LADM with Parallel Splitting and Adaptive Penalty (LADMPSAP)

- Model problem:

$$\min_{\mathbf{x}_1,\cdots,\mathbf{x}_n} \sum_{i=1}^{n} f_i(\mathbf{x}_i), \quad s.t. \quad \sum_{i=1}^{n} \mathcal{A}_i(\mathbf{x}_i) = \mathbf{b}.$$

$$\min_{\mathbf{X}} \|\mathbf{X}\|_* + \frac{1}{2\mu}\|\mathbf{b} - \mathcal{P}\ (\mathbf{X})\|^2, \quad s.t. \quad \mathbf{X} \geq 0,$$

$$\Downarrow$$

$$\min_{\mathbf{X},\mathbf{e}} \|\mathbf{X}\|_* + \frac{1}{2\mu}\|\mathbf{e}\|^2, \quad s.t. \quad \mathbf{b} = \mathcal{P}\ (\mathbf{X}) + \mathbf{e},\ \mathbf{X} \geq 0,$$

$$\Downarrow$$

$$\min_{\mathbf{X},\mathbf{Y},\mathbf{e}} \|\mathbf{X}\|_* + \frac{1}{2\mu}\|\mathbf{e}\|^2, \quad s.t. \quad \mathbf{b} = \mathcal{P}\ (\mathbf{Y}) + \mathbf{e},\ \mathbf{X} = \mathbf{Y},\ \mathbf{Y} \geq 0,$$

$$\Downarrow$$

$$\min_{\mathbf{X},\mathbf{Y},\mathbf{e}} \|\mathbf{X}\|_* + \frac{1}{2\mu}\|\mathbf{e}\|^2 + \chi_{\mathbf{Y}\geq 0}(\mathbf{Y}), \quad s.t. \quad \mathbf{b} = \mathcal{P}\ (\mathbf{Y}) + \mathbf{e},\ \mathbf{X} = \mathbf{Y}.$$

Lin et al., *Linearized Alternating Direction Method with Parallel Splitting and Adaptive Penalty for Separable Convex Programs in Machine Learning*, ML, 2015.

LADM with Parallel Splitting and Adaptive Penalty (LADMPSAP)

- Can we naively generalize two-block LADMAP for multi-block problems?

No!

Actually, the naive generalization of LADMAP may be divergent, e.g., when applied to the following problem with $n \geq 5$:

$$\min_{\mathbf{x}_1,\cdots,\mathbf{x}_n} \sum_{i=1}^{n} \|\mathbf{x}_i\|_1, \quad s.t. \quad \sum_{i=1}^{n} \mathbf{A}_i\mathbf{x}_i = \mathbf{b}.$$

Lin et al., *Linearized Alternating Direction Method with Parallel Splitting and Adaptive Penalty for Separable Convex Programs in Machine Learning*, ML, 2015.
C. Chen et al. *The Direct Extension of ADMM for Multi-block Convex Minimization Problems is Not Necessarily Convergent*. Preprint.

LADM with Parallel Splitting and Adaptive Penalty (LADMPSAP)

$$\mathbf{x}_i^{k+1} = \underset{\mathbf{x}_i}{\operatorname{argmin}} f_i(\mathbf{x}_i) + \frac{\eta_i \beta_k}{2} \left\| \mathbf{x}_i - \mathbf{x}_i^k + \mathcal{A}_i^* \left(\lambda^k + \beta_k \left(\sum_{j=1}^{n} \mathcal{A}_i(\mathbf{x}_j^k) - \mathbf{b} \right) \right) / (\eta_i \beta_k) \right\|^2,$$

$$i = 1, \cdots, n,$$

$$\sum_{j=1}^{i-1} \mathcal{A}_i(\mathbf{x}_j^{k+1}) + \sum_{j=i}^{n} \mathcal{A}_i(\mathbf{x}_j^k)$$

$$\lambda^{k+1} = \lambda^k + \beta_k \left(\sum_{i=1}^{n} \mathcal{A}_i(\mathbf{x}_i^{k+1}) - \mathbf{b} \right),$$

$$\beta_{k+1} = \min(\beta_{\max}, \rho \beta_k),$$

Parallel!

where

$$\rho = \begin{cases} \rho_0, & \text{if } \beta_k \max\left(\left\{ \sqrt{\eta_i} \left\| \mathbf{x}_i^{k+1} - \mathbf{x}_i^k \right\|, i = 1, \cdots, n \right\} \right) / \|\mathbf{b}\| < \varepsilon_2, \\ 1, & \text{otherwise,} \end{cases}$$

with $\rho_0 > 1$ being a constant and $0 < \varepsilon_2 \ll 1$ being a threshold.

Lin et al., *Linearized Alternating Direction Method with Parallel Splitting and Adaptive Penalty for Separable Convex Programs in Machine Learning*, ML, 2015.

LADM with Parallel Splitting and Adaptive Penalty (LADMPSAP)

Theorem: If $\{\beta_k\}$ is non-decreasing and upper bounded, $\eta_i > n\|\mathcal{A}_i\|^2$, $i = 1, \cdots, n$, then $\{(\{\mathbf{x}_i^k\}, \lambda^k)\}$ generated by LADMPSAP converges to a KKT point of the problem.

Remark: When $n = 2$, LADMPSAP is weaker than LADMAP:

$$\eta_i > 2\|A_i\|^2 \text{ vs. } \eta_i > \|A_i\|^2.$$

- Related work: He & Yuan, *Linearized Alternating Direction Method with Gaussian Back Substitution for Separable Convex Programming*, preprint.

Lin et al., *Linearized Alternating Direction Method with Parallel Splitting and Adaptive Penalty for Separable Convex Programs in Machine Learning*, ML, 2015.

LADM with Parallel Splitting and Adaptive Penalty (LADMPSAP)

- Model problem:

$$\min_{\mathbf{x}_1,\cdots,\mathbf{x}_n} \sum_{i=1}^{n} f_i(\mathbf{x}_i),\ s.t.\ \sum_{i=1}^{n} \mathcal{A}_i(\mathbf{x}_i) = \mathbf{b},\ \mathbf{x}_i \in X_i, i = 1,\cdots,n,$$

where $X_i \subseteq \mathbb{R}^{d_i}$ is a closed convex set.

$$\Downarrow$$

$$\min_{\mathbf{x}_1,\cdots,\mathbf{x}_{2n}} \sum_{i=1}^{n} f_i(\mathbf{x}_i) + \sum_{i=n+1}^{2n} \chi_{\mathbf{x}_i \in X_{i-n}}(\mathbf{x}_i),\ s.t.\ \sum_{i=1}^{n} \mathcal{A}_i(\mathbf{x}_i) = \mathbf{b}, \mathbf{x}_i = \mathbf{x}_{n+i}, i = 1,\cdots,n.$$

Theorem: If $\{\beta_k\}$ is non-decreasing and upper bounded, $\mathbf{x}_{n+1},\cdots,\mathbf{x}_{2n}$ are auxiliary variables, $\eta_i > n\|\mathcal{A}_i\|^2 + 2$, $\eta_{n+i} > 2$, $i = 1,\cdots,n$, then $\{(\{\mathbf{x}_i^k\}, \lambda^k)\}$ generated by LADMPSAP converges to a KKT point of the problem.

$$\eta_i > 2n(\|\mathcal{A}_i\|^2 + 1), \eta_{n+i} > 2n, i = 1,\cdots,n$$

Lin et al., *Linearized Alternating Direction Method with Parallel Splitting and Adaptive Penalty for Separable Convex Programs in Machine Learning*, ML, 2015.

