CPSC 540: Machine Learning
Convergence of Gradient Descent

Mark Schmidt
University of British Columbia

Winter 2018
Admin

- **Auditting/registration forms:**
  - Submit them at end of class, pick them up end of next class.
  - I need your prereq form before I’ll sign registration forms.
  - I wrote comments on the back of some forms.

- **Office hours:** start today after class.

- **Assignment 1** due Friday.
  - 1 late day to hand in Monday, 2 late days for Wednesday.
  - Instructions to hand in assignment on Piazza.
  - If you don’t have a CS account, sign up ASAP:
    https://www.cs.ubc.ca/getacct
We discussed convex optimization problems.

- Off-the-shelf solvers are available for solving medium-sized convex problems.

We discussed ways to show functions are convex:

- Show that $f$ is below chord for any convex combination of points.
- $f$ is constructed from operations that preserve convexity.
  - Non-negative scaling, sum, max, composition with affine map.
- Show that $\nabla^2 f(w)$ is positive semi-definite for all $w$,
  \[ \nabla^2 f(w) \succeq 0 \text{ (zero matrix)} \]

Formally, the notation $A \succeq B$ means that for any vector $v$ we have

\[ v^T A v \geq v^T B v, \]

and this is called a “generalized inequality”.

- It defines an “ordering” among some matrices, but not all matrices can be compared.
We say that a $C^2$ function is strictly convex iff for all $w$ we have

$$\nabla^2 f(w) \succ 0,$$

meaning that the Hessian is positive definite everywhere.

Equivalent definitions of a positive definite matrix $A$:
1. The eigenvalues of $A$ are all positive.
2. $v^T A v > 0$ for all $v \neq 0$.

Why do we care about strict convexity?
- Positive-definite matrices are invertible, so $[\nabla^2 f(w)]^{-1}$ exists.
- There can be at most one global optimum (so it’s unique, if one exists).
**Strict Convexity and L2-Regularized Least Squares**

- In L2-regularized least squares, the Hessian matrix is
  \[ \nabla^2 f(w) = (X^\top X + \lambda I). \]

- This matrix is positive-definite,
  \[ v^\top (X^\top X + \lambda I)v = \|Xv\|^2 + \lambda \|v\|^2 > 0, \]
  which follows from properties of norms:
  - Both terms are non-negative because they’re norms.
  - Second term \( \|v\| \) is positive because \( v \neq 0 \) and \( \lambda > 0 \).

- This implies that:
  - The matrix \((X^\top X + \lambda I)\) is invertible.
  - The solution is unique.
Cost of L2-Regularized Least Squares

- Two strategies from 340 for L2-regularized least squares:
  1. Closed-form solution,
     \[ w = (X^T X + \lambda I)^{-1} (X^T y), \]
     which costs \(O(n d^2 + d^3)\).
     - This is fine for \(d = 5000\), but may be too slow for \(d = 1,000,000\).
  2. Run \(t\) iterations of gradient descent,
     \[ w^{k+1} = w^k - \alpha_k \left( X^T (X w^k - y) + \lambda w^k \right), \]
     \(\nabla f(w^k)\)
     which costs \(O(n d t)\).
     - I'm using \(t\) as total number of iterations, and \(k\) as iteration number.

- Gradient descent is faster if \(t\) is not too big:
  - If we only do \(t < \max\{d, d^2/n\}\) iterations.
Cost of Logistic Regression

- Gradient descent can also be applied to other models like **logistic regression**,

\[ f(w) = \sum_{i=1}^{n} \log(1 + \exp(-y^i w^T x^i)), \]

which we can’t formulate as a linear system or linear program.
- Setting \( \nabla f(w) = 0 \) gives a system of transcendental equations.

- But this objective function is **convex and differentiable**.
  - So gradient descent converges to a global optimum.

