AI505

Optimization

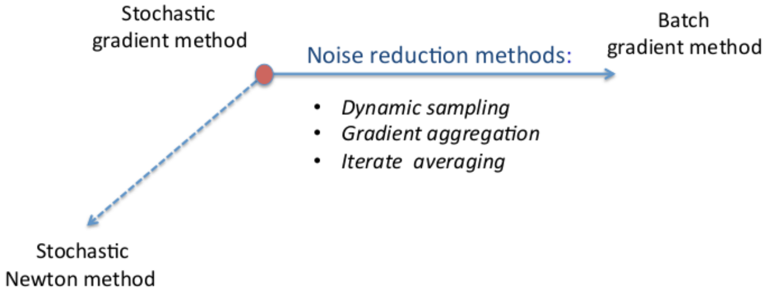
# Optimization in Machine Learning

Marco Chiarandini

Department of Mathematics & Computer Science
University of Southern Denmark

## Noise Reduction Methods

- SG as the ideal optimization approach for large-scale applications.
- SG suffers from the adverse effect of **noisy gradient estimates**.
  - when fixed stepsizes are used it prevents SG from converging to the solution
  - when a diminishing stepsize sequence $\{\alpha_k\}$ is employed it leads to a slow, sublinear rate of convergence.
- Remedies:



## Overview

Achieve linear rate of convergence to the optimal value using a fixed stepsize.

- **Dynamic sampling methods** achieve noise reduction by gradually increasing the mini-batch size used in the gradient computation, thus employing increasingly more accurate gradient estimates as the optimization process proceeds.

- **Gradient aggregation methods** improve the quality of the search directions by storing gradient estimates corresponding to samples employed in previous iterations, updating one (or some) of these estimates in each iteration, and defining the search direction as a weighted average of these estimates.

Rate of convergence remains sublinear but reduces variance of iterates

- **iterate averaging methods** maintain an average of iterates computed during the optimization process and employes a more aggressive stepsize sequence–of order $O(1/\sqrt{k})$ rather than $O(1/k)$.

## Reducing Noise at a Geometric Rate

rate of decrease in noise that allows a stochastic-gradient-type method to converge at a linear rate.

Consequence of Lipschitz assumption with $\ell$ constant:

$$\mathbb{E}_{\xi_k}[F(\boldsymbol{w}_{k+1})] - F(\boldsymbol{w}_k) \le -\alpha_k \nabla F(\boldsymbol{w}_k)^T \mathbb{E}_{\xi_k}[g(\boldsymbol{w}_k, \xi_k)] + \frac{1}{2}\alpha_k^2 \ell \, \mathbb{E}_{\xi_k}[\|g(\boldsymbol{w}_k, \xi_k)\|_2^2]$$

We want to make the left hand side small (sequence of expected optimality gaps).

<u>Theorem 5.1</u> (Strongly Convex Objective, Noise Reduction)
The SG method with a fixed stepsize $\bar{\alpha}$ and previous assumptions plus a variance of the stochastic vectors that decreases geometrically

$$\mathrm{Var}_{\xi_k}[g(\boldsymbol{w}_k, \xi_k)] \le M\zeta^{k-1}$$

has a sequence of expected optimality gaps that vanishes at a linear rate:

$$\mathbb{E}[F(\boldsymbol{w}_k) - F^*] \le \omega\rho^{k-1}$$

# Dynamic Sample Size Methods

Can we design efficient optimization methods attaining the critical bound on the variance?

