High Order Methods for Empirical Risk Minimization

Alejandro Ribeiro

Department of Electrical and Systems Engineering
University of Pennsylvania
aribeiro@seas.upenn.edu

IPAM Workshop of Emerging Wireless Systems

February 7th, 2017
Introduction

Incremental quasi-Newton algorithms

Adaptive sample size algorithms

Conclusions
Wireless channels characterized by random fading coefficients $h$

Want to assign power $p(h)$ as a function of fading to:

⇒ Satisfy prescribed constraints, e.g., average power, target SINRs
⇒ Optimize given criteria, e.g., maximize capacity, minimize power

Two challenges

⇒ Resultant optimization problems are infinite dimensional ($h$ is)
⇒ In most cases problems are not convex

However, duality gap is null under mild conditions (Ribeiro-Giannakis ’10)

And in the dual domain the problem is finite dimensional and convex

Motivate use of stochastic optimization algorithms that

⇒ Have manageable computational complexity per iteration
⇒ Use sample channel realizations instead of channel distributions
We aim to solve expected risk minimization problem \( \min_{w \in \mathbb{R}^p} \mathbb{E}_{\theta}[f(w, \theta)] \)

\( \Rightarrow \) The distribution is unknown

\( \Rightarrow \) We have access to \( N \) independent realizations of \( \theta \)

We settle for solving the Empirical Risk Minimization (ERM) problem

\[
\min_{w \in \mathbb{R}^p} F(w) := \min_{w \in \mathbb{R}^p} \frac{1}{N} \sum_{i=1}^{N} f(w, \theta_i)
\]

Large-scale optimization or machine learning: large \( N \), large \( p \)

\( \Rightarrow \) \( N \): number of observations (inputs)

\( \Rightarrow \) \( p \): number of parameters in the model

Not just wireless

\( \Rightarrow \) Many (most) machine learning algorithms reduce to ERM problems
Optimization methods

- **Stochastic methods**: a subset of samples is used at each iteration
  - SGD is the most popular; however, it is slow because of
    - Noise of stochasticity ➔ Variance reduction (SAG, SAGA, SVRG, ...)
    - Poor curvature approx. ➔ Stochastic QN (SGD-QN, RES, oLBFGS, ...)

- **Decentralized methods**: samples are distributed over multiple processors
  - Primal methods: DGD, Acc. DGD, NN, ...
  - Dual methods: DDA, DADMM, DQM, EXTRA, ESOM, ...

- **Adaptive sample size methods**: start with a subset of samples and increase the size of training set at each iteration ➔ Ada Newton
  - The solutions are close when the number of samples are close
Introduction

Incremental quasi-Newton algorithms

Adaptive sample size algorithms

Conclusions
Incremental Gradient Descent

Objective function gradients ⇒ \( s(w) := \nabla F(w) = \frac{1}{N} \sum_{i=1}^{N} \nabla f(w, \theta_i) \)

(Deterministic) gradient descent iteration ⇒ \( w_{t+1} = w_t - \epsilon_t s(w_t) \)

Evaluation of (deterministic) gradients is not computationally affordable

Incremental/Stochastic gradient ⇒ Sample average in lieu of expectations

\[ \hat{s}(w, \tilde{\theta}) = \frac{1}{L} \sum_{l=1}^{L} \nabla f(w, \theta_l) \quad \tilde{\theta} = [\theta_1; \ldots; \theta_L] \]

Functions are chosen cyclically or at random with or without replacement

Incremental gradient descent iteration ⇒ \( w_{t+1} = w_t - \epsilon_t \hat{s}(w_t, \tilde{\theta}_t) \)

(Incremental) gradient descent is (very) slow. Newton is impractical
BFGS quasi-Newton method

- Approximate function’s curvature with Hessian approximation matrix $B_t^{-1}$
  
  $$w_{t+1} = w_t - \epsilon_t B_t^{-1}s(w_t)$$

- Make $B_t$ close to $H(w_t) := \nabla^2 F(w_t)$. Broyden, DFP, BFGS

- Variable variation: $v_t = w_{t+1} - w_t$. Gradient variation: $r_t = s(w_{t+1}) - s(w_t)$

