Stochastic Quasi-Newton Methods

Donald Goldfarb

Department of IEOR
Columbia University

UCLA Distinguished Lecture Series
May 17-19, 2016

Outline

- Stochastic Approximation
- Stochastic Gradient Descent
- Variance Reduction Techniques
- Newton-like and quasi-Newton methods for convex stochastic optimization problems using limited memory block BFGS updates.
- Numerical results on problems from machine learning.
- Quasi-Newton methods for nonconvex stochastic optimization problems using damped and modified limited memory BFGS updates.

Stochastic optimization

- Stochastic optimization

$$\min \; f(x) = \mathbb{E}[f(x,\xi)], \quad \xi \text{ is random variable}$$

- Or finite sum (with $f_i(x) \equiv f(x,\xi_i)$ for $i = 1,\ldots,n$ and very large n)

$$\min \; f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x)$$

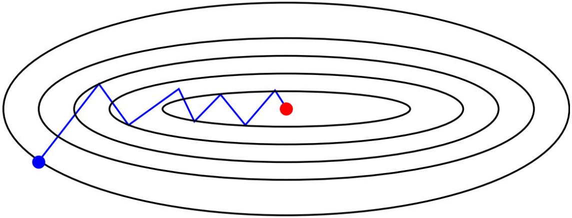
- f and ∇f are very expensive to evaluate; stochastic gradient descent (SGD) methods choose a random subset $\mathcal{S} \subset [n]$ and evaluate

$$f_{\mathcal{S}}(x) = \frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}} f_i(x) \quad \text{and} \quad \nabla f_{\mathcal{S}}(x) = \frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}} \nabla f_i(x)$$

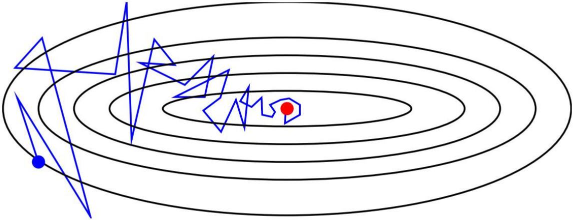
- Essentially, only noisy info about f, ∇f and $\nabla^2 f$ is available
- Challenge: how to smooth variability of stochastic methods
- Challenge: how to design methods that take advantage of noisy 2nd-order information?

Stochastic optimization

Deterministic gradient method

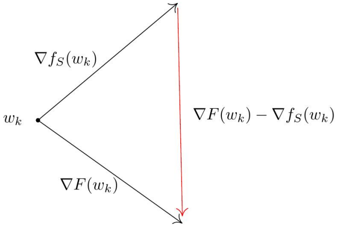


Stochastic gradient method



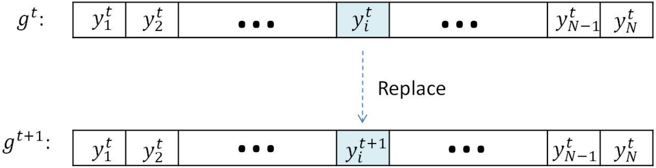
Stochastic Variance Reduced Gradients

- Stochastic methods converge slowly near the optimum due to the variance of the gradient estimates $\nabla f_{\mathcal{S}}(x)$; hence requiring a decreasing step size.
- We use the control variates approach of Johnson and Zhang (2013) for a SGD method SVRG.
- It uses $d = \nabla f_{\mathcal{S}}(x_t) - \nabla f_{\mathcal{S}}(w_k) + \nabla f(w_k)$, where w_k is a reference point, in place of $\nabla f_{\mathcal{S}}(x_t)$.
- w_k, and the full gradient, are computed after each full pass of the data, hence doubling the work of computing stochastic gradients.



Stochastic Average Gradient

- At iteration t
  - Sample i from $\{1, \ldots, N\}$
  - update $y_i^{t+1} = \nabla f_i(x^t)$ and $y_j^{t+1} = y_j^t$ for all $j \neq i$
  - Compute $g^{t+1} = \frac{1}{N} \sum_{j=1}^{N} y_i^{t+1}$
  - Set $x^{t+1} = x^t - \alpha^{t+1} g^{t+1}$



- Provable linear convergence in expectation.
- Other SGD variance reduction techniques have been recently proposes including the methods: SAGA, SDCA, S2GD.