- Alternately, another common approach is **Newton’s method**.
  - Requires computing Hessian \( \nabla^2 f(w^k) \), and known as “IRLS” in statistics.
Digression: Logistic Regression Gradient and Hessian

- With some tedious manipulations, gradient for logistic regression is
  \[ \nabla f(w) = X^T r. \]

  where vector \( r \) has \( r_i = -y^i h(-y^i w^T x^i) \) and \( h \) is the sigmoid function.

- We know the gradient has this form from the multivariate chain rule.
  - Functions for the form \( f(Xw) \) always have \( \nabla f(w) = X^T r \) (see bonus slide).

- With some more tedious manipulations we get
  \[ \nabla^2 f(w) = X^T DX. \]

  where \( D \) is a diagonal matrix with \( d_{ii} = h(y^i w^T x^i)h(-y^i w^T x^i) \).
  - The \( f(Xw) \) structure leads to a \( X^T DX \) Hessian structure.
  - For other problems \( D \) may not be diagonal.
Cost of Logistic Regression

- Gradient descent costs $O(nd)$ per iteration to compute $Xw^k$ and $X^Tr^k$.
- Newton costs $O(nd^2 + d^3)$ per iteration to compute and invert $\nabla^2f(w^k)$.

- Newton typically requires substantially fewer iterations.

- But for datasets with very large $d$, gradient descent might be faster.
  - If $t < \max\{d, d^2/n\}$ then we should use the “slow” algorithm with fast iterations.

- So, how many iterations $t$ of gradient descent do we need?
Outline

1. Gradient Descent Progress Bound
2. Gradient Descent Convergence Rate
Gradient Descent for Finding a Local Minimum

- A typical gradient descent algorithm:
  - Start with some initial guess, $w^0$.
  - Generate new guess $w^1$ by moving in the negative gradient direction:
    \[
    w^1 = w^0 - \alpha_0 \nabla f(w^0),
    \]
    where $\alpha_0$ is the step size.
  - Repeat to successively refine the guess:
    \[
    w^{k+1} = w^k - \alpha_k \nabla f(w^k), \quad \text{for } k = 1, 2, 3, \ldots
    \]
    where we might use a different step-size $\alpha_k$ on each iteration.
  - Stop if $\|\nabla f(w^k)\| \leq \epsilon$.
    - In practice, you also stop if you detect that you aren’t making progress.
Gradient Descent in 2D

Starting "guess" $w^0$
Lipschitz Continuity of the Gradient

- Let’s first show a basic property:
  - If the step-size $\alpha_t$ is small enough, then gradient descent decreases $f$.

- We’ll analyze gradient descent assuming gradient of $f$ is Lipschitz continuous.
  - There exists an $L$ such that for all $w$ and $v$ we have
    \[ \|\nabla f(w) - \nabla f(v)\| \leq L\|w - v\|. \]
  - “Gradient can’t change arbitrarily fast”.

- This is a fairly weak assumption: it’s true in almost all ML models.
  - Least squares, logistic regression, neural networks with sigmoid activations, etc.
Lipschitz Continuity of the Gradient

- For $C^2$ functions, Lipschitz continuity of the gradient is equivalent to

$$\nabla^2 f(w) \preceq LI,$$

for all $w$.

- Equivalently: “singular values of the Hessian are bounded above by $L$”.
  - For least squares, minimum $L$ is the maximum eigenvalue of $X^TX$.

- This means we can bound quadratic forms involving the Hessian using

$$d^T \nabla^2 f(u) d \leq d^T (LI) d = Ld^T d = L\|d\|^2.$$
Descent Lemma

- For a $C^2$ function, a variation on the multivariate Taylor expansion is that

$$f(v) = f(w) + \nabla f(w)^T (v - w) + \frac{1}{2} (v - w)^T \nabla^2 f(u) (v - w),$$

for any $w$ and $v$ (with $u$ being some convex combination of $w$ and $v$).