Mini-batch stochastic gradient:

$$\boldsymbol{w}_{k+1} \leftarrow \boldsymbol{w}_k - \bar{\alpha} g(\boldsymbol{w}_k, \xi_k)$$

where the stochastic directions are computed for some $\tau > 1$ as

$$g(\boldsymbol{w}_k, \xi_k) \overset{def}{=} \frac{1}{n_k} \sum_{i \in \mathcal{S}_k} \nabla f(\boldsymbol{w}_k; \xi_{k,i}) \qquad \text{with } n_k \overset{def}{=} |\mathcal{S}_k| = \lceil \tau^{k-1} \rceil.$$

the mini-batch size increases geometrically as a function of the iteration counter $k$

Corollary 5.2. Let $\{\boldsymbol{w}_k\}$ be the iterates generated with unbiased gradient estimates, i.e., $\mathbb{E}_{\xi_{k,i}}[\nabla f(\boldsymbol{w}_k; \xi_{k,i})] = \nabla F(\boldsymbol{w}_k)$ for all $k \in \mathbb{N}$ and $i \in \mathcal{S}_k$. Then, the variance condition is satisfied, and if all other assumptions of Theorem 5.1 hold, then the expected optimality gap vanishes linearly.

## Dynamic Sample Size Methods

Note: we described a method as linearly convergent but the per-iteration cost increases without bound.

Recall that SG method needs $\mathcal{T}(n, \epsilon) \leq 1/\epsilon$ evaluations to gurantee $\mathbb{E}[F(\boldsymbol{w}_k) - F^*] \leq \epsilon$

Theorem 5.3 Suppose that the dynamic sampling SG method is run with a stepsize $\bar{\alpha}$ satisfying "some" bounds and some $\tau$. In addition, suppose that all previous Assumptions hold. Then, the total number of evaluations of a stochastic gradient of the form $\nabla f(\boldsymbol{w}_k; \xi_{k,i})$ required to obtain $\mathbb{E}[F(\boldsymbol{w}_k) - F^*] \leq \epsilon$ is $O(\epsilon^{-1})$.

## Dynamic Sample Size Guidelines

Given the rate of convergence of a batch optimization algorithm on strongly convex functions (i.e., linear, superlinear, etc.), what should be the sampling rate so that the overall algorithm is **efficient** in the sense that it results in the lowest computational complexity?

- if the optimization method has a sublinear rate of convergence, then there is no sampling rate that makes the algorithm "efficient";
- if the optimization algorithm is linearly convergent, then the sampling rate must be geometric (with restrictions on the constant in the rate) for the algorithm to be "efficient";
- for superlinearly convergent methods, increasing the sample size at a rate that is slightly faster than geometric will yield an "efficient" method.

## Design in Practice

- presetting the sampling rate, ie, $\tau > 1$ before running the optimization algorithm, requires some experimentation. Care must be put in preventing the full sample set from being employed too soon

- adaptive mechanisms to produce descent directions sufficiently often
  - any direction $g(\boldsymbol{w}_k, \xi_k)$ is a descent direction for $F$ at $\boldsymbol{w}_k$ if, for some $\chi \in [0, 1)$, one has

    $$\delta(\boldsymbol{w}_k, \xi_k) \overset{def}{=} \|g(\boldsymbol{w}_k, \xi_k) - \nabla F(wk)\|_2 \leq \chi \|g(\boldsymbol{w}_k, \xi_k)\|_2$$

    verifying the inequality may be costly because involves the evaluation of $\nabla F(\boldsymbol{w}_k)$, one can estimate the left-hand side $\delta(\boldsymbol{w}_k, \xi_k)$, and then choose $n_k$ so it holds sufficiently often.
  - The sample variance obtained by sampling without replacements is bounded above by $\chi^2 \|g(\boldsymbol{w}_k, \xi_\epsilon)\|_2^2$
  - If this condition is not satisfied, then increase the sample size to a size that one might predict would satisfy such a condition.
  - no guarantee that the size $n_k$ increases at a geometric rate. Remedy: if the adaptive increasesthe sampling rate more slowly than a preset geometric sequence, then a growth in the sample size is imposed.

