AI505

Optimization

Optimization in Machine Learning

Marco Chiarandini

Department of Mathematics & Computer Science
University of Southern Denmark

Simplified Notation

Let ξ be a random seed or the realization of a single (or a set of) sample $(\boldsymbol{x}, y)$.
For a given $(\boldsymbol{w}, \xi)$ let $f(\boldsymbol{w}; \xi)$ be the composition of the loss function L and the prediction function h
Then:

$$R(\boldsymbol{w}) = \mathbb{E}_{\xi}[f(\boldsymbol{w}; \xi)] \qquad \text{Expected Risk}$$

Let $\{\xi_{[i]}\}_{i=1}^{n}$ be realizations of ξ corresponding to $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n}$ and $f_i(\boldsymbol{w}) \stackrel{def}{=} f(\boldsymbol{w}; \xi_{[i]})$
Then:

$$R_n(\boldsymbol{w}) \stackrel{def}{=} \frac{1}{n} \sum_{i=1}^{n} f_i(\boldsymbol{w}) \qquad \text{Empirical Risk}$$

Stochastic vs Batch Optimization Methods

Reduction to minimizing R_n, with $\boldsymbol{w}_0 \in \mathbb{R}^d$ given (deterministic problem)

Stochastic Approach: Stochastic Gradient (Robbins and Monro, 1951)

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

i_k is chosen randomly from $\{1, \ldots, n\}$, $\alpha_k > 0$.

- very cheap iteration only on one sample.
- $\{\boldsymbol{w}_k\}$ is a stochastic process determined by the random sequence $\{i_k\}$.
- the direction might not always be a descent but if it is a descent direction in **expectation**, then the sequence $\{\boldsymbol{w}_k\}$ can be guided toward a minimizer of R_n.

Batch Approach: batch gradient, steepest descent, full gradient method:

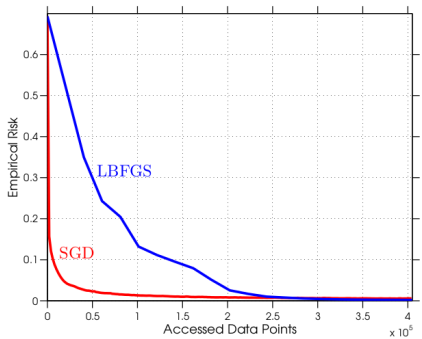
$$\boldsymbol{w}_{k+1} \leftarrow \boldsymbol{w}_k - \alpha_k \nabla R_n(\boldsymbol{w}_k) = \boldsymbol{w}_k - \frac{\alpha_k}{n} \sum_{i=1}^{n} \nabla f_i(\boldsymbol{w}_k)$$

- more expensive
- can use all deterministic gradient-based optimization methods
- the sum structure opens up to parallelization

Analogues in simulation: stochastic approximation (SA) and sample average approximation (SAA)

Stochastic Gradient

- In case of redundancy using all the sample data in every iteration is inefficient
- Comparison of the performance of a batch L-BFGS method on number of evaluations of a sample gradient $\nabla f_{i_k}(\boldsymbol{w}_k)$.
- Each set of n consecutive accesses is called an **epoch**.
- The batch method performs only one step per epoch while SG performs n steps per epoch.



the fast initial improvement achieved by SG, followed by a drastic slowdown after 1 or 2 epochs, is common in practice

SG more sensitive to α_k and starting point

if more epochs, batch may become better

Rate of Convergence

Let $\{x_k\}$ be a sequence in $\mathbb{R}^n$ that converges to x^*.
The convergence is said to be **Q-linear** (quotient-linear) if there is a constant $r \in (0, 1)$ such that

$$\frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} \leq r \qquad \text{for all } k \text{ sufficiently large}$$

ie, the distance to the solution x^* decreases at each iteration by at least a constant factor bounded away from 1 (ie, < 1).

Example:
sequence $\{1 + (0.5)^k\}$ converges Q-linearly to 1, with rate $r = 0.5$.

Rate of Convergence

The convergence is said to be **Q-superlinear** if

$$\lim_{k\to\infty} \frac{\|\boldsymbol{x}_{k+1} - \boldsymbol{x}^*\|}{\|\boldsymbol{x}_k - \boldsymbol{x}^*\|} = 0$$

Example: the sequence $\{1 + k^{-k}\}$ converges superlinearly to 1.

