Optimization methods

Unconstrained convex optimization

Unconstrained convex optimization problem

\[ \min_{w \in \mathbb{R}^d} f(w) \]

(\(f\) is the convex objective function; feasible region is \(\mathbb{R}^d\).

Gradient descent for differentiable objectives

- Start with some initial \(w^{(1)} \in \mathbb{R}^d\).
- For \(t = 1, 2, \ldots \) until some stopping condition is satisfied.
  - Compute gradient of \(f\) at \(w^{(t)}\):
    \[ \lambda^{(t)} := \nabla f(w^{(t)}) \]
  - Update:
    \[ w^{(t+1)} := w^{(t)} - \eta_t \lambda^{(t)} \]

(\(\eta_t > 0\) are step sizes.)

Two issues

1. Objective function not necessarily differentiable everywhere.
2. Feasible region not necessarily \(\mathbb{R}^d\).

(Projected) (sub)gradient descent
Non-differentiability

Non-differentiable convex objectives
Some convex functions $f$ are not differentiable everywhere; gradient descent not even well-specified for these problems.

Example: hinge loss
\[
f(w) = \ell_{\text{hinge}}(w, 0; x, y) = \left[1 - y\langle w, x \rangle\right]_+.
\]

Not differentiable at $w \in \mathbb{R}^d$ where $y\langle w, x \rangle = 1$.

Subgradient descent

Subgradient descent for general convex objectives
- Start with some initial $w^{(1)} \in \mathbb{R}^d$.
- For $t = 1, 2, \ldots$ until some stopping condition is satisfied.
  - Compute any subgradient $\lambda^{(t)} \in \partial f(w^{(t)})$.
  - Update:
    \[
    w^{(t+1)} := w^{(t)} - \eta^t \lambda^{(t)}.
    \]
    ($\eta^t > 0$ are step sizes.)
Constrained convex optimization

\[
\min_{w \in \mathbb{R}^d} f(w), \quad f(w) := \frac{1}{2} \|w\|^2 + \frac{1}{|S|} \sum_{(x,y) \in S} \left[1 - y \langle w, x \rangle \right]_+
\]

Question: How do we compute a subgradient \( g \) of \( f \) at a given point \( w_0 \in \mathbb{R}^d \)?

\[
g = \lambda w_0 + \frac{1}{|S|} \sum_{(x,y) \in S} \begin{cases} 
0 & \text{if } 1 - y \langle w_0, x \rangle < 0; \\
- y x & \text{if } 1 - y \langle w_0, x \rangle > 0; \\
\text{any conv. comb. of above} & \text{if } 1 - y \langle w_0, x \rangle = 0.
\end{cases}
\]

With “continuous” data, exact equalities like \( 1 - y \langle w_0, x \rangle = 0 \) never come up.

Example: soft-margin (homogeneous) SVM

Subgradient descent algorithm for soft-margin SVM:

- Start with some initial \( w^{(1)} \in \mathbb{R}^d \).
- For \( t = 1, 2, \ldots \) until some stopping condition is satisfied.

\[
w^{(t+1)} := w^{(t)} - \eta_t \left( \lambda w^{(t)} + \frac{1}{|S|} \sum_{(x,y) \in S} \begin{cases} 0 & \text{if } 1 - y \langle w^{(t)}, x \rangle < 0; \\
y x & \text{if } 1 - y \langle w^{(t)}, x \rangle \geq 0
\end{cases} \right)
\]

Note effect of regularization term \( \frac{1}{2} \|w\|^2 \) (whenever \( \eta_t < 1/\lambda \)):
Shrink \( w^{(t)} \) by a factor \( 1 - \lambda \eta_t \) before updating with subgradient of loss term.
Helps prevent \( w^{(t)} \) from becoming too long.

Example: box constraints

Suppose you want to prevent any feature from having a high weight:
- Constraints:
  \[ |w_i| \leq 1 \quad \text{for all } i = 1, 2, \ldots, d. \]
- Feasible region:
  \[ A := [-1,+1]^d \]
- \( \text{Proj}_A(w) \):
  - For each \( i = 1, 2, \ldots, d \):
    - If \( w_i \in [-1,+1] \), leave \( w_i \) alone.
    - If \( w_i > +1 \), set \( w_i := 1 \).
    - If \( w_i < -1 \), set \( w_i := -1 \).
Convergence and optimality conditions

Convergence of (projected) subgradient descent
In general, subgradient descent (with possibly non-differentiable convex objectives) can converge relatively slowly.
(Can help to return the average of the iterates $\frac{1}{T} \sum_{t=1}^{T} w(t)$.)