## Gradient Aggregation

- Rather than compute increasingly more **new** stochastic gradient information in each iteration, achieve a lower variance by **reusing and/or revising** previously computed information
- achieve a linear rate of convergence on strongly convex problems.
- improved rate is achieved primarily by either an increase in computation or an increase in storage.
- works on finite sums like $R_n$

## SVRG

```
Procedure SVRG ;                          # Methods for Minimizing an Empirical Risk $R_n$
Choose an initial iterate $\boldsymbol{w}_1 \in \mathbb{R}^d$, stepsize $\alpha > 0$ and a positive integer $m$;
for $k = 1, 2, \ldots$ do
    Compute the batch gradient $\nabla R_n(\boldsymbol{w}_k)$;
    Initialize $\tilde{\boldsymbol{w}}_1 \leftarrow \boldsymbol{w}_k$;
    for $j = 1, \ldots, m$ do
        $\tilde{\boldsymbol{g}}_j \leftarrow \nabla f_{i_j}(\tilde{\boldsymbol{w}}_j) - (\nabla f_{i_j}(\boldsymbol{w}_k) - \nabla R_n(\boldsymbol{w}_k))$ ;    # $\nabla R_n(\boldsymbol{w}_k)$ from batch gradient
        $\tilde{\boldsymbol{w}}_{j+1} \leftarrow \tilde{\boldsymbol{w}}_j - \alpha \tilde{\boldsymbol{g}}_j$;
    Option (a): Set $\boldsymbol{w}_{k+1} = \tilde{\boldsymbol{w}}_{m+1}$;
    Option (b): Set $\boldsymbol{w}_{k+1} = \frac{1}{m} \sum_{j=1}^{m} \tilde{w}_{j+1}$;
    Option (c): Choose $j$ uniformly from $\{1, \ldots, m\}$ and set $\boldsymbol{w}_{k+1} = \tilde{\boldsymbol{w}}_{j+1}$;
```

- since $E_{i_j}[\nabla f_{i_j}(\boldsymbol{w}_k)] = \nabla R_n(\boldsymbol{w}_k)$, one can view $\nabla f_{i_j}(\boldsymbol{w}_k) - \nabla R_n(\boldsymbol{w}_k)$ as the bias in the gradient estimate $\nabla f_{i_j}(\boldsymbol{w}_k)$.
- sampled gradient $\nabla f_{i_j}(\tilde{\boldsymbol{w}}_j)$ is corrected based on a perceived bias. Overall, $\tilde{\boldsymbol{g}}_j$ represents an unbiased estimator of $\nabla R_n(\tilde{\boldsymbol{w}}_j)$, but with a variance that one can expect to be smaller than as in simple SG

## SAGA

in each iteration, it computes a stochastic vector $\boldsymbol{g}_k$ as the average of stochastic gradients evaluated at previous iterates.

```
Procedure SAGA ;                                    # Method for Minimizing an Empirical Risk $R_n$
Choose an initial iterate $\boldsymbol{w}_1 \in \mathbb{R}^d$ and stepsize $\alpha > 0$;
for $i = 1, \ldots, n$ do
    Compute $\nabla f_i(\boldsymbol{w}_1)$;
    Store $\nabla f_i(\boldsymbol{w}_{[i]}) \leftarrow \nabla f_i(\boldsymbol{w}_1)$ ;          # $\boldsymbol{w}_{[i]}$ represents the latest iterate at which $\nabla f_i$
for $k = 1, 2, \ldots$ do
    Choose $j$ uniformly in $\{1, \ldots, n\}$;
    Compute $\nabla f_j(\boldsymbol{w}_k)$;
    Set $\boldsymbol{g}_k \leftarrow \nabla f_j(\boldsymbol{w}_k) - \nabla f_j(\boldsymbol{w}_{[j]}) + \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\boldsymbol{w}_{[i]})$;
    Store $\nabla f_j(\boldsymbol{w}_{[j]}) \leftarrow \nabla f_j(\boldsymbol{w}_k)$;
    Set $\boldsymbol{w}_{k+1} \leftarrow \boldsymbol{w}_k - \alpha \boldsymbol{g}_k$;
```

As in SVRG, the method employs unbiased gradient estimates, but with variances that are expected to be less than the stochastic gradients that would be employed in a basic SG routine