Experiment

$$\min_{\mathbf{X}} \|\mathbf{X}\|_* + \frac{1}{2\mu}\|\mathbf{b} - \mathcal{P}\ (\mathbf{X})\|^2, \quad s.t. \quad \mathbf{X} \geq 0,$$

$$\Downarrow$$

$$\min_{\mathbf{X},\mathbf{Y},\mathbf{e}} \|\mathbf{X}\|_* + \frac{1}{2\mu}\|\mathbf{e}\|^2 + \chi_{\mathbf{Y}\geq 0}(\mathbf{Y}), \quad s.t. \quad \mathbf{b} = \mathcal{P}\ (\mathbf{Y}) + \mathbf{e},\ \mathbf{X} = \mathbf{Y}.$$

(a) Original (b) Corrupted (c) FPCA (d) LADM (e) LADMPSAP

Lin et al., *Linearized Alternating Direction Method with Parallel Splitting and Adaptive Penalty for Separable Convex Programs in Machine Learning*, ML, 2015.

Experiment

Table 1: Numerical comparison on the NMC problem with synthetic data, average of 10 runs. q, t and d_r denote, respectively, sample ratio, the number of measurements $t = q(mn)$ and the "degree of freedom" defined by $d_r = r(m+n-r)$ for a matrix with rank r and q. Here we set $m = n$ and fix $r = 10$ in all the tests.

$\mathbf{X}$			LADM				LADMPSAP			
n	q	t/d_r	Iter.	Time(s)	RelErr	FA	Iter.	Time(s)	RelErr	FA
1000	20%	10.05	375	177.92	1.35E-5	6.21E-4	58	24.94	9.67E-6	0
	10%	5.03	1000	459.70	4.60E-5	6.50E-4	109	42.68	1.72E-5	0
5000	20%	50.05	229	1613.68	1.08E-5	1.93E-4	49	369.96	9.05E-6	0
	10%	25.03	539	2028.14	1.20E-5	7.70E-5	89	365.26	9.76E-6	0
10000	10%	50.03	463	6679.59	1.11E-5	4.18E-5	89	1584.39	1.03E-5	0

Table 1: Numerical comparison on the image inpainting problem.

Method	#Iter.	Time(s)	PSNR	FA
FPCA	179	228.99	27.77dB	9.41E-4
LADM	228	207.95	26.98dB	2.92E-3
LADMPSAP	143	134.89	31.39dB	0

Lin et al., *Linearized Alternating Direction Method with Parallel Splitting and Adaptive Penalty for Separable Convex Programs in Machine Learning*, ML, 2015.

LADM with Parallel Splitting and Adaptive Penalty (LADMPSAP)

Enhanced convergence results:

Theorem 1: If $\{\beta_k\}$ is non-decreasing and $\sum_{k=1}^{+\infty} \beta_k^{-1} = +\infty$, $\eta_i > n\|\mathcal{A}_i\|^2$, $\partial f_i(\mathbf{x})$ is bounded, $i = 1, \cdots, n$, then the sequence $\{\mathbf{x}_i^k\}$ generated by LADMPSAP converges to an optimal solution to the model problem.

Theorem 2: If $\{\beta_k\}$ is non-decreasing, $\eta_i > n\|\mathcal{A}_i\|^2$, $\partial f_i(\mathbf{x})$ is bounded, $i = 1, \cdots, n$, then $\sum_{k=1}^{+\infty} \beta_k^{-1} = +\infty$ is also the necessary condition for the global convergence of $\{\mathbf{x}_i^k\}$ generated by LADMPSAP to an optimal solution to the model problem.

With the above analysis, when all the subgradients of the component objective functions are bounded we can remove the upper bound $\beta_{\max}$.

Lin et al., *Linearized Alternating Direction Method with Parallel Splitting and Adaptive Penalty for Separable Convex Programs in Machine Learning*, ML, 2015.

LADM with Parallel Splitting and Adaptive Penalty (LADMPSAP)

Define $\mathbf{x} = (\mathbf{x}_1^T, \cdots, \mathbf{x}_n^T)^T$, $\mathbf{x}^* = ((\mathbf{x}_1^*)^T, \cdots, (\mathbf{x}_2^*)^T)^T$ and $f(\mathbf{x}) = \sum_{i=1}^{n} f_i(\mathbf{x}_i)$, where $(\mathbf{x}_1^*, \cdots, \mathbf{x}_2^*, \lambda^*)$ is a KKT point of the model problem.

Proposition: $\tilde{\mathbf{x}}$ is an optimal solution to the model problem iff there exists $\alpha > 0$, such that

$$f(\tilde{\mathbf{x}}) - f(\mathbf{x}^*) + \sum_{i=1}^{n} \langle \mathcal{A}_i^*(\lambda^*), \tilde{\mathbf{x}}_i - \mathbf{x}_i^* \rangle + \alpha \left\| \sum_{i=1}^{n} \mathcal{A}_i(\tilde{\mathbf{x}}_i) - \mathbf{b} \right\|^2 = 0.$$

Our criterion for checking the optimality of a solution is much simpler than that in He et al. 2011, which has to compare with all $(\mathbf{x}_1, \cdots, \mathbf{x}_n, \lambda) \in \mathbb{R}^{d_1} \times \cdots \times \mathbb{R}^{d_n} \times \mathbb{R}^m$.

Lin et al., *Linearized Alternating Direction Method with Parallel Splitting and Adaptive Penalty for Separable Convex Programs in Machine Learning*, ML, 2015.

B. S. He and X. Yuan. *On the O(1/t) convergence rate of alternating direction method*. Preprint, 2011.

LADM with Parallel Splitting and Adaptive Penalty (LADMPSAP)

Theorem 3: Define $\bar{\mathbf{x}}^K = \sum_{k=0}^{K} \gamma_k \mathbf{x}^{k+1}$, where $\gamma_k = \beta_k^{-1} / \sum_{j=0}^{K} \beta_j^{-1}$. Then

$$f(\bar{\mathbf{x}}^K) - f(\mathbf{x}^*) + \sum_{i=1}^{n} \left\langle \mathcal{A}_i^*(\lambda^*), \bar{\mathbf{x}}_i^K - \mathbf{x}_i^* \right\rangle + \frac{\alpha\beta_0}{2} \left\| \sum_{i=1}^{n} \mathcal{A}_i(\bar{\mathbf{x}}_i^K) - \mathbf{b} \right\|^2 <= C_0 / \left(2 \sum_{k=0}^{K} \beta_k^{-1} \right), \tag{1}$$

where $\alpha^{-1} = (n+1) \max\left(1, \left\{ \frac{\|\mathcal{A}_i\|^2}{\eta_i - n\|\mathcal{A}_i\|^2}, i = 1, \cdots, n \right\}\right)$ and $C_0 = \sum_{i=1}^{n} \eta_i \left\| \mathbf{x}_i^0 - \mathbf{x}_i^* \right\|^2 + \beta_0^{-2} \left\| \lambda^0 - \lambda^* \right\|^2$.