- Matrix $B_{t+1}$ satisfies secant condition $B_{t+1}v_t = r_t$. Underdetermined

- Resolve indeterminacy making $B_{t+1}$ closest to previous approximation $B_t$

- Using Gaussian relative entropy as proximity condition yields update

  $$B_{t+1} = B_t + \frac{r_tr_t^T}{v_t^Tr_t} - \frac{B_tv_tv_t^TB_t}{v_t^TB_tv_t}$$

- Superlinear convergence $\Rightarrow$ Close enough to quadratic rate of Newton

- BFGS requires gradients $\Rightarrow$ Use stochastic/incremental gradients
Stochastic/Incremental quasi-Newton methods

- Online (o)BFGS & Online Limited-Memory (oL)BFGS [Schraudolph et al '07]
- oBFGS may diverge because Hessian approximation gets close to singular
  ⇒ Regularized Stochastic BFGS (RES) [Mokhtari-Ribeiro '14]
- oLBFGS does (surprisingly) converge [Mokhtari-Ribeiro '15]
- Problem solved? Alas. RES and oLBFGS have sublinear convergence
- Same as stochastic gradient descent. Asymptotically not better
- Variance reduced stochastic L-BFGS (SVRG+oLBFGS) [Mortiz et al '16]
  ⇒ Linear convergence rate. But this is not better than SAG, SAGA, SVRG
- Computationally feasible quasi Newton method with superlinear convergence
Incremental aggregated gradient method

- Utilize memory to reduce variance of stochastic gradient approximation

\[
\nabla f_1^t \rightarrow \nabla f_{i_t}^t \rightarrow \nabla f_N^t \\
\downarrow \hspace{2cm} \downarrow \hspace{2cm} \downarrow \\
\nabla f_1^{t+1} \rightarrow \nabla f_{i_t}^{t+1} \rightarrow \nabla f_N^{t+1}
\]

- Descend along incremental gradient \( \Rightarrow w^{t+1} = w^t - \frac{\alpha}{N} \sum_{i=1}^{N} \nabla f_i^t = w^t - \alpha g_i^t \)

- Select update index \( i_t \) cyclically. Uniformly at random is similar

- Update gradient corresponding to function \( f_{i_t} \) \( \Rightarrow \nabla f_{i_t}^{t+1} = \nabla f_{i_t}^t(w^{t+1}) \)

- Sum easy to compute \( \Rightarrow g_i^{t+1} = g_i^t - \nabla f_{i_t}^{t+1} + \nabla f_{i_t}^{t+1} \). Converges linearly
Incremental BFGS method

- Keep memory of variables $z_i^t$, Hessian approximations $B_i^t$, and gradients $\nabla f_i^t$
  
  $\Rightarrow$ Functions indexed by $i$. Time indexed by $t$. Select function $f_{it}$ at time $t$

- All gradients, matrices, and variables used to update $w^{t+1}$
Incremental BFGS method

- Keep memory of variables $z^t_i$, Hessian approximations $B^t_i$, and gradients $\nabla f^t_i$.
  - Functions indexed by $i$. Time indexed by $t$. Select function $f_{it}$ at time $t$.

\[
\begin{align*}
  z^t_1 & \quad z^t_{it} & \quad z^t_N \\
  B^t_1 & \quad B^t_{it} & \quad B^t_N \\
  \nabla f^t_1 & \quad \nabla f^t_{it} & \quad \nabla f^t_N \\
  w^{t+1} & \quad \Rightarrow \quad \nabla f_{it}(w^{t+1})
\end{align*}
\]

- Updated variable $w^{t+1}$ used to update gradient $\nabla f^t_{it} = \nabla f_{it}(w^{t+1})$.
Incremental BFGS method

- Keep memory of variables $z_i^t$, Hessian approximations $B_i^t$, and gradients $\nabla f_i^t$

  ⇒ Functions indexed by $i$. Time indexed by $t$. Select function $f_{it}$ at time $t$

- Update $B_{it}^t$ to satisfy secant condition for function $f_{it}$ for variable variation $z_{it}^t - w_{t+1}^t$ and gradient variation $\nabla f_{it}^{t+1} - \nabla f_{it}^t$ (more later)
Incremental BFGS method

- Keep memory of variables $z^t_i$, Hessian approximations $B^t_i$, and gradients $\nabla f^t_i$.
  - Functions indexed by $i$. Time indexed by $t$. Select function $f^t_i$ at time $t$.

