First-Order Optimization Algorithms for Machine Learning

Linear and Superlinear Convergence

Mark Schmidt

University of British Columbia

Summer 2020

Last Time: Convergence Rate of Gradient Descent

- We discussed the iteration complexity of an algorithm for a problem class:
  - "How many iterations t before we guarantee an accuracy ϵ"?
- We showed that gradient descent requires $t = O(1/\epsilon)$ iterations.
  - For functions that are bounded below and have a Lipschitz-continuous gradient.
- We discussed different types of rates of convergence:
  - Sublinear rates like error being $O(1/t)$ (need $O(1/\epsilon)$ iterations).
  - Linear rates like error being $O(\rho^t)$ (need $O(\log(1/\epsilon))$ iterations).
  - Superlinear rates like error being $O(\rho^{2^t})$ (need $O(\log\log(1/\epsilon))$ iterations).

Polyak-Łojasiewicz (PL) Inequality

- For least squares, we have linear cost but we only showed sublinear rate.
- For many "nice" functions f, gradient descent actually has a linear rate.
- For example, for functions satisfying the Polyak-Łojasiewicz (PL) inequality,

$$\frac{1}{2}\|\nabla f(w)\|^2 \geq \mu(f(w) - f^*),$$

  for all w and some $\mu > 0$.
    - "Gradient grows as a quadratic function as we increase f".

Linear Convergence under the PL Inequality

- Recall our guaranteed progress bound

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

- Under the PL inequality we have $-\|\nabla f(w^k)\|^2 \leq -2\mu(f(w^k) - f^*)$, so

$$f(w^{k+1}) \leq f(w^k) - \frac{\mu}{L}(f(w^k) - f^*).$$

- Let's subtract f^* from both sides,

$$f(w^{k+1}) - f^* \leq f(w^k) - f^* - \frac{\mu}{L}(f(w^k) - f^*),$$

and factorizing the right side gives

$$f(w^{k+1}) - f^* \leq \left(1 - \frac{\mu}{L}\right)(f(w^k) - f^*).$$

Linear Convergence under the PL Inequality

- Applying this recursively:

$$\begin{aligned} f(w^k) - f^* &\leq \left(1 - \frac{\mu}{L}\right)[f(w^{k-1}) - f(w^*)] \\ &\leq \left(1 - \frac{\mu}{L}\right)\left[\left(1 - \frac{\mu}{L}\right)[f(w^{k-2}) - f^*]\right] \\ &= \left(1 - \frac{\mu}{L}\right)^2 [f(w^{k-2}) - f^*] \\ &\leq \left(1 - \frac{\mu}{L}\right)^3 [f(w^{k-3}) - f^*] \\ &\leq \left(1 - \frac{\mu}{L}\right)^k [f(w^0) - f^*] \end{aligned}$$

- We'll always have $0 < \mu \leq L$ so we have $(1 - \mu/L) < 1$.
  - So PL implies a linear convergence rate: $f(w^k) - f^* = O(\rho^k)$ for $\rho < 1$.

Linear Convergence under the PL Inequality

- We've shown that

$$f(w^k) - f^* \leq \left(1 - \frac{\mu}{L}\right)^k [f(w^0) - f^*]$$

- By using the inequality that

$$(1 - \gamma) \leq \exp(-\gamma),$$

we have that

$$f(w^k) - f^* \leq \exp\left(-k\frac{\mu}{L}\right)[f(w^0) - f^*],$$

which is why linear convergence is sometimes called "exponential convergence".

- We'll have $f(w^t) - f^* \leq \epsilon$ for any t where

$$t \geq \frac{L}{\mu}\log((f(w^0) - f^*)/\epsilon) = O(\log(1/\epsilon)).$$

Discussion of Linear Convergence under the PL Inequality

- PL is satisfied for many standard convex models like least squares (bonus).
  - So cost of least squares is $O(nd \log(1/\epsilon))$.

- PL is also satisfied for some non-convex functions like $w^2 + 3\sin^2(w)$.
  - It's satisfied for PCA on a certain "Riemann manifold".
  - But it's not satisfied for many models, like neural networks.

- The PL constant μ might be terrible.
  - For least squares μ is the smallest non-zero eigenvalue of the Hessian

.

- It may be hard to show that a function satisfies PL.
  - But regularizing a convex function gives a PL function with non-trivial μ...

