Optimization (and Learning)

Steve Wright[1]

[1]Computer Sciences Department,
University of Wisconsin,
Madison, WI, USA

MLSS, Tübingen, August 2013

Overview

Focus on optimization formulations and algorithms that are relevant to learning.

- Convex
- Regularized
- Incremental / stochastic
- Coordinate descent
- Constraints

Mention other optimization areas of potential interest, as time permits.

Mário Figueiredo (IST, Lisbon) collaborated with me on a tutorial on sparse optimization at ICCOPT last month. He wrote and edited many of these slides.

Matching Optimization Tools to Learning

(From KDD, Aug 2013) Optimization tools are often combined in different ways to addesss learning applications.

Linear Regression.

- Linear algebra for $\|\cdot\|_2$. (Traditional!)
- Stochastic gradient for $m \gg n$ (e.g. parallel).

Variable Selection & Compressed Sensing.

- Shrink algorithms (for ℓ_1 term) (Wright et al., 2009b).
- Accelerated Gradient (Beck and Teboulle, 2009b).
- ADMM (Zhang et al., 2010).
- Higher-order: reduced inexact Newton (Wen et al., 2010); interior-point (Fountoulakis and Gondzio, 2013)
- (Also homotopy in λ, LARS, ...) (Efron et al., 2004)

Support Vector Machines.

- Coordinate Descent (Platt, 1999; Chang and Lin, 2011).
- Stochastic gradient (Bottou and LeCun, 2004; Shalev-Shwartz et al., 2007).
- Higher-order methods (interior-point) (Ferris and Munson, 2002; Fine and Scheinberg, 2001); (on reduced space) (Joachims, 1999).
- Shrink Algorithms (Duchi and Singer, 2009; Xiao, 2010).
- Stochastic gradient + shrink + higher-order (Lee and Wright, 2012).

Logistic Regression (+ Regularization).

- Shrink algorithms + reduced Newton (Shevade and Keerthi, 2003; Shi et al., 2008).
- Newton (Lin et al., 2008; Lee et al., 2006)
- Stochastic gradient (many!)
- Coordinate Descent (Meier et al., 2008)

Matrix Completion.

- (Block) Coordinate Descent (Wen et al., 2012).
- Shrink (Cai et al., 2010a; Lee et al., 2010).
- Stochastic Gradient (Lee et al., 2010).

Inverse Covariance.

- Coordinate Descent (Friedman et al., 2008)
- Accelerated Gradient (d'Aspremont et al., 2008)
- ADMM (Goldfarb et al., 2012; Scheinberg and Ma, 2012)

Deep Belief Networks.

- Stochastic Gradient (Le et al., 2012)
- Higher-order (LBFGS, approximate Newton) (Martens, 2010).
- Shrinks
- Coordinate descent (pretraining) (Hinton et al., 2006).

Image Processing.

- Shrink algorithms, gradient projection (Figueiredo and Nowak, 2003; Zhu et al., 2010)
- Higher-order methods: interior-point (Chan et al., 1999), reduced Newton.
- Augmented Lagrangian and ADMM (Bregman) Yin et al. (2008)

Data Assimilation.

- Higher-order methods (L-BFGS, inexact Newton)
- + many other tools from scientific computing.

1. First-Order Methods for Smooth Functions

2. Stochastic Gradient Methods

3. Higher-Order Methods

4. Sparse Optimization

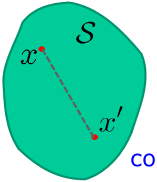
5. Augmented Lagrangian Methods

6. Coordinate Descent

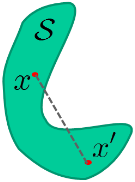
Just the Basics: Convex Sets

Convex and strictly convex sets

$\mathcal{S}$ is convex if $x, x' \in \mathcal{S} \;\Rightarrow \forall \lambda \in [0,1], \;\; \lambda x + (1-\lambda)x' \in \mathcal{S}$

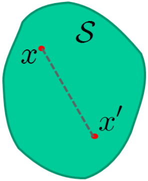


convex

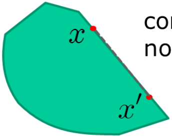


non-convex

$\mathcal{S}$ is strictly convex if $x, x' \in \mathcal{S} \;\Rightarrow \forall \lambda \in (0,1), \;\; \lambda x + (1-\lambda)x' \in \mathrm{int}(\mathcal{S})$



strictly convex



convex, but not strictly

Convex Functions

Extended real valued function: $f : \mathbb{R}^N \to \bar{\mathbb{R}} = \mathbb{R} \cup \{+\infty\}$

Domain: $\operatorname{dom}(f) = \{x : f(x) \neq +\infty\}$

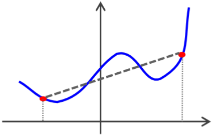
f is proper if $\operatorname{dom}(f) \neq \emptyset$

f is convex if

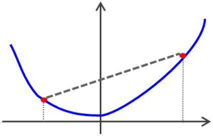
$$\forall \lambda \in [0,1], x, x' \in \operatorname{dom}(f) \;\; f(\lambda x + (1-\lambda)x') \leq \lambda f(x) + (1-\lambda) f(x')$$

f is strictly convex if

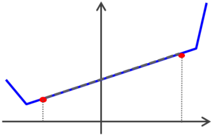
$$\forall \lambda \in (0,1), x, x' \in \operatorname{dom}(f) \;\; f(\lambda x + (1-\lambda)x') < \lambda f(x) + (1-\lambda) f(x')$$



non-convex



strictly convex



convex, not strictly

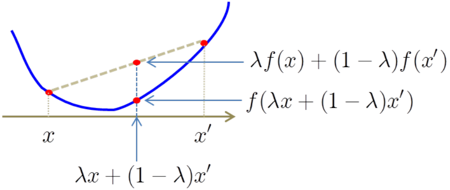
Strong Convexity

Recall the definition of convex function: $\forall \lambda \in [0,1]$,

$$f(\lambda x + (1-\lambda)x') \le \lambda f(x) + (1-\lambda) f(x')$$

A $\beta-$strongly convex function satisfies a stronger condition: $\forall \lambda \in [0,1]$

$$f(\lambda x + (1-\lambda)x') \le \lambda f(x) + (1-\lambda) f(x') - \frac{\beta}{2}\lambda(1-\lambda)\|x - x'\|_2^2$$


$\lambda f(x) + (1-\lambda)f(x')$
$f(\lambda x + (1-\lambda)x')$
x
x'
$\lambda x + (1-\lambda)x'$

convexity

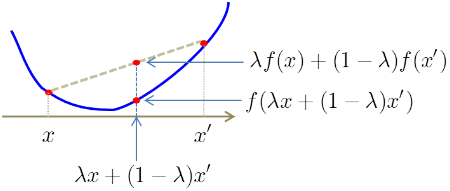
Strong Convexity

Recall the definition of convex function: $\forall \lambda \in [0,1]$,

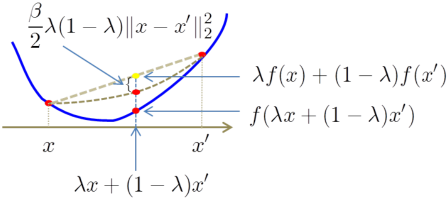
$$f(\lambda x + (1-\lambda)x') \leq \lambda f(x) + (1-\lambda) f(x')$$

A $\beta-$strongly convex function satisfies a stronger condition: $\forall \lambda \in [0,1]$

$$f(\lambda x + (1-\lambda)x') \leq \lambda f(x) + (1-\lambda) f(x') - \frac{\beta}{2}\lambda(1-\lambda)\|x - x'\|_2^2$$


$\lambda f(x) + (1-\lambda)f(x')$
$f(\lambda x + (1-\lambda)x')$
x
x'
$\lambda x + (1-\lambda)x'$

convexity


$\frac{\beta}{2}\lambda(1-\lambda)\|x-x'\|_2^2$
$\lambda f(x) + (1-\lambda)f(x')$
$f(\lambda x + (1-\lambda)x')$
x
x'
$\lambda x + (1-\lambda)x'$

strong convexity

A Little More on Convex Functions

Let $f_1, \dots, f_N : \mathbb{R}^n \to \bar{\mathbb{R}}$ be convex functions. Then

- $f : \mathbb{R}^n \to \bar{\mathbb{R}}$, defined as $f(x) = \max\{f_1(x), \dots, f_N(x)\}$, is convex.
- $g : \mathbb{R}^n \to \bar{\mathbb{R}}$, defined as $g(x) = f_1(L(x))$, where L is affine, is convex. ("Affine" means that L has the form $L(x) = Ax + b$.)
- $h : \mathbb{R}^n \to \bar{\mathbb{R}}$, defined as $h(x) = \sum_{j=1}^{N} \alpha_j f_j(x)$, for $\alpha_j > 0$, is convex.

An important function: the indicator of a set $C \subset \mathbb{R}^n$,

$$\iota_C : \mathbb{R}^n \to \bar{\mathbb{R}}, \quad \iota_C(x) = \begin{cases} 0 & \Leftarrow \quad x \in C \\ +\infty & \Leftarrow \quad x \notin C \end{cases}$$

If C is a closed convex set, ι_C is a lower semicontinuous convex function.

Smooth Functions

Let $f : \mathbb{R}^n \to \mathbb{R}$ be twice differentiable and consider its Hessian matrix at x, denoted $\nabla^2 f(x)$.

$$[\nabla^2 f(x)]_{ij} = \frac{\partial f}{\partial x_i \partial x_j}, \quad \text{for } i, j = 1, ..., n.$$

- f is convex $\Leftrightarrow$ its Hessian $\nabla^2 f(x)$ is positive semidefinite $\forall_x$
- f is strictly convex $\Leftarrow$ its Hessian $\nabla^2 f(x)$ is positive definite $\forall_x$
- f is β-strongly convex $\Leftrightarrow$ its Hessian $\nabla^2 f(x) \succeq \beta I$, with $\beta > 0$, $\forall_x$.

Norms: A Quick Review

Consider some real vector space $\mathcal{V}$, for example, $\mathbb{R}^n$ or $\mathbb{R}^{n\times n}$, ...

Some function $\|\cdot\| : \mathcal{V} \to \mathbb{R}$ is a norm if it satisfies:

- $\|\alpha x\| = |\alpha|\, \|x\|$, for any $x \in \mathcal{V}$ and $\alpha \in \mathbb{R}$ (homogeneity);
- $\|x + x'\| \leq \|x\| + \|x'\|$, for any $x, x' \in \mathcal{V}$ (triangle inequality);
- $\|x\| = 0 \;\Rightarrow\; x = 0$.

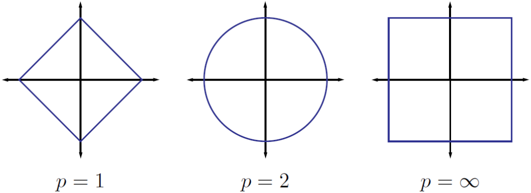
Examples:

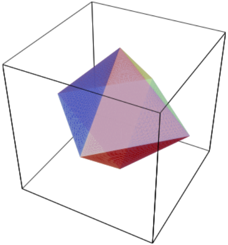
- $\mathcal{V} = \mathbb{R}^n$, $\|x\|_p = \left(\sum_i |x_i|^p\right)^{1/p}$ (called ℓ_p norm, for $p \geq 1$).
- $\mathcal{V} = \mathbb{R}^n$, $\|x\|_\infty = \lim_{p\to\infty} \|x\|_p = \max\{|x_1|, ..., |x_n|\}$
- $\mathcal{V} = \mathbb{R}^{n\times n}$, $\|X\|_* = \text{trace}\left(\sqrt{X^T X}\right)$ (matrix nuclear norm)

Also important (but not a norm): $\|x\|_0 = \lim_{p\to 0} \|x\|_p^p = |\{i : x_i \neq 0\}|$

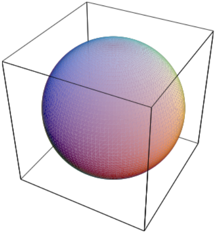
Norm balls

Radius r ball in ℓ_p norm: $\quad B_p(r) = \{x \in \mathbb{R}^n : \|x\|_p \le r\}$





$p = 1$



$p = 2$

I. First-Order Algorithms: Smooth Convex Functions

Consider $\min_{x \in \mathbb{R}^n} f(x)$, with f smooth and convex.

Usually assume $\mu I \preceq \nabla^2 f(x) \preceq LI, \;\; \forall_x$, with $0 \le \mu \le L$ (thus L is a Lipschitz constant of ∇f).

If $\mu > 0$, then f is μ-strongly convex (as seen in Part 1) and

$$f(y) \ge f(x) + \nabla f(x)^T (y - x) + \frac{\mu}{2} \|y - x\|_2^2.$$

Define conditioning (or condition number) as $\kappa := L/\mu$.

We are often interested in convex quadratics:

$$f(x) = \frac{1}{2} x^T A x, \qquad \mu I \preceq A \preceq LI \;\; \text{or}$$

$$f(x) = \frac{1}{2} \|Bx - b\|_2^2, \quad \mu I \preceq B^T B \preceq LI$$

What's the Setup?

We consider iterative algorithms

$$x_{k+1} = x_k + d_k,$$

where d_k depends on x_k or possibly (x_k, x_{k-1}),

For now, assume we can evaluate $f(x_t)$ and $\nabla f(x_t)$ at each iteration. We focus on algorithms that can be extended to a setting broader than convex, smooth, unconstrained:

- nonsmooth f;
- f not available (or too expensive to evaluate exactly);
- only an *estimate* of the gradient is available;
- a constraint $x \in \Omega$, usually for a simple Ω (e.g. ball, box, simplex);
- nonsmooth regularization; *i.e.*, instead of simply $f(x)$, we want to minimize $f(x) + \tau\psi(x)$.

Steepest Descent

Steepest descent (a.k.a. gradient descent):

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k), \qquad \text{for some } \alpha_k > 0.$$

Different ways to select an appropriate α_k.

1. Hard: interpolating scheme with safeguarding to identify an approximate minimizing α_k.
2. Easy: backtracking. $\bar{\alpha}$, $\frac{1}{2}\bar{\alpha}$, $\frac{1}{4}\bar{\alpha}$, $\frac{1}{8}\bar{\alpha}$, ... until sufficient decrease in f is obtained.
3. Trivial: don't test for function decrease; use rules based on L and μ.

Analysis for 1 and 2 usually yields global convergence at unspecified rate. The "greedy" strategy of getting good decrease in the current search direction may lead to better practical results.

Analysis for 3: Focuses on convergence rate, and leads to accelerated multi-step methods.

Line Search

Seek α_k that satisfies Wolfe conditions: "sufficient decrease" in f:

$$f(x_k - \alpha_k \nabla f(x_k)) \le f(x_k) - c_1 \alpha_k \|\nabla f(x_k)\|^2, \qquad (0 < c_1 \ll 1)$$

while "not being too small" (significant increase in the directional derivative):

$$\nabla f(x_{k+1})^T \nabla f(x_k) \ge -c_2 \|\nabla f(x_k)\|^2, \qquad (c_1 < c_2 < 1).$$

(works for nonconvex f.) Can show that accumulation points $\bar{x}$ of $\{x_k\}$ are stationary: $\nabla f(\bar{x}) = 0$ (thus minimizers, if f is convex)

Can do one-dimensional line search for α_k, taking minima of quadratic or cubic interpolations of the function and gradient at the last two values tried. Use brackets for reliability. Often finds suitable α within 3 attempts. (Nocedal and Wright, 2006, Chapter 3)

Backtracking

Try $\alpha_k = \bar{\alpha}, \frac{\bar{\alpha}}{2}, \frac{\bar{\alpha}}{4}, \frac{\bar{\alpha}}{8}$, ... until the sufficient decrease condition is satisfied.

No need to check the second Wolfe condition: the α_k thus identified is "within striking distance" of an α that's too large — so it is not too short.

Backtracking is widely used in applications, but doesn't work for *f* nonsmooth, or when *f* is not available / too expensive.

Constant (Short) Steplength

By elementary use of Taylor's theorem, and since $\nabla^2 f(x) \preceq LI$,

$$f(x_{k+1}) \le f(x_k) - \alpha_k \left(1 - \frac{\alpha_k}{2} L\right) \|\nabla f(x_k)\|_2^2$$

For $\alpha_k \equiv 1/L$, $\quad f(x_{k+1}) \le f(x_k) - \frac{1}{2L} \|\nabla f(x_k)\|_2^2,$

thus $\quad \|\nabla f(x_k)\|^2 \le 2L[f(x_k) - f(x_{k+1})]$

Summing for $k = 0, 1, \dots, N$, and telescoping the sum,

$$\sum_{k=0}^{N} \|\nabla f(x_k)\|^2 \le 2L[f(x_0) - f(x_{N+1})].$$

It follows that $\nabla f(x_k) \to 0$ if f is bounded below.

Rate Analysis

Suppose that the minimizer x^* is unique.

Another elementary use of Taylor's theorem shows that

$$\|x_{k+1} - x^*\|^2 \le \|x_k - x^*\|^2 - \alpha_k \left(\frac{2}{L} - \alpha_k \right) \|\nabla f(x_k)\|^2,$$

so that $\{\|x_k - x^*\|\}$ is decreasing.

Define for convenience: $\Delta_k := f(x_k) - f(x^*)$. By convexity, have

$$\Delta_k \le \nabla f(x_k)^T (x_k - x^*) \le \|\nabla f(x_k)\| \, \|x_k - x^*\| \le \|\nabla f(x_k)\| \, \|x_0 - x^*\|.$$

From previous page (subtracting $f(x^*)$ from both sides of the inequality), and using the inequality above, we have

$$\Delta_{k+1} \le \Delta_k - (1/2L)\|\nabla f(x_k)\|^2 \le \Delta_k - \frac{1}{2L\|x_0 - x^*\|^2} \Delta_k^2.$$

Weakly convex: $1/k$ sublinear; Strongly convex: linear

Take reciprocal of both sides and manipulate (using $(1-\epsilon)^{-1} \geq 1+\epsilon$):

$$\frac{1}{\Delta_{k+1}} \geq \frac{1}{\Delta_k} + \frac{1}{2L\|x_0 - x^*\|^2} \geq \frac{1}{\Delta_0} + \frac{k+1}{2L\|x_0 - x^*\|^2},$$

which yields

$$f(x_{k+1}) - f(x^*) \leq \frac{2L\|x_0 - x\|^2}{k+1}.$$

The classic $1/k$ convergence rate!

By assuming $\mu > 0$, can set $\alpha_k \equiv 2/(\mu + L)$ and get a linear (geometric) rate.

$$\|x_k - x^*\|^2 \leq \left(\frac{L-\mu}{L+\mu}\right)^{2k} \|x_0 - x^*\|^2 = \left(1 - \frac{2}{\kappa+1}\right)^{2k} \|x_0 - x^*\|^2.$$

Linear convergence is almost always better than sublinear!

INTERMISSION: Convergence rates

There's somtimes confusion about terminology for convergence. Here's what optimizers generally say, when talking about how fast a positive sequence $\{t_k\}$ of scalars is decreasing to zero.

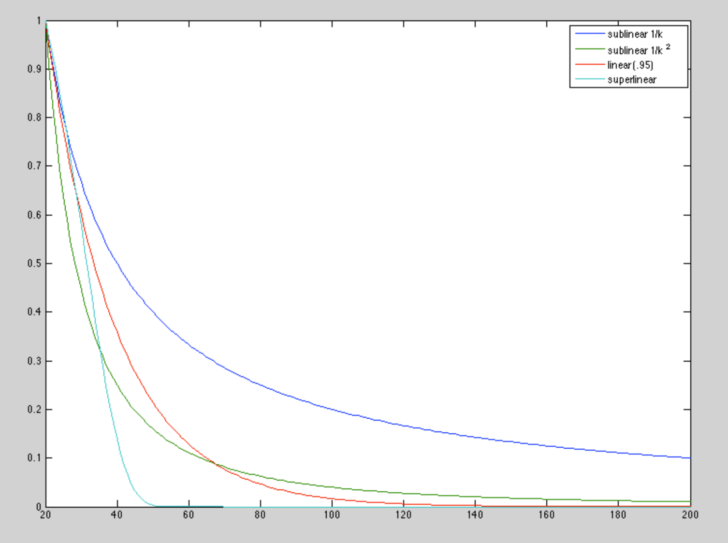
Sublinear: $t_k \to 0$, but $t_{k+1}/t_k \to 1$. Example: $1/k$ rate, where $t_k \le Q/k$ for some constant Q.

Linear: $t_{k+1}/t_k \le r$ for some $r \in (0,1)$. Thus typically $t_k \le Cr^k$. Also called "geometric" or "exponential" (but I hate the last one — it's oversell!)

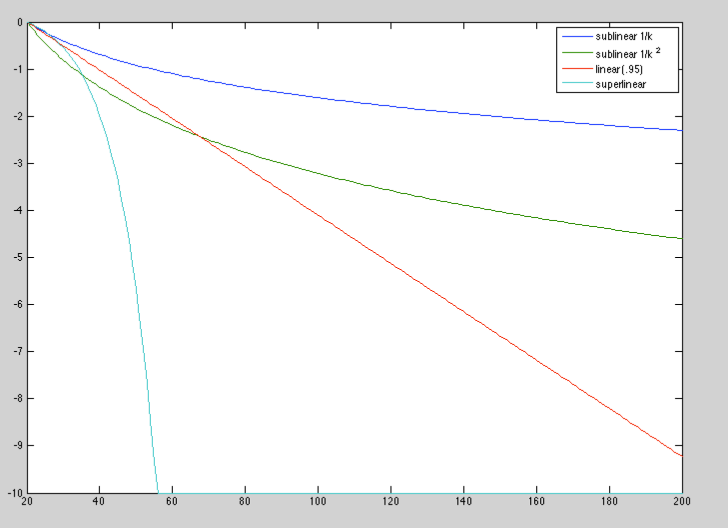
Superlinear: $t_{k+1}/t_k \to 0$. That's fast!

Quadratic: $t_{k+1} \le Ct_k^2$. Really fast, typical of Newton's method. The number of correct significant digits doubles at each iteration. There's no point being faster than this!

Comparing Rates



Comparing Rates: Log Plot



Back To Steepest Descent

Question: does taking α_k as the exact minimizer of f along $-\nabla f(x_k)$ yield a better rate of linear convergence than the $(1-2/\kappa)$ linear rate identified earlier?

Consider $f(x) = \frac{1}{2}x^T A x$ (thus $x^* = 0$ and $f(x^*) = 0$.)

We have $\nabla f(x_k) = A x_k$. Exactly minimizing w.r.t. α_k,

$$\alpha_k = \arg\min_{\alpha} \frac{1}{2}(x_k - \alpha A x_k)^T A (x_k - \alpha A x_k) = \frac{x_k^T A^2 x_k}{x_k^T A^3 x_k} \in \left[\frac{1}{L}, \frac{1}{\mu}\right]$$

Thus

$$f(x_{k+1}) \le f(x_k) - \frac{1}{2}\frac{(x_k^T A^2 x_k)^2}{(x_k^T A x_k)(x_k^T A^3 x_k)},$$

so, defining $z_k := A x_k$, we have

$$\frac{f(x_{k+1}) - f(x^*)}{f(x_k) - f(x^*)} \le 1 - \frac{\|z_k\|^4}{(z_k^T A^{-1} z_k)(z_k^T A z_k)}.$$

Exact minimizing α_k: Faster rate?

Using Kantorovich inequality:

$$(z^T Az)(z^T A^{-1}z) \le \frac{(L+\mu)^2}{4L\mu}\|z\|^4.$$

Thus

$$\frac{f(x_{k+1})-f(x^*)}{f(x_k)-f(x^*)} \le 1-\frac{4L\mu}{(L+\mu)^2} = \left(1-\frac{2}{\kappa+1}\right)^2,$$

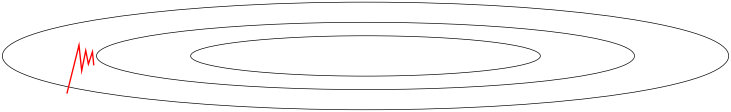
and so

$$f(x_k)-f(x^*) \le \left(1-\frac{2}{\kappa+1}\right)^{2k}[f(x_0)-f(x^*)].$$

No improvement in the linear rate over constant steplength!