A much simpler proof of convergence rate (in ergodic sense)!

Lin et al., *Linearized Alternating Direction Method with Parallel Splitting and Adaptive Penalty for Separable Convex Programs in Machine Learning*, ML, 2015.

B. S. He and X. Yuan. *On the O(1/t) convergence rate of alternating direction method*. Preprint, 2011.

Proximal LADMPSAP

- Even more general problem:

$$\min_{\mathbf{x}_1,\cdots,\mathbf{x}_n} \sum_{i=1}^{n} f_i(\mathbf{x}_i), \quad s.t. \quad \sum_{i=1}^{n} \mathcal{A}_i(\mathbf{x}_i) = \mathbf{b}.$$

$$f_i(\mathbf{x}_i) = g_i(\mathbf{x}_i) + h_i(\mathbf{x}_i),$$

where both g_i and h_i are convex, g_i is $C^{1,1}$:

$$\|\nabla g_i(\mathbf{x}) - \nabla g_i(\mathbf{y})\| \leq L_i \|\mathbf{x} - \mathbf{y}\|, \quad \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^{d_i},$$

and h_i may not be differentiable but its proximal operation is easily solvable.

Proximal LADMPSAP

- Linearize the augmented term to obtain:

$$\mathbf{x}_i^{k+1} = \underset{\mathbf{x}_i}{\operatorname{argmin}}\, h_i(\mathbf{x}_i) + g_i(\mathbf{x}_i) + \frac{\sigma_i^{(k)}}{2}\left\|\mathbf{x}_i - \mathbf{x}_i^k + \mathcal{A}_i^\dagger(\hat{\lambda}^k)/\sigma_i^{(k)}\right\|^2, \quad i = 1, \cdots, n,$$

- Further linearize g_i:

$$\begin{aligned}
\mathbf{x}_i^{k+1} &= \underset{\mathbf{x}_i}{\operatorname{argmin}}\, h_i(\mathbf{x}_i) + g_i(\mathbf{x}_i^k) + \frac{\sigma_i^{(k)}}{2}\left\|\mathcal{A}_i^\dagger(\hat{\lambda}^k)/\sigma_i^{(k)}\right\|^2 \\
&\quad + \langle \nabla g_i(\mathbf{x}_i^k) + \mathcal{A}_i^\dagger(\hat{\lambda}^k), \mathbf{x}_i - \mathbf{x}_i^k\rangle + \frac{\tau_i^{(k)}}{2}\left\|\mathbf{x}_i - \mathbf{x}_i^k\right\|^2 \\
&= \underset{\mathbf{x}_i}{\operatorname{argmin}}\, h_i(\mathbf{x}_i) + \frac{\tau_i^{(k)}}{2}\left\|\mathbf{x}_i - \mathbf{x}_i^k + \frac{1}{\tau_i^{(k)}}[\mathcal{A}_i^\dagger(\hat{\lambda}^k) + \nabla g_i(\mathbf{x}_i^k)]\right\|^2.
\end{aligned}$$

- Convergence condition:

$\tau_i^{(k)} = T_i + \beta_k\eta_i$, where $T_i \geq L_i$ and $\eta_i > n\|\mathcal{A}_i\|^2$ are both positive constants.

Experiment

- Group Sparse Logistic Regression with Overlap

$$\min_{\mathbf{w},b} \frac{1}{s}\sum_{i=1}^{s} \log\left(1+\exp\left(-y_i(\mathbf{w}^T\mathbf{x}_i+b)\right)\right)+\mu\sum_{j=1}^{t}\|\mathbf{S}_j\mathbf{w}\|, \tag{1}$$

where $\mathbf{x}_i$ and y_i, $i=1,\cdots,s$, are the training data and labels, respectively, and $\mathbf{w}$ and b parameterize the linear classifier. $\mathbf{S}_j$, $j=1,\cdots,t$, are the selection matrices, with only one 1 at each row and the rest entries are all zeros. The groups of entries, $\mathbf{S}_j\mathbf{w}$, $j=1,\cdots,t$, may overlap each other.

Introducing $\bar{\mathbf{w}}=(\mathbf{w}^T,b)^T$, $\bar{\mathbf{x}}_i=(\mathbf{x}_i^T,1)^T$, $\mathbf{z}=(\mathbf{z}_1^T,\mathbf{z}_2^T,\cdots,\mathbf{z}_t^T)^T$, and $\bar{\mathbf{S}}=(\mathbf{S},\mathbf{0})$, where $\mathbf{S}=(\mathbf{S}_1^T,\cdots,\mathbf{S}_t^T)^T$, (1) can be rewritten as

$$\min_{\bar{\mathbf{w}},\mathbf{z}} \frac{1}{s}\sum_{i=1}^{s} \log\left(1+\exp\left(-y_i(\bar{\mathbf{w}}^T\bar{\mathbf{x}}_i)\right)\right)+\mu\sum_{j=1}^{t}\|\mathbf{z}_j\|, \quad s.t. \quad \mathbf{z}=\bar{\mathbf{S}}\bar{\mathbf{w}}, \tag{2}$$

The Lipschitz constant of the gradient of logistic function with respect to $\bar{\mathbf{w}}$ can be proven to be $L_{\bar{w}} \cdot \frac{1}{4s}\|\bar{\mathbf{X}}\|_2^2$, where $\bar{\mathbf{X}}=(\bar{\mathbf{x}}_1,\bar{\mathbf{x}}_2,\cdots,\bar{\mathbf{x}}_s)$.

Experiment

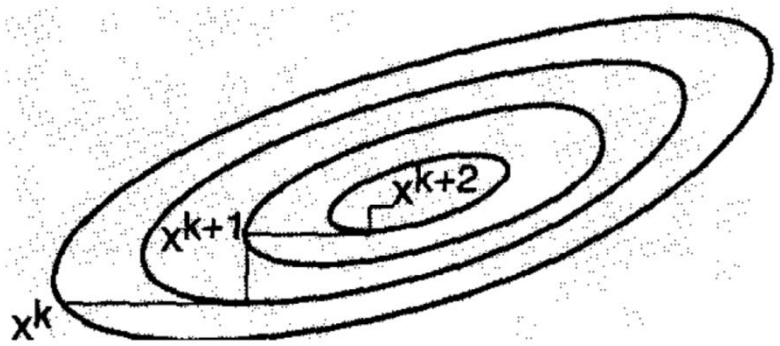
$(s,\ p,\ t,\ q)$	Method	Time	#Iter.	$\frac{\|\hat{\bar{\mathbf{w}}}-\bar{\mathbf{w}}^*\|}{\|\bar{\mathbf{w}}^*\|}$	$\frac{\|\hat{\mathbf{z}}-\mathbf{z}^*\|}{\|\mathbf{z}^*\|}$
(300, 901, 100, 10)	ADM	294.15	43	0.4800	0.4790
	LADM	229.03	43	0.5331	0.5320
	LADMPS	105.50	47	0.2088	0.2094
	LADMPSAP	57.46	39	0.0371	0.0368
	pLADMPSAP	**1.97**	141	**0.0112**	**0.0112**
(450, 1351, 150, 15)	ADM	450.96	33	0.4337	0.4343
	LADM	437.12	36	0.5126	0.5133
	LADMPS	201.30	39	0.1938	0.1937
	LADMPSAP	136.64	37	0.0321	0.0306
	pLADMPSAP	**4.16**	150	**0.0131**	**0.0131**
(600, 1801, 200, 20)	ADM	1617.09	62	1.4299	1.4365
	LADM	1486.23	63	1.5200	1.5279
	LADMPS	494.52	46	0.4915	0.4936
	LADMPSAP	216.45	32	0.0787	0.0783
	pLADMPSAP	**5.77**	127	**0.0276**	**0.0277**