Strong Convexity

- We say that a function f is strongly convex if the function
$$f(w) - \frac{\mu}{2}\|w\|^2,$$
is a convex function for some $\mu > 0$.
  - "If you 'un-regularize' by μ then it's still convex."

- For C^2 functions this is equivalent to assuming that
$$\nabla^2 f(w) \succeq \mu I,$$
that the eigenvalues of the Hessian are at least μ everywhere.

- Some nice properties of strongly-convex functions (see bonus):
  - A unique solution exists.
  - C^1 strongly-convex functions satisfy the PL inequality.
  - If $g(w) = f(Aw)$ for strongly-convex f and matrix A, then g is PL (least squares).

Strong Convexity Implies PL Inequality

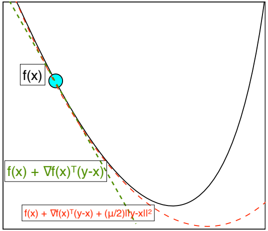
- As before, from Taylor's theorem we have for C^2 functions that

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

- By strong-convexity, $d^\top \nabla^2 f(u) d \geq \mu \|d\|^2$ for any d and u.

$$f(v) \geq f(w) + \nabla f(w)^\top (v - w) + \frac{\mu}{2}\|v - w\|^2$$

- Treating right side as function of v, we get a quadratic lower bound on f.



Strong Convexity Implies PL Inequality

- As before, from Taylor's theorem we have for C^2 functions that

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

- By strong-convexity, $d^\top \nabla^2 f(u) d \geq \mu \|d\|^2$ for any d and u.

$$f(v) \geq f(w) + \nabla f(w)^\top (v - w) + \frac{\mu}{2}\|v - w\|^2.$$

- Treating right side as function of v, we get a quadratic lower bound on f.

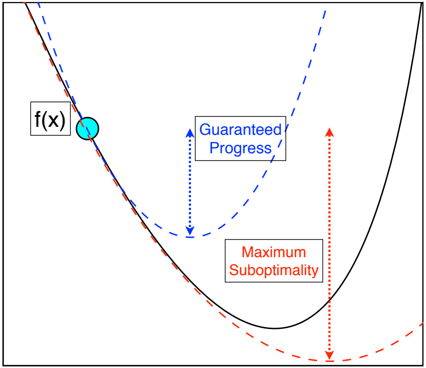
- Minimize both sides in terms of v gives

$$f^* \geq f(w) - \frac{1}{2\mu}\|\nabla f(w)\|^2,$$

which is the PL inequality (bonus slides show for C^1 functions).

Combining Lipschitz Continuity and Strong Convexity

- Lipschitz continuity of gradient gives guaranteed progress.
- Strong convexity of functions gives maximum sub-optimality.



- Progress on each iteration will be at least a fixed fraction of the sub-optimality.

Effect of Regularization on Convergence Rate

- We said that f is strongly convex if the function

$$f(w) - \frac{\mu}{2}\|w\|^2,$$

is a convex function for some $\mu > 0$.
    - For a C^2 univariate function, equivalent to $f''(w) \geq \mu$.

- If we have a convex loss f, adding L2-regularization makes it strongly-convex,

$$f(w) + \frac{\lambda}{2}\|w\|^2,$$

with μ being at least λ.

- So adding L2-regularization can improve rate from sublinear to linear.
    - Go from exponential $O(1/\epsilon)$ to polynomial $O(\log(1/\epsilon))$ iterations.
    - And guarantees a unique solution.

Effect of Regularization on Convergence Rate

- Our convergence rate under PL was

$$f(w^k) - f^* \leq \underbrace{\left(1 - \frac{\mu}{L}\right)^k}_{\rho^k} [f(w^0) - f^*].$$

- For L2-regularized least squares we have

$$\frac{L}{\mu} = \frac{\max\{\text{eig}(X^\top X)\} + \lambda}{\min\{\text{eig}(X^\top X)\} + \lambda}.$$

- So as λ gets larger ρ gets closer to 0 and we converge faster.

- The number $\frac{L}{\mu}$ is called the condition number of f.
  - For least squares, it's the "matrix condition number" of $\nabla^2 f(w)$.