The slow linear rate is typical!

Not just a pessimistic bound — it really is this slow!



Multistep Methods: Heavy-Ball

Enhance the search direction using a contribution from the previous step. (known as heavy ball, momentum, or two-step)

Consider first a constant step length α, and a second parameter β for the "momentum" term:

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

Analyze by defining a composite iterate vector:

$$w_k := \begin{bmatrix} x_k - x^* \\ x_{k-1} - x^* \end{bmatrix}.$$

Thus

$$w_{k+1} = B w_k + o(\|w_k\|), \qquad B := \begin{bmatrix} -\alpha \nabla^2 f(x^*) + (1+\beta) I & -\beta I \\ I & 0 \end{bmatrix}.$$

Multistep Methods: The Heavy-Ball

Matrix B has same eigenvalues as

$$\begin{bmatrix} -\alpha\Lambda + (1+\beta)I & -\beta I \\ I & 0 \end{bmatrix}, \qquad \Lambda = \operatorname{diag}(\lambda_1, \lambda_2, \dots, \lambda_n),$$

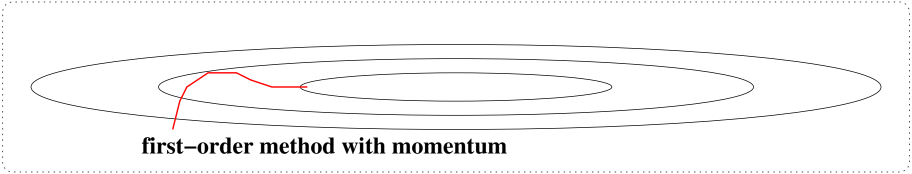
where λ_i are the eigenvalues of $\nabla^2 f(x^*)$.

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

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

Leads to linear convergence for $\|x_k - x^*\|$ with rate approximately

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



There's some damping going on! (Like the PID controllers in Schaal's talk yesterday.)

Summary: Linear Convergence, Strictly Convex f

- Steepest descent: Linear rate approx $\left(1 - \frac{2}{\kappa}\right)$;
- Heavy-ball: Linear rate approx $\left(1 - \frac{2}{\sqrt{\kappa}}\right)$.

Big difference! To reduce $\|x_k - x^*\|$ by a factor ϵ, need k large enough that

$$\left(1 - \frac{2}{\kappa}\right)^k \le \epsilon \quad \Leftarrow \quad k \ge \frac{\kappa}{2} |\log \epsilon| \quad \text{(steepest descent)}$$
$$\left(1 - \frac{2}{\sqrt{\kappa}}\right)^k \le \epsilon \quad \Leftarrow \quad k \ge \frac{\sqrt{\kappa}}{2} |\log \epsilon| \quad \text{(heavy-ball)}$$

A factor of $\sqrt{\kappa}$ difference; e.g. if $\kappa = 1000$ (not at all uncommon in inverse problems), need ~ 30 times fewer steps.

Conjugate Gradient

Basic conjugate gradient (CG) step is

$$x_{k+1} = x_k + \alpha_k p_k, \qquad p_k = -\nabla f(x_k) + \gamma_k p_{k-1}.$$

Can be identified with heavy-ball, with $\beta_k = \dfrac{\alpha_k \gamma_k}{\alpha_{k-1}}$.

However, CG can be implemented in a way that doesn't require knowledge (or estimation) of L and μ.

- Choose α_k to (approximately) miminize f along p_k;
- Choose γ_k by a variety of formulae (Fletcher-Reeves, Polak-Ribiere, etc), all of which are equivalent if f is convex quadratic. e.g.

$$\gamma_k = \frac{\|\nabla f(x_k)\|^2}{\|\nabla f(x_{k-1})\|^2}$$

Conjugate Gradient

Nonlinear CG: Variants include Fletcher-Reeves, Polak-Ribiere, Hestenes.

Restarting periodically with $p_k = -\nabla f(x_k)$ is useful (e.g. every n iterations, or when p_k is not a descent direction).

For quadratic f, convergence analysis is based on eigenvalues of A and Chebyshev polynomials, min-max arguments. Get

- Finite termination in as many iterations as there are distinct eigenvalues;
- Asymptotic linear convergence with rate approx $1 - \dfrac{2}{\sqrt{\kappa}}$. (like heavy-ball.)

(Nocedal and Wright, 2006, Chapter 5)

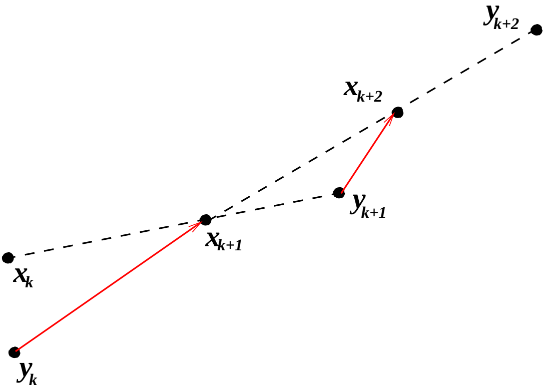
Accelerated First-Order Methods

Accelerate the rate to $1/k^2$ for weakly convex, while retaining the linear rate (related to $\sqrt{\kappa}$) for strongly convex case.

Nesterov (1983) describes a method that requires κ.

```
Initialize: Choose $x_0$, $\alpha_0 \in (0,1)$; set $y_0 \leftarrow x_0$.
   Iterate: $x_{k+1} \leftarrow y_k - \frac{1}{L}\nabla f(y_k)$; (*short-step*)
            find $\alpha_{k+1} \in (0,1)$: $\alpha_{k+1}^2 = (1-\alpha_{k+1})\alpha_k^2 + \frac{\alpha_{k+1}}{\kappa}$;
            set $\beta_k = \dfrac{\alpha_k(1-\alpha_k)}{\alpha_k^2 + \alpha_{k+1}}$;
            set $y_{k+1} \leftarrow x_{k+1} + \beta_k(x_{k+1} - x_k)$.
```

Still works for weakly convex ($\kappa = \infty$).



Separates the “gradient descent” and “momentum” step components.

Convergence Results: Nesterov

If $\alpha_0 \geq 1/\sqrt{\kappa}$, have

$$f(x_k) - f(x^*) \leq c_1 \min\left(\left(1 - \frac{1}{\sqrt{\kappa}}\right)^k, \frac{4L}{(\sqrt{L} + c_2 k)^2}\right),$$

where constants c_1 and c_2 depend on x_0, α_0, L.

- Linear convergence "heavy-ball" rate for strongly convex f;
- $1/k^2$ sublinear rate otherwise.

In the special case of $\alpha_0 = 1/\sqrt{\kappa}$, this scheme yields

$$\alpha_k \equiv \frac{1}{\sqrt{\kappa}}, \qquad \beta_k \equiv 1 - \frac{2}{\sqrt{\kappa} + 1}.$$

FISTA

Beck and Teboulle (2009a) propose a similar algorithm, with a fairly short and elementary analysis (though still not intuitive).

Initialize: Choose x_0; set $y_1 = x_0$, $t_1 = 1$;

Iterate:
$$x_k \leftarrow y_k - \tfrac{1}{L}\nabla f(y_k);$$
$$t_{k+1} \leftarrow \tfrac{1}{2}\left(1 + \sqrt{1 + 4t_k^2}\right);$$
$$y_{k+1} \leftarrow x_k + \frac{t_k - 1}{t_{k+1}}(x_k - x_{k-1}).$$

For (weakly) convex f, converges with $f(x_k) - f(x^*) \sim 1/k^2$.

When L is not known, increase an estimate of L until it's big enough.

Beck and Teboulle (2009a) do the convergence analysis in 2-3 pages: elementary, but technical.

A Nonmonotone Gradient Method: Barzilai-Borwein

Barzilai and Borwein (1988) (BB) proposed an unusual choice of α_k.
Allows f to increase (sometimes a lot) on some steps: non-monotone.

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k), \qquad \alpha_k := \arg\min_{\alpha} \|s_k - \alpha z_k\|^2,$$

where

$$s_k := x_k - x_{k-1}, \qquad z_k := \nabla f(x_k) - \nabla f(x_{k-1}).$$

Explicitly, we have

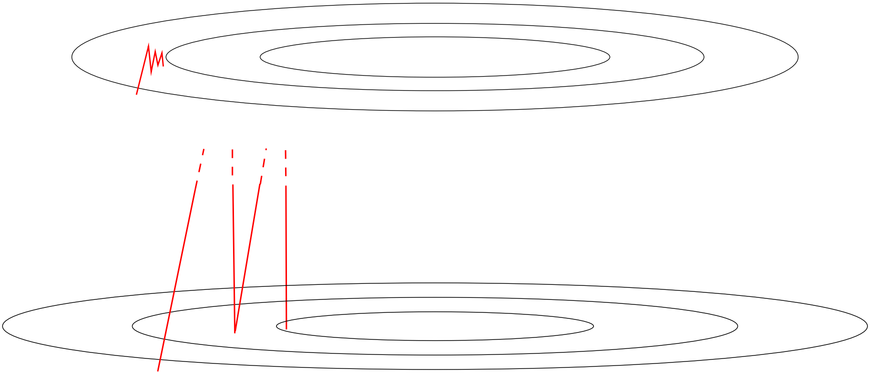
$$\alpha_k = \frac{s_k^T z_k}{z_k^T z_k}.$$

Note that for $f(x) = \frac{1}{2} x^T A x$, we have

$$\alpha_k = \frac{s_k^T A s_k}{s_k^T A^2 s_k} \in \left[\frac{1}{L}, \frac{1}{\mu} \right].$$

BB can be viewed as a quasi-Newton method, with the Hessian approximated by $\alpha_k^{-1} I$.

Comparison: BB vs Greedy Steepest Descent



There Are Many BB Variants

- use $\alpha_k = s_k^T s_k / s_k^T z_k$ in place of $\alpha_k = s_k^T z_k / z_k^T z_k$;
- alternate between these two formulae;
- hold α_k constant for a number (2, 3, 5) of successive steps;
- take α_k to be the steepest descent step from the previous iteration.

Nonmonotonicity appears essential to performance. Some variants get global convergence by requiring a sufficient decrease in f over the worst of the last M (say 10) iterates.

The original 1988 analysis in BB's paper is nonstandard and illuminating (just for a 2-variable quadratic).

In fact, most analyses of BB and related methods are nonstandard, and consider only special cases. The precursor of such analyses is Akaike (1959). More recently, see Ascher, Dai, Fletcher, Hager and others.

Adding a Constraint: $x \in \Omega$

How to change these methods to handle the constraint $x \in \Omega$?
(assuming that Ω is a closed convex set)

Some algorithms and theory stay much the same,
...if we can involve the constraint $x \in \Omega$ explicity in the subproblems.

Example: Nesterov's constant step scheme requires just one calculation to be changed from the unconstrained version.

```
Initialize: Choose $x_0$, $\alpha_0 \in (0,1)$; set $y_0 \leftarrow x_0$.
   Iterate: $x_{k+1} \leftarrow \arg\min_{y \in \Omega} \frac{1}{2}\|y - [y_k - \frac{1}{L}\nabla f(y_k)]\|_2^2$;
            find $\alpha_{k+1} \in (0,1)$: $\alpha_{k+1}^2 = (1-\alpha_{k+1})\alpha_k^2 + \frac{\alpha_{k+1}}{\kappa}$;
            set $\beta_k = \frac{\alpha_k(1-\alpha_k)}{\alpha_k^2+\alpha_{k+1}}$;
            set $y_{k+1} \leftarrow x_{k+1} + \beta_k(x_{k+1} - x_k)$.
```

Convergence theory is unchanged.

Extending to Regularized Optimization

How to change these methods to handle optimization with regularization:

$$\min_x \; f(x) + \tau\psi(x),$$

where f is convex and smooth, while ψ is convex but usually nonsmooth.

Often, all that is needed is to change the update step to

$$x_{k+1} = \arg\min_x \frac{1}{2\alpha_k} \|x - (x_k + \alpha_k d_k)\|_2^2 + \tau\psi(x).$$

where d_k could be a scaled gradient descent step, or something more complicated (heavy-ball, accelerated gradient), while α_k is the step length.

This is the shrinkage/tresholding step; how to solve it with a nonsmooth ψ? We'll come back to this topic after discussing the need for regularization.

II. Stochastic Gradient Methods

Deal with (weakly or strongly) convex f. We change the rules a bit in this section:

- Allow f nonsmooth.
- Can't get function values $f(x)$ easily.
- At any feasible x, have access only to a cheap unbiased estimate of an element of the subgradient ∂f.

Common settings are:

$$f(x) = E_\xi F(x, \xi),$$

where ξ is a random vector with distribution P over a set Ξ. Special case:

$$f(x) = \frac{1}{m} \sum_{i=1}^{m} f_i(x),$$

where each f_i is convex and nonsmooth.

(We focus on this finite-sum formulation, but the ideas generalize.)

Applications

This setting is useful for machine learning formulations. Given data $x_i \in \mathbb{R}^n$ and labels $y_i = \pm 1$, $i = 1, 2, \dots, m$, find w that minimizes

$$\tau \psi(w) + \frac{1}{m} \sum_{i=1}^{m} \ell(w; x_i, y_i),$$

where ψ is a regularizer, $\tau > 0$ is a parameter, and ℓ is a loss. For linear classifiers/regressors, have the specific form $\ell(w^T x_i, y_i)$.

Example: SVM with hinge loss $\ell(w^T x_i, y_i) = \max(1 - y_i(w^T x_i), 0)$ and $\psi = \| \cdot \|_1$ or $\psi = \| \cdot \|_2^2$.

Example: Logistic classification: $\ell(w^T x_i, y_i) = \log(1 + \exp(y_i w^T x_i))$. In regularized version may have $\psi(w) = \|w\|_1$.

Subgradients

Recall: For each x in domain of f, g is a *subgradient of f at x* if

$$f(z) \geq f(x) + g^T(z - x), \qquad \text{for all } z \in \operatorname{dom} f.$$

- Right-hand side is a *supporting hyperplane*.
- The set of subgradients is called the *subdifferential*, denoted by $\partial f(x)$.
- When f is differentiable at x, have $\partial f(x) = \{\nabla f(x)\}$.

We have strong convexity with modulus $\mu > 0$ if

$$f(z) \geq f(x) + g^T(z - x) + \frac{1}{2}\mu\|z - x\|^2, \quad \text{for all } x, z \in \operatorname{dom} f \text{ with } g \in \partial f(x).$$

Generalizes the assumption $\nabla^2 f(x) \succeq \mu I$ made earlier for smooth functions.

Classical Stochastic Gradient

For the finite-sum objective, get a cheap unbiased estimate of the gradient $\nabla f(x)$ by choosing an index $i \in \{1, 2, \dots, m\}$ uniformly at random, and using $\nabla f_i(x)$ to estimate $\nabla f(x)$.

Basic SA Scheme: At iteration k, choose i_k i.i.d. uniformly at random from $\{1, 2, \dots, m\}$, choose some $\alpha_k > 0$, and set

$$x_{k+1} = x_k - \alpha_k \nabla f_{i_k}(x_k).$$

Note that x_{k+1} depends on all random indices up to iteration k, i.e. $i_{[k]} := \{i_1, i_2, \dots, i_k\}$.

When f is strongly convex, the analysis of convergence of expected square error $E(\|x_k - x^*\|^2)$ is fairly elementary — see Nemirovski et al. (2009).

Define $a_k = \frac{1}{2}E(\|x_k - x^*\|^2)$. Assume there is $M > 0$ such that

$$\frac{1}{m}\sum_{i=1}^{m} \|\nabla f_i(x)\|_2^2 \le M.$$

Rate: $1/k$

Thus

$$\begin{aligned}
\frac{1}{2}&\|x_{k+1} - x^*\|_2^2 \\
&= \frac{1}{2}\|x_k - \alpha_k \nabla f_{i_k}(x_k) - x^*\|^2 \\
&= \frac{1}{2}\|x_k - x^*\|_2^2 - \alpha_k (x_k - x^*)^T \nabla f_{i_k}(x_k) + \frac{1}{2}\alpha_k^2 \|\nabla f_{i_k}(x_k)\|^2.
\end{aligned}$$

Taking expectations, get

$$a_{k+1} \le a_k - \alpha_k E[(x_k - x^*)^T \nabla f_{i_k}(x_k)] + \frac{1}{2}\alpha_k^2 M^2.$$

For middle term, have

$$\begin{aligned}
E[(x_k - x^*)^T \nabla f_{i_k}(x_k)] &= E_{i_{[k-1]}} E_{i_k}[(x_k - x^*)^T \nabla f_{i_k}(x_k) | i_{[k-1]}] \\
&= E_{i_{[k-1]}} (x_k - x^*)^T g_k,
\end{aligned}$$

... where

$$g_k := E_{i_k}[\nabla f_{i_k}(x_k) | i_{[k-1]}] \in \partial f(x_k).$$

By strong convexity, have

$$(x_k - x^*)^T g_k \geq f(x_k) - f(x^*) + \frac{1}{2}\mu \|x_k - x^*\|^2 \geq \mu \|x_k - x^*\|^2.$$

Hence by taking expectations, we get $E[(x_k - x^*)^T g_k] \geq 2\mu a_k$. Then, substituting above, we obtain

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

When

$$\alpha_k \equiv \frac{1}{k\mu},$$

a neat inductive argument (below) reveals the $1/k$ rate:

$$a_k \leq \frac{Q}{2k}, \qquad \text{for } Q := \max\left(\|x_1 - x^*\|^2, \frac{M^2}{\mu^2} \right).$$

Inductive Proof of $1/k$ Rate

Clearly true for $k = 1$. Otherwise:

$$\begin{aligned}
a_{k+1} &\le (1 - 2\mu\alpha_k)a_k + \frac{1}{2}\alpha_k^2 M^2 \\
&\le \left(1 - \frac{2}{k}\right) a_k + \frac{M^2}{2k^2\mu^2} \\
&\le \left(1 - \frac{2}{k}\right) \frac{Q}{2k} + \frac{Q}{2k^2} \\
&= \frac{(k-1)}{2k^2} Q \\
&= \frac{k^2-1}{k^2} \frac{Q}{2(k+1)} \\
&\le \frac{Q}{2(k+1)},
\end{aligned}$$

as claimed.

But... What if we don't know μ? Or if $\mu = 0$?

The choice $\alpha_k = 1/(k\mu)$ requires strong convexity, with knowledge of the modulus μ. An underestimate of μ can greatly degrade the performance of the method (see example in Nemirovski et al. (2009)).

Now describe a *Robust Stochastic Approximation* approach, which has a rate $1/\sqrt{k}$ (in function value convergence), and works for weakly convex nonsmooth functions and is not sensitive to choice of parameters in the step length.

This is the approach that generalizes to *mirror descent*, as discussed later.

Robust SA

At iteration k:

- set $x_{k+1} = x_k - \alpha_k \nabla f_{i_k}(x_k)$ as before;
- set

$$\bar{x}_k = \frac{\sum_{i=1}^k \alpha_i x_i}{\sum_{i=1}^k \alpha_i}.$$

For any $\theta > 0$, choose step lengths to be

$$\alpha_k = \frac{\theta}{M\sqrt{k}}.$$

Then $f(\bar{x}_k)$ converges to $f(x^*)$ in expectation with rate approximately $(\log k)/k^{1/2}$.

(The choice of θ is not critical.)

Analysis of Robust SA

The analysis is again elementary. As above (using i instead of k), have:

$$\alpha_i E[(x_i - x^*)^T g_i] \le a_i - a_{i+1} + \frac{1}{2}\alpha_i^2 M^2.$$

By convexity of f, and $g_i \in \partial f(x_i)$:

$$f(x^*) \ge f(x_i) + g_i^T(x^* - x_i),$$

thus

$$\alpha_i E[f(x_i) - f(x^*)] \le a_i - a_{i+1} + \frac{1}{2}\alpha_i^2 M^2,$$

so by summing iterates $i = 1, 2, \dots, k$, telescoping, and using $a_{k+1} > 0$:

$$\sum_{i=1}^{k} \alpha_i E[f(x_i) - f(x^*)] \le a_1 + \frac{1}{2} M^2 \sum_{i=1}^{k} \alpha_i^2.$$

Thus dividing by $\sum_{i=1} \alpha_i$:

$$E\left[\frac{\sum_{i=1}^k \alpha_i f(x_i)}{\sum_{i=1}^k \alpha_i} - f(x^*)\right] \le \frac{a_1 + \frac{1}{2}M^2 \sum_{i=1}^k \alpha_i^2}{\sum_{i=1}^k \alpha_i}.$$

By convexity, we have

$$f(\bar{x}_k) = f\left(\frac{\sum_{i=1}^k \alpha_i x_i}{\sum_{i=1}^k \alpha_i}\right) \le \frac{\sum_{i=1}^k \alpha_i f(x_i)}{\sum_{i=1}^k \alpha_i},$$

so obtain the fundamental bound:

$$E[f(\bar{x}_k) - f(x^*)] \le \frac{a_1 + \frac{1}{2}M^2 \sum_{i=1}^k \alpha_i^2}{\sum_{i=1}^k \alpha_i}.$$

By substituting $\alpha_i = \frac{\theta}{M\sqrt{i}}$, we obtain

$$\begin{aligned} E[f(\bar{x}_k) - f(x^*)] &\le \frac{a_1 + \frac{1}{2}\theta^2 \sum_{i=1}^k \frac{1}{i}}{\frac{\theta}{M} \sum_{i=1}^k \frac{1}{\sqrt{i}}} \\ &\le \frac{a_1 + \theta^2 \log(k+1)}{\frac{\theta}{M}\sqrt{k}} \\ &= M\left[\frac{a_1}{\theta} + \theta \log(k+1)\right]\frac{1}{\sqrt{k}}. \end{aligned}$$

That's it!

There are other variants — periodic restarting, averaging just over the recent iterates. These can be analyzed with the basic bound above.

Constant Step Size

We can also get rates of approximately $1/k$ for the strongly convex case, *without* performing iterate averaging. The tricks are to

- define the desired threshold ϵ for a_k in advance, and
- use a constant step size.

Recall the bound on a_{k+1} from a few slides back, and set $\alpha_k \equiv \alpha$:

$$a_{k+1} \leq (1 - 2\mu\alpha)a_k + \frac{1}{2}\alpha^2 M^2.$$

Apply this recursively to get

$$a_k \leq (1 - 2\mu\alpha)^k a_0 + \frac{\alpha M^2}{4\mu}.$$

Given $\epsilon > 0$, find α and K so that both terms on the right-hand side are less than $\epsilon/2$. The right values are:

$$\alpha := \frac{2\epsilon\mu}{M^2}, \qquad K := \frac{M^2}{4\epsilon\mu^2} \log\left(\frac{a_0}{2\epsilon}\right).$$

Constant Step Size, continued

Clearly the choice of α guarantees that the second term is less than $\epsilon/2$.

For the first term, we obtain k from an elementary argument:

$$\begin{aligned} & (1-2\mu\alpha)^k a_0 \le \epsilon/2 \\ \Leftrightarrow \quad & k\log(1-2\mu\alpha) \le -\log(2a_0/\epsilon) \\ \Leftarrow \quad & k(-2\mu\alpha) \le -\log(2a_0/\epsilon) \qquad \text{since } \log(1+x) \le x \\ \Leftrightarrow \quad & k \ge \frac{1}{2\mu\alpha}\log(2a_0/\epsilon), \end{aligned}$$

from which the result follows, by substituting for α in the right-hand side.

If μ is underestimated by a factor of β, we undervalue α by the same factor, and K increases by $1/\beta$. (Easy modification of the analysis above.)

Thus, underestimating μ gives a mild performance penalty.