An even more rapid convergence rate:
The convergence is said to be **Q-quadratic** if

$$\frac{\|\boldsymbol{x}_{k+1} - \boldsymbol{x}^*\|}{\|\boldsymbol{x}_k - \boldsymbol{x}^*\|^2} \leq M \qquad \text{for all } k \text{ sufficiently large}$$

where M is a positive constant, not necessarily less than 1.
Example: the sequence $\{1 + (0.5)^{2^k}\}$.
The values of r and M depend not only on the algorithm but also on the properties of the particular problem.
Regardless of these values a quadratically convergent sequence will always eventually converge faster than a linearly convergent sequence.

Rate of Convergence

Superlinear convergence (quadratic, cubic, quartic, etc) is regarded as fast and desirable, while sublinear convergence is usually impractical.

- Quasi-Newton methods for unconstrained optimization typically converge Q-superlinearly
- Newton's method converges Q-quadratically under appropriate assumptions.
- Steepest descent algorithms converge only at a Q-linear rate, and when the problem is ill-conditioned the convergence constant r in is close to 1.

Rate of Convergence

A slightly weaker form of convergence:
overall rate of decrease in the error, rather than the decrease over each individual step of the algorithm.
We say that convergence is **R-linear** (root-linear) if there is a sequence of nonnegative scalars $\{v_k\}$ such that

$$\|x_k - x^*\| \leq \{v_k\} \text{ for all } k, \text{ and } \{v_k\} \text{ converges Q-linearly to zero.}$$

Theoretical Motivations

- a batch approach can minimize R_n at a fast rate; e.g., if R_n is strongly convex. A batch gradient method, then there exists a constant $\rho \in (0,1)$ such that, for all $k \in \mathbb{N}$, the training error follows **linear convergence**

$$R_n(\boldsymbol{w}_k) - R_n^* \leq \mathcal{O}(\rho^k),$$

- rate of convergence of a basic stochastic method is slower than for a batch gradient; e.g., if R_n is strictly convex and each i_k is drawn uniformly from $\{1, \ldots, n\}$, then for all $k \in \mathbb{N}$, SG satisfies the **sublinear convergence property**

$$\mathbb{E}[R_n(\boldsymbol{w}_k) - R_n^*] = \mathcal{O}(1/k).$$

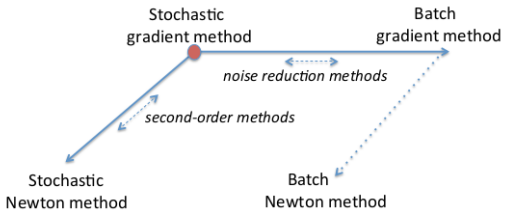
neither the per-iteration cost nor the right-hand side depends on the sample set size n

- in a stochastic optimization setting, SG yields for the expected risk the same convergence rate once substituted $\nabla f_{i_k}(\boldsymbol{w}_k)$ replaced by $\nabla f(\boldsymbol{w}_k; \xi_k)$ with each ξ_k drawn independently according to the distribution P

$$\mathbb{E}[R(\boldsymbol{w}_k) - R^*] = \mathcal{O}(1/k).$$

If $n \gg k$ up to iteration k minimizing R_n same as minimizing R

Beyond SG: Noise Reduction and Second-Order Methods



- on horizontal axis methods that try to improve rate of convergence
- on vertical axis, methods that try to overcome non-linearity and ill-conditioning

Mini-batch Approach small subset of samples, call it $\mathcal{S}_k \subseteq \{1, \ldots, n\}$, chosen randomly in each iteration:

$$\boldsymbol{w}_{k+1} \leftarrow \boldsymbol{w}_k - \frac{\alpha_k}{|\mathcal{S}_k|} \sum_{i \in \mathcal{S}} \nabla f_i(\boldsymbol{w}_k)$$

due to the reduced variance of the stochastic gradient estimates, the method is easier to tune in terms of choosing the stepsizes $\{\alpha_k\}$.
dynamic sample size and gradient aggregation methods, both of which aim to improve the rate of convergence from sublinear to linear

Outline

Theoretical Analysis — Preliminaries

convergence properties and worst-case iteration complexity bounds.

$$F(\boldsymbol{w}) = \begin{cases} R_n(\boldsymbol{w}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\boldsymbol{w}) & \text{Empirical Risk} \\ R(\boldsymbol{w}) = \mathbb{E}_{\xi}[f(\boldsymbol{w};\xi)] & \text{Expected Risk} \end{cases}$$

sampling uniformly with replacement from training set $\rightsquigarrow R_n$
sampling with $P(\xi)$ with replacement from training set $\rightsquigarrow R$.