Outline

Last Time: Iteration Complexity

- We discussed the iteration complexity of an algorithm for a problem class:
  - "How many iterations t before we guarantee an accuracy ϵ"?

- Iteration complexity of gradient descent when ∇f is Lipschitz continuous:

Assumption	Iteration Complexity	Quantity
Non-Convex	$t = O(1/\epsilon)$	$\min_{k=0,2,\dots,t-1} \|\nabla f(w^k)\|^2 \leq \epsilon$
Convex	$t = O(1/\epsilon)$	$f(w^t) - f^* \leq \epsilon$
Strongly-Convex	$t = O(\log(1/\epsilon))$	$f(w^t) - f^* \leq \epsilon$

- Adding L2-regularization to a convex function gives a strongly-convex function.
  - So L2-regularization can make gradient descent converge much faster.
- Can we go faster?

Nesterov Acceleration (Strongly-Convex Case)

- We showed that gradient descent for strongly-convex functions has

$$f(w^k) - f^* \leq \left(1 - \frac{\mu}{L}\right)^k [f(w^0) - f^*].$$

- Applying accelerated gradient methods to strongly-convex gives

$$f(w^k) - f^* \leq \left(1 - \sqrt{\frac{\mu}{L}}\right)^k [f(w^0) - f^*],$$

  which is a faster linear convergence rate
  $(\alpha_k = 1/L, \beta_k = (\sqrt{L} - \sqrt{\mu})/(\sqrt{L} + \sqrt{\mu}))$.

- This nearly acheives optimal possible dimension-independent rate.
  - For strictly-convex quadratics, conjugate gradient exactly achieves optimum possible.
  - There exist "restart" methods that converge slower but that don't need to know μ.

Newton's Method

- Newton's method is a second-order strategy.

(also called IRLS for functions of the form $f(Ax)$)

- Modern form uses the update

$$w^{k+1} = w^k - \alpha_k d^k,$$

where d^k is a solution to the system

$$\nabla^2 f(w^k) d^k = \nabla f(w^k).$$

(Assumes $\nabla^2 f(w^k) \succ 0$)

- Equivalent to minimizing the quadratic approximation:

$$f(v) \approx f(w^k) + \nabla f(w^k)^\top (v - w^k) + \frac{1}{2\alpha_k}(v - w^k)\nabla^2 f(w^k)(v - w^k).$$

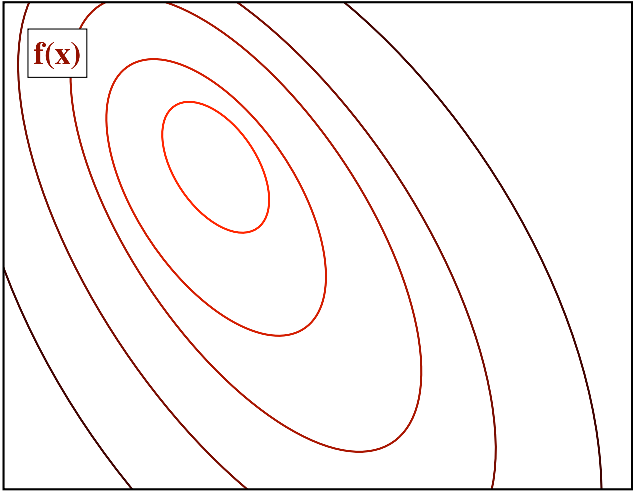
- To guarantee convergence, we can set the α_k using an Armijo condition:

$$f(w^{k+1}) \leq f(w^k) + \gamma \alpha_k \nabla f(w^k)^\top d^k.$$

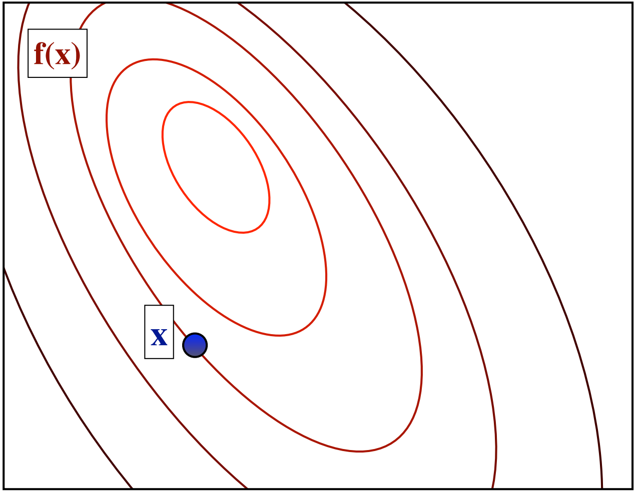
- From Taylor expansion, has a natural step length of $\alpha_k = 1$ if y and x^k are close.