Constant Step Size: Summary

PRO: Avoid averaging, $1/k$ sublinear convergence, insensitive to underestimates of μ.

CON: Need to estimate probably unknown quantities: besides μ, we need M (to get α) and a_0 (to get K).

We use constant size size in the parallel SG approach HOGWILD!, *to be described later.*

But the step is chosen by trying different options and seeing which seems to be converging fastest. We don't actually try to estimate all the quantities in the theory and construct α that way.

Mirror Descent

The step from x_k to x_{k+1} can be viewed as the solution of a subproblem:

$$x_{k+1} = \arg\min_z \, \nabla f_{i_k}(x_k)^T (z - x_k) + \frac{1}{2\alpha_k} \|z - x_k\|_2^2,$$

a linear estimate of f plus a prox-term. This provides a route to handling constrained problems, regularized problems, alternative prox-functions.

For the constrained problem $\min_{x \in \Omega} f(x)$, simply add the restriction $z \in \Omega$ to the subproblem above.

We may use other prox-functions in place of $(1/2)\|z - x\|_2^2$ above. Such alternatives may be particularly well suited to particular constraint sets Ω.

Mirror Descent is the term used for such generalizations of the SA approaches above.

Mirror Descent cont'd

Given constraint set Ω, choose a norm $\|\cdot\|$ (not necessarily Euclidean). Define the *distance-generating function* ω to be a strongly convex function on Ω with modulus 1 with respect to $\|\cdot\|$, that is,

$$(\omega'(x) - \omega'(z))^T(x - z) \geq \|x - z\|^2, \quad \text{for all } x, z \in \Omega,$$

where $\omega'(\cdot)$ denotes an element of the subdifferential.

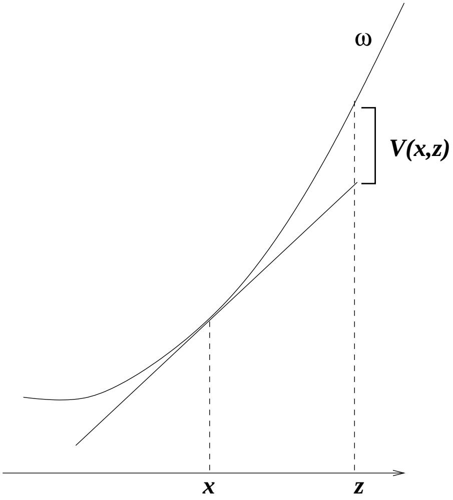
Now define the *prox-function* $V(x, z)$ as follows:

$$V(x, z) = \omega(z) - \omega(x) - \omega'(x)^T(z - x).$$

This is also known as the *Bregman distance*. We can use it in the subproblem in place of $\frac{1}{2}\|\cdot\|^2$:

$$x_{k+1} = \arg\min_{z\in\Omega} \nabla f_{i_k}(x_k)^T(z - x_k) + \frac{1}{\alpha_k} V(z, x_k).$$

Bregman distance is the deviation of ω from linearity:



Bregman Distances: Examples

For *any* Ω, we can use $\omega(x) := (1/2)\|x - \bar{x}\|_2^2$, leading to the "universal" prox-function

$$V(x,z) = (1/2)\|x - z\|_2^2$$

For the simplex

$$\Omega = \{x \in \mathbb{R}^n \,:\, x \ge 0,\ \sum_{i=1}^n x_i = 1\},$$

we can use instead the 1-norm $\|\cdot\|_1$, choose ω to be the entropy function

$$\omega(x) = \sum_{i=1}^n x_i \log x_i,$$

leading to Bregman distance (Kullback-Liebler divergence)

$$V(x,z) = \sum_{i=1}^n z_i \log(z_i/x_i)$$

(standard measure of distance between two probability distributions).

Applications to SVM

SA techniques have an obvious application to linear SVM classification. In fact, they were proposed in this context and analyzed independently by researchers in the ML community for some years.

Codes: SGD (Bottou, 2012), PEGASOS (Shalev-Shwartz et al., 2007).

Tutorial: *Stochastic Optimization for Machine Learning*, (Srebro and Tewari, 2010) for many more details on the connections between stochastic optimization and machine learning.

Related Work: Zinkevich (2003) on online convex programming. Aiming to approximate the minimize the average of a sequence of convex functions, presented sequentially. No i.i.d. assumption, regret-based analysis. Take steplengths of size $O(k^{-1/2})$ in gradient $\nabla f_k(x_k)$ of latest convex function. Average regret is $O(k^{-1/2})$.

Parallel Stochastic Gradient

Several approaches tried, for $f(x) = (1/m)\sum_{i=1}^{m} f_i(x)$.

- **Dual Averaging:** Average gradient estimates evaluated in parallel on different cores. Requires message passing / synchronization (Dekel et al., 2012), (Duchi et al., 2010).
- **Round-Robin:** Cores evaluate ∇f_i in parallel and update centrally stored x in round-robin fashion. Requires synchronization (Langford et al., 2009)
- **Asynchronous:** HOGWILD!**:** Each core grabs the centrally-stored x and evaluates $\nabla f_e(x_e)$ for some random e, then writes the updates back into x (Niu et al., 2011).

HOGWILD!: Each processor runs independently:

1. Sample i_j uniformly from $\{1, 2, \ldots, m\}$;
2. Read current state of x and evaluate $g_{i_j} = \nabla f_{i_j}(x)$;
3. Update $x \leftarrow x - \alpha g_{i_j}$;

HOGWILD! Convergence

- Updates can be old by the time they are applied, but we assume a bound τ on their age.
- Niu et al. (2011) analyze the case in which the update is applied to just one $v \in e$, but can be extended easily to update the full edge e, provided this is done atomically.
- Processors can overwrite each other's work, but sparsity of ∇f_e helps — updates to not interfere too much.

Analysis of Niu et al. (2011) recently simplified / generalized by Richtarik (2012).

In addition to L, μ, M, a_0 defined above, also define quantities that capture the size and interconnectivity of the subvectors x_e.

- $\rho_i = |\{j \, : \, f_i$ and f_j have overlapping support$\}|$;
- $\rho = \sum_{i=1}^{m} \rho_i / m^2$: average rate of overlapping subvectors.

HOGWILD! Convergence

Given $\epsilon \in (0, a_0/L)$, we have

$$\min_{0 \le j \le k} E(f(x_j) - f(x^*)) \le \epsilon,$$

for constant step size

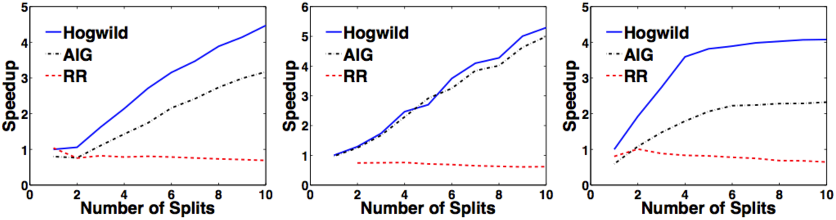
$$\alpha_k \equiv \frac{\mu\epsilon}{(1 + 2\tau\rho)LM^2|E|^2}$$

and $k \ge K$, where

$$K = \frac{(1 + 2\tau\rho)LM^2 m^2}{\mu^2 \epsilon} \log\left(\frac{2La_0}{\epsilon} - 1\right).$$

Broadly, recovers the sublinear $1/k$ convergence rate seen in regular SGD, with the delay τ and overlap measure ρ both appearing linearly.

HOGWILD! Performance



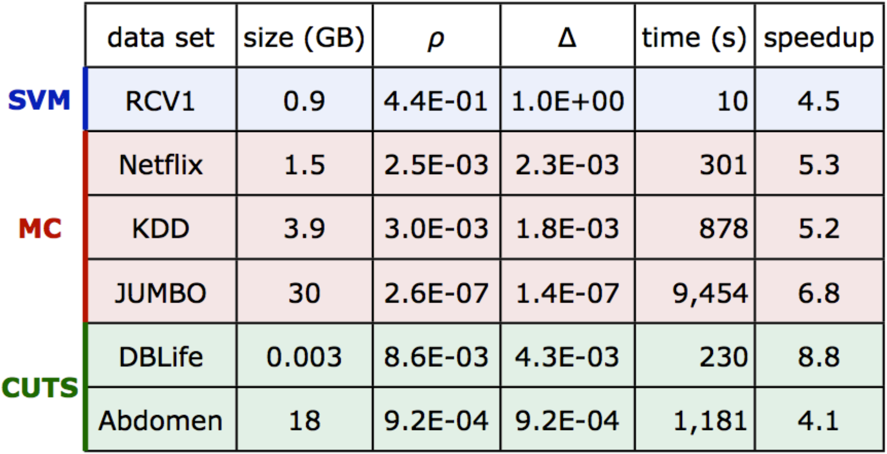
SVM
RCV1

MC
Netflix

CUTS
Abdomen

HOGWILD! compared with averaged gradient (AIG) and round-robin (RR). Experiments run on a 12-core machine. (10 cores used for gradient evaluations, 2 cores for data shuffling.)

Hogwild! Performance



	data set	size (GB)	ρ	Δ	time (s)	speedup
SVM	RCV1	0.9	4.4E-01	1.0E+00	10	4.5
MC	Netflix	1.5	2.5E-03	2.3E-03	301	5.3
	KDD	3.9	3.0E-03	1.8E-03	878	5.2
	JUMBO	30	2.6E-07	1.4E-07	9,454	6.8
CUTS	DBLife	0.003	8.6E-03	4.3E-03	230	8.8
	Abdomen	18	9.2E-04	9.2E-04	1,181	4.1

III. Higher-Order Methods

Methods whose steps use second-order information about the objective f and constraints. This information can be

- Calculated exactly (Newton);
- Approximated, by inspecting changes in the gradient over previous iterations (quasi-Newton);
- Approximated by finite-differencing on the gradient;
- Approximated by re-use from a nearby point;
- Approximated by sampling.

Newton's method is based on a quadratic Taylor-series approximation of a smooth objective f around the current point x:

$$f(x+d) \approx q(x;d) := f(x) + \nabla f(x)^T d + \frac{1}{2} d^T \nabla^2 f(x) d.$$

If $\nabla^2 f(x)$ is positive definite (as near the solution x^*), the Newton step is the d that minimizes $q(\cdot; x)$, explicitly:

$$d = -\nabla^2 f(x)^{-1} \nabla f(x).$$

Can it be practical?

Isn't it too expensive to compute $\nabla^2 f$? Too expensive to store it for large n? What about nonsmooth functions?

- For many learning applications, f is often composed of simple functions, so writing down second derivatives is often easy. Not that we always need them...
- $\nabla^2 f$ may have sparsity, or other structure that allows it to be computed and stored efficiently.
- In many learning applications, nonsmoothness often enters in a highly structured way that can be accounted for explicitly, e.g. $f(x) + \tau \|x\|_1$.
- Often don't need $\nabla^2 f$ — use approximations instead.

In learning and data analysis, the interesting stuff is often happening on a low-dimensional manifold. We might be able to home in on this manifold using first-order methods, then switch to higher-order to improve performance on the later stages.

Implementing Newton's Method

In some cases, it's practical to compute the Hessian $\nabla^2 f(x)$ explicitly, and solve for d by factoring and back-substitution:

$$\nabla^2 f(x) d = -\nabla f(x).$$

Newton's method has local quadratic convergence:

$$\|x_{k+1} - x^*\| = O(\|x_k - x^*\|^2).$$

When the Hessian is not positive definite, can modify the quadratic approximation to ensure that it still has a solution, e.g. by adding a multiple of I to the diagonal:

$$(\nabla^2 f(x) + \delta I) d = -\nabla f(x), \quad \text{for some } \delta \geq 0.$$

("Damped Newton" or "Levenberg-Marquardt")

If we replace $\nabla^2 f(x)$ by 0 and set $\delta = 1/\alpha_k$, we recover steepest descent.

Typically do a line search along d_k to ensure sufficient decrease in f.

Inexact Newton

We can use an iterative method to solve the Newton equations

$$\nabla^2 f(x) d = -\nabla f(x),$$

for example, conjugate gradient. These iterative methods often require matrix-vector multiplications as the key operations:

$$v \leftarrow \nabla^2 f(x) u.$$

These can be performed approximately without knowledge of $\nabla^2 f(x)$, by using finite differencing:

$$\nabla^2 f(x) u \approx [\nabla f(x + \epsilon u) - \nabla f(x)]/\epsilon,$$

for some small scalar ϵ. Each "inner iteration" of the iterative method costs one gradient evaluation.

Sampled Newton

A cheap estimate of $\nabla f(x)$ may be available by sampling.

When f has a partially separable form (common in learning applications):

$$f(x) = \frac{1}{m} \sum_{i=1}^{m} f_i(x),$$

(with m huge), we have

$$\nabla^2 f(x) = \frac{1}{m} \sum_{i=1}^{m} \nabla^2 f_i(x),$$

we can thus choose a subset $\mathcal{B} \subset \{1, 2, \ldots, m\}$ randomly, and define the approximate Hessian H_k as

$$H_k = \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \nabla^2 f_i(x_k),$$

Has been useful in deep learning / speech applications (Byrd et al., 2011).

Quasi-Newton Methods

Maintains an approximation to the Hessian that's filled in using information gained on successive steps.

Generate sequence $\{B_k\}$ of approximate Hessians alongside the iterate sequence $\{x_k\}$, and calculate steps d_k by solving

$$H_k d_k = -\nabla f(x_k).$$

Update from $B_k \rightarrow B_{k+1}$ so that

- Approx Hessian mimics the behavior of the *true* Hessian over this step:

  $$\nabla^2 f(x_{k+1}) s_k \approx y_k,$$

  where $s_k := x_{k+1} - x_k$, $y_k := \nabla f(x_{k+1}) - \nabla f(x_k)$, so we enforce

  $$B_{k+1} s_k = y_k$$

- Make the smallest change consistent with this property (Occam strikes again!)
- Maintain positive definiteness.

BFGS

These principles are satisfied by a family of updates, most famously, BFGS:

$$B_{k+1} = B_k - \frac{B_k s s^T B_k}{s^T B_k s} + \frac{y y^T}{y^T s}$$

where $s = s_k$ and $y = y_k$.

Can start the sequence with $B_0 = \rho I$ for some multiple ρ that's consistent with problem scaling, e.g. $s^T y / s^T s$.

Can maintain instead an approximation H_k to the inverse Hessian — makes the step calculation easier. Update formula is

$$H_{k+1} = (I - \rho s y^T) H_k (I - \rho y s^T) + \rho s s^T,$$

where $\rho = 1/(y^T s)$. Still perform a line search along d_k.

Can prove superlinear local convergence for BFGS and other quasi-Newton methods: $\|x_{k+1} - x^*\| / \|x_k - x^*\| \to 0$. Not as fast as Newton, but fast!

LBFGS

LBFGS doesn't store the $n \times n$ matrices H_k or B_k from BFGS explicitly but rather keeps track of s_k and y_k from the last few iterations (say, $m = 5$ or 10), and reconstructs these matrices as needed.

That is, take an initial matrix (B_0 or H_0) and assume that m steps have been taken since. A simple procedure computes $B_k u$ via a series of inner and outer products with the matrices s_{k-j} and y_{k-j} from the last m iterations: $j = 0, 1, 2, \ldots, m - 1$.

Suitable for problems where n is large. Requires $2mn$ storage and $O(mn)$ linear algebra operations, plus the cost of function and gradient evaluations, and line search.

No superlinear convergence proved, but good behavior is observed on a wide range of applications.

(Liu and Nocedal, 1989)

Newton's Method for Nonlinear Equations

Given the smooth, square nonlinear equations $F(x) = 0$ where $F : \mathbb{R}^n \to \mathbb{R}^n$ (not an optimization problem, but the algorithms are related!) we can use a *first-order* Taylor series expansion:

$$F(x + d) = F(x) + J(x)d + O(\|d\|^2),$$

where $J(x)$ is the Jacobian (the $n \times n$ matrix of first partial derivatives) whose (i, j) element is $\partial F_i / \partial x_j$. (Not necessarily symmetric.)

At iteration k, define the Newton step to be d_k such that

$$F(x_k) + J(x_k)d_k = 0.$$

If there is a solution x^* at which $F(x^*) = 0$ and $J(x^*)$ is nonsingular, then Newton's method again has local quadratic convergence. (Kantorovich.)

Useful in interior-point methods — see below.

Interior-Point Methods

Primal-dual interior-point methods are a powerful class of algorithms for linear programming and convex quadratic programming — both in theory and practice.

Also ~~simple~~ elementary to motivate and implement!

$$\min_x \frac{1}{2}x^T Qx + c^T x \ \text{ s.t. } \ Ax = b, \ x \geq 0,$$

where Q is symmetric positive semidefinite. (LP is a special case.) Optimality conditions are that there exist vectors λ and s such that

$$Qx + c - A^T\lambda - s = 0, \quad Ax = b, \quad (x, s) \geq 0, \quad x_i s_i = 0, \ i = 1, 2, \ldots, n.$$

Defining

$$X = \text{diag}(x_1, x_2, \ldots, x_n), \quad S = \text{diag}(s_1, s_2, \ldots, s_n),$$

we can write the last condition as $XSe = 0$, where $e = (1, 1, \ldots, 1)^T$.

Thus can write the optimality conditions as a square system of constrained, nonlinear equations:

$$\begin{bmatrix} Qx + c - A^T\lambda - s \\ Ax - b \\ XSe \end{bmatrix} = 0, \quad (x, s) \geq 0.$$

Primal-dual interior-point methods generate iterates (x^k, λ^k, s^k) with

- $(x^k, s^k) > 0$ (interior).
- Each step $(\Delta x^k, \Delta\lambda^k, \Delta s^k)$ is a Newton step on a perturbed version of the equations. (The perturbation eventually goes to zero.)
- Use steplength α_k to maintain $(x^{k+1}, s^{k+1}) > 0$. Set

$$(x^{k+1}, \lambda^{k+1}, s^{k+1}) = (x^k, \lambda^k, s^k) + \alpha_k(\Delta x^k, \Delta\lambda^k, \Delta s^k).$$

Perturbed Newton step is a linear system:

$$\begin{bmatrix} Q & -A^T & -I \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x^k \\ \Delta \lambda^k \\ \Delta s^k \end{bmatrix} = \begin{bmatrix} r_x^k \\ r_\lambda^k \\ r_c^k \end{bmatrix}$$

where

$$\begin{aligned} r_x^k &= -(Qx^k + c - A^T \lambda^k - s^k) \\ r_\lambda^k &= -(Ax^k - b) \\ r_c^k &= -X^k S^k e + \sigma_k \mu_k e \end{aligned}$$

r_x^k, r_λ^k, r_c^k are current redisuals, $\mu_k = (x^k)^T s^k / n$ is the current duality gap, and $\sigma_k \in (0, 1]$ is a centering parameter.

Typically there is a lot of structure in the system than can be exploited. Do block elimination, for a start.

See Wright (1997) for a description of primal-dual methods, Gertz and Wright (2003) for a software framework.

Interior-Point Methods for Classification

Interior-point methods were tried early for compressed sensing, regularized least squares, support vector machines.

- SVM with hinge loss formulated as a QP, solved with a primal-dual interior-point method. Included in the OOQP distribution (Gertz and Wright, 2003); see also (Fine and Scheinberg, 2001; Ferris and Munson, 2002).
- Compressed sensing and LASSO variable selection formulated as bound-constrained QPs and solved with primal-dual; or second-order cone programs solved with barrier (Candès and Romberg, 2005)

However they were mostly superseded by first-order methods.

- Stochastic gradient in machine learning (low accuracy, simple data access);
- Gradient projection (GPSR) and prox-gradient (SpaRSA, FPC) in compressed sensing (require only matrix-vector multiplications).

Is it time to reconsider interior-point methods?

Compressed Sensing: Splitting and Conditioning

Consider the ℓ_2-ℓ_1 problem

$$\min_x \frac{1}{2}\|Bx - b\|_2^2 + \tau\|x\|_1,$$

where $B \in \mathbb{R}^{m\times n}$. Recall the bound constrained convex QP formulation:

$$\min_{u\geq 0, v\geq 0} \frac{1}{2}\|B(u-v) - b\|_2^2 + \tau\mathbf{1}^T(u+v).$$

B has special properties associated with compressed sensing matrices (e.g. RIP) that make the problem well conditioned.

(Though the objective is only weakly convex, RIP ensures that when restricted to the optimal support, the active Hessian submatrix is well conditioned.)

Compressed Sensing via Primal-Dual Interior-Point

Fountoulakis et al. (2012) describe an approach that solves the bounded-QP formulation.

- Uses a vanilla primal-dual interior-point framework.
- Solves the linear system at each interior-point iteration with a conjugate gradient (CG) method.
- Preconditions CG with a simple matrix that exploits the RIP properties of B.

Matrix for each linear system in the interior point solver has the form

$$\mathcal{M} := \begin{bmatrix} B^T B & -B^T B \\ -B^T B & B^T B \end{bmatrix} + \begin{bmatrix} U^{-1} S & 0 \\ 0 & V^{-1} T \end{bmatrix},$$

where $U = \text{diag}(u)$, $V = \text{diag}(v)$, and $S = \text{diag}(s)$ and $T = \text{diag}(t)$ are constructed from the Lagrange multipliers for the bound $u \geq 0$, $v \geq 0$.

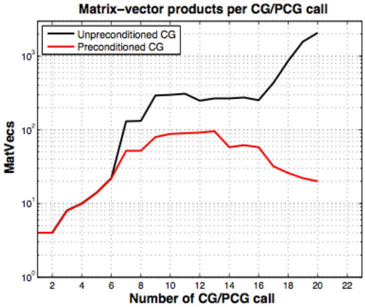
The preconditioner replaces $B^T B$ by $(m/n)I$. Makes sense according to the RIP properties of B.

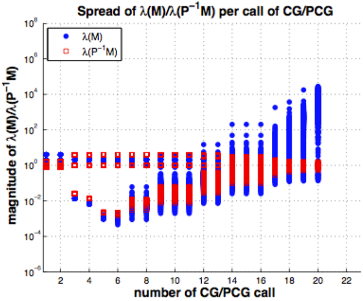
$$\mathcal{P} := \frac{m}{n}\begin{bmatrix} I & -I \\ -I & I \end{bmatrix} + \begin{bmatrix} U^{-1}S & 0 \\ 0 & V^{-1}T \end{bmatrix},$$

Convergence of preconditioned CG depends on the eigenvalue distribution of $\mathcal{P}^{-1}\mathcal{M}$. Gondzio and Fountoulakis (2013) shows that the gap between largest and smallest eigenvalues actually decreases as the interior-point iterates approach a solution. (The gap blows up to ∞ for the non-preconditioned system.)

Overall, the strategy is competitive with first-order methods, on random test problems.

Preconditioning: Effect on Eigenvalue Spread / Solve Time





Red = preconditioned, Blue = non-preconditioned.

Inference via Optimization

Many **inference** problems are formulated as **optimization** problems:

- image reconstruction
- image restoration/denoising
- supervised learning
- unsupervised learning
- statistical inference
- ...

Standard formulation:

- observed data: y
- unknown object (signal, image, vector, matrix,...): x
- inference criterion:

$$\widehat{x} \in \arg\min_{x} g(x, y)$$

Inference via Optimization

Inference criterion: $\widehat{x} \in \arg\min_{x} g(x, y)$

g comes from the application domain (machine learning, signal processing, inverse problems, statistics, bioinformatics,...); examples ahead.

Typical structure of g: $g(x, y) = h(x, y) + \tau\psi(x)$

Inference via Optimization

Inference criterion: $\widehat{x} \in \arg\min_{x} g(x, y)$

g comes from the application domain (machine learning, signal processing, inverse problems, statistics, bioinformatics,...); examples ahead.

Typical structure of g: $g(x, y) = h(x, y) + \tau\psi(x)$

- $h(x, y) \rightarrow$ how well x “fits” / “explains” the data y; (data term, log-likelihood, loss function, observation model,...)