Quasi-Newton Method for $\min f(x) : f \in C^1$

- Gradient method:

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

- Newton's method:

$$x_{k+1} = x_k - \alpha_k [\nabla^2 f(x_k)]^{-1} \nabla f(x_k)$$

- Quasi-Newton method:

$$x_{k+1} = x_k - \alpha_k B_k^{-1} \nabla f(x_k)$$

  where $B_k \succ 0$ approximates the Hessian matrix

- Update

$$B_{k+1} s_k = y_k, \qquad \text{(Secant equation)}$$

  where $s_k = x_{k+1} - x_k = \alpha_k d_k$, and $y_k = \nabla f_{k+1} - \nabla f_k$

BFGS

- BFGS quasi-Newton method

$$B_{k+1} = B_k + \frac{y_k^\top y_k}{s_k^\top y_k} - \frac{B_k s_k s_k^\top B_k}{s_k^\top B_k s_k}$$

  where $s_k := x_{k+1} - x_k$ and $y_k := \nabla f(x_{k+1}) - \nabla f(x_k)$
- $B_{k+1} \succ 0$ if $B_k \succ 0$ and $s_k^\top y_k > 0$ (curvature condition)
- Secant equation has a solution if $s_k^\top y_k > 0$
- When f is strongly convex, $s_k^\top y_k > 0$ holds automatically
- If f is nonconvex, use line search to guarantee $s_k^\top y_k > 0$
- $H_{k+1} = (I - \frac{s_k y_k^\top}{s_k^\top y_k}) H_k (I - \frac{y_k s_k^\top}{s_k^\top y_k}) + \frac{s_k s_k^\top}{s_k^\top y^k}$

Prior work on Quasi-Newton Methods for Stochastic Optimization

P1 N.N. Schraudolph, J. Yu and S.Günter. A stochastic quasi-Newton method for online convex optim. Int'l. Conf. AI & Stat., 2007
Modifies BFGS and L-BFGS updates by reducing the step s_k and the last term in the update of H_k, uses step size $\alpha_k = \beta/k$ for small $\beta > 0$.

P2 A. Bordes, L. Bottou and P. Gallinari. SGD-QN: Careful quasi-Newton stochastic gradient descent. JMLR vol. 10, 2009
Uses a diagonal matrix approximation to $[\nabla^2 f(\cdot)]^{-1}$ which is updated (hence, the name SGD-QN) on each iteration, $\alpha_k = 1/(k + \alpha)$.

Prior work on Quasi-Newton Methods for Stochastic Optimization

P3 A. Mokhtari and A. Ribeiro. RES: Regularized stochastic BFGS algorithm. IEEE Trans. Signal Process., no. 10, 2014.
Replaces y_k by $y_k - \delta s_k$ for some $\delta > 0$ in BFGS update and also adds δI to the update; uses $\alpha_k = \beta/k$; converges in expectation at sub-linear rate $\mathbb{E}(f(x_k) - f^*) \leq C/k$

P4 A. Mokhtari and A. Ribeiro. Global convergence of online limited memory BFGS. to appear in J. Mach. Learn. Res., 2015.
Uses L-BFGS without regularization and $\alpha_k = \beta/k$; converges in expectation at sub-linear rate $\mathbb{E}(f(x^k) - f^*) \leq C/k$

Prior work on Quasi-Newton Methods for Stochastic Optimization

- P5 R.H. Byrd, S.L. Hansen, J. Nocedal, and Y. Singer. A stochastic quasi-Newton method for large-scale optim. arXiv1401.7020v2, 2015
  Averages iterates over L steps keeping H_k fixed; uses average iterates to update H_k using subsampled Hessian to compute y_k; $\alpha_k = \beta/k$; converges in expectation at a sub-linear rate $\mathbb{E}(f(x^k) - f^*) \leq C/k$

- P6 P. Moritz, R. Nishihara, M.I. Jordan. A linearly-convergent stochastic L-BFGS Algorithm, 2015 arXiv:1508.02087v1
  Combines [P5] with SVRG; uses fixed step size α; converges in expectation at a linear rate.