## SAGA

- Same per-iteration costs as basic SG
- on strongly convex $R_n$ can achieve a linear rate of convergence but needs knowledge of at least $\ell$.
- More effective initialization instead of evaluating all the gradients $\{\nabla f_i\}_{i=1}^n$ at the initial point. For example, one could perform one epoch of simple SG, or one can assimilate iterates one-by-one and compute $\boldsymbol{g}_k$ only using the gradients available up to that point.
- SAGA needs to store $n$ stochastic gradient vectors
- for very large n, gradient aggregation methods are comparable to batch algorithms and therefore cannot beat SG in this regime

## Iterated Averaging Methods

- for minimizing a continuously differentiable $F$ with unbiased gradient estimates, the idea is to employ the iteration:

$$
\begin{aligned}
\boldsymbol{w}_{k+1} &\leftarrow \boldsymbol{w}_k - \alpha g(\boldsymbol{w}_k, \xi_k) \\
\tilde{\boldsymbol{w}}_{k+1} &\leftarrow \frac{1}{k+1} \sum_{j=1}^{k+1} \boldsymbol{w}_j
\end{aligned}
$$

  where $\{\tilde{\boldsymbol{w}}_k\}$ has no effect on the computation of the SG iterate sequence $\{\boldsymbol{w}_k\}$

- with stepsizes diminishing at a slow rate of $\mathcal{O}(1/(k^a))$ for some $a \in (\frac{1}{2}, 1)$ on strongly convex objectives, yields that $\mathbb{E}[\|\boldsymbol{w}_k - w^*\|_2^2] = \mathcal{O}(1/(k^a))$ while $\mathbb{E}[\|\tilde{\boldsymbol{w}}_k - w^*\|_2^2] = \mathcal{O}(1/k)$.

- in certain cases this combination of long steps and averaging yields an optimal constant in $\mathbb{E}[\|\tilde{\boldsymbol{w}}_k - w^*\|_2^2]$ in the sense that no rescaling of the steps—through multiplication with a positive definite matrix (second order methods) can improve the asymptotic rate or constant.

## Second Order Methods

- Address the adverse effects of high nonlinearity and ill-conditioning of the objective function through the use of second-order information.
- improve convergence rates of batch methods or the constants involved in the sublinear convergence rate of stochastic methods
- First-order methods are **not scale invariant**. Consider:
  $F$ continously differentiable function $F : \mathbb{R}^d \to \mathbb{R}$

  $$\boldsymbol{w}_{k+1} \leftarrow \boldsymbol{w}_k - \alpha_k \nabla F(\boldsymbol{w}_k)$$

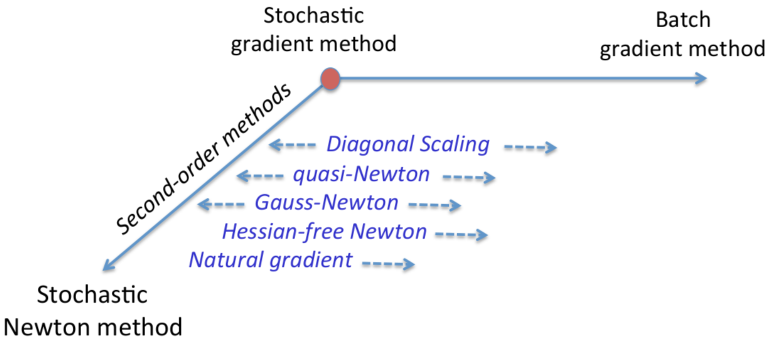
  linear transformation of the variables $\{\boldsymbol{w}_k\} = \{B\tilde{\boldsymbol{w}}_k\}$. $\min_{\tilde{\boldsymbol{w}}} F(B\tilde{\boldsymbol{w}}_k)$

  $$\tilde{\boldsymbol{w}}_{k+1} \leftarrow \tilde{\boldsymbol{w}}_k - \alpha_k B\nabla F(B\tilde{\boldsymbol{w}}_k) \qquad \Longrightarrow \qquad \boldsymbol{w}_{k+1} \leftarrow \boldsymbol{w}_k - \alpha_k B^2 \nabla F(\boldsymbol{w}_k)$$

  They will perform differently. With $\alpha = 1$ and $B = (\nabla^2 F(\boldsymbol{w}_1))^{-1/2}$ we get Newton's method

- Newton's method achieves a quadratic rate of convergence if $w_1$ is sufficiently close to a strong minimizer. On the other hand, stochastic methods like the SG method cannot achieve a convergence rate that is faster than sublinear, regardless of the choice of $B$.