Inference via Optimization

Inference criterion: $\widehat{x} \in \arg\min_{x} g(x, y)$

g comes from the application domain (machine learning, signal processing, inverse problems, statistics, bioinformatics,...); examples ahead.

Typical structure of g: $g(x, y) = h(x, y) + \tau\psi(x)$

- $h(x, y) \rightarrow$ how well x "fits" / "explains" the data y;
  (data term, log-likelihood, loss function, observation model,...)
- $\psi(x) \rightarrow$ knowledge/constraints/structure: the **regularizer**
- $\tau \geq 0$: the **regularization parameter**.

Inference via Optimization

Inference criterion: $\widehat{x} \in \arg\min_{x} g(x, y)$

g comes from the application domain (machine learning, signal processing, inverse problems, statistics, bioinformatics,...); examples ahead.

Typical structure of g: $g(x, y) = h(x, y) + \tau\psi(x)$

- $h(x, y)$ → how well x "fits" / "explains" the data y; (data term, log-likelihood, loss function, observation model,...)
- $\psi(x)$ → knowledge/constraints/structure: the **regularizer**
- $\tau \geq 0$: the **regularization parameter**.
- Since y is fixed, we often write simply $f(x) = h(x, y)$.

IV-A. Sparse Optimization: Formulations

Inference criterion, with regularizer ψ: $\min_x f(x) + \tau\psi(x)$

Typically, the unknown is a vector $x \in \mathbb{R}^n$ or a matrix $x \in \mathbb{R}^{n \times m}$.

Common regularizers impose/encourage one (or a combination of) the following characteristics:

- small norm (vector or matrix)
- sparsity (few nonzeros)
- specific nonzero patterns (*e.g.*, group/tree structure)
- low-rank (matrix)
- smoothness or piece-wise smoothness

Alternative Formulations for Sparse Optimization

- Tikhonov regularization: $$\min_x \; f(x) + \tau\psi(x)$$

- Morozov regularization: $$\begin{array}{ll} \min_x & \psi(x) \\ \text{subject to} & f(x) \le \varepsilon \end{array}$$

- Ivanov regularization: $$\begin{array}{ll} \min_x & f(x) \\ \text{subject to} & \psi(x) \le \delta \end{array}$$

Under mild conditions, these are all equivalent.

Morozov and Ivanov can be written as Tikhonov using indicator functions.

Which one to use? Depends on problem and context.

Cardinality (ℓ_0) is hard to deal with!

Finding the sparsest solution is NP-hard (Muthukrishnan, 2005).

$$\widehat{w} = \arg\min_{w} \|w\|_0 \quad \text{s.t. } \|Aw - y\|_2^2 \leq \delta.$$

Cardinality (ℓ_0) is hard to deal with!

Finding the sparsest solution is NP-hard (Muthukrishnan, 2005).

$$\widehat{w} = \arg\min_{w} \|w\|_0 \quad \text{s.t. } \|Aw - y\|_2^2 \leq \delta.$$

The related best subset selection problem is also NP-hard (Amaldi and Kann, 1998; Davis et al., 1997).

$$\widehat{w} = \arg\min_{w} \|Aw - y\|_2^2 \quad \text{s.t. } \|w\|_0 \leq \tau.$$

Cardinality (ℓ_0) is hard to deal with!

Finding the sparsest solution is NP-hard (Muthukrishnan, 2005).

$$\widehat{w} = \arg\min_{w} \|w\|_0 \quad \text{s.t. } \|Aw - y\|_2^2 \le \delta.$$

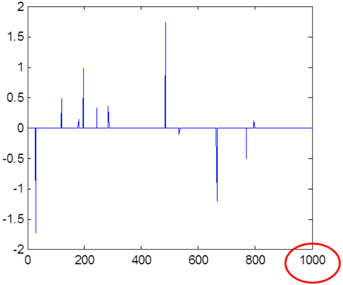
The related best subset selection problem is also NP-hard (Amaldi and Kann, 1998; Davis et al., 1997).

$$\widehat{w} = \arg\min_{w} \|Aw - y\|_2^2 \quad \text{s.t. } \|w\|_0 \le \tau.$$

Under conditions, we can use ℓ_1 as a proxy for ℓ_0! This is the central issue in compressive sensing (CS) (Candès et al., 2006a; Donoho, 2006)

Compressed Sensing — Reconstructions with ℓ_1

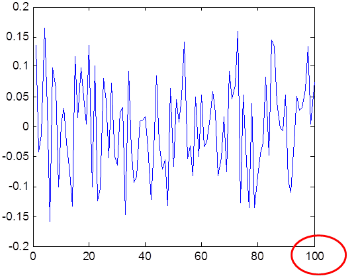
Sparse vector $\mathbf{x}$



$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{n}$$

Random matrix

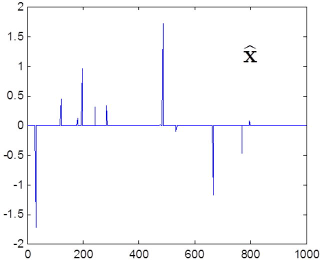
Observed data $\mathbf{y}$



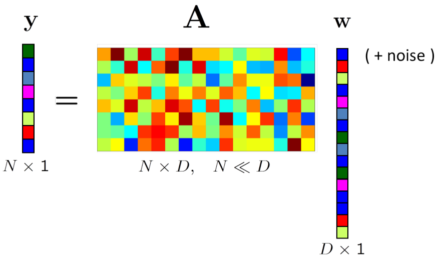
Under certain conditions, "perfect" recovery is possible

$$\widehat{\mathbf{x}} = \arg\min_{\mathbf{x}} \left\{ \|\mathbf{y} - \mathbf{A}\mathbf{x}\|^2 + 2\,\lambda\,\|\mathbf{x}\|_1 \right\}$$

[Candès, Romberg, Tao, 2004 – 2006]
[Donoho, 2006]

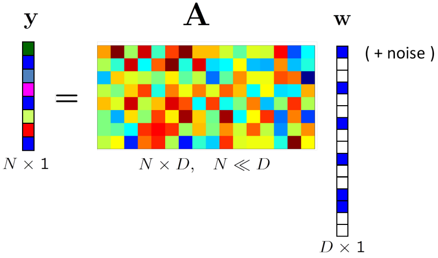


Compressive Sensing in a Nutshell



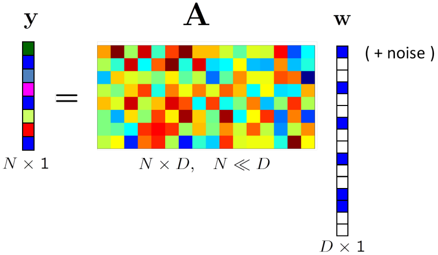
Even in the noiseless case, it seems impossible to recover w from y

Compressive Sensing in a Nutshell



Even in the noiseless case, it seems impossible to recover w from y ...unless, w is sparse and A has some properties.

Compressive Sensing in a Nutshell



Even in the noiseless case, it seems impossible to recover w from y ...unless, w is sparse and A has some properties.

If w is sparse enough and A has certain properties, then w is stably recovered via (Haupt and Nowak, 2006)

$$\begin{aligned} \widehat{w} &= \arg\min_{w} \|w\|_0 \\ &\quad \text{s. t. } \|Aw - y\| \leq \delta \end{aligned}$$

NP-hard!

Compressive Sensing in a Nutshell

The key assumption on A is defined variously as incoherence or restricted isometry (RIP). When such a property holds, we can replace ℓ_0 with ℓ_1 in the formulation: (Candès et al., 2006b):

$$\begin{aligned} \widehat{w} &= \arg\min_{w} \|w\|_1 \\ &\quad \text{subject to } \|Aw - y\| \le \delta \qquad\qquad \text{convex problem} \end{aligned}$$

Compressive Sensing in a Nutshell

The key assumption on A is defined variously as incoherence or restricted isometry (RIP). When such a property holds, we can replace ℓ_0 with ℓ_1 in the formulation: (Candès et al., 2006b):

$$\begin{aligned} \widehat{w} &= \arg\min_{w} \|w\|_1 \\ &\quad \text{subject to } \|Aw - y\| \leq \delta \end{aligned}$$

convex problem

Matrix A satisfies the RIP of order k, with constant $\delta_k \in (0, 1)$, if

$$\|w\|_0 \leq k \;\Rightarrow\; (1 - \delta_k)\|w\|_2^2 \leq \|Aw\| \leq (1 + \delta_k)\|w\|_2^2$$

...i.e., for k-sparse vectors, A is approximately an isometry. Alternatively, say that "all k-column submatrices of A are nearly orthonormal."

Compressive Sensing in a Nutshell

The key assumption on A is defined variously as incoherence or restricted isometry (RIP). When such a property holds, we can replace ℓ_0 with ℓ_1 in the formulation: (Candès et al., 2006b):

$$\begin{aligned} \widehat{w} &= \arg\min_{w} \|w\|_1 \\ & \quad \text{subject to } \|Aw - y\| \le \delta \end{aligned}$$

convex problem

Matrix A satisfies the RIP of order k, with constant $\delta_k \in (0, 1)$, if

$$\|w\|_0 \le k \;\Rightarrow\; (1 - \delta_k)\|w\|_2^2 \le \|Aw\| \le (1 + \delta_k)\|w\|_2^2$$

...i.e., for k-sparse vectors, A is approximately an isometry. Alternatively, say that "all k-column submatrices of A are nearly orthonormal."

Other properties (spark and null space property (NSP)) can be used.

Caveat: checking RIP, NSP, spark is **NP-hard** (Tillmann and Pfetsch, 2012), but these properties are usually satisfied by random matrices.

Underconstrained Systems

Let $\bar{x}$ be the sparsest solution of $Ax = y$, where $A \in \mathbb{R}^{m \times n}$ and $m < n$.

$$\bar{x} = \arg\min \|x\|_0 \;\text{ s.t. }\; Ax = y.$$

Consider instead the convex, ℓ_1 norm version:

$$\min_x \|x\|_1 \;\text{ s.t. }\; Ax = y.$$

Of course, $\bar{x}$ solves this problem too, if $\|\bar{x} + v\|_1 \geq \|\bar{x}\|_1, \;\; \forall v \in \ker(A)$.

Recall: $\ker(A) = \{x \in \mathbb{R}^n : \; Ax = 0\}$ is the kernel (a.k.a. null space) of A.

Coming Up: elementary analysis by Yin and Zhang (2008), based on work by Kashin (1977) and Garnaev and Gluskin (1984).

Equivalence Between ℓ_1 and ℓ_0 Optimization

Minimum ℓ_0 (sparsest) solution: $\bar{x} \in \arg\min \|x\|_0$ s.t. $Ax = y$.

$\bar{x}$ is also a solution of min $\|x\|_1$ s.t. $Ax = y$, provided that

$$\|\bar{x} + v\|_1 \geq \|\bar{x}\|_1, \quad \forall v \in \ker(A).$$

Letting $S = \{i : \bar{x}_i \neq 0\}$ and $Z = \{1, ..., n\} \setminus S$, we have

$$\begin{aligned} \|\bar{x} + v\|_1 &= \|\bar{x}_S + v_S\|_1 + \|v_Z\|_1 \\ &\geq \|\bar{x}_S\|_1 + \|v_Z\|_1 - \|v_S\|_1 \\ &= \|\bar{x}\|_1 + \|v\|_1 - 2\|v_S\|_1 \\ &\geq \|\bar{x}\|_1 + \|v\|_1 - 2\sqrt{k}\|v\|_2. \end{aligned}$$

Hence, $\bar{x}$ minimizes the convex formuation if $\frac{1}{2}\frac{\|v\|_1}{\|v\|_2} \geq \sqrt{k}, \quad \forall v \in \ker(A)$

...but, in general, we have only: $1 \leq \frac{\|v\|_1}{\|v\|_2} \leq \sqrt{n}$.

However, we may have $\frac{\|v\|_1}{\|v\|_2} \gg 1$, if v is restricted to a random subspace.

Bounding the ℓ_1/ℓ_2 Ratio in Random Matrices

If the elements of $A \in \mathbb{R}^{m \times n}$ are sampled i.i.d. from $\mathcal{N}(0,1)$ (zero mean, unit variance Gaussian), then, with high probability,

$$\frac{\|v\|_1}{\|v\|_2} \geq \frac{C\sqrt{m}}{\sqrt{\log(n/m)}}, \quad \text{for all } v \in \ker(A),$$

for some constant C (based on concentration of measure phenomena).

Thus, with high probability, $\bar{x} \in G$, if

$$m \geq \frac{4}{C^2} k \log n.$$

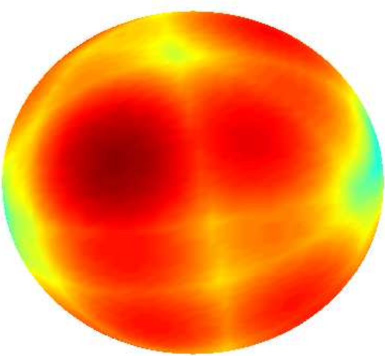
Conclusion: Can solve under-determined system, where A has i.i.d. $\mathcal{N}(0,1)$ elements, by solving the convex problem

$$\min_x \|x\|_1 \;\; s.t. \;\; Ax = b,$$

if the solution is sparse enough, and A is random with $O(k \log n)$ rows.

Ratio $\|v\|_1/\|v\|_2$ on Random Null Spaces

Random $A \in \mathbb{R}^{4\times 7}$, showing ratio $\|v\|_1$ for $v \in \ker(A)$ with $\|v\|_2 = 1$

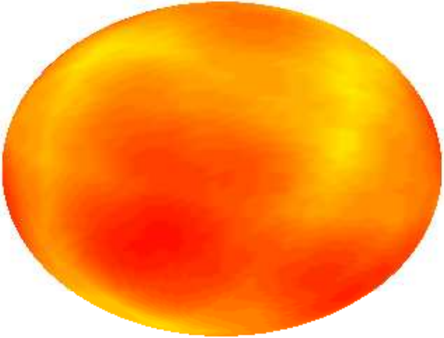


Blue: $\|v\|_1 \approx 1$. Red: ratio $\approx \sqrt{7}$. Note that $\|v\|_1$ is well away from the lower bound of 1 over the whole nullspace.

Ratio $\|v\|_1/\|v\|_2$ on Random Null Spaces

The effect grows more pronounced as m/n grows.

Random $A \in \mathbb{R}^{17\times 20}$, showing ratio $\|v\|_1$ for $v \in N(A)$ with $\|v\|_2 = 1$.



Blue: $\|v\|_1 \approx 1$. Red: $\|v\|_1 \approx \sqrt{20}$. Note that $\|v\|_1$ is closer to upper bound throughout.

The Ubiquitous ℓ_1 Norm

- Lasso (*least absolute shrinkage and selection operator*) (Tibshirani, 1996) a.k.a. *basis pursuit denoising* (Chen et al., 1995):

  $$\min_x \frac{1}{2}\|Ax - y\|_2^2 + \tau\|x\|_1 \;\text{ or }\; \min_x \|Ax - y\|_2^2 \;\text{ s.t. }\; \|x\|_1 \le \delta$$

  or, more generally,

  $$\min_x f(x) + \tau\|x\|_1 \;\text{ or }\; \min_x f(x) \;\text{ s.t. }\; \|x\|_1 \le \delta$$

The Ubiquitous ℓ_1 Norm

- Lasso (*least absolute shrinkage and selection operator*) (Tibshirani, 1996) a.k.a. *basis pursuit denoising* (Chen et al., 1995):

  $$\min_x \frac{1}{2}\|Ax - y\|_2^2 + \tau\|x\|_1 \;\text{ or }\; \min_x \|Ax - y\|_2^2 \;\text{ s.t. }\; \|x\|_1 \le \delta$$

  or, more generally,

  $$\min_x f(x) + \tau\|x\|_1 \;\text{ or }\; \min_x f(x) \;\text{ s.t. }\; \|x\|_1 \le \delta$$

- Widely used outside and much earlier than compressive sensing (statistics, signal processing, neural networks, ...).

The Ubiquitous

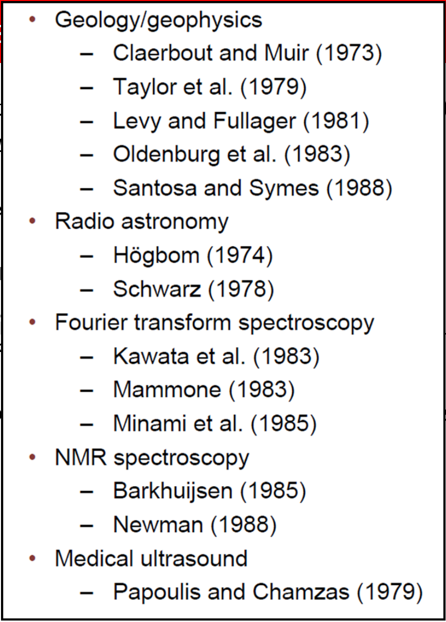
- Lasso (*least ab*) (Tibshirani, 1996)
  a.k.a. *basis pu*

$$\min_x \frac{1}{2}\|A \qquad \text{s.t.} \quad \|x\|_1 \le \delta$$

or, more gene

$$\min_x \qquad x\|_1 \le \delta$$

- Widely used o ssive sensing
  (statistics, sig



- Geology/geophysics
  - Claerbout and Muir (1973)
  - Taylor et al. (1979)
  - Levy and Fullager (1981)
  - Oldenburg et al. (1983)
  - Santosa and Symes (1988)
- Radio astronomy
  - Högbom (1974)
  - Schwarz (1978)
- Fourier transform spectroscopy
  - Kawata et al. (1983)
  - Mammone (1983)
  - Minami et al. (1985)
- NMR spectroscopy
  - Barkhuijsen (1985)
  - Newman (1988)
- Medical ultrasound
  - Papoulis and Chamzas (1979)

from (Goyal et al, 2010)

The Ubiquitous ℓ_1 Norm

- Lasso (*least absolute shrinkage and selection operator*) (Tibshirani, 1996) a.k.a. *basis pursuit denoising* (Chen et al., 1995):

  $$\min_x \frac{1}{2}\|Ax - y\|_2^2 + \tau\|x\|_1 \text{ or } \min_x \|Ax - y\|_2^2 \text{ s.t. } \|x\|_1 \leq \delta$$

  or, more generally,

  $$\min_x f(x) + \tau\|x\|_1 \text{ or } \min_x f(x) \text{ s.t. } \|x\|_1 \leq \delta$$

- Widely used outside and much earlier than compressive sensing (statistics, signal processing, neural networks, ...).

- Many extensions: namely to express structured sparsity (more later).

The Ubiquitous ℓ_1 Norm

- Lasso (*least absolute shrinkage and selection operator*) (Tibshirani, 1996) a.k.a. *basis pursuit denoising* (Chen et al., 1995):

  $$\min_x \frac{1}{2}\|Ax - y\|_2^2 + \tau\|x\|_1 \ \text{ or } \ \min_x \|Ax - y\|_2^2 \ \text{ s.t. } \ \|x\|_1 \leq \delta$$

  or, more generally,

  $$\min_x f(x) + \tau\|x\|_1 \ \text{ or } \ \min_x f(x) \ \text{ s.t. } \ \|x\|_1 \leq \delta$$

- Widely used outside and much earlier than compressive sensing (statistics, signal processing, neural networks, ...).
- Many extensions: namely to express structured sparsity (more later).
- Why does ℓ_1 yield sparse solutions? (next slides)

The Ubiquitous ℓ_1 Norm

- Lasso (*least absolute shrinkage and selection operator*) (Tibshirani, 1996) a.k.a. *basis pursuit denoising* (Chen et al., 1995):

$$\min_x \frac{1}{2}\|Ax - y\|_2^2 + \tau\|x\|_1 \text{ or } \min_x \|Ax - y\|_2^2 \text{ s.t. } \|x\|_1 \le \delta$$

  or, more generally,

$$\min_x f(x) + \tau\|x\|_1 \text{ or } \min_x f(x) \text{ s.t. } \|x\|_1 \le \delta$$

- Widely used outside and much earlier than compressive sensing (statistics, signal processing, neural networks, ...).

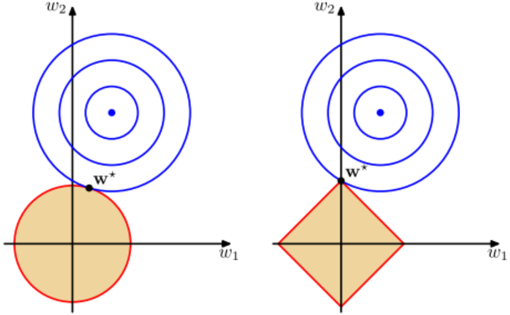
- Many extensions: namely to express structured sparsity (more later).

- Why does ℓ_1 yield sparse solutions? (next slides)

- How to solve these problems? (this tutorial)

Why ℓ_1 Yields Sparse Solution

$$w^* = \arg\min_w \|Aw - y\|_2^2 \quad \text{s.t.} \quad \|w\|_2 \le \delta \qquad \text{vs} \qquad w^* = \arg\min_w \|Aw - y\|_2^2 \quad \text{s.t.} \quad \|w\|_1 \le \delta$$



"Shrinking" with ℓ_1

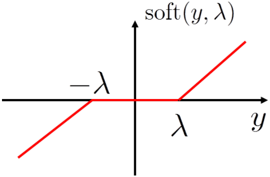
The simplest problem with ℓ_1 regularization

$$\widehat{w} = \arg\min_w \frac{1}{2}(w-y)^2 + \tau|w| = \text{soft}(y,\tau) = \begin{cases} y-\tau & \Leftarrow \quad y > \tau \\ 0 & \Leftarrow \quad |y| \le \tau \\ y+\tau & \Leftarrow \quad y < -\tau \end{cases}$$

"Shrinking" with ℓ_1

The simplest problem with ℓ_1 regularization

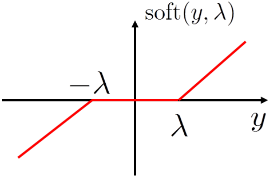
$$\widehat{w} = \arg\min_w \frac{1}{2}(w-y)^2 + \tau|w| = \text{soft}(y,\tau) = \begin{cases} y-\tau & \Leftarrow \quad y > \tau \\ 0 & \Leftarrow \quad |y| \le \tau \\ y+\tau & \Leftarrow \quad y < -\tau \end{cases}$$



“Shrinking” with ℓ_1

The simplest problem with ℓ_1 regularization

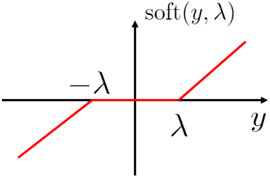
$$\widehat{w} = \arg\min_{w} \frac{1}{2}(w-y)^2 + \tau|w| = \mathrm{soft}(y,\tau) = \begin{cases} y-\tau & \Leftarrow \quad y > \tau \\ 0 & \Leftarrow \quad |y| \le \tau \\ y+\tau & \Leftarrow \quad y < -\tau \end{cases}$$



“Shrinking” with ℓ_1

The simplest problem with ℓ_1 regularization

$$\widehat{w} = \arg\min_w \frac{1}{2}(w-y)^2 + \tau|w| = \text{soft}(y,\tau) = \begin{cases} y-\tau & \Leftarrow \quad y > \tau \\ 0 & \Leftarrow \quad |y| \le \tau \\ y+\tau & \Leftarrow \quad y < -\tau \end{cases}$$



Contrast with the squared ℓ_2 (ridge) regularizer (linear scaling):

$$\widehat{w} = \arg\min_w \frac{1}{2}(w-y)^2 + \frac{\tau}{2}w^2 = \frac{1}{1+\tau}y.$$

(No sparsification.)

Machine/Statistical Learning: Linear Classification

Data N pairs $(x_1, y_1), ..., (x_N, y_N)$, where $x_i \in \mathbb{R}^d$ (feature vectors) and $y_i \in \{-1, +1\}$ (labels).