- Update variable, Hessian approximation, and gradient memory for function $f^t_i$. 

$$
\begin{align*}
  z^t_1 & \quad z^t_i & \quad z^t_N \\
  B^t_1 & \quad B^t_i & \quad B^t_N \\
  \nabla f^t_1 & \quad \nabla f^t_i & \quad \nabla f^t_N \\
  w^{t+1} & \quad B^{t+1}_i & \quad \nabla f^t_i (w^{t+1}) \\
  z^{t+1}_1 & \quad z^{t+1}_i & \quad z^{t+1}_N \\
  B^{t+1}_1 & \quad B^{t+1}_i & \quad B^{t+1}_N \\
  \nabla f^{t+1}_1 & \quad \nabla f^{t+1}_i & \quad \nabla f^{t+1}_N
\end{align*}
$$
Update of Hessian approximation matrices

- Variable variation at time $t$ for function $f_i = f_{i_t}$ \(\Rightarrow v_t^i := z_t^{i+1} - z_t^i\)
- Gradient variation at time $t$ for function $f_i = f_{i_t}$ \(\Rightarrow r_t^i := \nabla f_{i_t}^{i+1} - \nabla f_{i_t}^i\)
- Update $B_t^i = B_{i_t}^i$ to satisfy secant condition for variations $v_t^i$ and $r_t^i$

\[
B_{i+1} = B_i + \frac{r_t^i r_t^{iT}}{r_t^{iT}v_t^i} - \frac{B_i v_t^{iT}v_t^i}{v_t^{iT}B_i v_t^i} B_i
\]

- We want $B_t^i$ to approximate the Hessian of the function $f_i = f_{i_t}$
A naive (in hindsight) incremental BFGS method

- The key is in the update of $w^t$. Use memory in stochastic quantities

$$w^{t+1} = w^t - \left( \frac{1}{N} \sum_{i=1}^{N} B_i^t \right)^{-1} \left( \frac{1}{N} \sum_{i=1}^{N} \nabla f_i^t \right)$$

- It doesn’t work $\Rightarrow$ Better than incremental gradient but not superlinear

- Optimization updates are solutions of function approximations

- In this particular update we are minimizing the quadratic form

$$f(w) \approx \frac{1}{n} \sum_{i=1}^{n} \left[ f_i(z_i^t) + \nabla f_i(z_i^t)^T (w - w_t) + \frac{1}{2} (w - w^t)^T B_i^t (w - w^t) \right]$$

- Gradients evaluated at $z_i^t$. Secant condition verified at $z_i^t$

- The quadratic form is centered at $w^t$. Not a reasonable Taylor series
A proper Taylor series expansion

- Each individual function $f_i$ is being approximated by the quadratic

$$f_i(w) \approx f_i(z_t^i) + \nabla f_i(z_t^i)^T (w - w_t) + \frac{1}{2} (w - w_t)^T B_i^t (w - w_t)$$

- To have a proper expansion we have to recenter the quadratic form at $z_t^i$

$$f_i(w) \approx f_i(z_t^i) + \nabla f_i(z_t^i)^T (w - z_t^i) + \frac{1}{2} (w - z_t^i)^T B_i^t (w - z_t^i)$$

- I.e., we approximate $f(w)$ with the aggregate quadratic function

$$f(w) \approx \frac{1}{N} \sum_{i=1}^{N} \left[ f_i(z_t^i) + \nabla f_i(z_t^i)^T (w - z_t^i) + \frac{1}{2} (w - z_t^i)^T B_i^t (w - z_t^i) \right]$$

- This is now a reasonable Taylor series that we use to derive an update
Solving this quadratic program yields the update for the IQN method

\[ w^{t+1} = \left( \frac{1}{N} \sum_{i=1}^{N} B_i^t \right)^{-1} \left[ \frac{1}{N} \sum_{i=1}^{N} B_i^t z_i^t - \frac{1}{N} \sum_{i=1}^{N} \nabla f_i(z_i^t) \right] \]

Looks difficult to implement but it is more similar to BFGS than apparent.