- Lipschitz continuity implies the green term is at most $L \|v - w\|^2$,

$$f(v) \leq f(w) + \nabla f(w)^T (v - w) + \frac{L}{2} \|v - w\|^2,$$

which is called the descent lemma.

- The descent lemma also holds for $C^1$ functions (bonus slide).
The descent lemma gives us a convex quadratic upper bound on $f$:

$$f(x) + \nabla f(x)^T(y-x) + \frac{L}{2}\|y-x\|^2$$

This bound is minimized by a gradient descent step from $w$ with $\alpha_k = 1/L$. 
Gradient Descent decreases $f$ for $\alpha_k = 1/L$

- So let’s consider doing gradient descent with a step-size of $\alpha_k = 1/L$,

$$w^{k+1} = w^k - \frac{1}{L} \nabla f(w^k).$$

- If we substitute $w^{k+1}$ and $w^k$ into the descent lemma we get

$$f(w^{k+1}) \leq f(w^k) + \nabla f(w^k)^T (w^{k+1} - w^k) + \frac{L}{2} \|w^{k+1} - w^k\|^2.$$

- Now if we use that $(w^{k+1} - w^k) = -\frac{1}{L} \nabla f(w^k)$ in gradient descent,

$$f(w^{k+1}) \leq f(w^k) - \frac{1}{L} \nabla f(w^k)^T \nabla f(w^k) + \frac{L}{2} \|\frac{1}{L} \nabla f(w^k)\|^2$$

$$= f(w^k) - \frac{1}{L} \|\nabla f(w^k)\|^2 + \frac{1}{2L} \|\nabla f(w^k)\|^2$$

$$= f(w^k) - \frac{1}{2L} \|\nabla f(w^k)\|^2.$$
We’ve derived a \textit{bound on guaranteed progress} when using $\alpha_k = 1/L$.

$$f(w^{k+1}) \leq f(w^k) - \frac{1}{2L} \| \nabla f(w^k) \|^2.$$ 

- If gradient is non-zero, $\alpha_k = 1/L$ is guaranteed to decrease objective.
- Amount we decrease grows with the size of the gradient.
- Same argument shows that any $\alpha_k < 2/L$ will decrease $f$. 
Choosing the Step-Size in Practice

- In practice, you should never use $\alpha_k = 1/L$.
  - $L$ is usually expensive to compute, and this step-size is really small.
  - You only need a step-size this small in the worst case.

- One practical option is to approximate $L$:
  - Start with a small guess for $\hat{L}$ (like $\hat{L} = 1$).
  - Before you take your step, check if the progress bound is satisfied:
    \[
    f(w^k - (1/\hat{L})\nabla f(w^k)) \leq f(w^k) - \frac{1}{2\hat{L}} \|\nabla f(w^k)\|^2.
    \]
    - Double $\hat{L}$ if it’s not satisfied, and test the inequality again.
    - Worst case: eventually have $L \leq \hat{L} < 2L$ and you decrease $f$ at every iteration.
    - Good case: $\hat{L} << L$ and you are making way more progress than using $1/L$. 

Choosing the Step-Size in Practice

- An approach that usually works better is a backtracking line-search:
  - Start each iteration with a large step-size $\alpha$.
  - So even if we took small steps in the past, be optimistic that we’re not in worst case.
  - Decrease $\alpha$ until if Armijo condition is satisfied (this is what `findMin.jl` does),
    \[
    f(w^k - \alpha \nabla f(w^k)) \leq f(w^k) - \alpha \gamma \|\nabla f(w^k)\|^2 \quad \text{for} \quad \gamma \in (0, 1/2],
    \]
    
    \[
    \text{potential} \quad w^{k+1}
    \]
    
    often we choose $\gamma$ to be very small like $\gamma = 10^{-4}$.
    - We would rather take a small decrease instead of trying many $\alpha$ values.