Goal: find "good" linear classifier (*i.e.*, find the optimal weights):

$$\widehat{y} = \text{sign}([x^T 1]w) = \text{sign}\Big(w_{d+1} + \sum_{j=1}^{d} w_j x_j\Big)$$

Machine/Statistical Learning: Linear Classification

Data N pairs $(x_1, y_1), ..., (x_N, y_N)$, where $x_i \in \mathbb{R}^d$ (feature vectors) and $y_i \in \{-1, +1\}$ (labels).

Goal: find "good" linear classifier (*i.e.*, find the optimal weights):

$$\widehat{y} = \text{sign}([x^T 1]w) = \text{sign}\left(w_{d+1} + \sum_{j=1}^{d} w_j x_j\right)$$

Assumption: data generated i.i.d. by some underlying distribution $P_{X,Y}$

Expected error: $\min_{w \in \mathbb{R}^{d+1}} \mathbb{E}\left(1_{Y([X^T 1]w)<0}\right)$ impossible! $P_{X,Y}$ unknown

Machine/Statistical Learning: Linear Classification

Data N pairs $(x_1, y_1), \ldots, (x_N, y_N)$, where $x_i \in \mathbb{R}^d$ (feature vectors) and $y_i \in \{-1, +1\}$ (labels).

Goal: find "good" linear classifier (*i.e.*, find the optimal weights):

$$\widehat{y} = \text{sign}([x^T 1]w) = \text{sign}\Big(w_{d+1} + \sum_{j=1}^{d} w_j x_j \Big)$$

Assumption: data generated i.i.d. by some underlying distribution $P_{X,Y}$

Expected error: $\min_{w \in \mathbb{R}^{d+1}} \mathbb{E}\big(1_{Y([X^T 1]w)<0}\big)$ impossible! $P_{X,Y}$ unknown

Empirical error (EE): $\min_w \frac{1}{N} \sum_{i=1}^{N} h\big(\underbrace{y_i \; ([x^T 1]w)}_{\text{margin}} \big)$, where $h(z) = 1_{z<0}$.

Machine/Statistical Learning: Linear Classification

Data N pairs $(x_1, y_1), ..., (x_N, y_N)$, where $x_i \in \mathbb{R}^d$ (feature vectors) and $y_i \in \{-1, +1\}$ (labels).

Goal: find "good" linear classifier (*i.e.*, find the optimal weights):

$$\widehat{y} = \text{sign}([x^T 1]w) = \text{sign}\Big(w_{d+1} + \sum_{j=1}^{d} w_j x_j\Big)$$

Assumption: data generated i.i.d. by some underlying distribution $P_{X,Y}$

Expected error: $\min\limits_{w \in \mathbb{R}^{d+1}} \mathbb{E}\big(1_{Y([X^T 1]w)<0}\big)$ impossible! $P_{X,Y}$ unknown

Empirical error (EE): $\min\limits_{w} \frac{1}{N} \sum\limits_{i=1}^{N} h\big(\underbrace{y_i\ ([x^T 1]w)}_{\text{margin}}\big)$, where $h(z) = 1_{z<0}$.

Convexification: EE neither convex nor differentiable (NP-hard problem).
Solution: replace $h : \mathbb{R} \to \{0, 1\}$ with convex loss $L : \mathbb{R} \to \mathbb{R}_+$.

Machine/Statistical Learning: Linear Classification

Criterion: $$\underbrace{\min_w \sum_{i=1}^{N} L\big(\underbrace{y_i\,(w^T x_i + b)}_{\text{margin}}\big)}_{f(w)} + \tau\psi(w)$$

Regularizer: $\psi = \ell_1 \quad \Rightarrow$ encourage sparseness $\quad \Rightarrow$ feature selection

Convex losses: $L : \mathbb{R} \to \mathbb{R}_+$ is a (preferably convex) loss function.

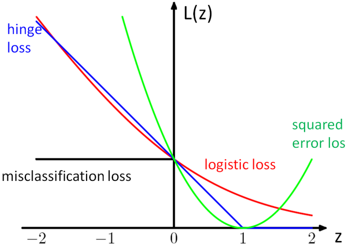
Machine/Statistical Learning: Linear Classification

Criterion: $$\min_w \underbrace{\sum_{i=1}^{N} L\big(\underbrace{y_i\,(w^T x_i + b)}_{\text{margin}}\big)}_{f(w)} + \tau\psi(w)$$

Regularizer: $\psi = \ell_1 \quad \Rightarrow$ encourage sparseness $\Rightarrow$ feature selection

Convex losses: $L : \mathbb{R} \to \mathbb{R}_+$ is a (preferably convex) loss function.

- Misclassification loss: $L(z) = 1_{z<0}$
- Hinge loss: $L(z) = \max\{1 - z, 0\}$
- Logistic loss: $L(z) = \frac{\log\big(1+\exp(-z)\big)}{\log 2}$
- Squared loss: $L(z) = (z-1)^2$



Machine/Statistical Learning: General Formulation

This formulation cover a wide range of linear ML methods:

$$\min_{w} \underbrace{\sum_{i=1}^{N} L\big(y_i \; ([x^T \; 1]w)\big)}_{f(w)} + \tau\psi(w)$$

- Least squares regression: $L(z) = (z-1)^2$, $\psi(w) = 0$.
- Ridge regression: $L(z) = (z-1)^2$, $\psi(w) = \|w\|_2^2$.
- Lasso regression: $L(z) = (z-1)^2$, $\psi(w) = \|w\|_1$
- Logistic classification: $L(z) = \log(1 + \exp(-z))$ (ridge, if $\psi(w) = \|w\|_2^2$
- Sparse logistic regression: $L(z) = \log(1 + \exp(-z))$, $\psi(w) = \|w\|_1$
- Support vector machines: $L(z) = \max\{1 - z, 0\}$, $\psi(w) = \|w\|_2^2$
- Boosting: $L(z) = \exp(-z)$,
- ...

Machine/Statistical Learning: Nonlinear Problems

What about non-linear functions?

Simply use $\widehat{y} = \phi(x,w) = \sum_{j=1}^{D} w_j \, \phi_j(x)$, where $\phi_j : \mathbb{R}^d \to \mathbb{R}$

Essentially, nothing changes; computationally, a lot may change!

$$\min_w \underbrace{\sum_{i=1}^{N} L\big(y_i \; \phi(x,w)\big)}_{f(w)} + \tau\psi(w)$$

Key feature: $\phi(x,w)$ is still linear with respect to w, thus f inherits the convexity of L.

Examples: polynomials, radial basis functions, wavelets, splines, kernels,...

Structured Sparsity — Groups

Main goal: to promote structural patterns, not just penalize cardinality

Structured Sparsity — Groups

Main goal: to promote structural patterns, not just penalize cardinality

Group sparsity: discard/keep entire *groups* of features (Bach et al., 2012)

- density inside each group
- sparsity with respect to the groups which are selected
- choice of groups: prior knowledge about the intended *sparsity patterns*

Structured Sparsity — Groups

Main goal: to promote structural patterns, not just penalize cardinality

Group sparsity: discard/keep entire *groups* of features (Bach et al., 2012)

- density inside each group
- sparsity with respect to the groups which are selected
- choice of groups: prior knowledge about the intended *sparsity patterns*

Yields statistical gains if the assumption is correct (Stojnic et al., 2009)

Structured Sparsity — Groups

Main goal: to promote structural patterns, not just penalize cardinality

Group sparsity: discard/keep entire *groups* of features (Bach et al., 2012)

- density inside each group
- sparsity with respect to the groups which are selected
- choice of groups: prior knowledge about the intended *sparsity patterns*

Yields statistical gains if the assumption is correct (Stojnic et al., 2009)

Many applications:

- feature template selection (Martins et al., 2011)
- multi-task learning (Caruana, 1997; Obozinski et al., 2010)
- learning the structure of graphical models (Schmidt and Murphy, 2010)

Example: Sparsity with Multiple Classes

In multi-class (more than just 2 classes) classification, a common formulation is

$$\widehat{y} = \arg \max_{y \in \{1,...,K\}} x^T w_y$$

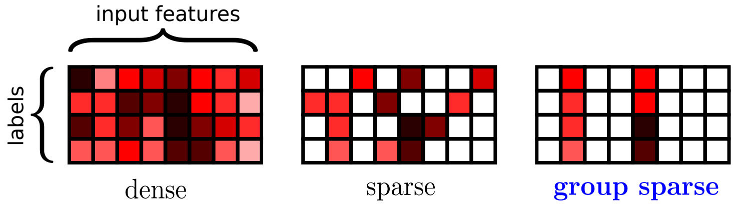
Weight vector $w = (w_1, ..., w_K) \in \mathbb{R}^{Kd}$ has a natural group/grid organization:

Example: Sparsity with Multiple Classes

In multi-class (more than just 2 classes) classification, a common formulation is

$$\widehat{y} = \arg \max_{y \in \{1,\ldots,K\}} x^T w_y$$

Weight vector $w = (w_1, \ldots, w_K) \in \mathbb{R}^{Kd}$ has a natural group/grid organization:



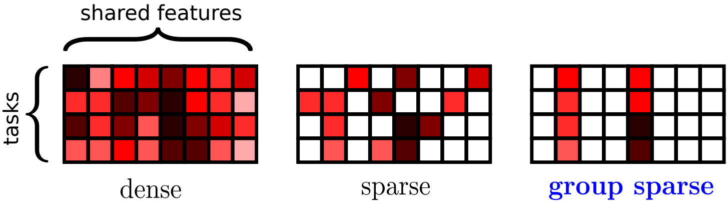
Simple sparsity is wasteful: may still need to keep all the features

Structured sparsity: discard some input features (feature selection)

Example: Multi-Task Learning

Same thing, except now rows are tasks and columns are **features**

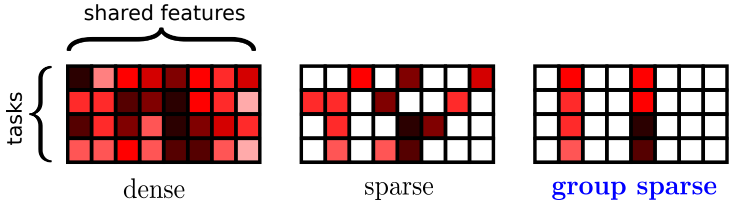
Example: simultaneous regression (seek function into $\mathbb{R}^d \to \mathbb{R}^b$)



Example: Multi-Task Learning

Same thing, except now rows are tasks and columns are **features**

Example: simultaneous regression (seek function into $\mathbb{R}^d \to \mathbb{R}^b$)



Goal: discard features that are irrelevant for *all* tasks

Approach: one group per feature (Caruana, 1997; Obozinski et al., 2010)

Mixed Norm Regularizer

$$\psi(x) = \sum_{i=1}^{M} \|x_{[i]}\|_2$$

where $x_{[i]}$ is a subvector of x (Zhao et al., 2009).

Sum of infinity norms also used (Turlach et al., 2005).

Three scenarios for groups, progressively harder to deal with:

- Non-overlapping Groups
- Tree-structured Groups
- Graph-structured Groups

Matrix Inference Problems

Sparsest solution:

- From $Bx = b \in \mathbb{R}^p$, find $x \in \mathbb{R}^n$ $(p < n)$.
- $\min_x \|x\|_0$ s.t. $Bx = b$
- Yields exact solution, under some conditions.

Matrix Inference Problems

Sparsest solution:

- From $Bx = b \in \mathbb{R}^p$, find $x \in \mathbb{R}^n$ $(p < n)$.
- $\min_x \|x\|_0$ s.t. $Bx = b$
- Yields exact solution, under some conditions.

Lowest rank solution:

- From $\mathcal{B}(X) = b \in \mathbb{R}^p$, find $X \in \mathbb{R}^{m \times n}$ $(p < m\,n)$.
- $\min_X \operatorname{rank}(X)$ s.t. $\mathcal{B}(X) = b$
- Yields exact solution, under some conditions.

Both $NP-$hard (in general); the same is true of noisy versions:

$$\min_{X \in \mathbb{R}^{m \times n}} \operatorname{rank}(X) \;\text{ s.t. }\; \|\mathcal{B}(X) - b\|_2^2$$

Matrix Inference Problems

Sparsest solution:

- From $Bx = b \in \mathbb{R}^p$, find $x \in \mathbb{R}^n$ $(p < n)$.
- $\min_x \|x\|_0$ s.t. $Bx = b$
- Yields exact solution, under some conditions.

Lowest rank solution:

- From $\mathcal{B}(X) = b \in \mathbb{R}^p$, find $X \in \mathbb{R}^{m \times n}$ $(p < m\,n)$.
- $\min_X \operatorname{rank}(X)$ s.t. $\mathcal{B}(X) = b$
- Yields exact solution, under some conditions.

Both $NP-$hard (in general); the same is true of noisy versions:

$$\min_{X \in \mathbb{R}^{m \times n}} \operatorname{rank}(X) \;\text{ s.t. }\; \|\mathcal{B}(X) - b\|_2^2$$

Under some conditions, the same solution is obtained by replacing rank(X) by the nuclear norm $\|X\|_*$ (as any norm, it is convex) (Recht et al., 2010)

Nuclear Norm Regularization

Tikhonov formulation: $\min_X \underbrace{\|\mathcal{B}(X) - b\|_2^2}_{f(X)} + \underbrace{\tau \|X\|_*}_{\tau\psi(X)}$

Nuclear Norm Regularization

Tikhonov formulation: $\min_X \underbrace{\|\mathcal{B}(X) - b\|_2^2}_{f(X)} + \underbrace{\tau \|X\|_*}_{\tau\psi(X)}$

Linear observations: $\mathcal{B} : \mathbb{R}^{m \times n} \to \mathbb{R}^p$, $\big(\mathcal{B}(X)\big)_i = \langle B_{(i)}, X \rangle$,

$$B_{(i)} \in \mathbb{R}^{m \times n}, \text{ and } \langle B, X \rangle = \sum_{ij} B_{ij} X_{ij} = \text{trace}(B^T X)$$

Nuclear Norm Regularization

Tikhonov formulation: $\min_X \underbrace{\|\mathcal{B}(X) - b\|_2^2}_{f(X)} + \underbrace{\tau\|X\|_*}_{\tau\psi(X)}$

Linear observations: $\mathcal{B} : \mathbb{R}^{m\times n} \to \mathbb{R}^p, \ \ \big(\mathcal{B}(X)\big)_i = \langle B_{(i)}, X\rangle,$

$$B_{(i)} \in \mathbb{R}^{m\times n}, \text{ and } \langle B, X\rangle = \sum_{ij} B_{ij}X_{ij} = \text{trace}(B^T X)$$

Particular case: matrix completion, each matrix $B_{(i)}$ has one 1 and is zero everywhere else.

Nuclear Norm Regularization

Tikhonov formulation: $\min_X \underbrace{\|\mathcal{B}(X) - b\|_2^2}_{f(X)} + \underbrace{\tau\|X\|_*}_{\tau\psi(X)}$

Linear observations: $\mathcal{B} : \mathbb{R}^{m\times n} \to \mathbb{R}^p, \;\; \big(\mathcal{B}(X)\big)_i = \langle B_{(i)}, X\rangle,$

$$B_{(i)} \in \mathbb{R}^{m\times n}, \text{ and } \langle B, X\rangle = \sum_{ij} B_{ij}X_{ij} = \text{trace}(B^T X)$$

Particular case: matrix completion, each matrix $B_{(i)}$ has one 1 and is zero everywhere else.

Why does the nuclear norm favor low rank solutions?

Nuclear Norm Regularization

Tikhonov formulation: $\min_X \underbrace{\|\mathcal{B}(X) - b\|_2^2}_{f(X)} + \underbrace{\tau\|X\|_*}_{\tau\psi(X)}$

Linear observations: $\mathcal{B} : \mathbb{R}^{m\times n} \to \mathbb{R}^p$, $\big(\mathcal{B}(X)\big)_i = \langle B_{(i)}, X\rangle$,

$$B_{(i)} \in \mathbb{R}^{m\times n}, \text{ and } \langle B, X\rangle = \sum_{ij} B_{ij}X_{ij} = \text{trace}(B^TX)$$

Particular case: matrix completion, each matrix $B_{(i)}$ has one 1 and is zero everywhere else.

Why does the nuclear norm favor low rank solutions? Let $Y = U\Lambda V^T$ be the singular value decomposition, where $\Lambda = \text{diag}\big(\sigma_1, ..., \sigma_{\min\{m,n\}}\big)$; then

$$\arg\min_X \frac{1}{2}\|Y - X\|_F^2 + \tau\|X\|_* = U \underbrace{\text{soft}(\Lambda, \tau)}_{\text{may yield zeros}} V^T$$

...singular value thresholding (Ma et al., 2011; Cai et al., 2010b)

Another Matrix Inference Problem: Inverse Covariance

Consider n samples $y_1, ..., y_n \in \mathbb{R}^d$ of a Gaussian r.v. $Y \sim \mathcal{N}(\mu, C)$; the log-likelihood is

$$L(P) = \log p(y_1, ..., y_n | P) = \log\det(P) - \operatorname{trace}(SP) + \text{constant}$$

where $S = \frac{1}{n}\sum_{i=1}^{n}(y_i - \mu)(y_i - \mu)^T$ and $P = C^{-1}$ (inverse covariance).

Another Matrix Inference Problem: Inverse Covariance

Consider n samples $y_1, ..., y_n \in \mathbb{R}^d$ of a Gaussian r.v. $Y \sim \mathcal{N}(\mu, C)$; the log-likelihood is

$$L(P) = \log p(y_1, ..., y_n|P) = \log\det(P) - \text{trace}(SP) + \text{constant}$$

where $S = \frac{1}{n}\sum_{i=1}^{n}(y_i - \mu)(y_i - \mu)^T$ and $P = C^{-1}$ (inverse covariance).

Zeros in P reveal conditional independencies between components of Y:

$$P_{ij} = 0 \quad \Leftrightarrow \quad Y_i \perp\!\!\!\perp Y_j | \{Y_k,\ k \neq i, j\}$$

...exploited to infer (in)dependencies among Gaussian variables. Widely used in computational biology and neuroscience, social network analysis, ...

Another Matrix Inference Problem: Inverse Covariance

Consider n samples $y_1, ..., y_n \in \mathbb{R}^d$ of a Gaussian r.v. $Y \sim \mathcal{N}(\mu, C)$; the log-likelihood is

$$L(P) = \log p(y_1, ..., y_n | P) = \log\det(P) - \text{trace}(SP) + \text{constant}$$

where $S = \frac{1}{n}\sum_{i=1}^n (y_i - \mu)(y_i - \mu)^T$ and $P = C^{-1}$ (inverse covariance).

Zeros in P reveal conditional independencies between components of Y:

$$P_{ij} = 0 \quad \Leftrightarrow \quad Y_i \perp\!\!\!\perp Y_j | \{Y_k,\ k \neq i, j\}$$

...exploited to infer (in)dependencies among Gaussian variables. Widely used in computational biology and neuroscience, social network analysis, ...

Sparsity (presence of zeros) in P is encouraged by solving

$$\min_{P \succ 0} \underbrace{-\log\det(P) + \text{trace}(SP)}_{f(P)} + \tau \underbrace{\|\text{vect}(P)\|_1}_{\psi(P)}$$

where $\text{vect}(P) = [P_{1,1}, ..., P_{d,d}]^T$.

Atomic-Norm Regularization

The atomic norm

$$\begin{aligned} \|x\|_{\mathcal{A}} &= \inf\big\{ t > 0 :\ x \in t\,\mathrm{conv}(\mathcal{A}) \big\} \\ &= \inf\Big\{ \sum_{a \in \mathcal{A}} c_a :\ x = \sum_{a \in \mathcal{A}} c_a a,\ c_a \geq 0 \Big\} \end{aligned}$$

...assuming that the centroid of $\mathcal{A}$ is at the origin.

Atomic-Norm Regularization

The atomic norm

$$\begin{aligned}\|x\|_{\mathcal{A}} &= \inf\left\{t > 0 :\ x \in t\,\mathrm{conv}(\mathcal{A})\right\}\\ &= \inf\left\{\sum_{a\in\mathcal{A}} c_a :\ x = \sum_{a\in\mathcal{A}} c_a a,\ c_a \geq 0\right\}\end{aligned}$$

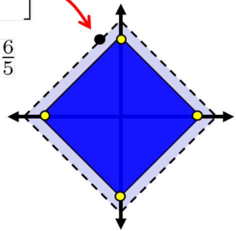
...assuming that the centroid of $\mathcal{A}$ is at the origin.

Example: the ℓ_1 norm as an atomic norm

- $\mathcal{A} = \left\{ \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ -1 \end{bmatrix}, \begin{bmatrix} -1 \\ 0 \end{bmatrix} \right\}$
- $\mathrm{conv}(\mathcal{A}) = B_1(1)$ (ℓ_1 unit ball).
- $\begin{aligned}\|x\|_{\mathcal{A}} &= \inf\left\{t > 0 :\ x \in t\ B_1(1)\right\}\\ &= \|x\|_1\end{aligned}$

$$x = \left[\begin{array}{c} -1/5 \\ 1 \end{array}\right]$$

$$\|x\|_{\mathcal{A}} = \frac{6}{5}$$



Atomic Norms: More Examples

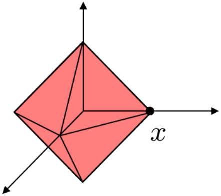
Examples with easy forms:

- *sparse vectors*

  $\mathcal{A} = \{\pm e_i\}_{i=1}^N$

  $\operatorname{conv}(\mathcal{A}) = \text{cross-polytope}$

  $\|x\|_{\mathcal{A}} = \|x\|_1$

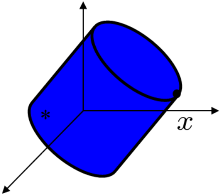


- *low-rank matrices*

  $\mathcal{A} = \{A : \operatorname{rank}(A) = 1, \|A\|_F = 1\}$

  $\operatorname{conv}(\mathcal{A}) = \text{nuclear norm ball}$

  $\|x\|_{\mathcal{A}} = \|x\|_\star$



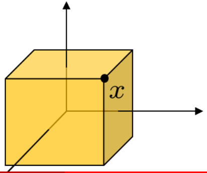
*symmetric matrices

- *binary vectors*

  $\mathcal{A} = \{\pm 1\}^N$

  $\operatorname{conv}(\mathcal{A}) = \text{hypercube}$

  $\|x\|_{\mathcal{A}} = \|x\|_\infty$



Atomic-Norm Regularization

Given an atomic set $\mathcal{A}$, we can adopt an Ivanov formulation

$$\min f(x) \text{ s.t. } \|x\|_{\mathcal{A}} \le \delta$$

(for some $\delta > 0$) tends to recover x with sparse atomic representation.

Atomic-Norm Regularization

Given an atomic set $\mathcal{A}$, we can adopt an Ivanov formulation

$$\min f(x) \text{ s.t. } \|x\|_{\mathcal{A}} \leq \delta$$

(for some $\delta > 0$) tends to recover x with sparse atomic representation.

Can formulate algorithms for the various special cases — but is a general approach available for this formulation?

Atomic-Norm Regularization

Given an atomic set $\mathcal{A}$, we can adopt an Ivanov formulation

$$\min f(x) \text{ s.t. } \|x\|_{\mathcal{A}} \le \delta$$

(for some $\delta > 0$) tends to recover x with sparse atomic representation.

Can formulate algorithms for the various special cases — but is a general approach available for this formulation?

Yes! The conditional gradient! (more later.)

It is also possible to tackle the Tikhonov and Morozov formulations

$$\min_x f(x) + \tau \|x\|_{\mathcal{A}} \quad \text{and} \quad \min_x \|x\|_{\mathcal{A}} \text{ s.t. } f(x) \le \epsilon$$

(more later).

Summary: Sparse Optimization Formulations

- Many inference, learning, signal/image processing problems can be formulated as optimization problems.
- Sparsity-inducing regularizers play an important role in these problems
- There are several way to induce sparsity
- It is possible to formulate structured sparsity
- It is possible to extend the sparsity rationale to other objects, namely matrices
- Atomic norms provide a unified framework for sparsity/simplicity regularization

Now discuss *Algorithms* for Sparse / Regularized Optimization (Section IV-B).