Coordinate Descent and Block Coordinate Descent

- Coordinate Descent

The cost is minimized along one coordinate direction at each iteration. The order in which coordinates are chosen may vary in the course of the algorithm. In the case where this order is cyclical, given $\mathbf{x}^k$, the i-th coordinate of $\mathbf{x}^{k+1}$ is determined by

$$x_i^{k+1} = \underset{x_i}{\operatorname{argmin}}\, f(x_1^{k+1}, x_2^{k+1}, \cdots, x_{i-1}^{k+1}, x_i, x_{i+1}^{k}, \cdots, x_n^{k}). \tag{1}$$



Coordinate Descent and Block Coordinate Descent

- Coordinate Descent - Parallel computation

Suppose that there is a subset of coordinates $x_{i_1}, x_{i_2}, \cdots, x_{i_m}$, which are not coupled through the cost function, that is, $f(\mathbf{x})$ can be written as $\sum_{i=1}^{m} f_{i_r}(\mathbf{x})$, where for each r, $fi_r(\mathbf{x})$ does not depend on the coordinates x_{i_s} for all $s \neq r$. Then one can perform the m coordinate descent iterations

$$x_{i_r}^{k+1} = \underset{\xi}{\operatorname{argmin}}\, f_{i_r}(\mathbf{x}^k + \xi \mathbf{e}_{i_r}), \quad r = 1, \cdots, m,$$

independently and in parallel.

Coordinate Descent and Block Coordinate Descent

- Coordinate Descent - Convergence

The coordinate descent method generally has similar convergence properties to steepest descent. For continuously differentiable functions, it can be shown to generate sequences whose limit points are stationary, although the proof of this is sometimes complicated and requires some additional assumptions. The convergence rate of coordinate descent to nonsingular and singular local minima can be shown to be linear and sublinear, respectively, similar to steepest descent. Often, the choice between coordinate descent and steepest descent is dictated by the structure of the objective function. Both methods can be very slow, but for many practical contexts, they can be quite effective.

Coordinate Descent and Block Coordinate Descent

- Block Coordinate Descent

$$\begin{aligned} &\min_{\mathbf{x}} \ f(\mathbf{x}), \\ &s.t. \ \mathbf{x} \in \mathcal{X}, \end{aligned} \tag{2}$$

where $\mathcal{X}$ is a Cartesian product of closed convex sets $\mathcal{X}_1, \cdots, \mathcal{X}_m$:

$$\mathcal{X} = \mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_m. \tag{3}$$

We assume that $\mathcal{X}_i$ is a closed convex subset of $\mathbb{R}^{n_i}$ and $n = n_1 + \cdots + n_m$. The vector $\mathbf{x}$ is partitioned as

$$\mathbf{x} = (\mathbf{x}_1^T, \mathbf{x}_2^T, \cdots, \mathbf{x}_m^T)^T,$$

where each $\mathbf{x}_i$ belong to $\mathbb{R}^{n_i}$, so the constraint $\mathbf{x} \in \mathcal{X}$ is equivalent to

$$\mathbf{x}_i \in \mathcal{X}_i, \quad i = 1 \cdots, m.$$

Coordinate Descent and Block Coordinate Descent

- Block Coordinate Descent

Let us assume that for every $\mathbf{x} \in \mathcal{X}$ and every $i = 1, \cdots, m$, the optimization problem:

$$\min_{\mathbf{x}} f(\mathbf{x}_1, \cdots, \mathbf{x}_{i-1}, \boldsymbol{\xi}, \mathbf{x}_{i+1}, \cdots, \mathbf{x}_m),$$
$$s.t.\ \boldsymbol{\xi} \in \mathcal{X}_i,$$

has at least one solution. The following algorithm, known as *block coordinate descent* or *nonlinear Gauss-Seidel* method, generates the next iterate $\mathbf{x}^{k+1} = (\mathbf{x}_1^{k+1}, \mathbf{x}_2^{k+1}, \cdots, \mathbf{x}_m^{k+1})^T$, given the current iterate $\mathbf{x}^k = (\mathbf{x}_1^k, \mathbf{x}_2^k, \cdots, \mathbf{x}_m^k)^T$, according to the iteration

$$\mathbf{x}_i^{k+1} = \underset{\boldsymbol{\xi} \in \mathbf{X}_i}{\operatorname{argmin}}\ f(\mathbf{x}_1^{k+1}, \cdots, \mathbf{x}_{i-1}^{k+1}, \boldsymbol{\xi}, \mathbf{x}_{i+1}^{k}, \cdots, \mathbf{x}_m^{k}), \quad i = 1, \cdots, m. \tag{4}$$

Coordinate Descent and Block Coordinate Descent

- Block Coordinate Descent - Convergence

Proposition 1 (Convergence of Block Coordinate Descent)**.** *Suppose that f is continuously differentiable over the set $\mathcal{X}$ of equation (3). Furthermore, suppose that for each i and $\mathbf{x} \in \mathcal{X}$, the minimum below*

$$\min_{\boldsymbol{\xi} \in \mathcal{X}_i} f(\mathbf{x}_1, \cdots, \mathbf{x}_{i-1}, \boldsymbol{\xi}, \mathbf{x}_{i+1}, \cdots, \mathbf{x}_m)$$

is uniquely attained. Let $\{\mathbf{x}^k\}$ be the sequence generated by the block coordinate descent method (4). Then every accumulate point of $\{\mathbf{x}^k\}$ is a stationary point.

Coordinate Descent and Block Coordinate Descent

- Block Coordinate Descent - Examples

Dictionary learning:

$$\min_{\mathbf{D},\mathbf{X}} \frac{1}{2}\|\mathbf{Y}-\mathbf{D}\mathbf{X}\|_F^2 + \lambda\|\mathbf{X}\|_1, \quad s.t. \quad \|\mathbf{d}_i\|_2 = 1, i = 1, \cdots, K.$$

Low-rank matrix completion:

$$\min_{\mathbf{U},\mathbf{V},\mathbf{A}} \frac{1}{2}\|\mathbf{U}\mathbf{V}^T-\mathbf{A}\|_F^2, \quad s.t. \quad \mathcal{P}_\Omega(\mathbf{A}) = \mathcal{P}_\Omega(\mathbf{D}).$$

Robust Matrix Factorization:

$$\min_{\mathbf{U},\mathbf{V}} \|\mathbf{W}\odot(\mathbf{U}\mathbf{V}^T-\mathbf{M})\|_1 + R_u(\mathbf{U}) + R_v(\mathbf{V}), \quad s.t. \quad \mathbf{U}\in\mathcal{U}, \mathbf{V}\in\mathcal{V}.$$

Truncated Nuclear Norm Minimization:

$$\min_{\mathbf{X}} \|\mathbf{X}\|_r + f(\mathbf{X}), \text{ where } \|\mathbf{X}\|_r = \sum_{i=r+1}^{\min(m,n)} \sigma_i(\mathbf{X}).$$

Chapter 8. Randomized Algorithms

- Stochastic Gradient Descent (SGD)

Stochastic Gradient Descent (SGD)

- Why randomized algorithms?