Optimality condition for general convex optimization problems also considerably more involved (e.g., more than just “$\nabla f_0(w) = 0$”).

But for machine learning purposes, we can use alternative criteria for stopping (e.g., hold-out error rate).

Minimizing sums of convex functions

In machine learning, we typically have objectives of the following form:

$$F(w) := \frac{1}{n} \sum_{i=1}^{n} \phi_i(w)$$

for convex functions $\phi_i : \mathbb{R}^d \to \mathbb{R}$ for $i = 1, 2, \ldots, n$.

Example: $\phi_i$ is loss on $i$-th training example.

Computational cost of computing $\nabla F(w)$: linear in $n$. (Typically $O(nd)$).

Stochastic gradient method

Key idea
Let $I$ be a uniform random variable in $\{1, 2, \ldots, n\}$.
Then for any $w$,

$$F(w) = \frac{1}{n} \sum_{i=1}^{n} \phi_i(w) = \mathbb{E}[\phi_I(w)]$$

and by linearity,

$$\nabla F(w) = \nabla \left\{ \frac{1}{n} \sum_{i=1}^{n} \phi_i(w) \right\} = \frac{1}{n} \sum_{i=1}^{n} \nabla \phi_i(w) = \mathbb{E}[\nabla \phi_I(w)].$$

Upshot: $\nabla \phi_I(w)$ is an unbiased estimate of gradient of $F$ at $w$.

Computational cost of computing $\nabla \phi_I(w)$: independent of $n$. (E.g., $O(d)$).
Stochastic gradient method

Consider convex functions $\phi_i : \mathbb{R}^d \to \mathbb{R}$, $i = 1, 2, \ldots, n$:

$$
\min_{w \in \mathbb{R}^d} F(w), \quad F(w) := \frac{1}{n} \sum_{i=1}^{n} \phi_i(w).
$$

Stochastic (projected) (sub)gradient “descent” method

- Start with some initial $w^{(1)} \in A$.
- For $t = 1, 2, \ldots$ until some stopping condition is satisfied.
  - Pick $I_t$ uniformly at random from $\{1, 2, \ldots, n\}$.
  - Compute any subgradient $\lambda^{(t)} \in \partial \phi_{I_t}(w^{(t)})$.
  - Update:
    $$
w^{(t+0.5)} := w^{(t)} - \eta_t \lambda^{(t)}.
    $$
  - Project to feasible region $A$:
    $$
w^{(t+1)} := \text{Proj}_A(w^{(t+0.5)}).
    $$

**Example: soft-margin (homogeneous) SVMs**

$$
\min_{w \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \left( \lambda \|w\|^2 + \left[ 1 - y_i \langle w, x_i \rangle \right]_+ \right)
$$

Stochastic gradient method for soft-margin SVMs:

- Start with some initial $w^{(1)} \in \mathbb{R}^d$.
- For $t = 1, 2, \ldots$ until some stopping condition is satisfied.
  - Pick $I_t$ uniformly at random from $\{1, 2, \ldots, n\}$;
    $$
w^{(t+1)} := w^{(t)} - \eta_t \lambda^{(t)}.
    $$
  - Project to feasible region $A$:
    $$
w^{(t+1)} := \text{Proj}_A(w^{(t+0.5)}).
    $$

**Example: logistic regression**

Maximum likelihood estimation for logistic regression:

$$
\min_{w \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \ln \left( 1 + \exp(-y_i \langle w, x_i \rangle) \right).
$$

Stochastic gradient method for logistic regression:

- Start with some initial $w^{(1)} \in \mathbb{R}^d$.
- For $t = 1, 2, \ldots$ until some stopping condition is satisfied.
  - Pick $I_t$ uniformly at random from $\{1, 2, \ldots, n\}$;
    $$
w^{(t+1)} := w^{(t)} + \eta_t \left( 1 - P_{w(t)}(Y=y_{I_t} | X=x_{I_t}) \right) y_{I_t} x_{I_t}.
    $$

Why should this work?