More Basics: Subgradients of Convex Functions

Convexity $\Rightarrow$ continuity (on the domain of the function).

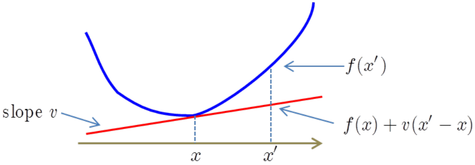
Convexity $\not\Rightarrow$ differentiability. (Example: $\|x\|_1$ is convex but not differentiable when any components of x are zero).

Subgradients generalize gradients for general convex functions:

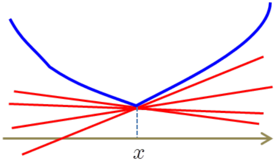
$$\boxed{v \text{ is a subgradient of } f \text{ at } x \text{ if } \; f(x') \geq f(x) + v^T(x' - x)}$$

Subdifferential: $\partial f(x) = \{\text{all subgradients of } f \text{ at } x\}$

If f is differentiable, $\partial f(x) = \{\nabla f(x)\}$



linear lower bound



nondifferentiable case

More on Subgradients and Subdifferentials

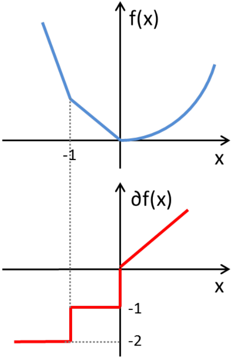
The subdifferential is a set-valued function:

$$f : \mathbb{R}^d \to \mathbb{R} \quad \Rightarrow \quad \partial f : \mathbb{R}^d \to 2^{\mathbb{R}^d} \quad \text{(power set of } \mathbb{R}^d\text{)}$$

Example:

$$f(x) = \begin{cases} -2x-1, & x \le -1 \\ -x, & -1 < x \le 0 \\ x^2/2, & x > 0 \end{cases}$$

$$\partial f(x) = \begin{cases} \{-2\}, & x < -1 \\ [-2,\, -1], & x = -1 \\ \{-1\}, & -1 < x < 0 \\ [-1,\, 0], & x = 0 \\ \{x\}, & x > 0 \end{cases}$$



$$\boxed{x \in \arg\min_x f(x) \;\Leftrightarrow\; 0 \in \partial f(x)}$$

A Key Tool: Moreau's Proximity Operators: "Shrinks"

Moreau (1962) proximity operator

$$\boxed{\widehat{x} \in \arg\min_{x} \frac{1}{2}\|x - y\|_2^2 + \psi(x) =: \text{prox}_{\psi}(y)}$$

...well defined for convex ψ, since $\| \cdot - y\|_2^2$ is coercive and strictly convex.

A Key Tool: Moreau's Proximity Operators: "Shrinks"

Moreau (1962) proximity operator

$$\boxed{\widehat{x} \in \arg\min_x \frac{1}{2}\|x - y\|_2^2 + \psi(x) =: \operatorname{prox}_\psi(y)}$$

...well defined for convex ψ, since $\|\cdot - y\|_2^2$ is coercive and strictly convex.

Example: (seen above) $\operatorname{prox}_{\tau|\cdot|}(y) = \operatorname{soft}(y, \tau) = \operatorname{sign}(y) \max\{|y| - \tau, 0\}$

A Key Tool: Moreau's Proximity Operators: "Shrinks"

Moreau (1962) proximity operator

$$\boxed{\widehat{x} \in \arg\min_{x} \frac{1}{2}\|x - y\|_2^2 + \psi(x) =: \operatorname{prox}_{\psi}(y)}$$

...well defined for convex ψ, since $\| \cdot - y\|_2^2$ is coercive and strictly convex.

Example: (seen above) $\operatorname{prox}_{\tau|\cdot|}(y) = \operatorname{soft}(y, \tau) = \operatorname{sign}(y) \max\{|y| - \tau, 0\}$

Block separability: $x = (x_{[1]}, ..., x_{[M]})$ (a partition of the components of x)

$$\psi(x) = \sum_{i=1}^{M} \psi_i(x_{[i]}) \;\Rightarrow\; (\operatorname{prox}_{\psi}(y))_i = \operatorname{prox}_{\psi_i}(y_{[i]})$$

Relationship with subdifferential: $z = \operatorname{prox}_{\psi}(y) \;\Leftrightarrow\; y - z \in \partial\psi(z)$

Important Proximity Operators

- Soft-thresholding is the proximity operator of the ℓ_1 norm.

Important Proximity Operators

- Soft-thresholding is the proximity operator of the ℓ_1 norm.
- Consider the indicator $\iota_{\mathcal{S}}$ of a convex set $\mathcal{S}$;

$$\text{prox}_{\iota_{\mathcal{S}}}(u) = \arg\min_{x} \frac{1}{2}\|x-u\|_2^2 + \iota_{\mathcal{S}}(x) = \arg\min_{x\in\mathcal{S}} \frac{1}{2}\|x-y\|_2^2 = P_{\mathcal{S}}(u)$$

...the Euclidean projection on $\mathcal{S}$.

Important Proximity Operators

- Soft-thresholding is the proximity operator of the ℓ_1 norm.
- Consider the indicator $\iota_{\mathcal{S}}$ of a convex set $\mathcal{S}$;

$$\text{prox}_{\iota_{\mathcal{S}}}(u) = \arg\min_x \frac{1}{2}\|x-u\|_2^2 + \iota_{\mathcal{S}}(x) = \arg\min_{x\in\mathcal{S}} \frac{1}{2}\|x-y\|_2^2 = P_{\mathcal{S}}(u)$$

...the Euclidean projection on $\mathcal{S}$.

- Squared Euclidean norm (separable, smooth):

$$\text{prox}_{(\tau/2)\|\cdot\|_2^2}(y) = \arg\min_x \frac{1}{2}\|x-y\|_2^2 + \frac{\tau}{2}\|x\|_2^2 = \frac{y}{1+\tau}$$

Important Proximity Operators

- Soft-thresholding is the proximity operator of the ℓ_1 norm.
- Consider the indicator $\iota_{\mathcal{S}}$ of a convex set $\mathcal{S}$;

$$\text{prox}_{\iota_{\mathcal{S}}}(u) = \arg\min_{x} \frac{1}{2}\|x-u\|_2^2 + \iota_{\mathcal{S}}(x) = \arg\min_{x\in\mathcal{S}} \frac{1}{2}\|x-y\|_2^2 = P_{\mathcal{S}}(u)$$

...the Euclidean projection on $\mathcal{S}$.

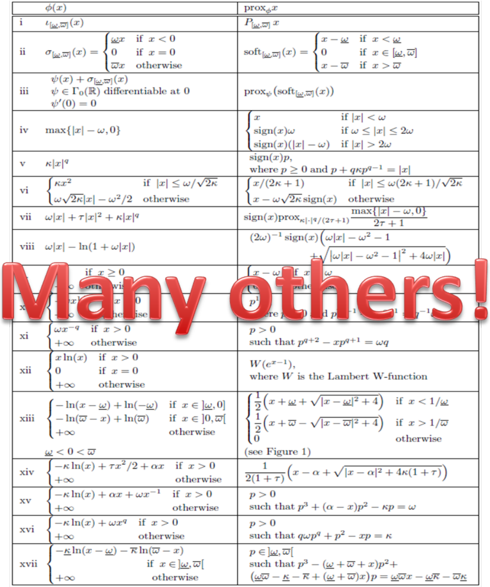
- Squared Euclidean norm (separable, smooth):

$$\text{prox}_{(\tau/2)\|\cdot\|_2^2}(y) = \arg\min_{x} \frac{1}{2}\|x-y\|_2^2 + \frac{\tau}{2}\|x\|_2^2 = \frac{y}{1+\tau}$$

- Euclidean norm (not separable, nonsmooth):

$$\text{prox}_{\tau\|\cdot\|_2}(y) = \begin{cases} \frac{x}{\|x\|_2}(\|x\|_2 - \tau), & \text{if } \|x\|_2 > \tau \\ 0 & \text{if } \|x\|_2 \le \tau \end{cases}$$

More Proximity Operators



	$\phi(x)$	$\mathrm{prox}_\phi x$
i	$\iota_{[\underline{\omega},\overline{\omega}]}(x)$	$P_{[\underline{\omega},\overline{\omega}]}\, x$
ii	$\sigma_{[\underline{\omega},\overline{\omega}]}(x) = \begin{cases} \underline{\omega}x & \text{if } x < 0 \\ 0 & \text{if } x = 0 \\ \overline{\omega}x & \text{otherwise} \end{cases}$	$\mathrm{soft}_{[\underline{\omega},\overline{\omega}]}(x) = \begin{cases} x - \underline{\omega} & \text{if } x < \underline{\omega} \\ 0 & \text{if } x \in [\underline{\omega},\overline{\omega}] \\ x - \overline{\omega} & \text{if } x > \overline{\omega} \end{cases}$
iii	$\psi(x) + \sigma_{[\underline{\omega},\overline{\omega}]}(x)$ $\psi \in \Gamma_0(\mathbb{R})$ differentiable at 0 $\psi'(0) = 0$	$\mathrm{prox}_\psi\big(\mathrm{soft}_{[\underline{\omega},\overline{\omega}]}(x)\big)$
iv	$\max\{\lvert x\rvert - \omega, 0\}$	$\begin{cases} x & \text{if } \lvert x\rvert < \omega \\ \mathrm{sign}(x)\omega & \text{if } \omega \le \lvert x\rvert \le 2\omega \\ \mathrm{sign}(x)(\lvert x\rvert - \omega) & \text{if } \lvert x\rvert > 2\omega \end{cases}$
v	$\kappa\lvert x\rvert^q$	$\mathrm{sign}(x)p$, where $p \ge 0$ and $p + q\kappa p^{q-1} = \lvert x\rvert$
vi	$\begin{cases} \kappa x^2 & \text{if } \lvert x\rvert \le \omega/\sqrt{2\kappa} \\ \omega\sqrt{2\kappa}\lvert x\rvert - \omega^2/2 & \text{otherwise} \end{cases}$	$\begin{cases} x/(2\kappa+1) & \text{if } \lvert x\rvert \le \omega(2\kappa+1)/\sqrt{2\kappa} \\ x - \omega\sqrt{2\kappa}\,\mathrm{sign}(x) & \text{otherwise} \end{cases}$
vii	$\omega\lvert x\rvert + \tau\lvert x\rvert^2 + \kappa\lvert x\rvert^q$	$\mathrm{sign}(x)\mathrm{prox}_{\kappa\lvert\cdot\rvert^q/(2\tau+1)} \dfrac{\max\{\lvert x\rvert - \omega, 0\}}{2\tau+1}$
viii	$\omega\lvert x\rvert - \ln(1 + \omega\lvert x\rvert)$	$(2\omega)^{-1}\,\mathrm{sign}(x)\Big(\omega\lvert x\rvert - \omega^2 - 1 + \sqrt{\big\lvert\omega\lvert x\rvert - \omega^2 - 1\big\rvert^2 + 4\omega\lvert x\rvert}\Big)$
[illegible]	$\begin{cases} [illegible] & \text{if } x \ge 0 \\ +\infty & \text{otherwise} \end{cases}$	$\begin{cases} x - \omega & \text{if } x\ [illegible]\ \omega \\ [illegible] & \text{otherwise} \end{cases}$
x	$\begin{cases} [illegible] & \text{if } x\ [illegible]\ 0 \\ +\infty & \text{otherwise} \end{cases}$	p^1 [illegible] where p [illegible] 0 and p [illegible]
xi	$\begin{cases} \omega x^{-q} & \text{if } x > 0 \\ +\infty & \text{otherwise} \end{cases}$	$p > 0$ such that $p^{q+2} - xp^{q+1} = \omega q$
xii	$\begin{cases} x\ln(x) & \text{if } x > 0 \\ 0 & \text{if } x = 0 \\ +\infty & \text{otherwise} \end{cases}$	$W(e^{x-1})$, where W is the Lambert W-function
xiii	$\begin{cases} -\ln(x - \underline{\omega}) + \ln(-\underline{\omega}) & \text{if } x \in]\underline{\omega}, 0] \\ -\ln(\overline{\omega} - x) + \ln(\overline{\omega}) & \text{if } x \in]0, \overline{\omega}[\\ +\infty & \text{otherwise} \end{cases}$ $\underline{\omega} < 0 < \overline{\omega}$	$\begin{cases} \frac{1}{2}\Big(x + \underline{\omega} + \sqrt{\lvert x - \underline{\omega}\rvert^2 + 4}\Big) & \text{if } x < 1/\underline{\omega} \\ \frac{1}{2}\Big(x + \overline{\omega} - \sqrt{\lvert x - \overline{\omega}\rvert^2 + 4}\Big) & \text{if } x > 1/\overline{\omega} \\ 0 & \text{otherwise} \end{cases}$ (see Figure 1)
xiv	$\begin{cases} -\kappa\ln(x) + \tau x^2/2 + \alpha x & \text{if } x > 0 \\ +\infty & \text{otherwise} \end{cases}$	$\dfrac{1}{2(1+\tau)}\Big(x - \alpha + \sqrt{\lvert x - \alpha\rvert^2 + 4\kappa(1+\tau)}\Big)$
xv	$\begin{cases} -\kappa\ln(x) + \alpha x + \omega x^{-1} & \text{if } x > 0 \\ +\infty & \text{otherwise} \end{cases}$	$p > 0$ such that $p^3 + (\alpha - x)p^2 - \kappa p = \omega$
xvi	$\begin{cases} -\kappa\ln(x) + \omega x^q & \text{if } x > 0 \\ +\infty & \text{otherwise} \end{cases}$	$p > 0$ such that $q\omega p^q + p^2 - xp = \kappa$
xvii	$\begin{cases} -\underline{\kappa}\ln(x - \underline{\omega}) - \overline{\kappa}\ln(\overline{\omega} - x) & \\ & \text{if } x \in]\underline{\omega}, \overline{\omega}[\\ +\infty & \text{otherwise} \end{cases}$	$p \in]\underline{\omega}, \overline{\omega}[$ such that $p^3 - (\underline{\omega} + \overline{\omega} + x)p^2 + (\underline{\omega}\overline{\omega} - \underline{\kappa} - \overline{\kappa} + (\underline{\omega} + \overline{\omega})x)p = \underline{\omega}\overline{\omega}x - \underline{\omega}\overline{\kappa} - \overline{\omega}\underline{\kappa}$

Many others!

(Combettes and Pesquet, 2011)

Matrix Nuclear Norm and its Prox Operator

- Recall the trace/nuclear norm: $\|X\|_* = \sum_{i=1}^{\min\{m,n\}} \sigma_i.$
- The dual of a Schatten p-norm is a Schatten q-norm, with $\frac{1}{q} + \frac{1}{p} = 1.$ Thus, the dual of the nuclear norm is the spectral norm:

$$\|X\|_\infty = \max\left\{\sigma_1, ..., \sigma_{\min\{m,n\}}\right\}.$$

- If $Y = U \Lambda V^T$ is the SVD of Y, we have

$$\begin{aligned} \text{prox}_{\tau\|\cdot\|_*}(Y) &= U\Lambda V^T - P_{\{X:\max\{\sigma_1,...,\sigma_{\min\{m,n\}}\}\leq\tau\}}(U\Lambda V^T) \\ &= U \,\text{soft}\big(\Lambda, \tau\big)\, V^T. \end{aligned}$$

Basic Proximal-Gradient Algorithm

Take this step at iteration k:

$$x_{k+1} = \mathrm{prox}_{\alpha_k \psi}\big(x_k - \alpha_k \nabla f(x_k)\big).$$

This approach goes by many names, such as

- "proximal gradient algorithm" (PGA),
- "iterative shrinkage/thresholding" (IST),
- "forward-backward splitting" (FBS)

It it has been reinvented several times in different communities: optimization, partial differential equations, convex analysis, signal processing, machine learning.

Convergence of the Proximal-Gradient Algorithm

- Basic algorithm: $x_{k+1} = \text{prox}_{\alpha_k \psi}\big(x_k - \alpha_k \nabla f(x_k)\big)$
- Generalized (possibly inexact) version:

$$x_{k+1} = (1 - \lambda_k) x_k + \lambda_k \Big(\text{prox}_{\alpha_k \psi}\big(x_k - \alpha_k \nabla f(x_k) + b_k\big) + a_k \Big)$$

  where a_k and b_k are “errors” in computing the prox and the gradient; λ_k is an over-relaxation parameter.

- Convergence is guaranteed (Combettes and Wajs, 2006) if
  - ✓ $0 < \inf \alpha_k \leq \sup \alpha_k < \frac{2}{L}$
  - ✓ $\lambda_k \in (0, 1]$, with $\inf \lambda_k > 0$
  - ✓ $\sum_k^\infty \|a_k\| < \infty$ and $\sum_k^\infty \|b_k\| < \infty$

More on IST/FBS/PGA for the ℓ_2-ℓ_1 Case

- Problem: $\widehat{x} \in G = \arg\min_{x \in \mathbb{R}^n} \frac{1}{2}\|Bx - b\|_2^2 + \tau\|x\|_1$ (recall $B^T B \preceq LI$)
- IST/FBS/PGA becomes $\boxed{x_{k+1} = \text{soft}\big(x_k - \alpha B^T(Bx - b), \alpha\tau\big)}$ with $\alpha < 2/L$.

More on IST/FBS/PGA for the ℓ_2-ℓ_1 Case

- Problem: $\widehat{x} \in G = \arg\min_{x\in\mathbb{R}^n} \frac{1}{2}\|B\,x - b\|_2^2 + \tau\|x\|_1$ (recall $B^TB \preceq LI$)
- IST/FBS/PGA becomes $\boxed{x_{k+1} = \text{soft}\big(x_k - \alpha B^T(B\,x - b), \alpha\tau\big)}$ with $\alpha < 2/L$.
- The zero set: $\mathcal{Z} \subseteq \{1, ..., n\}$: $\widehat{x} \in G \Rightarrow \widehat{x}_{\mathcal{Z}} = 0$
- Zeros are found in a finite number of iterations (Hale et al., 2008): after a finite number of iterations $(x_k)_{\mathcal{Z}} = 0$.

More on IST/FBS/PGA for the ℓ_2-ℓ_1 Case

- Problem: $\widehat{x} \in G = \arg\min_{x \in \mathbb{R}^n} \frac{1}{2}\|B\,x - b\|_2^2 + \tau\|x\|_1$ (recall $B^TB \preceq LI$)
- IST/FBS/PGA becomes $\boxed{x_{k+1} = \text{soft}\big(x_k - \alpha B^T(B\,x - b), \alpha\tau\big)}$ with $\alpha < 2/L$.
- The zero set: $\mathcal{Z} \subseteq \{1, ..., n\}$: $\widehat{x} \in G \Rightarrow \widehat{x}_{\mathcal{Z}} = 0$
- Zeros are found in a finite number of iterations (Hale et al., 2008): after a finite number of iterations $(x_k)_{\mathcal{Z}} = 0$.
- After discovery of the zero set, reduces to minimizing an unconstrained quadratic over the nonzero elements of x: $\mathcal{N} := \{1, 2, \ldots, n\} \setminus \mathcal{Z}$. By RIP property, the submatrix $B_{\mathcal{N}}^T B_{\mathcal{N}}$ is well conditioned, so convergence is typically fast linear.

Heavy Ball Acceleration: FISTA

- FISTA (*fast iterative shrinkage-thresholding algorithm*) is heavy-ball-type acceleration of IST (based on Nesterov (1983)) (Beck and Teboulle, 2009a).

Initialize: Choose $\alpha \le 1/L$, x_0; set $y_1 = x_0$, $t_1 = 1$;

Iterate:

$$x_k \leftarrow \text{prox}_{\tau\alpha\psi}\big(y_k - \alpha \nabla f(y_k)\big);$$

$$t_{k+1} \leftarrow \tfrac{1}{2}\left(1 + \sqrt{1 + 4t_k^2}\right);$$

$$y_{k+1} \leftarrow x_k + \frac{t_k - 1}{t_{k+1}}(x_k - x_{k-1}).$$

Heavy Ball Acceleration: FISTA

- FISTA (*fast iterative shrinkage-thresholding algorithm*) is heavy-ball-type acceleration of IST (based on Nesterov (1983)) (Beck and Teboulle, 2009a).

```
Initialize: Choose $\alpha \le 1/L$, $x_0$; set $y_1 = x_0$, $t_1 = 1$;

Iterate:    $x_k \leftarrow \text{prox}_{\tau\alpha\psi}(y_k - \alpha\nabla f(y_k))$;
            $t_{k+1} \leftarrow \frac{1}{2}\left(1 + \sqrt{1 + 4t_k^2}\right)$;
            $y_{k+1} \leftarrow x_k + \frac{t_k - 1}{t_{k+1}}(x_k - x_{k-1})$.
```

- Acceleration:

$$\text{FISTA: } f(x_k) - f(\widehat{x}) \sim O\left(\frac{1}{k^2}\right) \qquad \text{IST: } f(x_k) - f(\widehat{x}) \sim O\left(\frac{1}{k}\right).$$

Heavy Ball Acceleration: FISTA

- FISTA (*fast iterative shrinkage-thresholding algorithm*) is heavy-ball-type acceleration of IST (based on Nesterov (1983)) (Beck and Teboulle, 2009a).

```
Initialize: Choose $\alpha \le 1/L$, $x_0$; set $y_1 = x_0$, $t_1 = 1$;
Iterate:    $x_k \leftarrow \mathrm{prox}_{\tau\alpha\psi}(y_k - \alpha\nabla f(y_k))$;
            $t_{k+1} \leftarrow \frac{1}{2}\left(1 + \sqrt{1 + 4t_k^2}\right)$;
            $y_{k+1} \leftarrow x_k + \frac{t_k - 1}{t_{k+1}}(x_k - x_{k-1})$.
```

- Acceleration:

$$\text{FISTA: } f(x_k) - f(\widehat{x}) \sim O\left(\frac{1}{k^2}\right) \qquad \text{IST: } f(x_k) - f(\widehat{x}) \sim O\left(\frac{1}{k}\right).$$

- When L is not known, increase an estimate of L until it's big enough.

Acceleration via Larger Steps: SpaRSA

The standard step-size $\alpha_k \leq \frac{2}{L}$ in IST too timid.

The SpARSA (sparse reconstruction by separable approximation) framework proposes bolder choices of α_k (Wright et al., 2009a):

- Barzilai-Borwein (see above), to mimic Newton steps — or at least get the scaling right.
- keep increasing α_k until monotonicity is violated: backtrack.

Convergence to critical points (minima in the convex case) is guaranteed for a safeguarded version: ensure sufficient decrease w.r.t. the worst value in previous M iterations.

Acceleration by Continuation

- IST/FBS/PGA can be very slow if τ is very small and/or f is poorly conditioned.

Acceleration by Continuation

- IST/FBS/PGA can be very slow if τ is very small and/or f is poorly conditioned.
- A very simple acceleration strategy: continuation/homotopy

```
Initialization: Set $\tau_0 \gg \tau$, starting point $\bar{x}$, factor $\sigma \in (0,1)$, and $k = 0$.
Iterations: Find approx solution $x(\tau_k)$ of $\min_x f(x) + \tau_k \psi(x)$, starting from $\bar{x}$;
            if $\tau_k = \tau_f$ STOP;
            Set $\tau_{k+1} \leftarrow \max(\tau_f, \sigma\tau_k)$ and $\bar{x} \leftarrow x(\tau_k)$;
```

Acceleration by Continuation

- IST/FBS/PGA can be very slow if τ is very small and/or f is poorly conditioned.