As in BFGS, it can be implemented with \( O(p^2) \) operations

- Write as rank-2 update, use matrix inversion lemma
- Independently of \( N \). True incremental method.
The functions $f_i$ are $m$-strongly convex.

The gradients $\nabla f_i$ are $M$-Lipschitz continuous.

The Hessians $\nabla^2 f_i$ are $L$-Lipschitz continuous.

**Theorem**  The sequence of residuals $\|w^t - w^*\|$ in the IQN method converges to zero at a superlinear rate,

$$\lim_{t \to \infty} \frac{\|w^t - w^*\|}{(1/N)(\|w^{t-1} - w^*\| + \cdots + \|w^{t-N} - w^*\|)} = 0.$$

Incremental method with small cost per iteration converging at superlinear rate

$\Rightarrow$ Resulting from the use of memory to reduce stochastic variances
Numerical results

- Quadratic programming $f(w) := (1/N) \sum_{i=1}^{N} w^T A_i w/2 + b_i^T w$
- $A_i \in \mathbb{R}^{p \times p}$ is a diagonal positive definite matrix
- $b_i \in \mathbb{R}^p$ is a random vector from the box $[0, 10^3]^p$
- $N = 1000$, $p = 100$, and condition number $(10^2, 10^4)$
- Relative error $\|w^t - w^*\|/\|w^0 - w^*\|$ of SAG, SAGA, IAG, and IQN

![Diagram](a) small condition number (b) large condition number
Adaptive sample size algorithms

Introduction

Incremental quasi-Newton algorithms

Adaptive sample size algorithms

Conclusions
Back to the ERM problem

Our original goal was to solve the statistical loss problem

$$w^* := \arg\min_{w \in \mathbb{R}^p} L(w) = \arg\min_{w \in \mathbb{R}^p} \mathbb{E} [f(w, Z)]$$

But since the distribution of $Z$ is unknown we settle for the ERM problem

$$w^\dagger_N := \arg\min_{w \in \mathbb{R}^p} L_N(w) = \arg\min_{w \in \mathbb{R}^p} \frac{1}{N} \sum_{k=1}^{N} f(w, z_k)$$

Where the samples $z_k$ are drawn from a common distribution

ERM approximates actual problem  $\Rightarrow$ Don’t need perfect solution
From statistical learning we know that there exists a constant $V_N$ such that
\[
\sup_{\mathbf{w}} |L(\mathbf{w}) - L_N(\mathbf{w})| \leq V_N, \quad \text{w.h.p.}
\]

$V_N = O(1/\sqrt{N})$ from CLT. $V_N = O(1/N)$ sometimes [Bartlett et al ’06]

There is no need to minimize $L_N(\mathbf{w})$ beyond accuracy $O(V_N)$

This is well known. In fact, this is why we can add regularizers to ERM
\[
\mathbf{w}_N^* := \arg\min_{\mathbf{w}} R_N(\mathbf{w}) = \arg\min_{\mathbf{w}} L_N(\mathbf{w}) + \frac{cV_N}{2} \|\mathbf{w}\|^2
\]

Adding the term $(cV_N/2)\|\mathbf{w}\|^2$ “moves” the optimum of the ERM problem
But the optimum $\mathbf{w}_N^*$ is still in a ball of order $V_N$ around $\mathbf{w}^*$

Goal: Minimize the risk $R_N$ within its statistical accuracy $V_N$
Adaptive sample size methods