- Good codes use clever tricks to initialize and decrease the $\alpha$ values.
  - Usually only try 1 value per iteration.
- Even more fancy line-search: Wolfe conditions (makes sure $\alpha$ is not too small).
  - Good reference on these tricks: Nocedal and Wright’s Numerical Optimization book.
Outline

1. Gradient Descent Progress Bound
2. Gradient Descent Convergence Rate
Convergence Rate of Gradient Descent

- In 340, we claimed that \( \nabla f(w^k) \) converges to zero as \( k \) goes to \( \infty \).
  - For convex functions, this means it converges to a global optimum.
  - However, we may not have \( \nabla f(w^k) = 0 \) for any finite \( k \).

- Instead, we’re usually happy with \( \|\nabla f(w^k)\| \leq \epsilon \) for some small \( \epsilon \).
  - Given an \( \epsilon \), how many iterations does it take for this to happen?

- We’ll first answer this question only assuming that
  1. Gradient \( \nabla f \) is Lipschitz continuous (as before).
  2. Step-size \( \alpha_k = 1/L \) (this is only to make things simpler).
  3. Function \( f \) can’t go below a certain value \( f^* \) (“bounded below”).

- Most ML objectives \( f \) are bounded below (like the squared error being at least 0).
  - We’re not assuming convexity (argument will work for any smooth problem).
Convergence Rate of Gradient Descent

Key ideas:
1. We start at some $f(w^0)$, and at each step we decrease $f$ by at least $\frac{1}{2L} \| \nabla f(w^k) \|^2$.
2. But we can't decrease $f(w^k)$ below $f^\ast$.
3. So $\| \nabla f(w^k) \|^2$ must be going to zero "fast enough".

Let's start with our guaranteed progress bound,

$$f(w^k) \leq f(w^{k-1}) - \frac{1}{2L} \| \nabla f(w^{k-1}) \|^2.$$

Since we want to bound $\| \nabla f(w^k) \|$, let's rearrange as

$$\| \nabla f(w^{k-1}) \|^2 \leq 2L(f(w^{k-1}) - f(w^k)).$$
Convergence Rate of Gradient Descent

- So for each iteration $k$, we have
  \[ \| \nabla f(w^{k-1}) \|^2 \leq 2L [f(w^{k-1}) - f(w^k)]. \]

- Let’s sum up the squared norms of all the gradients up to iteration $t$,
  \[ \sum_{k=1}^{t} \| \nabla f(w^{k-1}) \|^2 \leq 2L \sum_{k=1}^{t} [f(w^{k-1}) - f(w^k)]. \]

- Now we use two tricks:
  1. On the left, use that all $\| \nabla f(w^{k-1}) \|$ are at least as big as their minimum.
  2. On the right, use that this is a telescoping sum:
     \[
     \sum_{k=1}^{t} [f(w^{k-1}) - f(w^k)] = f(w^0) - \underbrace{f(w^1) + f(w^1)}_{0} - \underbrace{f(w^2) + f(w^2)}_{0} - \ldots - f(w^t)
     = f(w^0) - f(w^t).
     \]
Convergence Rate of Gradient Descent

- With these substitutions we have
  \[
  \sum_{k=1}^{t} \min_{j \in \{0, \ldots, t-1\}} \left\{ \| \nabla f(w^j) \|^2 \right\} \leq 2L[f(w^0) - f(w^t)].
  \]
  no dependence on \( k \)

- Now using that \( f(w^t) \geq f^* \) we get
  \[
  \min_{k \in \{0,1,\ldots,t-1\}} \left\{ \| \nabla f(w^k) \|^2 \right\} \leq 2L[f(w^0) - f^*],
  \]
  and finally that
  \[
  \min_{k \in \{0,1,\ldots,t-1\}} \left\{ \| \nabla f(w^k) \|^2 \right\} \leq \frac{2L[f(w^0) - f^*]}{t} = O(1/t),
  \]
  so if we run for \( t \) iterations, we'll find least one \( k \) with \( \| \nabla f(w^k) \|^2 = O(1/t). \)
Convergence Rate of Gradient Descent