Using Stochastic 2nd-order information

- Assumption: $f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x)$ is strongly convex and twice continuously differentiable.
- Choose (compute) a sketching matrix S_k (the columns of S_k are a set of directions).
- We do not use differences in noisy gradients to estimate curvature, but rather compute the action of the sub-sampled Hessian on S_k. i.e.,
- compute $Y_k = \frac{1}{|\mathcal{T}|}\sum_{i\in\mathcal{T}} \nabla^2 f_i(x) S_k,$ where $\mathcal{T} \subset [n]$.

Example of Hessian-Vector Computation

In binary classification problem, sample function (logistic loss)

$$f(w; x_i, z_i) = z_i \log(c(w; x_i)) + (1 - z_i) \log(1 - c(w; x_i))$$

where

$$c(w; x_i) = \frac{1}{1 + \exp(-x_i^\top w)}, \quad x_i \in \mathbb{R}^n, w \in \mathbb{R}^n, z_i \in \{0, 1\},$$

Gradient:

$$\nabla f(w; x_i, z_i) = (c(w; x_i) - z_i)x_i$$

Action of Hessian on s :

$$\nabla^2 f(w; x_i, z_i)s = c(w; x_i)(1 - c(w; x_i))(x_i^\top s)x_i$$

block BFGS

The block BFGS method computes a "least change" update to the current approximation H_k to the inverse Hessian matrix $\nabla^2 f(x)$ at the current point x, by solving

$$\begin{array}{ll} \min & \|H - H_k\| \\ \text{s.t.,} & H = H^\top, \quad HY_k = S_k. \end{array}$$

where $\|A\| = \|(\nabla^2 f(x_k))^{1/2} A (\nabla^2 f(x_k))^{1/2}\|_F$ (F = Frobenius)
This gives the updating formula (analgous to the updates derived by Broyden, Fletcher, Goldfarb and Shanno, 1970).

$$H_{k+1} = (I - S_k[S_k^\top Y_k]^{-1} Y_k^\top) H_k (I - Y_k [S_k^\top Y_k]^{-1} S_k^\top) + S_k [S_k^\top Y_k]^{-1} S_k^\top$$

or, by the Sherman-Morrison-Woodbury formula:

$$B_{k+1} = B_k - B_k S_k [S_k^\top B_k S_k]^{-1} S_k^\top B_k + Y_k [S_k^\top Y_k]^{-1} Y_k^\top$$

Limited Memory Block BFGS

After M block BFGS steps starting from H_{k+1-M}, one can express H_{k+1} as

$$\begin{aligned} H_{k+1} &= V_k H_k V_k^T + S_k \Lambda_k S_k^T \\ &= V_k V_{k-1} H_{k-1} V_{k-1}^T V_k + V_k S_{k-1} \Lambda_{k-1} S_{k-1}^T V_k^T + S_k \Lambda_k S_k^T \\ &\vdots \\ &= V_{k:k+1-M} H_{k+1-M} V_{k:k+1-M}^T + \sum_{i=k}^{k+1-M} V_{k:i+1} S_i \Lambda_i S_i^T V_{k:i+1}^T, \end{aligned}$$

where

$$V_k = (I - S_k \Lambda_k Y_k^T) \tag{1}$$

and $\Lambda_k = (S_k^T Y_k)^{-1}$ and $V_{k:i} = V_k \cdots V_i$.

Limited Memory Block BFGS

- Hence, when the number of variables d is large, instead of storing the $d \times d$ matrix H_k, we store the previous M block curvature triples

$$(S_{k+1-M}, Y_{k+1-M}, \Lambda_{k+1-M}), \ldots, (S_k, Y_k, \Lambda_k).$$

- Then, analogously to the standard L-BFGS method, for any vector $v \in \mathbb{R}^d$, $H_k v$ can be computed efficiently using a two-loop block recursion (in $O(Mp(d+p)+p^3)$ operations), if all $S_i \in \mathbb{R}^{d \times p}$.

Intuition

- Limited memory - least change aspect of BFGS is important
- Each block update acts like a sketching procedure.