- A very simple acceleration strategy: continuation/homotopy

```
Initialization: Set $\tau_0 \gg \tau$, starting point $\bar{x}$, factor $\sigma \in (0,1)$, and $k = 0$.
Iterations: Find approx solution $x(\tau_k)$ of $\min_x f(x) + \tau_k \psi(x)$, starting from $\bar{x}$;
            if $\tau_k = \tau_f$ STOP;
            Set $\tau_{k+1} \leftarrow \max(\tau_f, \sigma\tau_k)$ and $\bar{x} \leftarrow x(\tau_k)$;
```

- Often the solution path $x(\tau)$, for a range of values of τ is desired, anyway (e.g., within an outer method to choose an optimal τ)

Acceleration by Continuation

- IST/FBS/PGA can be very slow if τ is very small and/or f is poorly conditioned.

- A very simple acceleration strategy: continuation/homotopy

Initialization: Set $\tau_0 \gg \tau$, starting point $\bar{x}$, factor $\sigma \in (0,1)$, and $k = 0$.

```
Iterations: Find approx solution $x(\tau_k)$ of $\min_x f(x) + \tau_k \psi(x)$, starting from $\bar{x}$;
            if $\tau_k = \tau_f$ STOP;
            Set $\tau_{k+1} \leftarrow \max(\tau_f, \sigma\tau_k)$ and $\bar{x} \leftarrow x(\tau_k)$;
```

- Often the solution path $x(\tau)$, for a range of values of τ is desired, anyway (e.g., within an outer method to choose an optimal τ)

- Shown to be very effective in practice (Hale et al., 2008; Wright et al., 2009a). Recently analyzed by Xiao and Zhang (2012).

A Final Touch: Debiasing

Consider problems of the form $\widehat{x} \in \arg\min_{x \in \mathbb{R}^n} \frac{1}{2}\|Bx - b\|_2^2 + \tau\|x\|_1$

Often, the original goal was to minimize the quadratic term, after the support of x had been found. But the ℓ_1 term can cause the nonzero values of x_i to be "suppressed."

Debiasing:

- ✓ find the zero set (complement of the support of $\widehat{x}$): $\mathcal{Z}(\widehat{x}) = \{1, ..., n\} \setminus \text{supp}(\widehat{x})$.
- ✓ solve $\min_x \|Bx - b\|_2^2$ s.t. $x_{\mathcal{Z}(\widehat{x})} = 0$. (Fix the zeros and solve an unconstrained problem over the support.)

  Often, this problem has to be solved using an algorithm that only involves products by B and B^T, since this matrix cannot be partitioned.

Identifying Optimal Manifolds

Identification of the manifold of the regularizer ψ on which x^* lies can improve algorithm performance, by focusing attention on a reduced space. We can thus evaluate *partial* gradients and Hessians, restricted to just this space.

For nonsmooth regularizer ψ, the optimal manifold is a smooth surface passing through x^* along which the restriction of ψ is smooth.

Example: for $\psi(x) = \|x\|_1$, have manifold consisting of z with

$$z_i \begin{cases} \geq 0 & \text{if } x_i^* > 0 \\ \leq 0 & \text{if } x_i^* < 0 \\ = 0 & \text{if } x_i^* = 0. \end{cases}$$

If we know the optimal nonzero components, we know the manifold. We could restrict the search to just this set of nonzeros.

Identification Properties of Prox-Gradient Algorithms

When the optimal manifold is partly smooth (that is, parametrizable by smooth functions and otherwise well behaved) and prox-regular, and the minimizer is nondegenerate, then the shrink approach can identify it from any sufficiently close x. That is,

$$S_\tau(x - \alpha \nabla f(x), \alpha)$$

lies on the optimal manifold, for α bounded away from 0 and x in a neighborhood of x^*. (Consequence of Lewis and Wright (2008).)

For $\psi(x) = \|x\|_1$, shrink algorithms identify the correct nonzero set, provided there are no “borderline” components (that is, the optimal nonzero set would not change with an arbitrarily small perturbation to the data).

Can use a heuristic to identify when the nonzero set settles down, then switch to second phase to conduct a search on the reduced space of “possible nonzeros.”

Conditional Gradient

Also known as "Frank-Wolfe" after the authors who devised it in the 1950s. Later analysis by Dunn (around 1990). Suddenly a topic of enormous renewed interest; see for example (Jaggi, 2013).

$$\min_{x \in \Omega} f(x),$$

where f is a convex function and Ω is a closed, bounded, convex set.

Start at $x_0 \in \Omega$. At iteration k:

$$v_k := \arg\min_{v \in \Omega} v^T \nabla f(x_k);$$

$$x_{k+1} := x_k + \alpha_k (v_k - x_k), \quad \alpha_k = \frac{2}{k+2}.$$

- Potentially useful when it is easy to minimize a linear function over the *original* constraint set Ω;
- Admits an elementary convergence theory: $1/k$ sublinear rate.
- Same convergence theory holds if we use a line search for α_k.

Conditional Gradient for Atomic-Norm Constraints

Conditional Gradient is particularly useful for optimization over atomic-norm constraints.

$$\min \ f(x) \ \text{ s.t. } \ \|x\|_{\mathcal{A}} \le \tau.$$

Reminder: Given the set of atoms $\mathcal{A}$ (possibly infinite) we have

$$\|x\|_{\mathcal{A}} := \inf \left\{ \sum_{a \in \mathcal{A}} c_a \ : \ x = \sum_{a \in \mathcal{A}} c_a a, \ c_a \ge 0 \right\}.$$

The search direction v_k is $\tau \bar{a}_k$, where

$$\bar{a}_k := \arg\min_{a \in \mathcal{A}} \langle a, \nabla f(x_k) \rangle.$$

That is, we seek the atom that lines up best with the negative gradient direction $-\nabla f(x_k)$.

Generating Atoms

We can think of each step as the "addition of a new atom to the basis." Note that x_k is expressed in terms of $\{\bar{a}_0, \bar{a}_1, \ldots, \bar{a}_k\}$.

If few iterations are needed to find a solution of acceptable accuracy, then we have an approximate solution that's represented in terms of few atoms, that is, sparse or compactly represented.

For many atomic sets $\mathcal{A}$ of interest, the new atom can be found cheaply.

Example: For the constraint $\|x\|_1 \le \tau$, the atoms are $\{\pm e_i \, : \, i = 1, 2, \ldots, n\}$. if i_k is the index at which $|[\nabla f(x_k)]_i|$ attains its maximum, we have

$$\bar{a}_k = -\text{sign}([\nabla f(x_k)]_{i_k})\, e_{i_k}$$

Example: For the constraint $\|x\|_\infty \le \tau$, the atoms are the 2^n vectors with entries ± 1. We have

$$[\bar{a}_k]_i = -\text{sign}[\nabla f(x_k)]_i, \quad i = 1, 2, \ldots, n.$$

V. Augmented Lagrangian Methods

- Consider a linearly constrained problem,

$$\min\ f(x)\ \text{ s.t. }\ Ax = b.$$

  where f is a proper, lower semi-continuous, convex function.

- The augmented Lagrangian is (with $\rho_k > 0$)

$$\mathcal{L}(x, \lambda; \rho) := \underbrace{f(x) + \lambda^T (Ax - b)}_{\text{Lagrangian}} + \underbrace{\frac{\rho_k}{2} \|Ax - b\|_2^2}_{\text{“augmentation”}}$$

V. Augmented Lagrangian Methods

- Consider a linearly constrained problem,

$$\min\ f(x)\ \text{ s.t. }\ Ax = b.$$

where f is a proper, lower semi-continuous, convex function.

- The augmented Lagrangian is (with $\rho_k > 0$)

$$\mathcal{L}(x, \lambda; \rho) := \underbrace{f(x) + \lambda^T(Ax - b)}_{\text{Lagrangian}} + \underbrace{\frac{\rho_k}{2}\|Ax - b\|_2^2}_{\text{“augmentation”}}$$

- Basic augmented Lagrangian (a.k.a. method of multipliers) is

$$x_k = \arg\min_x \mathcal{L}(x, \lambda_{k-1}; \rho_k);$$
$$\lambda_k = \lambda_{k-1} + \rho(Ax_k - b);$$

(possibly increase ρ_k.

(Hestenes, 1969; Powell, 1969)

A Favorite Derivation

...more or less rigorous for convex f.

- Write the problem as

$$\min_{x} \max_{\lambda} f(x) + \lambda^T (Ax - b).$$

  Obviously, the max w.r.t. λ will be $+\infty$, unless $Ax = b$, so this is equivalent to the original problem.

A Favorite Derivation

...more or less rigorous for convex f.

- Write the problem as

$$\min_x \max_\lambda f(x) + \lambda^T(Ax - b).$$

  Obviously, the max w.r.t. λ will be $+\infty$, unless $Ax = b$, so this is equivalent to the original problem.

- This equivalence is not very useful, computationally: the $\max_\lambda$ function is highly nonsmooth w.r.t. x. Smooth it by adding a proximal term, penalizing deviations from a prior estimate $\bar{\lambda}$:

$$\min_x \left\{ \max_\lambda f(x) + \lambda^T(Ax - b) - \frac{1}{2\rho}\|\lambda - \bar{\lambda}\|^2 \right\}.$$

A Favorite Derivation

...more or less rigorous for convex f.

- Write the problem as

$$\min_x \max_\lambda f(x) + \lambda^T(Ax - b).$$

  Obviously, the max w.r.t. λ will be $+\infty$, unless $Ax = b$, so this is equivalent to the original problem.

- This equivalence is not very useful, computationally: the $\max_\lambda$ function is highly nonsmooth w.r.t. x. Smooth it by adding a proximal term, penalizing deviations from a prior estimate $\bar{\lambda}$:

$$\min_x \left\{ \max_\lambda f(x) + \lambda^T(Ax - b) - \frac{1}{2\rho}\|\lambda - \bar{\lambda}\|^2 \right\}.$$

- Maximization w.r.t. λ is now trivial (a concave quadratic), yielding

$$\lambda = \bar{\lambda} + \rho(Ax - b).$$

A Favorite Derivation (Cont.)

Inserting $\lambda = \bar{\lambda} + \rho(Ax - b)$ leads to

$$\min_x f(x) + \bar{\lambda}^T(Ax - b) + \frac{\rho}{2}\|Ax - b\|^2 = \mathcal{L}(x, \bar{\lambda}; \rho).$$

Hence can view the augmented Lagrangian process as:

- ✓ $\min_x \mathcal{L}(x, \bar{\lambda}; \rho)$ to get new x;
- ✓ Shift the "prior" on λ by updating to the latest max: $\bar{\lambda} + \rho(Ax - b)$;
- ✓ Increase ρ_k if not happy with improvement in feasibility;
- ✓ repeat until convergence.

Add subscripts, and we recover the augmented Lagrangian algorithm of the first slide!

Inequality Constraints

- The same derivation can be used for inequality constraints:

$$\min \, f(x) \;\; \text{s.t.} \;\; Ax \geq b.$$

- Apply the same reasoning to the constrained min-max formulation:

$$\min_{x} \max_{\lambda \geq 0} f(x) - \lambda^T (Ax - b).$$

Inequality Constraints

- The same derivation can be used for inequality constraints:

$$\min f(x) \text{ s.t. } Ax \geq b.$$

- Apply the same reasoning to the constrained min-max formulation:

$$\min_{x} \max_{\lambda \geq 0} f(x) - \lambda^T (Ax - b).$$

- After the prox-term is added, can find the minimizing λ in closed form (as for prox-operators). Leads to update formula:

$$\lambda \leftarrow \max\left(\bar{\lambda} + \rho(Ax - b), 0\right).$$

- This derivation extends immediately to nonlinear constraints $c(x) = 0$ or $c(x) \geq 0$.

“Explicit” Constraints, Inequality Constraints

- There may be other constraints on x (such as $x \in \Omega$) that we prefer to handle explicitly in the subproblem.

- For the formulation $\min_x f(x)$, s.t. $Ax = b$, $x \in \Omega$, the $\min_x$ step can enforce $x \in \Omega$ explicitly:

$$x_k = \arg\min_{x \in \Omega} \mathcal{L}(x, \lambda_{k-1}; \rho);$$
$$\lambda_k = \lambda_{k-1} + \rho(Ax_k - b);$$

“Explicit” Constraints, Inequality Constraints

- There may be other constraints on x (such as $x \in \Omega$) that we prefer to handle explicitly in the subproblem.

- For the formulation $\min\limits_x \; f(x), \;\text{ s.t. }\; Ax = b, \;\; x \in \Omega,$ the $\min_x$ step can enforce $x \in \Omega$ explicitly:

$$x_k = \arg\min_{x \in \Omega} \mathcal{L}(x, \lambda_{k-1}; \rho);$$
$$\lambda_k = \lambda_{k-1} + \rho(Ax_k - b);$$

- This gives an alternative way to handle inequality constraints: introduce slacks s, and enforce them explicitly. That is, replace

$$\min_x \; f(x) \;\text{ s.t. }\; c(x) \geq 0,$$

by

$$\min_{x,s} \; f(x) \;\text{ s.t. }\; c(x) = s, \;\; s \geq 0,$$

and enforce $s \geq 0$ explicitly in the subproblems.

Quick History of Augmented Lagrangian

- Dates from at least 1969: Hestenes, Powell.
- Developments in 1970s, early 1980s by Rockafellar, Bertsekas, and others.
- Lancelot code for nonlinear programming (Conn et al., 1992).
- Lost favor somewhat as an approach for general nonlinear programming during the next 15 years.
- Recent revival in the context of sparse optimization and its many applications, in conjunction with splitting / coordinate descent.

Alternating Direction Method of Multipliers (ADMM)

- Consider now problems with a separable objective of the form

$$\min_{(x,z)} f(x) + h(z) \quad \text{s.t.} \quad Ax + Bz = c,$$

for which the augmented Lagrangian is

$$\mathcal{L}(x, z, \lambda; \rho) := f(x) + h(z) + \lambda^T(Ax + Bz - c) + \frac{\rho}{2}\|Ax - Bz - c\|_2^2.$$

Alternating Direction Method of Multipliers (ADMM)

- Consider now problems with a separable objective of the form

  $$\min_{(x,z)} f(x) + h(z) \quad \text{s.t.} \quad Ax + Bz = c,$$

  for which the augmented Lagrangian is

  $$\mathcal{L}(x,z,\lambda;\rho) := f(x) + h(z) + \lambda^T(Ax + Bz - c) + \frac{\rho}{2}\|Ax - Bz - c\|_2^2.$$

- Standard AL would minimize $\mathcal{L}(x,z,\lambda;\rho)$ w.r.t. (x,z) jointly. However, these are coupled in the quadratic term, separability is lost

Alternating Direction Method of Multipliers (ADMM)

- Consider now problems with a separable objective of the form

$$\min_{(x,z)} f(x) + h(z) \quad \text{s.t.} \quad Ax + Bz = c,$$

for which the augmented Lagrangian is

$$\mathcal{L}(x, z, \lambda; \rho) := f(x) + h(z) + \lambda^T(Ax + Bz - c) + \frac{\rho}{2}\|Ax - Bz - c\|_2^2.$$

- Standard AL would minimize $\mathcal{L}(x, z, \lambda; \rho)$ w.r.t. (x, z) jointly. However, these are coupled in the quadratic term, separability is lost
- In ADMM, minimize over x and z separately and sequentially:

$$\begin{aligned} x_k &= \arg\min_x \mathcal{L}(x, z_{k-1}, \lambda_{k-1}; \rho_k); \\ z_k &= \arg\min_z \mathcal{L}(x_k, z, \lambda_{k-1}; \rho_k); \\ \lambda_k &= \lambda_{k-1} + \rho_k(Ax_k + Bz_k - c). \end{aligned}$$

ADMM

Main features of ADMM:

- Does one cycle of block-coordinate descent in (x, z).
- The minimizations over x and z add only a quadratic term to f and h, respectively. Usually does not alter the cost much.
- Can perform the (x, z) minimizations inexactly.
- Can add explicit (separated) constraints: $x \in \Omega_x$, $z \in \Omega_z$.
- Many (many!) recent applications to compressed sensing, image processing, matrix completion, sparse principal components analysis....

ADMM has a rich collection of antecendents, dating even to the 1950s (operator splitting).

For an comprehensive recent survey, including a diverse collection of machine learning applications, see Boyd et al. (2011).

ADMM: A Simpler Form

- Often, a simpler version is enough: $\min_{(x,z)} f(x) + h(z)$ s.t. $Ax = z$, equivalent to $\min_x f(x) + h(Ax)$, often the one of interest.

ADMM: A Simpler Form

- Often, a simpler version is enough: $\min_{(x,z)} f(x) + h(z)$ s.t. $Ax = z$, equivalent to $\min_{x} f(x) + h(Ax)$, often the one of interest.

- In this case, the ADMM can be written as

$$
\begin{aligned}
x_k &= \arg\min_x \; f(x) + \tfrac{\rho}{2}\|A\,x - z_{k-1} - d_{k-1}\|_2^2 \\
z_k &= \arg\min_z \; h(z) + \tfrac{\rho}{2}\|A\,x_{k-1} - z - d_{k-1}\|_2^2 \\
d_k &= d_{k-1} - (A\,x_k - z_k)
\end{aligned}
$$

the so-called "scaled version" (Boyd et al., 2011).

ADMM: A Simpler Form

- Often, a simpler version is enough: $\min_{(x,z)} f(x) + h(z)$ s.t. $Ax = z$, equivalent to $\min_{x} f(x) + h(Ax)$, often the one of interest.
- In this case, the ADMM can be written as

$$x_k = \arg\min_x \; f(x) + \tfrac{\rho}{2}\|A\,x - z_{k-1} - d_{k-1}\|_2^2$$
$$z_k = \arg\min_z \; h(z) + \tfrac{\rho}{2}\|A\,x_{k-1} - z - d_{k-1}\|_2^2$$
$$d_k = d_{k-1} - (A\,x_k - z_k)$$

the so-called "scaled version" (Boyd et al., 2011).

- Updating z_k is a proximity computation: $z_k = \operatorname{prox}_{h/\rho}\big(A\,x_{k-1} - d_{k-1}\big)$

ADMM: A Simpler Form

- Often, a simpler version is enough: $\min_{(x,z)} f(x) + h(z)$ s.t. $Ax = z$, equivalent to $\min_x f(x) + h(Ax)$, often the one of interest.

- In this case, the ADMM can be written as

$$
\begin{aligned}
x_k &= \arg\min_x \; f(x) + \tfrac{\rho}{2}\|A\,x - z_{k-1} - d_{k-1}\|_2^2 \\
z_k &= \arg\min_z \; h(z) + \tfrac{\rho}{2}\|A\,x_{k-1} - z - d_{k-1}\|_2^2 \\
d_k &= d_{k-1} - (A\,x_k - z_k)
\end{aligned}
$$

  the so-called "scaled version" (Boyd et al., 2011).

- Updating z_k is a proximity computation: $z_k = \text{prox}_{h/\rho}\big(A\,x_{k-1} - d_{k-1}\big)$

- Updating x_k may be hard: if f is quadratic, involves matrix inversion; if f is not quadratic, may be as hard as the original problem.

ADMM: Convergence

- Consider the problem $\min_x f(x) + h(Ax)$, where f and h are lower semi-continuous, proper, convex functions and A has full column rank.

ADMM: Convergence

- Consider the problem $\min_{x} f(x) + h(Ax)$, where f and h are lower semi-continuous, proper, convex functions and A has full column rank.

- The ADMM algorithm presented in the previous slide converges (for any $\rho > 0$) to a solution x^*, if one exists, otherwise it diverges.

  This is a cornerstone result by Eckstein and Bertsekas (1992).

ADMM: Convergence

- Consider the problem $\min_x f(x) + h(Ax)$, where f and h are lower semi-continuous, proper, convex functions and A has full column rank.

- The ADMM algorithm presented in the previous slide converges (for any $\rho > 0$) to a solution x^*, if one exists, otherwise it diverges.

  This is a cornerstone result by Eckstein and Bertsekas (1992).

- As in IST/FBS/PGA, convergence is still guaranteed with inexactly solved subproblems, as long as the errors are absolutely summable.

ADMM: Convergence

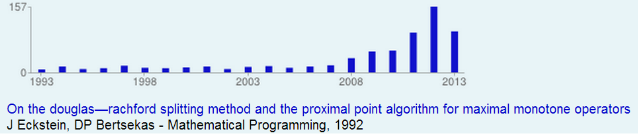
- Consider the problem $\min_{x} f(x) + h(Ax)$, where f and h are lower semi-continuous, proper, convex functions and A has full column rank.

- The ADMM algorithm presented in the previous slide converges (for any $\rho > 0$) to a solution x^*, if one exists, otherwise it diverges.

  This is a cornerstone result by Eckstein and Bertsekas (1992).

- As in IST/FBS/PGA, convergence is still guaranteed with inexactly solved subproblems, as long as the errors are absolutely summable.

- The recent burst of interest in ADMM is evident from the citation record of Eckstein and Bertsekas (1992).



On the douglas—rachford splitting method and the proximal point algorithm for maximal monotone operators
J Eckstein, DP Bertsekas - Mathematical Programming, 1992

Special Case: ℓ_2-ℓ_1

Standard problem: $\min_x \frac{1}{2}\|Ax-b\|_2^2 + \|x\|_1$. ADMM becomes

$$\begin{aligned}
x_k &= \arg\min_x \; \|x\|_1 + \tfrac{\rho}{2}\|A\,x - z_{k-1} - d_{k-1}\|_2^2 \\
z_k &= \arg\min_z \; h(z) + \tfrac{\rho}{2}\|A\,x_{k-1} - z - d_{k-1}\|_2^2 \\
d_k &= d_{k-1} - (A\,x_k - z_k)
\end{aligned}$$

Subproblems are

$$\begin{aligned}
x_k &:= (A^TA + \rho_k I)^{-1}(A^Tb + \rho_k z_{k-1} - \lambda_k), \\
z_k &:= \min_z \; \tau\|z\|_1 + (\lambda_k)^T(x_k - z) + \frac{\rho_k}{2}\|z - x_k\|_2^2 \\
&\;= \mathrm{prox}_{\tau/\rho_k}(x_k + \lambda_k/\rho_k) \\
\lambda_{k+1} &:= \lambda_k + \rho_k(x_k - z_k).
\end{aligned}$$

Solving for x_k is the most complicated part of the calculation. If the least-squares part is underdetermined (A is $m \times n$ with $n > m$), can make use of the Sherman-Morrison-Woodbury formula:

Moreover, in some compressed sensing applications, we have $AA^T = I$. In this case, x_k can be recovered at the cost of two matrix-vector multiplications involving A.

Otherwise, can solve for x_k inexactly, using a few steps of an iterative method.

The YALL1 code solves this problem, and other problems with more general regularizers (e.g. groups).

ADMM for Sparse Inverse Covariance

$$\max_{X \succ 0} \log\det(X) - \langle X, S \rangle - \tau \|X\|_1,$$

Reformulate as

$$\max_{X \succ 0} \log\det(X) - \langle X, S \rangle - \tau \|Z\|_1 \quad \text{s.t. } X - Z = 0.$$

Subproblems are:

$$\begin{aligned} X_k &:= \arg\max_X \log\det(X) - \langle X, S \rangle - \langle U_{k-1}, X - Z_{k-1} \rangle \\ &\quad - \frac{\rho_k}{2}\|X - Z_{k-1}\|_F^2 \\ &:= \arg\max_X \log\det(X) - \langle X, S \rangle - \frac{\rho_k}{2}\|X - Z_{k-1} + U_k/\rho_k\|_F^2 \\ Z_k &:= \text{prox}_{\tau/\rho_k \|\cdot\|_1}(X_k + U_k); \\ U_{k+1} &:= U_k + \rho_k(X_k - Z_k). \end{aligned}$$

Solving for X

Get optimality condition for the X subproblem by using $\nabla_X \log\det(X) = X^{-1}$, when X is s.p.d. Thus,

$$X^{-1} - S - \rho_k(X - Z_{k-1} + U_k/\rho_k) = 0,$$

which is equivalent to

$$X^{-1} - \rho_k X - (S - \rho_k Z_{k-1} + U_k) = 0.$$

Form eigendecomposition

$$(S - \rho_k Z_{k-1} + U_k) = Q\Lambda Q^T,$$

where Q is $n \times n$ orthogonal and Λ is diagonal with elements λ_i. Seek X with the form $Q\tilde{\Lambda}Q^T$, where $\tilde{\Lambda}$ has diagonals $\tilde{\lambda}_i$. Must have

$$\frac{1}{\tilde{\lambda}_i} - \rho_k\tilde{\lambda}_i - \lambda_i = 0, \quad i = 1, 2, \ldots, n.$$

Take positive roots: $\tilde{\lambda}_i = [\lambda_i + \sqrt{\lambda_i^2 + 4\rho_k}]/(2\rho_k)$, $i = 1, 2, \ldots, n$.