- Our “error on iteration $t$” bound:
  \[
  \min_{k \in \{0, 1, \ldots, t-1\}} \left\{ \| \nabla f(w^k) \|^2 \right\} \leq \frac{2L[f(w^0) - f^*]}{t}.
  \]

- We want to know when the norm is below $\epsilon$, which is guaranteed if:
  \[
  \frac{2L[f(w^0) - f^*]}{t} \leq \epsilon.
  \]

- Solving for $t$ gives that this is guaranteed for every $t$ where
  \[
  t \geq \frac{2L[f(w^0) - f^*]}{\epsilon},
  \]

so gradient descent requires $t = O(1/\epsilon)$ iterations to achieve $\| \nabla f(w^k) \|^2 \leq \epsilon$. 
Summary

- **Gradient descent** can be suitable for solving high-dimensional problems.
- **Guaranteed progress bound** if gradient is Lipschitz, based on norm of gradient.
- **Practical step size strategies** based on the progress bound.
- **Error on iteration** \( t \) of \( O(1/t) \) for functions that are bounded below.
  - Implies that we need \( t = O(1/\epsilon) \) iterations to have \( \|\nabla f(x^k)\| \leq \epsilon \).

- Next time: didn’t I say that regularization makes gradient descent go faster?
Strictly-Convex Functions

- A function is strictly-convex if the convexity definitions hold strictly:

\[
\begin{align*}
    f(\theta w + (1 - \theta) v) &< \theta f(w) + (1 - \theta) f(v), \quad 0 < \theta < 1 \quad (C^0) \\
    f(v) &> f(w) + \nabla f(w)^\top (v - w) \quad (C^1) \\
    \nabla^2 f(w) &> 0 \quad (C^2)
\end{align*}
\]

- Function is always strictly below any chord, strictly above any tangent, and curved upwards in every direction.

- Strictly-convex function have at most one global minimum:
  - If \( w \) and \( v \) can’t both be global minima if \( w \neq v \):
    - it would imply convex combinations \( u \) of \( w \) and \( v \) would have \( f(u) \) below the global minimum.
Checking Derivative Code

- Gradient descent codes require you to **write objective/gradient code**.
  - This tends to be error-prone, although automatic differentiation codes are helping.

- Make sure to **check your derivative code**:
  - Numerical approximation to partial derivative:
    \[
    \nabla_if(x) \approx \frac{f(x + \delta e_i) - f(x)}{\delta}
    \]
    
  - For large-scale problems you can check a random direction \(d\):
    \[
    \nabla f(x)^T d \approx \frac{f(x + \delta d) - f(x)}{\delta}
    \]
    
  - If the left side coming from your code is very different from the right side, there is likely a bug.
Multivariate Chain Rule

- If \( g : \mathbb{R}^d \mapsto \mathbb{R}^n \) and \( f : \mathbb{R}^n \mapsto \mathbb{R} \), then \( h(x) = f(g(x)) \) has gradient

\[
\nabla h(x) = \nabla g(x)^T \nabla f(g(x)),
\]

where \( \nabla g(x) \) is the Jacobian (since \( g \) is multi-output).

- If \( g \) is an affine map \( x \mapsto Ax + b \) so that \( h(x) = f(Ax + b) \) then we obtain

\[
\nabla h(x) = A^T \nabla f(Ax + b).
\]

- Further, for the Hessian we have

\[
\nabla^2 h(x) = A^T \nabla^2 f(Ax + b) A.
\]
Convexity of Logistic Regression

- Logistic regression Hessian is
  \[ \nabla^2 f(w) = X^T DX. \]
  
  where \( D \) is a diagonal matrix with \( d_{ii} = h(y_i w^T x^i)h(-y_i w^T x^i) \).