Procedure SG(...);
Choose an initial iterate $\boldsymbol{w}_0$;
for $k = 0, 1, \ldots$ **do**
  Generate a realization of the random variable xi_k;
  Compute a stochastic vector $g(\boldsymbol{w}_k, \xi_k)$;
  Choose a stepsize $\alpha_k > 0$;
  Set the new iterate as $\boldsymbol{w}_{k+1} \leftarrow \boldsymbol{w}_k - \alpha_k g(\boldsymbol{w}_k, \xi_k)$;

Theoretical Analysis — Preliminaries

ξ_k may represent a single sample or a mini-batch
g may represent a stochastic gradient (biased estimator of $\nabla F(\boldsymbol{w}_k)$ or a stochastic Netwon or quasi-Newton direction).

$$g(\boldsymbol{w}_k, \xi_k) = \begin{cases} \nabla f(\boldsymbol{w}_k; \xi_k) \\ \frac{1}{n_k} \sum_{i=1}^{n_k} \nabla f(\boldsymbol{w}_k; \xi_{k,i}) \\ H_k \frac{1}{n_k} \sum_{i=1}^{n_k} \nabla f(\boldsymbol{w}_k; \xi_{k,i}) \end{cases}$$

H_k a symmetric positive definite scaling matrix
α_k fixed stepsize or diminishing stepsizes
$\boldsymbol{w}_k$ can have influence on the sample selection (active learning)

Convergence Analysis – Assumptions

- Assumption 4.1 Lipschitz-continuous objective gradients
- Assumption 4.3 First and second moment limits. The objective function and SG (Algorithm 4.1) satisfy the following:
  - objective function to be bounded below by a scalar F_{inf} over the region explored by the algorithm.
  - in expectation, the vector $-g(\boldsymbol{w}_k, \xi_k)$ is a direction of sufficient descent for F from $\boldsymbol{w}_k$ with a norm comparable to the norm of the gradient
  - the variance of $g(\boldsymbol{w}_k, \xi_k)$ is restricted, but in a relatively minor manner.

$$\mathrm{Var}_{\xi_k}[g(\boldsymbol{w}_k, \xi_k)] \leq M + M_V \left\|\nabla F(\boldsymbol{w}_k)\right\|_2^2, \; M > 0, M_V > 0 \text{ for all } k \in \mathbb{N}$$

- Lemma: Markovian manner in the sense that $\boldsymbol{w}_{k+1}$ is a random variable that depends only on the iterate $\boldsymbol{w}_k$, the seed ξ_k, and the stepsize α_k and not on any past iterates.

Convergence Analysis – Assumptions

- Assumption 4.5 Strong convexity. The objective function $F : \mathbb{R}^d \to \mathbb{R}$ is **strongly convex** in that there exists a constant $c > 0$ such that

$$F(\bar{\boldsymbol{w}}) \geq F(\boldsymbol{w}) + \nabla F(\boldsymbol{w})^T(\bar{\boldsymbol{w}} - \boldsymbol{w}) + \frac{1}{2}c\,\|\bar{\boldsymbol{w}} - \boldsymbol{w}\|_2^2 \qquad \text{for all } (\bar{\boldsymbol{w}}, \boldsymbol{w}) \in \mathbb{R}^d \times \mathbb{R}^d$$

(grows at least quadratically, for univariate case: $f''(w) \geq c$, $c \geq 0$)
Hence, F has a unique minimizer, denoted as $\boldsymbol{w}^* \in \mathbb{R}^d$ with $F^* \stackrel{def}{=} F(w^*)$.

