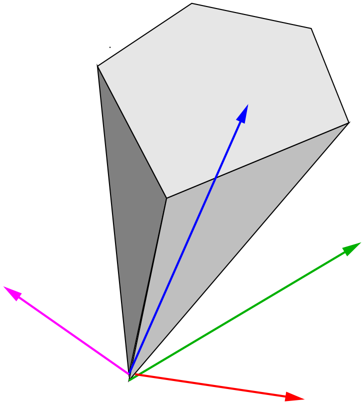
Summary

- high-dimensional data: challenges and opportunities
- optimization at large scales:
  - Need fast methods...
  - But approximate answers are OK
  - randomized algorithms (with strong control) are useful
- this talk:
  - the power of random projection
  - information-theoretic analysis reveals deficiency of classical sketch
  - Newton sketch: a fast and randomized Newton-type method

Papers/pre-prints:

- Pilanci & W. (2015): Randomized sketches of convex programs with sharp guarantees, *IEEE Transactions on Information Theory*
- Pilanci & W. (2016a): Iterative Hessian Sketch: Fast and accurate solution approximation for constrained least-squares, *Journal of Machine Learning Research*
- Pilanci & W. (2016b): Newton Sketch: A linear-time optimization algorithm with linear-quadratic convergence. To appear in *SIAM Journal of Optimization.*

Gaussian width of transformed tangent cone

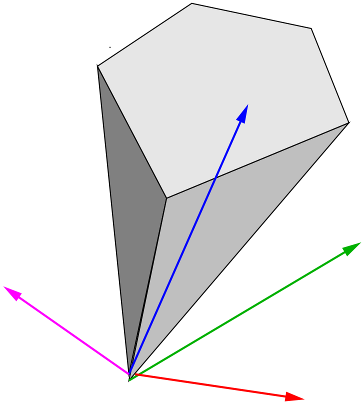


Gaussian width of set
$A\mathcal{K} \cap \mathcal{S}^{n-1} = \{A\Delta \mid \Delta \in \mathcal{K}, \|A\Delta\|_2 = 1\}$

$$\mathcal{W}(A\mathcal{K}) := \mathbb{E}\Big[\sup_{z \in A\mathcal{K} \cap \mathcal{S}^{n-1}} \langle g, z\rangle\Big]$$

where $g \sim N(0, I_{n \times n})$.

Gaussian width of transformed tangent cone



Gaussian width of set
$A\mathcal{K} \cap \mathcal{S}^{n-1} = \{A\Delta \mid \Delta \in \mathcal{K}, \|A\Delta\|_2 = 1\}$

$$\mathcal{W}(A\mathcal{K}) := \mathbb{E}\Big[\sup_{z \in A\mathcal{K} \cap \mathcal{S}^{n-1}} \langle g,\, z\rangle \Big]$$

where $g \sim N(0, I_{n \times n})$.

Gaussian widths used in many areas:

- Banach space theory: Pisier, 1986, Gordon 1988
- Empirical process theory: Ledoux & Talagrand, 1991, Bartlett et al., 2002
- Geometric analysis, compressed sensing: Mendelson, Pajor & Tomczak-Jaegermann, 2007

Fast Johnson-Lindenstrauss sketch

Step 1: Choose some fixed orthonormal matrix $H \in \mathbb{R}^{n \times n}$.
Example: Hadamard matrices

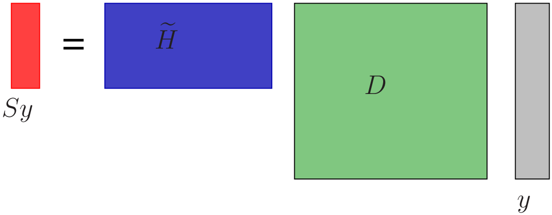
$$H_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \qquad H_{2^t} = \underbrace{H_2 \otimes H_2 \otimes \cdots \otimes H_2}_{\text{Kronecker product } t \text{ times}}$$

(E.g., Ailon & Liberty, 2010)

Fast Johnson-Lindenstrauss sketch

Step 1: Choose some fixed orthonormal matrix $H \in \mathbb{R}^{n \times n}$.
Example: Hadamard matrices

$$H_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \qquad H_{2^t} = \underbrace{H_2 \otimes H_2 \otimes \cdots \otimes H_2}_{\text{Kronecker product } t \text{ times}}$$



Step 2:

(A) Multiply data vector y with a diagonal matrix of random signs $\{-1, +1\}$

(B) Choose m rows of H to form sub-sampled matrix $\widetilde{H} \in \mathbb{R}^{m \times n}$

(C) Requires $\mathcal{O}(n \log m)$ time to compute sketched vector $Sy = \widetilde{H}\, Dy$.

(E.g., Ailon & Liberty, 2010)