- careful use of successive re-scalings based on (approximate) second-order derivatives can be beneficial between the stochastic and batch regimes.



## Hessian-Free Inexact Newton Methods

- Newton's method:

  $$\boldsymbol{w}_{k+1} \leftarrow \boldsymbol{w}_k + \alpha_k \boldsymbol{s}_k$$

  where $\boldsymbol{s}_k$ satisfies $\nabla^2 F(\boldsymbol{w}_k)\boldsymbol{s}_k = -\nabla F(\boldsymbol{w}_k)$.

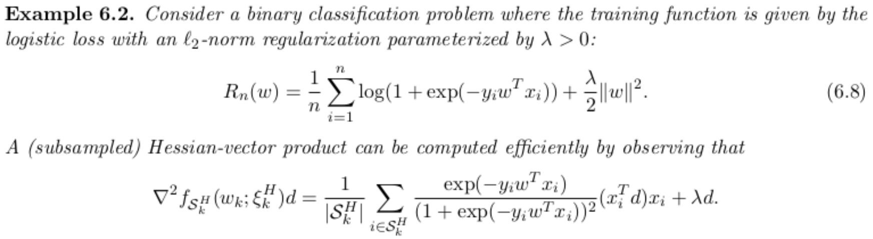
- one can solve the linear system inexactly through an iterative approach such as the conjugate gradient (CG) method.

- By ensuring that the linear solves are accurate enough, such an inexact Newton-CG method can enjoy a superlinear rate of convergence

- For a smooth objective function $F$, one can compute $\nabla^2 F(\boldsymbol{w})\boldsymbol{d}$ at a cost that is a small multiple of the cost of evaluating $\nabla F$, and without forming the Hessian, which would require $\mathcal{O}(d^2)$ storage

- exploit structure of risk measures

- iterations are more tolerant to noise in the hessian estimate than it is to noise in the gradient estimate
- employs a smaller, conditionally (given $\boldsymbol{w}_k$) uncorrelated, sample for defining the Hessian than for the stochastic gradient estimate
- can be combined with a backtracking (Armijo) line search or trust region
- (subsampled) Hessian-vector products can be computed efficiently in ML tasks



**Example 6.2.** *Consider a binary classification problem where the training function is given by the logistic loss with an $\ell_2$-norm regularization parameterized by $\lambda > 0$:*

$$R_n(w) = \frac{1}{n}\sum_{i=1}^{n} \log(1 + \exp(-y_i w^T x_i)) + \frac{\lambda}{2}\|w\|^2. \tag{6.8}$$

*A (subsampled) Hessian-vector product can be computed efficiently by observing that*

$$\nabla^2 f_{\mathcal{S}_k^H}(w_k; \xi_k^H)d = \frac{1}{|\mathcal{S}_k^H|}\sum_{i\in\mathcal{S}_k^H} \frac{\exp(-y_i w^T x_i)}{(1+\exp(-y_i w^T x_i))^2}(x_i^T d)x_i + \lambda d.$$

- Stochastic Quasi-Newton Methods:
  Like BFGS

- Gauss-Newton Methods
  constructs an approximation to the Hessian using only first-order information, and this approximation is guaranteed to be positive semidefinite, even when the full Hessian itself may be indefinite.
  The price to pay for this convenient representation is that it ignores second-order interactions between elements of the parameter vector $\boldsymbol{w}$, which might mean a loss of curvature information that could be useful for the optimization process.

## Summary

- Ways to cope with the problems in machine learning
- SG might not be the best choice for parallelization
- How about other methods like CMA-ES?