Two Loop Recursion

Algorithm 0.1: Two Loop Recursion

Input: $g_t \in \mathbb{R}^d$, S_i, $Y_i \in \mathbb{R}^{d \times q}$ and $\Lambda_i \in \mathbb{R}^{q \times q}$ for $i \in \{t+1-M, \dots, t\}$

initiate: $v = g_t$

for $i = t, \dots, t - M + 1$ **do**

  $\alpha_i = \Lambda_i S_i^\top v$

  $v = v - Y_i \alpha_i$

end

for $i = t - M + 1, \dots, t$ **do**

  $\beta_i = \Lambda_i Y_i^\top v$

  $v = v + S_i(\alpha_i - \beta_i)$

end

output: $H_t g_t = v$

Choices for the Sketching Matrix S_k

We employ one of the following strategies

- Gaussian: $S_k \sim \mathcal{N}(0, I)$ has Gaussian entries sampled i.i.d at each iteration.
- Previous search directions s_i delayed: Store the previous L search directions $S_k = [s_{k+1-L}, \ldots, s_k]$ then update H_k only once every L iterations.
- Self-conditioning: Sample the columns of the Cholesky factors L_k of H_k (i.e., $L_k L_k^T = H_k$) uniformly at random. Fortunately we can maintain and update L_k efficiently with limited memory.

The matrix S is a sketching matrix, in the sense that we are sketching the, possibly very large equation $\nabla^2 f(x) H = I$ to which the solution is the inverse Hessian. Right multiplying by S compresses/sketches the equation yielding $\nabla^2 f(x) HS = S$.

The Basic Algorithm

Algorithm 0.1: Stochastic Variable Metric Learning with SVRG

Input: $H_{-1} \in \mathbb{R}^{d \times d}$, $w_0 \in \mathbb{R}^d$, $\eta \in \mathbb{R}_+$, s = subsample size, q = sample action size and m

for $k = 0, \ldots, max\_iter$ **do**
  $\mu = \nabla f(w_k)$
  $x_0 = w_k$
  for $t = 0, \ldots, m-1$ **do**
    Sample $\mathcal{S}_t, \mathcal{T}_t \subseteq [n]$ i.i.d from a distribution $\mathcal{S}$
    Compute the sketching matrix $S_t \in \mathbb{R}^{d \times q}$
    Compute $\nabla^2 f_{\mathcal{S}}(x_t) S_t$
    $H_t =$update_metric$(H_{t-1}, S_t, \nabla^2 f_{\mathcal{T}}(x_t) S_t)$
    $d_t = -H_t \left(\nabla f_{\mathcal{S}}(x_t) - \nabla f_{\mathcal{S}}(w_k) + \mu\right)$
    $x_{t+1} = x_t + \eta d_t$
  end
  Option I: $w_{k+1} = x_m$
  Option II: $w_{k+1} = x_i$, i selected uniformly at random from $[m]$;
end

Convergence - Assumptions

There exist constants $\lambda, \Lambda \in \mathbb{R}_+$ such that

- f is λ–strongly convex

$$f(w) \geq f(x) + \nabla f(x)^T (w - x) + \frac{\lambda}{2} \|w - x\|_2^2 , \tag{2}$$

- f is Λ–smooth

$$f(w) \leq f(x) + \nabla f(x)^T (w - x) + \frac{\Lambda}{2} \|w - x\|_2^2 , \tag{3}$$

- These assumptions imply that

$$\lambda I \preceq \nabla^2 f_{\mathcal{S}}(w) \preceq \Lambda I, \quad \text{for all } x \in \mathbb{R}^d, \mathcal{S} \subseteq [n], \tag{4}$$

- from which we can prove that there exist constants $\gamma, \Gamma \in \mathbb{R}_+$ such that for all k we have

$$\gamma I \preceq H_k \preceq \Gamma I. \tag{5}$$

Bounds on Spectrum of H_k

Lemma

Assuming $\exists\, 0 < \lambda < \Lambda$ *such that*

$$\lambda I \preceq \nabla^2 f_T(x) \preceq \Lambda I$$

for all $x \in \mathbb{R}^d$ *and* $T \in [n]$,

$$\gamma I \preceq H_k \preceq \Gamma I$$

where

$\frac{1}{1+M\Lambda} \leq \gamma$, $\Gamma \leq (1+\sqrt{\kappa})^{2M}(1+\frac{1}{\lambda(2\sqrt{\kappa}+\kappa)})$ *and* $\kappa = \Lambda/\lambda$

- bounds in MNJ depend on problem dimension $\frac{1}{(d+M)\Lambda} \leq \gamma$ and $\Gamma \leq \frac{[(d+M)\Lambda]^{d+M-1}}{\lambda^{d+M}} \approx (d\kappa)^{d+M}$