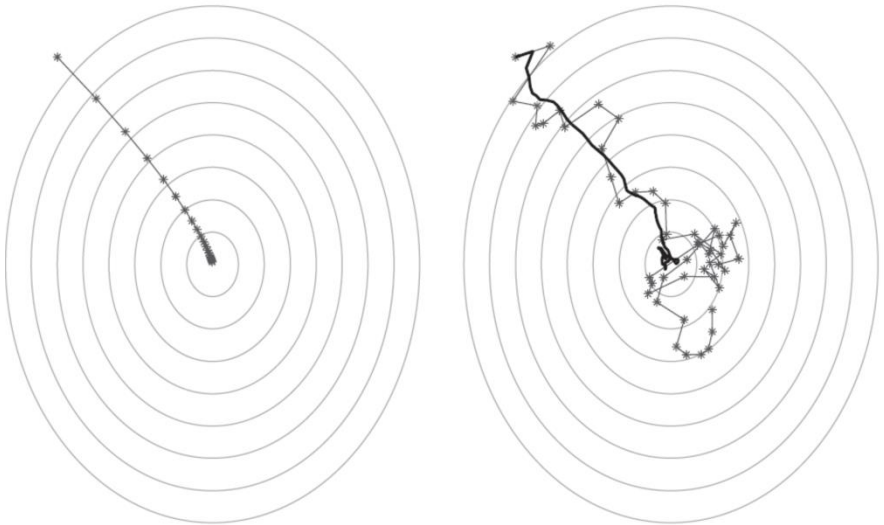
$$f(\mathbf{w}) = \frac{1}{N}\sum_{i=1}^{N} \ell(\mathbf{w}, \boldsymbol{\xi}_i).$$

$$L_{\mathcal{D}}(\mathbf{w}) = \mathbb{E}_{z\sim\mathcal{D}}[\ell(\mathbf{w}, z).$$

($\mathcal{D}$ unknown)

$$\mathbf{v}_k \approx \nabla f(\mathbf{w}_k) \qquad \mathbf{v}_k = \frac{1}{N_k}\sum_{j=1}^{N_k} \nabla \ell(\mathbf{w}, \boldsymbol{\xi}_{i_j}).$$

$$\mathbb{E}(\mathbf{v}_k) = \nabla f(\mathbf{w}_k)$$



Stochastic Gradient Descent (SGD)

- SGD

Stochastic Gradient Descent (SGD) for minimizing $f(\mathbf{w})$
parameters: Scalar $\eta > 0$, integer $T > 0$
parameters: $\mathbf{w}^{(1)} = \mathbf{0}$
for $t = 1, 2, \cdots, T$
choose $\mathbf{v}_t$ at random from a distribution such that $\mathbb{E}[\mathbf{v}_t | \mathbf{w}^{(t)}] \in \partial f(\mathbf{w}^{(t)})$
update $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \mathbf{v}_t$
output $\bar{\mathbf{w}} = \frac{1}{T} \sum_{t=1}^{T} \mathbf{w}^{(t)}$.

Stochastic Gradient Descent (SGD)

- Convergence

Theorem 1. *Let $B, \rho > 0$. Let f be a convex function and let*

$$\mathbf{w}^* \in \arg \min_{\mathbf{w}:\|\mathbf{w}\| \leq B} f(\mathbf{w}).$$

Assume that SGD is run for T iterations with $\eta = \sqrt{\frac{B^2}{\rho^2 T}}$. Assume also that for all t, $\|\mathbf{v}_t\| \leq \rho$ with probability 1. Then,

$$\mathbb{E}[f(\bar{\mathbf{w}})] - f(\mathbf{w}^*) \leq \frac{B\rho}{\sqrt{T}}.$$

Therefore, for any $\epsilon > 0$, to achieve $\mathbb{E}[f(\bar{\mathbf{w}})] - f(\mathbf{w}^*) \leq \epsilon$, it suffices to run the SGD algorithm for a number of iterations that satisfies

$$T \geq \frac{B^2 \rho^2}{\epsilon^2}.$$

Stochastic Gradient Descent (SGD)

- Convergence

Lemma 1. *Let $\mathbf{v}_1, \cdots, \mathbf{v}_T$ be an arbitrary sequence of vectors. Any algorithm with an initialization $\mathbf{w}^{(1)} = \mathbf{0}$ and an update rule of the form*

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \mathbf{v}_t \tag{1}$$

satisfies

$$\sum_{t=1}^{T} \left\langle \mathbf{w}^{(t)} - \mathbf{w}^*, \mathbf{v}_t \right\rangle \leq \frac{\|\mathbf{w}^*\|^2}{2\eta} + \frac{\eta}{2} \sum_{t=1}^{T} \|\mathbf{v}_t\|^2. \tag{2}$$

In particular, for every $B, \rho > 0$, if for all t we have that $\|\mathbf{v}_t\| \leq \rho$ and if we set $\eta = \frac{B}{\rho\sqrt{T}}$, then for every $\mathbf{w}^*$ with $\|\mathbf{w}^*\| \leq B$ we have

$$\sum_{t=1}^{T} \left\langle \mathbf{w}^{(t)} - \mathbf{w}^*, \mathbf{v}_t \right\rangle \leq \frac{B\rho}{\sqrt{T}}$$

Stochastic Gradient Descent (SGD)

- Variants - Adding a Projection Step

1. $\mathbf{w}^{(t+\frac{1}{2})} = \mathbf{w}^{(t)} - \eta \mathbf{v}_t$
2. $\mathbf{w}^{(t+1} = \arg\min_{\mathbf{w} \in \mathcal{H}} \|\mathbf{w} - \mathbf{w}^{(t+\frac{1}{2})}\|$

Lemma 1 (Projection Lemma)**.** *Let $\mathcal{H}$ be a closed convex set and let $\mathbf{v}$ be the projection of $\mathbf{w}$ onto $\mathcal{H}$, namely,*

$$\mathbf{v} = \arg\min_{\mathbf{x} \in \mathcal{H}} \|\mathbf{x} - \mathbf{w}\|^2.$$

Then, for every $\mathbf{u} \in \mathcal{H}$,

$$\|\mathbf{w} - \mathbf{u}\|^2 - \|\mathbf{v} - \mathbf{u}\|^2 \geq 0.$$

Stochastic Gradient Descent (SGD)

- Variants - Adding a Projection Step

Equipped with the preceding lemma, we can easily adapt the analysis of SGD to the case in which we add projection steps on a closed and convex set. Simply note that for every t,

$$\begin{aligned}
&\|\mathbf{w}^{(t+1)} - \mathbf{w}^*\|^2 - \|\mathbf{w}^{(t)} - \mathbf{w}^*\|^2 \\
=&\|\mathbf{w}^{(t+1)} - \mathbf{w}^*\|^2 - \|\mathbf{w}^{(t+\frac{1}{2})} - \mathbf{w}^*\|^2 + \|\mathbf{w}^{(t+\frac{1}{2})} - \mathbf{w}^*\|^2 - \|\mathbf{w}^{(t)} - \mathbf{w}^*\|^2 \\
\leq&\|\mathbf{w}^{(t+\frac{1}{2})} - \mathbf{w}^*\|^2 - \|\mathbf{w}^{(t)} - \mathbf{w}^*\|^2.
\end{aligned}$$

Therefore, Lemma 1 holds when we add projection steps and hence the rest of the analysis follows directly.