VI. Coordinate Descent

Consider first $\min f(x)$ with $f : \mathbb{R}^n \to \mathbb{R}$.

Iteration j of basic coordinate descent:

- Choose index $i_j \in \{1, 2, \dots, n\}$;
- Fix all components $i \neq i_j$, change x_{i_j} in a way that (hopefully) reduces f.

Variants for the reduced step:

- take a reduced gradient step: $-\nabla_{i_j} f(x)$;
- do a more rigorous search in the subspace defined by i_j;
- actually minimize f in the i_j component.

Many extensions of this basic idea.

An old approach, revived recently because of useful applications in machine learning, and interesting possibilities as stochastic and parallel algorithms.

Block Coordinate Descent

At iteration j, choose a subset $\mathcal{G}_j \subset \{1, 2, \ldots, n\}$ and allow only the components in $\mathcal{G}_j$ to change. Fix the components x_i for $i \notin \mathcal{G}_j$.

Again, the step could be a reduced gradient step along $-\nabla_{\mathcal{G}_j} f(x)$, or a more elaborate search.

There are many different heuristics for choosing $\mathcal{G}_j$, often arising naturally from the application.

Constraints and regularizers complicate things! Make block partition consistent with separablity of constraints / regularizers.

Deterministic and Stochastic CD

Step

$$x_{j+1,i_j} = x_{j,i_j} - \alpha_j [\nabla f(x_j)]_{i_j}.$$

- Deterministic CD: choose i_k in some fixed order e.g. cyclic;
- Stochastic CD: choose i_k at random from $\{1, 2, \ldots, n\}$.

CD is a reasonable choice when it's cheap to evaluate individual elements of $\nabla f(x)$ (e.g. at $1/n$ of the cost of a full gradient, say).

Convergence: Deterministic (Luo and Tseng, 1992; Tseng, 2001), linear rate (Beck and Tetruashvili, 2013). Stochastic, linear rate: (Nesterov, 2012).

Coordinate Descent in Dual SVM

Coordinate descent has long been popular for solving the dual (QP) formulation of support vector machines.

$$\min_{\alpha \in \mathbb{R}^N} \frac{1}{2}\alpha^T K\alpha - \mathbf{1}^T\alpha \ \text{ s.t. } \ 0 \le \alpha \le C\mathbf{1}, \ \ y^T\alpha = 0.$$

SMO: Each $\mathcal{G}_k$ has two components. (Thus can maintain feasibility with respect to the single linear constraint $y^T\alpha = 0$.)

LIBSVM: SMO approach (still $|\mathcal{G}_k| = 2$), with different heuristic.

LASVM: Again $|\mathcal{G}_k| = 2$, with focus on online setting.

SVM-light: Small $|\mathcal{G}_k|$ (default 10).

GPDT: Larger $|\mathcal{G}_k|$ (default 400) with gradient projection solver as the subproblem solver.

Asynchronous Stochastic Coordinate Descent (ASCD)

Consider $\min f(x)$, where $f : \mathbb{R}^n \to \mathbb{R}^n$ is smooth and convex.

Each processor (independently, without synchronization) does:

1. Choose $i \in \{1, 2, \dots, n\}$ uniformly at random;
2. Read x and evaluate $g = [\nabla f(x)]_i$;
3. Update $x_i \leftarrow x_i - \frac{\gamma}{L_{\max}} g$;

Here γ is a steplength (more below) and $L_{\max}$ is a bound on the diagonals of the Hessian $\nabla^2 f(x)$.

Assume that not more than τ cycles pass between when x is read (step 2) and updated (step 3).

How to choose γ to achieve good convergence?

Constants and "Diagonalicity"

Several constants are critical to the analysis.

- τ: maximum delay;
- $L_{\max}$: maximum diagonal of $\nabla^2 f(x)$;
- L_{res}: maximum row norm of Hessian;
- μ: lower bound on eigenvalues of $\nabla^2 f(x)$ (assumed positive).

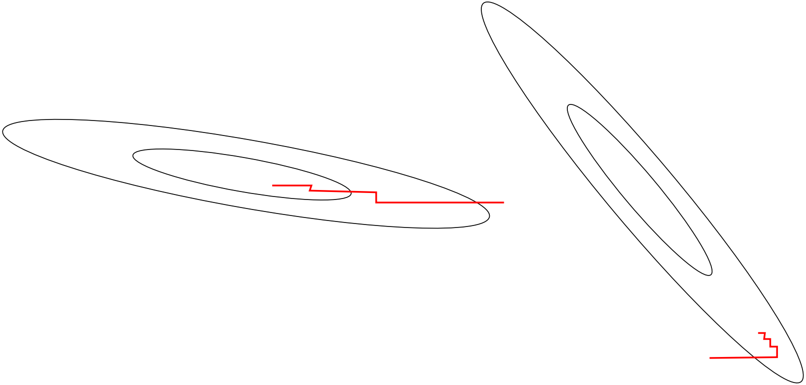
The ratio $L_{\text{res}}/L_{\max}$ is particularly important — it measures the degree of diagonal dominance in the Hessian $\nabla^2 f(x)$ (Diagonalicity).

By convexity, we have

$$1 \le \frac{L_{\text{res}}}{L_{\max}} \le \sqrt{n}.$$

Closer to 1 if Hessian is nearly diagonally dominant (eigenvectors close to principal coordinate axes). Smaller is better for parallelism.

Diagonalicity Illustrated



Left figure is better. It can tolerate a higher delay parameter τ and thus more cores working asynchronously.

How to choose γ?

Choose some $\rho > 1$ and pick γ small enough to ensure that

$$\rho^{-1} \leq \frac{\mathbb{E}(\|\nabla f(x_{j+1})\|^2)}{\mathbb{E}(\|\nabla f(x_j)\|^2)} \leq \rho.$$

Not too much change in gradient over each iteration, so not too much price to pay for using old information, in the asynchronous setting.

Choose γ small enough to satisfy this property but large enough to get a linear rate.

Assuming that

$$\tau + 1 \leq \frac{\sqrt{n} L_{\text{max}}}{2eL_{\text{res}}},$$

and choosing $\rho = 1 + \frac{2eL_{\text{res}}}{\sqrt{n}L_{\text{max}}}$, we can take $\gamma = 1$. Then have

$$\mathbb{E}(f(x_j) - f^*) \leq \left(1 - \frac{\mu}{2nL_{\text{max}}}\right)^j (f(x_0) - f^*).$$

(Liu and Wright, 2013)

ASCD Discussion

Linear rate is close to the rate attained by short-step steepest descent.

Bound on τ is a measure of potential parallelization. When ratio $L_{\text{res}}/L_{\max}$ is favorable, get $\tau = O(\sqrt{n})$. Thus, expect near-linear speedup on to $O(\sqrt{n})$ cores running asynchronously in parallel.

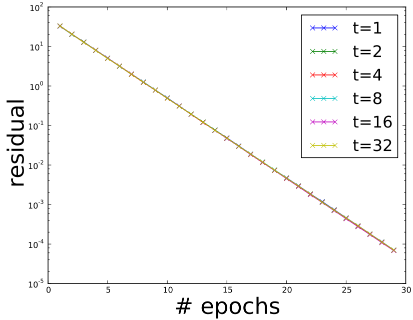
Can extend algorithm and analysis to

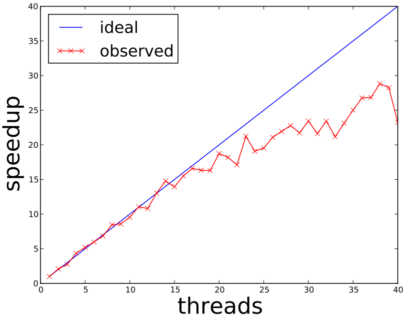
- "Essentially strongly convex" and "weakly convex" cases;
- Separable constraints. Have $\tau = O(n^{1/4})$ — less potential parallelism.

Implemented on 4-socket, 40-core Intel Xeon

$$\min_{x} \quad \|Ax - b\|^2 + 0.5\|x\|^2$$

where $A \in \mathbb{R}^{m\times n}$ is a Gaussian random matrix ($m = 6000$, $n = 20000$, columns are normalized to 1). $L_{\text{res}}/L_{\text{max}} \approx 2.2$. Choose $\gamma = 1$.





(Thanks: C. Ré, V. Bittorf, S. Sridhar)

Conclusions

- We've seen a sample of optimization tools that are relevant to learning, as currently practiced.
- Many holes in this discussion! Check the "matching" slides at the start for further leads.
- Besides being a source of algorithmic tools, optimization offers some interesting perspectives on formulation.
- Optimizers get a lot of brownie points for collaborating with learning and data analysis researchers, so don't hesitate to talk to us!

References I

Akaike, H. (1959). On a successive transformation of probability distribution and its application to the analysis fo the optimum gradient method. *Annals of the Institute of Statistics and Mathematics of Tokyo*, 11:1–17.

Amaldi, E. and Kann, V. (1998). On the approximation of minimizing non zero variables or unsatisfied relations in linear systems. *Theoretical Computer Science*, 209:237–260.

Bach, F., Jenatton, R., Mairal, J., and Obozinski, G. (2012). Structured sparsity through convex optimization. *Statistical Science*, 27:450–468.

Barzilai, J. and Borwein, J. (1988). Two point step size gradient methods. *IMA Journal of Numerical Analysis*, 8:141–148.

Beck, A. and Teboulle, M. (2009a). A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202.

Beck, A. and Teboulle, M. (2009b). A fast iterative shrinkage-threshold algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202.

Beck, A. and Tetruashvili, L. (2013). On the convergence of block coordinate descent methods. Technical report, Technion-Israel Institute of Technology.

Bottou, L. (2012). `leon.bottou.org/research/stochastic`.

Bottou, L. and LeCun, Y. (2004). Large-scale online learning. In *Advances in Neural Information Processing Systems*.

References II

Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122.

Byrd, R. H., Chin, G. M., Neveitt, W., and Nocedal, J. (2011). On the use of stochastic hessian information in unconstrained optimization. *SIAM Journal on Optimization*, 21:977–995.

Cai, J.-F., Candès, E., and Shen, Z. (2010a). A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4).

Cai, J.-F., Candès, E., and Shen, Z. (2010b). A singular value thresholding algorithm for matrix completion. *SIAM JOurnal on Optimization*, 20:1956–1982.

Candès, E. and Romberg, J. (2005). ℓ_1-MAGIC: Recovery of sparse signals via convex programming. Technical report, California Institute of Technology.

Candès, E., Romberg, J., and Tao, T. (2006a). Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52:489–509.

Candès, E., Romberg, J., and Tao, T. (2006b). Stable signal recovery from incomplete and inaccurate measurements. *Communications in Pure and Applied Mathematics*, 59:1207–1223.

Caruana, R. (1997). Multitask learning. *Machine Learning*, 28(1):41–75.

Chan, T. F., Golub, G. H., and Mulet, P. (1999). A nonlinear primal-dual method for total variation based image restoration. *SIAM Journal of Scientific Computing*, 20:1964–1977.

References III

Chang, C.-C. and Lin, C. (2011). LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2.

Chen, S., Donoho, D., and Saunders, M. (1995). Atomic decomposition by basis pursuit. Technical report, Department of Statistics, Stanford University.

Combettes, P. and Pesquet, J.-C. (2011). Signal recovery by proximal forward-backward splitting. In *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pages 185–212. Springer.

Combettes, P. and Wajs, V. (2006). Proximal splitting methods in signal processing. *Multiscale Modeling and Simulation*, 4:1168–1200.

Conn, A., Gould, N., and Toint, P. (1992). *LANCELOT: a Fortran package for large-scale nonlinear optimization (Release A)*. Springer Verlag, Heidelberg.

d'Aspremont, A., Banerjee, O., and El Ghaoui, L. (2008). First-order methods for sparse covariance selection. *SIAM Journal on Matrix Analysis and Applications*, 30:56–66.

Davis, G., Mallat, S., and Avellaneda, M. (1997). Greedy adaptive approximation. *Journal of Constructive Approximation*, 13:57–98.

Dekel, O., Gilad-Bachrach, R., Shamir, O., and Xiao, L. (2012). Optimal distributed online prediction using mini-batches. *Journal of Machine Learning Research*, 13:165–202.

Donoho, D. (2006). Compressed sensing. *IEEE Transactions on Information Theory*, 52:1289–1306.

References IV

Duchi, J., Agarwal, A., and Wainwright, M. J. (2010). Distributed dual averaging in networks. In *Advances in Neural Information Processing Systems*.

Duchi, J. and Singer, Y. (2009). Efficient learning using forward-backward splitting. In *Advances in Neural Information Processing Systems*. http://books.nips.cc.

Eckstein, J. and Bertsekas, D. (1992). On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical Programming*, 5:293–318.

Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. (2004). Least angle regression. *Annals of Statistics*, 32(2):407–499.

Ferris, M. C. and Munson, T. S. (2002). Interior-point methods for massive support vector machines. *SIAM Journal on Optimization*, 13(3):783–804.

Figueiredo, M. A. T. and Nowak, R. D. (2003). An EM algorithm for wavelet-based image restoration. *IEEE Transactions on Image Processing*, 12:906–916.

Fine, S. and Scheinberg, K. (2001). Efficient svm training using low-rank kernel representations. *Journal of Machine Learning Research*, 2:243–264.

Fountoulakis, K. and Gondzio, J. (2013). A second-order method for strongly convex ell_1-regularization problems. Technical Report ERGO-13-011, University of Edinburgh.

Fountoulakis, K., Gondzio, J., and Zhlobich, P. (2012). Matrix-free interior point method for compressed sensing problems. Technical Report, University of Edinburgh.

References V

Friedman, J., Hastie, T., and Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441.

Garnaev, A. and Gluskin, E. (1984). The widths of an Euclidean ball. *Doklady Akademii Nauk*, 277:1048–1052.

Gertz, E. M. and Wright, S. J. (2003). Object-oriented software for quadratic programming. *ACM Transations on Mathematical Software*, 29:58–81.

Goldfarb, D., Ma, S., and Scheinberg, K. (2012). Fast alternating linearization methods for minimizing the sum of two convex functions. *Mathematical Programming, Series A*. DOI 10.1007/s10107-012-0530-2.

Gondzio, J. and Fountoulakis, K. (2013). Second-order methods for l1-regularization. Talk at *Optimization and Big Data* Workshop, Edinburgh.

Hale, E., Yin, W., and Zhang, Y. (2008). Fixed-point continuation for l1-minimization: Methodology and convergence. *SIAM Journal on Optimization*, 19:1107–1130.

Haupt, J. and Nowak, R. (2006). Signal reconstruction from noisy random projections. *IEEE Transactions on Information Theory*, 52:4036–4048.

Hestenes, M. (1969). Multiplier and gradient methods. *Journal of Optimization Theory and Applications*, 4:303–320.

Hinton, G. E., Osindero, S., and Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554.

References VI

Jaggi, M. (2013). Revisiting frank-wolfe: Projection-free sparse convex optimization. In *Proceedings of ICML 2013*.

Joachims, T. (1999). Making large-scale support vector machine learning practical. In Schölkopf, B., Burges, C., and Smola, A., editors, *Advances in Kernel Methods - Support Vector Learning*, chapter 11, pages 169–184. MIT Press, Cambridge, MA.

Kashin, B. (1977). Diameters of certain finite-dimensional sets in classes of smooth functions. *Izvestiya Akademii Nauk. SSSR: Seriya Matematicheskaya*, 41:334–351.

Langford, J., Li, L., and Zhang, T. (2009). Sparse online learning via truncated gradient. *Journal of Machine Learning Research*, 10:777–801.

Le, Q. V., Ranzato, M., Monga, R., Devin, M., Chen, K., Corrado, G. S., Dean, J., and Ng, A. Y. (2012). Building high-level features using large scale unsupervised learning. In *Proceedings of the Conference on Learning Theory*. arXiv:1112.6209v5.

Lee, J., Recht, B., Salakhutdinov, R., Srebro, N., and Tropp, J. (2010). Practical large-scale optimization for max-norm regularization. In *Advances in Neural Information Processing Systems*. NIPS.

Lee, S. and Wright, S. J. (2012). Manifold identification in dual averaging methods for regularized stochastic online learning. *Journal of Machine Learning Research*, 13:1705–1744.

Lee, S. I., Lee, H., Abbeel, P., and Ng, A. Y. (2006). Efficient ℓ_1 regularized logistic regerssion. In *Proceedings of the National Conference of the American Association for Artificial Intelligence*.

References VII

Lin, C., Weng, R. C., and Keerthi, S. S. (2008). Trust-region newton method for logistic regression. *Journal of Machine Learning Research*, 9:627–650.

Liu, D. C. and Nocedal, J. (1989). On the limited-memory BFGS method for large scale optimization. *Mathematical Programming, Series A*, 45:503–528.

Liu, J. and Wright, S. J. (2013). An asynchronous parallel stochastic coordinate descent method. In preparation.

Luo, Z. Q. and Tseng, P. (1992). On the convergence of the coordinate descent method for convex differentiable minimization. *Journal of Optimization Theory and Applications*, 72(1):7–35.

Ma, S., Goldfarb, D., and Chen, L. (2011). Fixed point and Bregman iterative methods for matrix rank minimization. *Mathematical Programming (Series A)*, 128:321–353.

Martens, J. (2010). Deep learning via Hessian-free optimization. In *Proceedings of the 27th International Conference on Machine Learning*.

Martins, A. F. T., Smith, N. A., Aguiar, P. M. Q., and Figueiredo, M. A. T. (2011). Structured Sparsity in Structured Prediction. In *Proc. of Empirical Methods for Natural Language Processing*.

Meier, L., van de Geer, S., and Buhlmann, P. (2008). The group lasso for logistic regression. *Journal of the Royal Statistical Society B*, 70(1):53–71.

Moreau, J. (1962). Fonctions convexes duales et points proximaux dans un espace hilbertien. *CR Acad. Sci. Paris Sér. A Math*, 255:2897–2899.

References VIII

Muthukrishnan, S. (2005). *Data Streams: Algorithms and Applications*. Now Publishers, Boston, MA.

Nemirovski, A., Juditsky, A., Lan, G., and Shapiro, A. (2009). Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, 19(4):1574–1609.

Nesterov, Y. (1983). A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Soviet Math. Doklady*, 27:372–376.

Nesterov, Y. (2012). Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22:341–362.

Niu, F., Recht, B., Ré, C., and Wright, S. J. (2011). HOGWILD!: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in Neural Information Processing Systems*.

Nocedal, J. and Wright, S. J. (2006). *Numerical Optimization*. Springer, New York.

Obozinski, G., Taskar, B., and Jordan, M. (2010). Joint covariate selection and joint subspace selection for multiple classification problems. *Statistics and Computing*, 20(2):231–252.

Platt, J. C. (1999). Fast training of support vector machines using sequential minimal optimization. In Schölkopf, B., Burges, C. J. C., and Smola, A. J., editors, *Advances in Kernel Methods — Support Vector Learning*, pages 185–208, Cambridge, MA. MIT Press.

Powell, M. (1969). A method for nonlinear constraints in minimization problems. In Fletcher, R., editor, *Optimization*, pages 283–298. Academic Press, New York.

References IX

Recht, B., Fazel, M., and Parrilo, P. (2010). Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*, 52:471–501.

Richtarik, P. (2012). HOGWILD! and other hybrid beasts. Notes.

Scheinberg, K. and Ma, S. (2012). Optimization methods for sparse inverse covariance selection. In Sra, S., Nowozin, S., and Wright, S. J., editors, *Optimization for Machine Learning*, Neural Information Processing Series. MIT Press.

Schmidt, M. and Murphy, K. (2010). Convex structure learning in log-linear models: Beyond pairwise potentials. In *Proc. of AISTATS*.

Shalev-Shwartz, S., Singer, Y., and Srebro, N. (2007). Pegasos: Primal Estimated sub-GrAdient SOlver for SVM. In *Proceedings of the 24th International Confernece on Machine Learning*.

Shevade, S. K. and Keerthi, S. S. (2003). A simple and efficient algorithm for gene selection using sparse logistic regression. *Bioinformatics*, 19(17):2246–2253.

Shi, W., Wahba, G., Wright, S. J., Lee, K., Klein, R., and Klein, B. (2008). LASSO-Patternsearch algorithm with application to opthalmology data. *Statistics and its Interface*, 1:137–153.

Srebro, N. and Tewari, A. (2010). Stochastic optimization for machine learning. Tutorial, ICML 2010, Haifa, Israel.

Stojnic, M., Parvaresh, F., and Hassibi, B. (2009). On the reconstruction of block-sparse signals with an optimal number of measurements. *Signal Processing, IEEE Transactions on*, 57(8):3075–3085.

References X

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society B.*, pages 267–288.

Tillmann, A. and Pfetsch, M. (2012). The computational complexity of RIP, NSP, and related concepts in compressed sensing. Technical report, arXiv/1205.2081.

Tseng, P. (2001). Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109(3):475–494.

Turlach, B., Venables, W. N., and Wright, S. J. (2005). Simultaneous variable selection. *Technometrics*, 47(3):349–363.

Wen, Z., Yin, W., Goldfarb, D., and Zhang, Y. (2010). A fast algorithms for sparse reconstruction based on shrinkage, subspace optimization, and continuation. *SIAM Journal on Scientific Computing*, 32(4):1832–1857.

Wen, Z., Yin, W., and Zhang, Y. (2012). Solving a low-rank factorization model for matrix completion by a nonlinear successive over-relaxation algorithm. *Mathematical Programming Computation*, 4:333–361.

Wright, S., Nowak, R., and Figueiredo, M. (2009a). Sparse reconstruction by separable approximation. *IEEE Transactions on Signal Processing*, 57:2479–2493.

Wright, S. J. (1997). *Primal-Dual Interior-Point Methods*. SIAM, Philadelphia, PA.

Wright, S. J., Nowak, R. D., and Figueiredo, M. A. T. (2009b). Sparse reconstruction by separable approximation. *IEEE Transactions on Signal Processing*, 57:2479–2493.

References XI

Xiao, L. (2010). Dual averaging methods for regularized stochastic learning and online optimization. *Journal of Machine Learning Research*, 11:2543–2596.

Xiao, L. and Zhang, T. (2012). A proximal-gradient homotopy method for the sparse least-squares problem. *SIAM Journal on Optimization*. (to appear; available at http://arxiv.org/abs/1203.3002).

Yin, W., Osher, S., Goldfarb, D., and Darbon, J. (2008). Bregman iterative algorithms for ℓ_1-minimization with applications to compressed sensing. *SIAM Journal on Imaging Sciences*, 1(1):143–168.

Yin, W. and Zhang, Y. (2008). Extracting salient features from less data via ℓ_1-minimization authors. *SIAG/OPT Views-and-News*, 19:11–19.

Zhang, Y., Yang, J., and Yin, W. (2010). User's guide for YALL1: Your algorithms for L1 optimization. CAAM Technical Report TR09-17, CAAM Department, Rice University.

Zhao, P., Rocha, G., and Yu, B. (2009). Grouped and hierarchical model selection through composite absolute penalties. *Annals of Statistics*, 37(6A):3468–3497.

Zhu, M., Wright, S. J., and Chan, T. F. (2010). Duality-based algorithms for total variation image restoration. *Computational Optimization and Applications*, 47(3):377–400. DOI: 10.1007/s10589-008-9225-2.

Zinkevich, M. (2003). Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning*, pages 928–936.