Linear Convergence

Theorem

Suppose that the Assumptions hold. Let w_* be the unique minimizer of $f(w)$. Then in our Algorithm, we have for all $k \geq 0$ that

$$\mathbb{E}f(w_k) - f(w_*) \leq \rho^k \mathbb{E}f(w_0) - f(w_*),$$

where the convergence rate is given by

$$\rho = \frac{1/2m\eta + \eta\Gamma^2\Lambda(\Lambda - \lambda)}{\gamma\lambda - \eta\Gamma^2\Lambda^2} < 1,$$

assuming we have chosen $\eta < \gamma\lambda/(2\Gamma^2\Lambda^2)$ and that we choose m large enough to satisfy

$$m \geq \frac{1}{2\eta\left(\gamma\lambda - \eta\Gamma^2\Lambda(2\Lambda - \lambda)\right)},$$

which is a positive lower bound given our restriction on η.

Numerical Experiments

Empirical Risk Minimization Test Problems

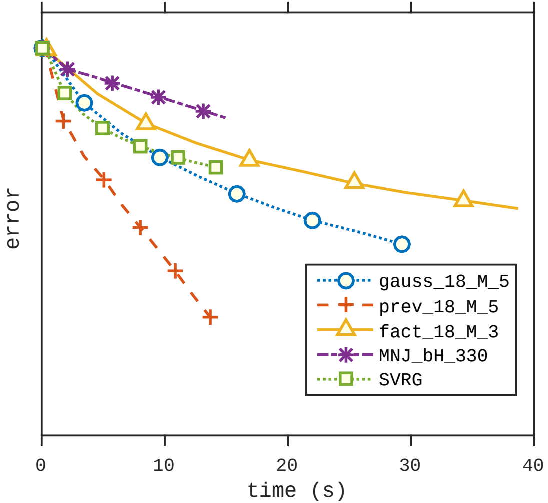
- logistic loss with l_2 regularizer

$$\min_w \sum_{i=1}^{n} \log(1 + \exp(-y_i \langle a^i, w \rangle)) + L\|w\|_2^2$$

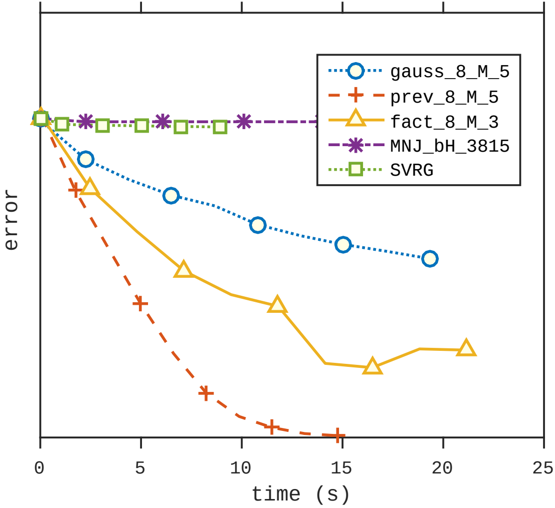
given data: $A = [a^1, a^2, \cdots, a^n] \in \mathbb{R}^{d \times n}$ $y \in \{0, 1\}^n$.

- For each method, chose step size $\eta \in \{1, .5, .1, .05, \ldots, 5 \times 10^{-8}, 10^{-8}\}$ that gave best results
- Computed full gradient after each full data pass.
- Vertical axis in figures below: log(relative error)