ERM problem $R_n^*$ for subset of $n \leq N$ unif. chosen samples

$$w_n^* := \arg\min_w R_n(w) = \arg\min_w L_n(w) + \frac{cV_n}{2} \|w\|^2$$

Solutions $w_m^*$ for $m$ samples and $w_n^*$ for $n$ samples are close

Find approx. solution $w_m$ for the risk $R_m$ with $m$ samples

Increase sample size to $n > m$ samples

Use $w_m$ as a warm start to find approx. solution $w_n$ for $R_n$

If $m < n$, it is easier to solve $R_m$ comparing to $R_n$ since

⇒ The condition number of $R_m$ is smaller than $R_n$
⇒ The required accuracy $V_m$ is larger than $V_n$
⇒ The computation cost of solving $R_m$ is lower than $R_n$
Ada Newton is a specific adaptive sample size method using Newton steps

- Find $w_m$ that solves $R_m$ to its statistical accuracy $V_m$
- Apply single Newton iteration $\Rightarrow w_n = w_m - \nabla^2 R_n(w_m)^{-1} \nabla R_n(w_m)$
- If $m$ and $n$ close, we have $w_n$ within statistical accuracy of $R_n$

This works if statistical accuracy ball of $R_m$ is within Newton quadratic convergence ball of $R_n$.

Then, $w_m$ is within Newton quadratic convergence ball of $R_n$

A single Newton iteration yields $w_n$ within statistical accuracy of $R_n$

Question: How should we choose $\alpha$?
Assumptions

- The functions $f(w, z)$ are convex
- The gradients $\nabla f(w, z)$ are $M$-Lipschitz continuous
  \[ \|\nabla f(w, z) - \nabla f(w', z)\| \leq M\|w - w'\|, \quad \text{for all } z. \]
- The functions $f(w, z)$ are self-concordant with respect to $w$ for all $z$
Theoretical result

**Theorem** Consider $w_m$ as a $V_m$-optimal solution of $R_m$, i.e.,

$$R_m(w_m) - R_m(w^*_m) \leq V_m,$$

and let $n = \alpha m$. If the inequalities

$$\left[ \frac{2(M+cV_m)V_m}{cV_n} \right]^{\frac{1}{2}} + \frac{2(n-m)}{nc^{\frac{1}{2}}} + \frac{(2+\sqrt{2})c^{\frac{1}{2}} + c\|w^*\|)(V_m-V_n)}{(cV_n)^{\frac{1}{2}}} \leq \frac{1}{4},$$

$$144 \left[ V_m + \frac{2(n-m)}{n} (V_{n-m} + V_m) + \frac{4 + c\|w^*\|^2}{2} (V_m - V_n) \right]^2 \leq V_n$$

are satisfied, then $w_n$ has sub-optimality error $V_n$ w.h.p., i.e.,

$$R_n(w_n) - R_n(w^*_n) \leq V_n, \quad \text{w.h.p.}$$

- **Condition 1** ⇒ $w_m$ is in the Newton quadratic convergence ball of $R_n$
- **Condition 2** ⇒ $w_n$ is in the statistical accuracy of $R_n$
- **Condition 2** becomes redundant for large $m$
Doubling the size of training set

**Proposition** Consider a learning problem in which the statistical accuracy satisfies $V_m \leq \alpha V_n$ for $n = \alpha m$ and $\lim_{n \to \infty} V_n = 0$. If $c$ is chosen so that

$$\left( \frac{2\alpha M}{c} \right)^{1/2} + \frac{2(\alpha - 1)}{\alpha c^{1/2}} \leq \frac{1}{4},$$

then, there exists a sample size $\tilde{m}$ such that the conditions in Theorem 1 are satisfied for all $m > \tilde{m}$ and $n = \alpha m$.