Convergence Analysis – Results

- Theorem 4.6 (Strongly Convex Objective, Fixed Stepsize).
- Theorem 4.7 (Strongly Convex Objective, Diminishing Stepsizes)
  - role of strong convexity
  - role of initial point
  - trade-offs of mini batches
- Theorem 4.8 (Nonconvex Objective, Fixed Stepsize)
  - While one cannot bound the expected optimality gap as in the convex case, inequality (4.28b) bounds the average norm of the gradient of the objective function observed on $\{\boldsymbol{w}_k\}$ visited during the first K iterations.
  - classical result for the full gradient method applied to nonconvex functions, namely, that the sum of squared gradients remains finite, implying that

$$\{\|\nabla F(\boldsymbol{w}_k)\|_2\} \to 0.$$

- Theorem 4.9 (Nonconvex Objective, Diminishing Stepsizes)
  - for the SG method with diminishing stepsizes, the expected gradient norms cannot stay bounded away from zero
  - the weighted average norm of the squared gradients converges to zero even if the gradients are noisy, (i.e., if $M > 0$ in the Variance upper bounding assumption) one can still conclude that the expected gradient norms cannot asymptotically stay far from zero.

Computational Complexity Analysis

- consider a big data scenario with an infinite supply of training examples, but a **limited computational time budget**. what type of algorithm — e.g., a simple SG or batch gradient method — would provide the best guarantees in terms of achieving a low expected risk?

- $\boldsymbol{w}^* \in \operatorname{argmin} R(\boldsymbol{w})$; $\boldsymbol{w}_n \in \operatorname{argmin} R_n(\boldsymbol{w})$, $\tilde{\boldsymbol{w}}_n$ approximate empirical risk minimizer returned by a given optimization algorithm at $\mathcal{T}_{max}$

- The tradeoffs associated with this scenario can be formalized as choosing the family of prediction functions $\mathcal{H}$, the number of examples n, and the optimization accuracy $\epsilon \stackrel{def}{=} E[R_n(\tilde{\boldsymbol{w}}_n) - R_n(\boldsymbol{w}_n)]$ in order to minimize the total error:

$$\underset{\mathcal{H}, n \in \mathbb{N}, \epsilon}{\text{minimize}} \; E[R(\tilde{\boldsymbol{w}}_n)] = \overbrace{R(\boldsymbol{w}^*)}^{\mathcal{E}_{app}(\mathcal{H})} + \overbrace{E[R(\boldsymbol{w}_n) - R(\boldsymbol{w}^*)]}^{\mathcal{E}_{est}(\mathcal{H}, n)} + \overbrace{E[R(\tilde{\boldsymbol{w}}_n) - R(\boldsymbol{w}_n)]}^{\mathcal{E}_{opt}(\mathcal{H}, n, \epsilon)}$$

$$\text{subject to } \mathcal{T}(n, \epsilon) \leq \mathcal{T}_{max}$$

Computational Complexity Analysis

- SG, with its sublinear rate of convergence, is more efficient for large-scale learning than (full, batch) gradient-based methods that have a linear rate of convergence.

- reducing the optimization error $\mathcal{E}_{opt}(H, n, \epsilon)$ (evaluated with respect to R rather than R_n) one might need to make up for the additional computing time by: (i) reducing the sample size n, potentially increasing the estimation error $\mathcal{E}_{est}(H, n)$; or (ii) simplifying the function family $\mathcal{H}$, potentially increasing the approximation error $\mathcal{E}_{app}(\mathcal{H})$.

Computational Complexity Analysis

Keep fixed $\mathcal{H}$ carrying out a worst-case analysis on the influence of the sample size n and optimization tolerance ϵ, which together only influence the estimation and optimization errors.

	Batch	Stochastic
$\mathcal{T}(n, \epsilon)$	$\sim n \log\left(\frac{1}{\epsilon}\right)$	$\frac{1}{\epsilon}$
$\mathcal{E}^*$	$\sim \frac{\log(\mathcal{T}_{\max})}{\mathcal{T}_{\max}} + \frac{1}{\mathcal{T}_{\max}}$	$\frac{1}{\mathcal{T}_{\max}}$

A stochastic optimization algorithm performs better that batch stochastic in terms of expected error

Large gap between asymptotical behavior and practical realities.

Remarks

- Fragility of the Asymptotic Performance of SG
  ok if objective function it includes a squared L_2-norm regularizer (related to constant c) but regularization parameter should be lowered when the number of samples increases.

- SG good for GPUs but ill-conditioning erodes efficiency of SG

- Distributed computing not working with basic SG because of too frequent updates of $\boldsymbol{w}$, more promising with mini-batch.

- Alternatives with Faster Convergence: minimizing empirical risk R_n there is information from previous gradients.
  - gradient aggregation methods
  - dynamic sampling approach