Stochastic Gradient Descent (SGD)

- Variants - Variable Step Size

Another variant of SGD is decreasing the step size as a function of t. That is, rather than updating with a constant η ,we use η_t. For instance, we can set $\eta_t = \frac{B}{\rho\sqrt{t}}$ and achieve a bound similar to Theorem 1. The idea is that when we are closer to the minimum of the function, we take our steps more carefully, so as not to "overshoot" the minimum.

Stochastic Gradient Descent (SGD)

- Variants - Other Averaging Techniques

We have set the output vector to be $\hat{\mathbf{w}} = \frac{1}{T}\sum_{t=1}^{T}\mathbf{w}^{(t)}$. There are alternative approaches such as outputting $\mathbf{w}^{(t)}$ for some random $t \in [t]$, or outputting the average of $\mathbf{w}^{(t)}$ over the last αT iterations, for some $\alpha \in (0, 1)$. One can also take a weighted average of the last few iterates. These more sophisticated averaging schemes can improve the convergence speed in some situations, such as in the case of strongly convex functions.

Stochastic Gradient Descent (SGD)

- Case of Strongly Convex Function

SGD for minimizing a λ-strongly convex function
Goal: Solve $\min_{\mathbf{w}\in\mathcal{H}} f(\mathbf{w})$
parameters: T
parameters: $\mathbf{w}^{(1)} = \mathbf{0}$
for $t = 1, 2, \cdots, T$
Choose a random vector $\mathbf{v}_t$ s.t. $\mathbb{E}[\mathbf{v}_t|\mathbf{w}^{(t)}] \in \partial f(\mathbf{w}^{(t)})$
Set $\eta_t = 1/(\lambda t)$
Set $\mathbf{w}^{(t+\frac{1}{2})} = \mathbf{w}^{(t)} - \eta_t \mathbf{v}_t$
Set $\mathbf{w}^{(t+1)} = \arg\min_{\mathbf{w}\in\mathcal{H}} \left\|\mathbf{w} - \mathbf{w}^{(t+\frac{1}{2})}\right\|^2$
output $\bar{\mathbf{w}} = \frac{1}{T}\sum_{t=1}^{T} \mathbf{w}^{(t)}$.

Stochastic Gradient Descent (SGD)

- Case of Strongly Convex Function

Theorem 2. *Assume that f is λ-strongly convex and that $\mathbb{E}[\|\mathbf{v}_t\|^2] \leq \rho^2$. Let $\mathbf{w}^* \in \arg\min_{\mathbf{w}\in\mathcal{H}} f(\mathbf{w})$ be an optimal solution. Then,*

$$\mathbb{E}[f(\hat{\mathbf{w}})] - f(\mathbf{w}^*) \leq \frac{\rho^2}{2\lambda T}(1 + \log(T)).$$

Stochastic Gradient Descent (SGD)

- Learning with SGD - SGD for Risk Minimization

Recall that in learning we face the problem of minimizing the risk function

$$L_{\mathcal{D}}(\mathbf{w}) = \mathbb{E}_{z\sim\mathcal{D}}[\ell(\mathbf{w}, z)].$$

We have seen the method of empirical risk minimization, where we minimize the empirical risk, $L_S(\mathbf{w})$, as an estimate to minimizing $L_{\mathcal{D}}(\mathbf{w})$. SGD allows us to take a different approach and minimize $L_{\mathcal{D}}(\mathbf{w})$ directly. Since we do not know $\mathcal{D}$, we cannot simply calculate $\nabla L_{\mathcal{D}}(\mathbf{w}^{(t)})$ and minimize it with the GD method. With SGD, however, all we need is to find an unbiased estimate of the gradient of $L_{\mathcal{D}}(\mathbf{w})$, that is, a random vector whose conditional expected value is $\nabla L_{\mathcal{D}}(\mathbf{w}^{(t)})$. We shall now see how such an estimate can be easily constructed.

Stochastic Gradient Descent (SGD)

- Learning with SGD - SGD for Risk Minimization

We first consider the case of differentiable loss functions. Hence the risk function $L_{\mathcal{D}}$ is also differentiable. The construction of the random vector $\mathbf{v}_t$ will be as follows: First, sample $z \sim \mathcal{D}$. Then, define $\mathbf{v}_t$ to be the gradient of the function $\ell(\mathbf{w}, z)$ with respect to $\mathbf{w}$, at the point $\mathbf{w}^{(t)}$. Then, by the linearity of the gradient we have

$$\mathbb{E}[\mathbf{v}_t|\mathbf{w}^{(t)}] = \mathbb{E}_{z\sim\mathcal{D}}[\nabla\ell(\mathbf{w}^{(t)}, z)] = \nabla\mathbb{E}_{z\sim\mathcal{D}}[\ell(\mathbf{w}^{(t)}, z)] = \nabla L_{\mathcal{D}}(\mathbf{w}^{(t)}). \quad (1)$$

The gradient of the loss function $\ell(\mathbf{w}, z)$ at $\mathbf{w}^{(t)}$ is therefore an unbiased estimate of the gradient of the risk function $L_{\mathcal{D}}(\mathbf{w}^{(t)})$ and is easily constructed by sampling a single fresh example $z \sim \mathcal{D}$ at each iteration t.

Stochastic Gradient Descent (SGD)

- Learning with SGD - SGD for Risk Minimization

For nondifferentiable loss functions, we simply let $\mathbf{v}_t$ be a subgradient of $\ell(\mathbf{w}, z)$ at $\mathbf{w}^{(t)}$. Then, for every $\mathbf{u}$ we have

$$\ell(\mathbf{u}, z) - \ell(\mathbf{w}^{(t)}, z) \geq \langle \mathbf{u} - \mathbf{w}^{(t)}, \mathbf{v}_t \rangle.$$

Taking expectation on both sides with respect to $z \sim \mathcal{D}$ and conditioned on the value $\mathbf{w}^{(t)}$ we obtain

$$\begin{aligned} L_{\mathcal{D}}(\mathbf{u}) - L_{\mathcal{D}}(\mathbf{w}^{(t)}) &= \mathbb{E}[\ell(\mathbf{u}, z) - \ell(\mathbf{w}^{(t)}, z) | \mathbf{w}^{(t)}] \\ &\geq \mathbb{E}[\langle \mathbf{u} - \mathbf{w}^{(t)}, \mathbf{v}_t \rangle | \mathbf{w}^{(t)}] \\ &= \langle \mathbf{u} - \mathbf{w}^{(t)}, \mathbb{E}[\mathbf{v}_t | \mathbf{w}^{(t)}] \rangle. \end{aligned}$$

It follows that $\mathbb{E}[\mathbf{v}_t | \mathbf{w}^{(t)}]$ is a subgradient of $L_{\mathcal{D}}(\mathbf{w})$ at $\mathbf{w}^{(t)}$.