- Since the sigmoid function is non-negative, we can compute \( D^{\frac{1}{2}} \), and
  \[ v^T X^T DX v = v^T X^T D^{\frac{1}{2}} D^{\frac{1}{2}} X v = (D^{\frac{1}{2}} X v)^T (D^{\frac{1}{2}} X v) = \|XD^{\frac{1}{2}}v\|^2 \geq 0, \]

  so \( X^T DX \) is positive semidefinite and logistic regression is convex.

  - It becomes strictly convex if you add L2-regularization, making solution unique.
Lipschitz Continuity of Logistic Regression Gradient

- Logistic regression Hessian is
  \[
  \nabla^2 f(w) = \sum_{i=1}^{n} \left[ h(y_i w^T x^i) h(-y_i w^T x^i) x^i (x^i)^T \right] d_{ii}
  \]
  \[
  \leq 0.25 \sum_{i=1}^{n} x^i (x^i)^T
  \]
  \[
  = 0.25 X^T X.
  \]

- In the second line we use that \( h(\alpha) \in (0, 1) \) and \( h(-\alpha) = 1 - \alpha \).
  - This means that \( d_{ii} \leq 0.25 \).

- So for logistic regression, we can take \( L = \frac{1}{4} \max\{\text{eig}(X^T X)\} \).
Why the gradient descent iteration?

- For a $C^2$ function, a variation on the multivariate Taylor expansion is that
  \[
  f(v) = f(w) + \nabla f(w)^T (v - w) + \frac{1}{2} (v - w)^T \nabla^2 f(u)(v - w),
  \]
  for any $w$ and $v$ (with $u$ being some convex combination of $w$ and $v$).

- If $w$ and $v$ are very close to each other, then we have
  \[
  f(v) = f(w) + \nabla f(w)^T (v - w) + O(\|v - w\|^2),
  \]
  and the last term becomes negligible.

- Ignoring the last term, for a fixed $\|v - w\|$ I can minimize $f(v)$ by choosing
  \[
  (v - w) \propto -\nabla f(w).
  \]
  So if we’re moving a small amount the optimal choice is gradient descent.
Descent Lemma for $C^1$ Functions

- Let $\nabla f$ be $L$-Lipschitz continuous, and define $g(\alpha) = f(x + \alpha z)$ for a scalar $\alpha$.

$$f(y) = f(x) + \int_0^1 \nabla f(x + \alpha(y - x))^T(y - x) d\alpha \quad \text{(fund. thm. calc.)}$$

$$(\pm \text{ const.}) = f(x) + \nabla f(x)^T(y - x) + \int_0^1 (\nabla f(x + \alpha(y - x)) - \nabla f(x))^T(y - x) d\alpha$$

$$(\text{CS ineq.}) \leq f(x) + \nabla f(x)^T(y - x) + \int_0^1 \|\nabla f(x + \alpha(y - x)) - \nabla f(x)\|\|y - x\| d\alpha$$

$$(\text{Lipschitz}) \leq f(x) + \nabla f(x)^T(y - x) + \int_0^1 L\|x + \alpha(y - x) - x\|\|y - x\| d\alpha$$

$$(\text{homog.}) = f(x) + \nabla f(x)^T(y - x) + \int_0^1 L\alpha\|y - x\|^2 d\alpha$$

$$\left(\int_0^1 \alpha = \frac{1}{2}\right) = f(x) + \nabla f(x)^T(y - x) + \frac{L}{2}\|y - x\|^2.$$
We said that Lipschitz continuity of the gradient
\[ \|\nabla f(w) - \nabla f(v)\| \leq L\|w - v\|, \]
is equivalent for $C^2$ functions to having
\[ \nabla^2 f(w) \preceq LI. \]

There are a lot of other equivalent definitions, see here:
- [http://xingyuzhou.org/blog/notes/Lipschitz-gradient](http://xingyuzhou.org/blog/notes/Lipschitz-gradient).