Consider differentiable convex functions $\phi_i : \mathbb{R}^d \to \mathbb{R}$, $i = 1, 2, \ldots, n$:

$$
\min_{w \in \mathbb{R}^d} F(w), \quad F(w) := \frac{1}{n} \sum_{i=1}^{n} \phi_i(w).
$$

Then

$$
\lambda^{(t)} = \nabla \phi_{I_t}(w^{(t)}) = \nabla F(w^{(t)}) + \left( \nabla \phi_{I_t}(w^{(t)}) - \mathbb{E} \left[ \nabla \phi_{I_t}(w^{(t)}) \right] \right).
$$

“Noise” is “averaged away” when step sizes $\eta_t$ are small (e.g., $\eta_t \sim 1/\sqrt{t}$).
**Convergence theory**

Consider convex functions \( \phi_i : \mathbb{R}^d \to \mathbb{R} \), \( i = 1, 2, \ldots, n \);

\[
\min_{w \in A} F(w), \quad F(w) := \frac{1}{n} \sum_{i=1}^{n} \phi_i(w).
\]

Run stochastic gradient method for \( T \) iterations, get iterates \( w_1, w_2, \ldots, w_{T+1} \in A \).

Under some suitable conditions (e.g., on step sizes \( \eta_t \)),

\[
E[F(w_{T+1})] \leq \min_{w \in A} F(w) + O\left( \frac{\log T}{\sqrt{T}} \right).
\]

Let \( \bar{w} := \frac{1}{T+1} \sum_{t=1}^{T+1} w_t \) (average of the iterates). Under same conditions,

\[
E[F(\bar{w})] \leq \min_{w \in A} F(w) + O\left( \frac{1}{\sqrt{T}} \right).
\]

Often really helps in practice!

**Efficient implementation trick**

Special case:
- \( \phi_i(w) = \lambda \||w||_2^2 / 2 + \ell(y_i \langle w, x_i \rangle) \) for some convex function \( \ell : \mathbb{R} \to \mathbb{R} \).
- \( x_i \) has few non-zero entries (i.e., \( x_i \) is “sparse”).

Gradient of \( \phi_i \) at \( w \):

\[
\nabla \phi_i(w) = \lambda w + \ell'(y_i \langle w, x_i \rangle) y_i x_i.
\]

Update \( (I_t = i) \):

\[
w_{t+1} := (1 - \lambda \eta_t) w_t - \eta_t \ell'(y_i \langle w, x_i \rangle) y_i x_i.
\]

- Represent weight vector \( w_t \) as \( c_t v_t \) for scalar \( c_t \in \mathbb{R} \) and vector \( v_t \in \mathbb{R}^d \).
- Sparsity-respecting update:

\[
\begin{align*}
g_t & := \ell'(y_t c_t \langle v_t, x_i \rangle) \\
c_{t+1} & := (1 - \lambda \eta_t) c_t \\
v_{t+1} & := v_t - \frac{\eta_t g_t}{c_{t+1}} y_i x_i
\end{align*}
\]

**Other practical tips**

- Pick \( I_1, I_2, \ldots, I_t \) u.a.r. **without replacement** from \( \{1, 2, \ldots, n\} \).
  - I.e., randomly shuffle order of training data, then process in that order.
  - If possible, re-shuffle after each pass through training data.
- For debugging purposes, periodically check overall objective value.
  - (Should generally be improving, though not necessarily at every step.)
- Use hold-out error rate as a stopping criterion.
  - (Stop when hold-out error rate does not improve after a while.)
- Use hold-out set to tune step sizes.

**Other solvers**

Many other algorithms for solving convex optimization problems
- **Newton-Raphson**: use Hessian to pick better descent directions.
- **Quasi-Newton methods** (e.g., conjugate gradient, “BFGS”, “L-BFGS”): use efficient approximations of Hessians.
- Techniques for dealing with constraints:
  - **Barrier methods**: add penalties for constraint violations, slowly relax.
  - **Primal-dual methods**: start with dual-feasible point, iteratively improve until corresponding primal point is feasible.
  - \ldots

**But remember**: end goal in machine learning is **not** to minimize training error rate or training surrogate loss.
Key takeaways

1. Concept of subgradients, and generalizing gradient descent to non-differentiable objectives.
2. Subgradients for maximum of affine functions.
3. Extending gradient descent to constrained optimization problems.
4. Concept of stochastic gradients; stochastic gradient descent method; high-level idea of convergence theory.
5. Efficient implementation for sparse gradients.