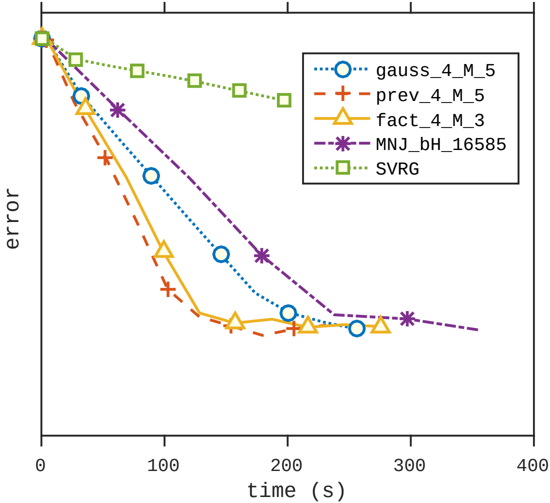
gisette-scale $d = 5,000$, $n = 6,000$



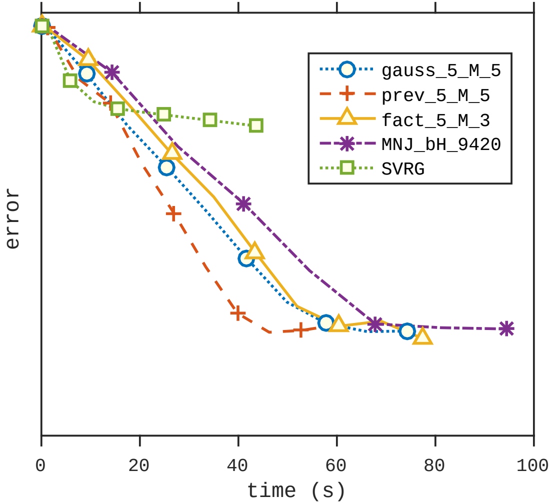
covtype-libsvm-binary $d = 54, n = 581,012$



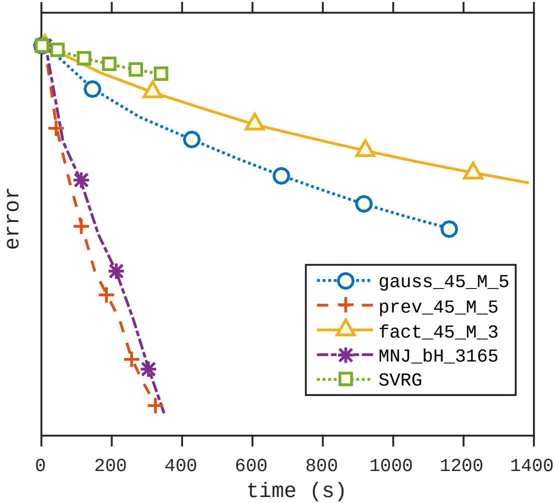
Higgs $d = 28, n = 11,000,000$



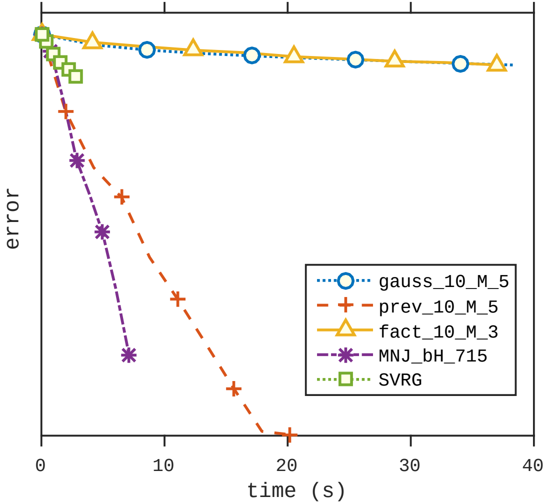
SUSY $d = 18, n = 3,548,466$



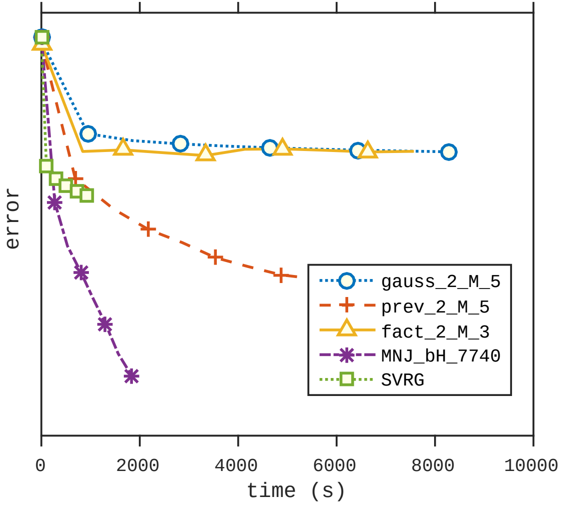
epsilon-normalized $d = 2,000$, $n = 400,000$



rcv1-training $d = 47,236$, $n = 20,242$



url-combined $d = 3,231,961, n = 2,396,130$



Contributions

- *New metric learning framework.* A block BFGS framework for gradually learning the metric of the underlying function using a sketched form of the subsampled Hessian matrix
- *New limited memory block BFGS method.* May also be of interest for non-stochastic optimization
- *Several sketching matrix possibilities.*
- *More reasonable bounds on eigenvalues of* H_k
  $\Rightarrow$ *more reasonable conditions for step size*