Stochastic Gradient Descent (SGD)

- Learning with SGD - SGD for Risk Minimization

Stochastic Gradient Descent (SGD) for minimizing $L_{\mathcal{D}}(\mathbf{w})$
parameters: Scalar $\eta > 0$, integer $T > 0$
initialize: $\mathbf{w}^{(1)} = \mathbf{0}$
for $t = 1, 2, \cdots, T$
sample $z \sim \mathcal{D}$
pick $\mathbf{v}_t \in \partial \ell(\mathbf{w}^{(t)}, z)$
update $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \mathbf{v}_t$
output $\bar{\mathbf{w}} = \frac{1}{T} \sum_{t=1}^{T} \mathbf{w}^{(t)}$.

Stochastic Gradient Descent (SGD)

- Learning with SGD - SGD for Risk Minimization

Corollary 2. *Consider a convex-Lipschitz-bounded learning problem with parameters* ρ, B. *Then, for every* $\epsilon > 0$, *if we run the SGD method for minimizing* $L_{\mathcal{D}}(\mathbf{w})$ *with a number of iterations (i.e., number of examples)*

$$T \geq \frac{B^2 \rho^2}{\epsilon^2}$$

and with $\eta = \sqrt{\frac{B^2}{\rho^2 T}}$, *then the output of SGD satisfies*

$$\mathbb{E}[L_{\mathcal{D}}(\bar{\mathbf{w}})] \leq \min_{\mathbf{w} \in \mathcal{H}} L_{\mathcal{D}}(\mathbf{w}) + \epsilon.$$

Stochastic Gradient Descent (SGD)

- SGD for Regularized Loss Minimization

$$\min_{\mathbf{w}} \left(\frac{\lambda}{2} \|\mathbf{w}\|^2 + L_S(\mathbf{w}) \right). \tag{1}$$

Define $f(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|^2 + L_S(\mathbf{w})$. Note that f is a λ-strongly convex function; therefore, we can apply the SGD variant for strongly convex functions. To apply this algorithm, we only need to find a way to construct an unbiased estimate of a subgradient of f at $\mathbf{w}^{(t)}$. This is easily done by noting that if we pick z uniformly at random from S, and choose $\mathbf{v}_t \in \partial \ell(\mathbf{w}^{(t)}, z)$ then the expected value of $\lambda \mathbf{w}^{(t)} + \mathbf{v}_t$ is a subgradient of f at $\mathbf{w}^{(t)}$.

Stochastic Gradient Descent (SGD)

- Sum of smooth and strongly convex functions

Consider unconstrained minimization of

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

where $f_1, \ldots, f_m$ are β-smooth and convex functions, and f is α-strongly convex. Typically in machine learning α can be as small as $1/m$, while β is of order of a constant. In other words the condition number $\kappa = \beta/\alpha$ can be as large as $\Omega(m)$. Let us now compare the basic gradient descent, that is

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \frac{\eta}{m}\sum_{i=1}^{m} \nabla f_i(\mathbf{x}),$$

to SGD

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \eta \nabla f_{i_t}(\mathbf{x}),$$

where i_t is drawn uniformly at random in $[m]$ (independently of everything else).

Stochastic Gradient Descent (SGD)

- Sum of smooth and strongly convex functions

Theorem 2. *Let f be α-strongly convex and β-smooth on $\mathcal{X}$. Then projected gradient descent with $\eta = \frac{1}{\beta}$ satisfies for $t \geq 0$,*

$$\|\mathbf{x}_{t+1} - \mathbf{x}^*\|^2 \leq \exp\left(-\frac{t}{\kappa}\right) \|\mathbf{x}_1 - \mathbf{x}^*\|^2, \quad \textit{where } \kappa = \beta/\alpha.$$

Theorem 3. *Let f be α-strongly convex, and assume that the stochastic oracle is such that $\mathbb{E}\|\tilde{g}(\mathbf{x})\|_*^2 \leq B^2$. Then SGD with $\eta_s = \frac{2}{\alpha(s+1)}$ satisfies*

$$f\left(\sum_{s=1}^{t} \frac{2s}{t(t+1)} \mathbf{x}_s\right) - f(\mathbf{x}^*) \leq \frac{2B^2}{\alpha(t+1)}.$$

Stochastic Gradient Descent (SGD)

- Sum of smooth and strongly convex functions

Theorem 2 shows that gradient descent requires $\mathcal{O}(m\kappa \log(1/\varepsilon))$ gradient computations (which can be improved to $\mathcal{O}(m\sqrt{\kappa} \log(1/\varepsilon))$ with Nesterov's accelerated gradient descent), while Theorem 3 shows that SGD (with appropriate averaging) requires $\mathcal{O}(1/(\alpha\varepsilon))$ gradient computations. Thus one can obtain a low accuracy solution reasonably fast with SGD, but for high accuracy the basic gradient descent is more suitable. Can we get the best of both worlds? This question was answered positively with SAG (Stochastic Averaged Gradient) and with SDCA (Stochastic Dual Coordinate Ascent). These methods require only $\mathcal{O}((m+\kappa) \log(1/\varepsilon))$ gradient computations. We describe below the SVRG (Stochastic Variance Reduced Gradient descent) algorithm which makes the main ideas of SAG and SDCA more transparent.

Stochastic Gradient Descent (SGD)

- Stochastic Variance Reduced Gradient (SVRG) descent

To obtain a linear rate of convergence one needs to make "big steps", that is the step-size should be of order of a constant. In SGD the stepsize is typically of order $1/\sqrt{t}$ because of the variance introduced by the stochastic oracle. The idea of SVRG is to "center" the output of the stochastic oracle in order to reduce the variance. Precisely instead of feeding $\nabla f_i(\mathbf{x})$ into the gradient descent one would use $\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y}) + \nabla f(\mathbf{y})$ where $\mathbf{y}$ is a centering sequence. This is a sensible idea since, when $\mathbf{x}$ and $\mathbf{y}$ are close to the optimum, one should have that $\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})$ will have a small variance, and of course $\nabla f(\mathbf{y})$ will also be small (note that $\nabla f_i(\mathbf{x})$ by itself is not necessarily small). This intuition is made formal with the following lemma.