- We can double the size of training set $\alpha = 2$
  - If the size of training set is large enough
  - If the constant $c$ satisfies $c > 16(2\sqrt{M} + 1)^2$

- We achieve the S.A. of the full training set in about 2 passes over the data
  - After inversion of about $3.32 \log_{10} N$ Hessians
Adaptive sample size Newton (Ada Newton)

- **Parameters:** $\alpha_0 = 2$ and $0 < \beta < 1$.
- **Initialize:** $n = m_0$ and $w_n = w_{m_0}$ with $R_n(w_n) - R_n(w^*_n) \leq V_n$
- **while** $n \leq N$ **do**
  
  Update $w_m = w_n$ and $m = n$. Reset factor $\alpha = \alpha_0$

  **repeat** [sample size backtracking loop]

  1. Increase sample size: $n = \min\{\alpha m, N\}$.

  2. Comp. gradient: $\nabla R_n(w_m) = \frac{1}{n} \sum_{k=1}^{n} \nabla f(w_m, z_k) + cV_n w_m$

  3. Comp. Hessian: $\nabla^2 R_n(w_m) = \frac{1}{n} \sum_{k=1}^{n} \nabla^2 f(w_m, z_k) + cV_n I$

  4. Update the variable: $w_n = w_m - \nabla^2 R_n(w_m)^{-1} \nabla R_n(w_m)$

  5. Backtrack sample size increase $\alpha = \beta \alpha$.

  **until** $R_n(w_n) - R_n(w^*_n) \leq V_n$

- **end** while
Numerical results

- LR problem $\Rightarrow$ Protein homology dataset provided (KDD cup 2004)
- Number of samples $N = 145,751$, dimension $p = 74$
- Parameters $\Rightarrow V_n = 1/n$, $c = 20$, $m_0 = 124$, and $\alpha = 2$

- Ada Newton achieves the statistical accuracy of the full training set with about two passes over the dataset
Numerical results

- We use A9A and SUSY datasets to train a LR problem
  - A9A: $N = 32,561$ samples with dimension $p = 123$
  - SUSY: $N = 5,000,000$ samples with dimension $p = 18$

- The green line shows the iteration at which Ada Newton reached convergence on the test set

![Figure: Suboptimality vs No. of effective passes. A9A (left) and SUSY (right)](image)

- Ada Newton achieves the accuracy of $R_N(w) - R^*_N < 1/N$
  - by less than 2.3 passes over the full training set
Numerical results

- We use A9A and SUSY datasets to train a LR problem
  - A9A: \( N = 32,561 \) samples with dimension \( p = 123 \)
  - SUSY: \( N = 5,000,000 \) samples with dimension \( p = 18 \)

- The green line shows the iteration at which Ada Newton reached convergence on the test set

![Graphs showing suboptimality vs runtime for A9A and SUSY datasets.]

**Figure:** Suboptimality vs runtime. A9A (left) and SUSY (right)
We use A9A and SUSY datasets to train a LR problem
- A9A: \( N = 32,561 \) samples with dimension \( p = 123 \)
- SUSY: \( N = 5,000,000 \) samples with dimension \( p = 18 \)

Figure: Test error vs No. of effective passes. A9A (left) and SUSY (right)
Ada Newton and the challenges for Newton in ERM

There are four reasons why it is impractical to use Newton’s method in ERM

- It is costly to compute Hessians (and gradients). Order $O(Np^2)$ operations.
- It is costly to invert Hessians. Order $O(p^3)$ operations.
- A line search is needed to moderate stepsize outside of quadratic region.
- Quadratic convergence is advantageous close to the optimum but we don’t want to optimize beyond statistical accuracy.

Ada Newton (mostly) overcomes these four challenges

- Compute Hessians for a subset of samples. Two passes over dataset.
- Hessians are inverted in a logarithmic number of steps. But still.
- There is no line search.
- We enter quadratic regions without going beyond statistical accuracy.
Introduction

Incremental quasi-Newton algorithms

Adaptive sample size algorithms

Conclusions
Conclusions

- We studied different approaches to solve large-scale ERM problems.
- An incremental quasi-Newton BFGS method (IQN) was presented.
  - IQN only computes the information of a single function at each step.
    - Low computation cost
- IQN aggregates variable, gradient, and BFGS approximation.
  - Reduce the noise ⇒ Superlinear convergence
- Ada Newton resolves the Newton-type methods drawbacks.
  - Unit stepsize ⇒ No line search
  - Not sensitive to initial point ⇒ less Hessian inversions
  - Exploits quadratic convergence of Newton’s method at each iteration
- Ada Newton achieves statistical accuracy with about two passes over the data.