Nonconvex stochastic optimization

- Most stochastic quasi-Newton optimization methods are for strongly convex problems; this is needed to ensure a curvature condition required for the positive definiteness of B_k (H_k)
- This is not possible for problems $\min f(x) \equiv \mathbb{E}[F(x, \xi)]$, where f is nonconvex
- In deterministic setting, one can do line search to guarantee the curvature condition, and hence the positive definiteness of B_k (H_k)
- Line search is not possible for stochastic optimization
- To address these issues we develop a stochastic damped and a stochastic modified L-BFGS method.

Stochastic Damped BFGS (Wang, Ma, G, Liu, 2015)

- Let $y_k = \frac{1}{m}\sum_{i=1}^{m}(\nabla f(x_{k+1}, \xi_{k,i}) - \nabla f(x_k, \xi_{k,i}))$ and define
$$\bar{y}_k = \theta_k y_k + (1-\theta_k)B_k s_k,$$
where
$$\theta_k = \begin{cases} 1, & \text{if } s_k^\top y_k \geq 0.25 s_k^\top B_k s_k, \\ (0.75 s_k^\top B_k s_k)/(s_k^\top B_k s_k - s_k^\top y_k), & \text{if } s_k^\top y_k < 0.25 s_k^\top B_k s_k. \end{cases}$$
- Update H_k: (replace y_k by $\bar{y}_k$)
$$H_{k+1} = (I - \rho_k s_k \bar{y}_k^\top)H_k(I - \rho_k \bar{y}_k s_k^\top) + \rho_k s_k s_k^\top$$
where $\rho_k = 1/s_k^\top \bar{y}_k$
- Implemented in a limited memory version
- Work in progress: combine with variance reduced stochastic gradients (SVRG)

Convergence of Stochastic Damped BFGS Method

Assumptions

[AS1] $f \in C^1$, bounded below, ∇f is $L-$Lipschitz continuous

[AS2] For any iteration k, the stochastic gradient satisfies

$$\mathbb{E}_{\xi_k}[\nabla f(x_k, \xi_k)] = \nabla f(x_k)$$
$$\mathbb{E}_{\xi_k}[\|\nabla f(x_k, \xi_k) - \nabla f(x_k)\|^2] \leq \sigma^2$$

Theorem (Global convergence): Assume AS1-AS2 hold, (and $\alpha_k = \beta/k \leq \gamma/(L\Gamma^2)$ for all k), then there exist positive constants γ, Γ, such that $\gamma I \preceq H_k \preceq \Gamma I$, for all k, and

$$\liminf_{k\to\infty} \|\nabla f(x_k)\| = 0, \text{ with probability 1.}$$

- Under additional assumption $\mathbb{E}_{\xi_k}\left[\|\nabla f(x_k, \xi_k)\|^2\right] \leq M$

$$\lim_{k\to\infty} \|\nabla f(x_k)\| = 0, \quad \text{with probability 1.}$$

- We do not need to assume convexity of f

Block-L-BFGS Method for Non-Convex Stochastic Optimization

- Block-update

$$H_{k+1} = (I - S_k \Lambda_k^{-1} Y_k^\top) H_k (I - Y_k \Lambda_k^{-1} S_k^\top) + S_k \Lambda_k^{-1} S_k^\top$$

where $\Lambda_k = S_k^\top Y_k = S_k^\top \nabla^2 f(x_k) S_k$

- In non-convex case $\Lambda_k = \Lambda_k^\top$ may not be positive definite.
- $\Lambda_k \not\succeq 0$ discovered while computing Cholesky factorization $LDL^\top$ of Λ_k.
  If during Cholesky, $d_j \geq \delta$ or $|(LD^{1/2})_{ij}| \leq \beta$ are not satisfied, d_j is increased by τ_j.
  $\implies (\Lambda_k)_{jj} \leftarrow (\Lambda_k)_{jj} + \tau_j$
- has the effect of moving search direction $H_{k+1} \nabla f(x_{k+1})$ toward one of negative curvature.
- Modification based on Gershgorin disc also possible.