When $\mathbf{x} \approx \mathbf{y} \approx \mathbf{x}^*$,

$$\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y}) + \nabla f(\mathbf{y}) \approx \nabla f_i(\mathbf{x}^*) - \nabla f_i(\mathbf{x}^*) + \nabla f(\mathbf{b}^*) = \mathbf{0}$$

vs.

$$\nabla f_i(\mathbf{x}) \approx \nabla f_i(\mathbf{x}^*) \neq \mathbf{0}$$

Stochastic Gradient Descent (SGD)

- Stochastic Variance Reduced Gradient (SVRG) descent

Lemma 1. *Let $f_1, \ldots, f_m$ be β-smooth convex functions on $\mathbb{R}^n$, and i be a random variable uniformly distributed in $[m]$. Then*

$$\mathbb{E}\|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{x}^*)\|_2^2 \leq 2\beta(f(\mathbf{x}) - f(\mathbf{x}^*)).$$

Proof. Let $g_i(\mathbf{x}) = f_i(\mathbf{x}) - f_i(\mathbf{x}^*) - \nabla f_i(\mathbf{x}^*)^\top(\mathbf{x} - \mathbf{x}^*)$. By convexity of f_i one has $g_i(\mathbf{x}) \geq 0$ for any $\mathbf{x}$ and in particular using ($f\left(\mathbf{x} - \frac{1}{\beta}\nabla f(\mathbf{x})\right) - f(\mathbf{x}) \leq -\frac{1}{2\beta}\|\nabla f(\mathbf{x})\|^2$) this yields $-g_i(\mathbf{x}) \leq -\frac{1}{2\beta}\|\nabla g_i(\mathbf{x})\|_2^2$ which can be equivalently written as

$$\|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{x}^*)\|_2^2 \leq 2\beta(f_i(\mathbf{x}) - f_i(\mathbf{x}^*) - \nabla f_i(\mathbf{x}^*)^\top(\mathbf{x} - \mathbf{x}^*)).$$

Taking expectation with respect to i and observing that $\mathbb{E}\nabla f_i(\mathbf{x}^*) = \nabla f(\mathbf{x}^*) = 0$ yields the claimed bound. □

Stochastic Gradient Descent (SGD)

- Stochastic Variance Reduced Gradient (SVRG) descent

On the other hand the computation of $\nabla f(\mathbf{y})$ is expensive (it requires m gradient computations), and thus the centering sequence should be updated more rarely than the main sequence. These ideas lead to the following epoch-based algorithm.

Let $\mathbf{y}^{(1)} \in \mathbb{R}^n$ be an arbitrary initial point. For $s = 1, 2, \ldots,$ let $\mathbf{x}_1^{(s)} = \mathbf{y}^{(s)}$. For $t = 1, \ldots, k$ let

$$\mathbf{x}_{t+1}^{(s)} = \mathbf{x}_t^{(s)} - \eta \left(\nabla f_{i_t^{(s)}}(\mathbf{x}_t^{(s)}) - \nabla f_{i_t^{(s)}}(\mathbf{y}^{(s)}) + \nabla f(\mathbf{y}^{(s)}) \right),$$

where $i_t^{(s)}$ is drawn uniformly at random (and independently of everything else) in $[m]$. Also let

$$y^{(s+1)} = \frac{1}{k} \sum_{t=1}^{k} \mathbf{x}_t^{(s)}.$$

Stochastic Gradient Descent (SGD)

- Stochastic Variance Reduced Gradient (SVRG) descent

Theorem 4. *Let $f_1, \ldots, f_m$ be β-smooth convex functions on $\mathbb{R}^n$ and f be α-strongly convex. Then SVRG with $\eta = \frac{1}{10\beta}$ and $k = 20\kappa$ satisfies*

$$\mathbb{E} f(\mathbf{y}^{(s+1)}) - f(\mathbf{x}^*) \leq 0.9^s (f(\mathbf{y}^{(1)}) - f(\mathbf{x}^*)).$$

Chapter 9. Acceleration Techniques

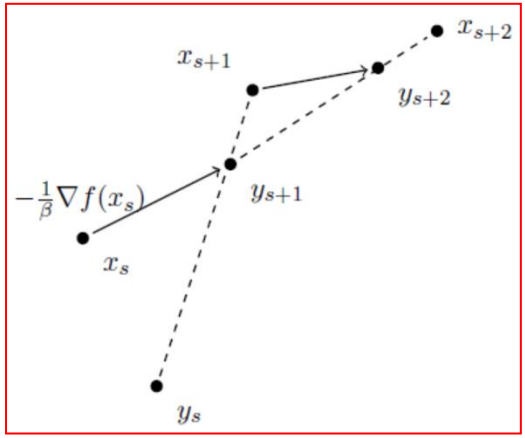
- Nesterov's accelerated gradient descent

The Smooth and Strongly Convex Case

Theorem 1. *Let f be α-strongly convex and L-smooth. Then gradient descent with $\eta = 2(\alpha + L)^{-1}$ satisfies for all $t \geq 0$,*

$$f(\mathbf{x}_{t+1}) - f(\mathbf{x}^*) \leq \frac{L}{2} \exp\left(-4t/(\kappa+1)\right) \|\mathbf{x}_1 - \mathbf{x}^*\|^2,$$

where $\kappa = L/\alpha$ is the condition number.



Nesterov's Accelerated algorithm: Start at an arbitrary initial point $\mathbf{x}_1 = \mathbf{y}_1$ and then iterate the following equations for $t \geq 1$,

$$\mathbf{y}_{t+1} = \mathbf{x}_t - \frac{1}{\beta}\nabla f(\mathbf{x}_t),$$

$$\mathbf{x}_{t+1} = \left(1 + \frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)\mathbf{y}_{t+1} - \frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\mathbf{y}_t.$$

The Smooth and Strongly Convex Case

Theorem 2. *Let f be α-strongly convex and L-smooth. Then Nesterov's accelerated gradient descent satisfies*

$$f(\mathbf{y}_t) - f(\mathbf{x}^*) \leq \frac{\alpha + \beta}{2} \|\mathbf{x}_1 - \mathbf{x}^*\|^2 \exp\left(-\frac{t-1}{\sqrt{\kappa}}\right).$$

The Smooth and Convex Case

Theorem 3. *Let f be convex and L-smooth. Then gradient descent with $\eta = L^{-1}$ satisfies*

$$f(\mathbf{x}_t) - f(\mathbf{x}^*) \leq \frac{2L\|\mathbf{x}_1 - \mathbf{x}^*\|^2}{t-1}.$$

Nesterov's Accelerated algorithm:

$$\lambda_0 = 0, \lambda_t = \frac{1 + \sqrt{1 + 4\lambda_{t-1}^2}}{2}, \text{ and } \gamma_t = \frac{1 - \lambda_t}{\lambda_{t+1}}.$$

(Note that $\gamma_t \leq 0$.) Now the algorithm is simply defined by the following equations, with $\mathbf{x}_1 = \mathbf{y}_1$ an arbitrary initial point,

$$\mathbf{y}_{t+1} = \mathbf{x}_t - \frac{1}{\beta}\nabla f(\mathbf{x}_t),$$

$$\mathbf{x}_{t+1} = (1 - \gamma_s)\mathbf{y}_{t+1} + \gamma_t \mathbf{y}_t.$$

The Smooth and Convex Case

Theorem 4. *Let f be a convex and L-smooth function, then Nesterov's accelerated gradient descent satisfies*

$$f(\mathbf{y}_t) - f(\mathbf{x}^*) \leq \frac{2L\|\mathbf{x}_1 - \mathbf{x}^*\|^2}{t^2}.$$