($\alpha_k = 1$ is always accepted when close to a minimizer)

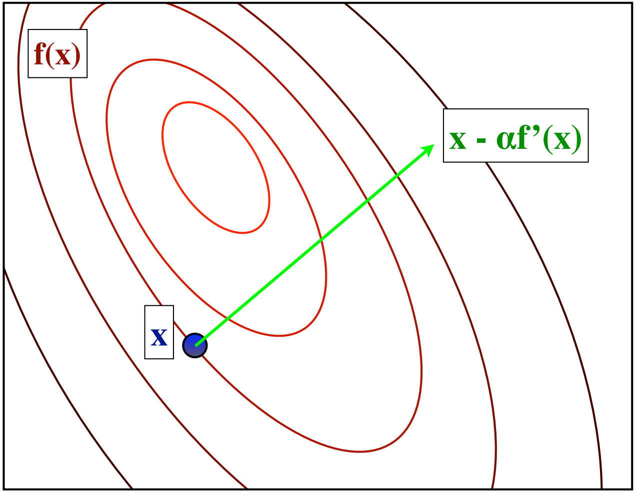
Newton's Method



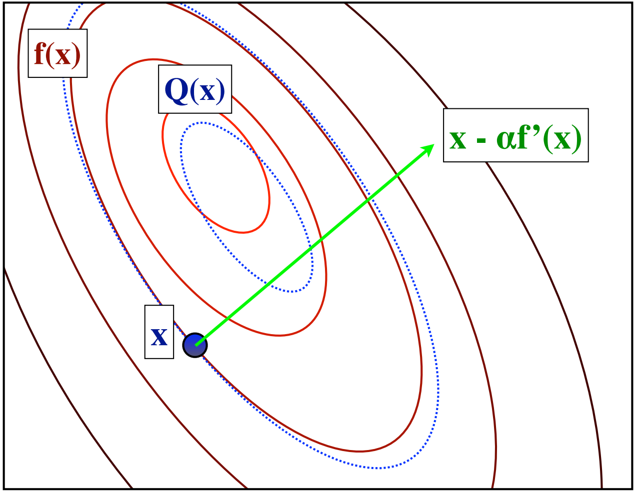
Newton's Method



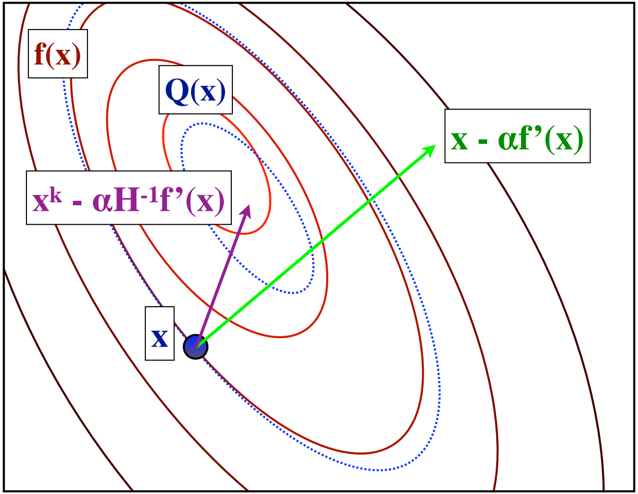
Newton's Method



Newton's Method



Newton's Method



Convergence Rate of Newton's Method

- If $\mu I \preceq \nabla^2 f(w) \preceq LI$ and $\nabla^2 f(x)$ is Lipschitz-continuous, then close to w^* Newton's method has local superlinear convergence:

$$f(w^{k+1}) - f(w^*) \leq \rho_k[f(w^k) - f(w^*)],$$

  with $\lim_{k\to\infty} \rho_k = 0$.
- Converges very fast, use it if you can!
- But Newton's method is expensive if dimension d is large:
  - Requires solving $\nabla^2 f(w^k)d^k = \nabla f(w^k)$.

Practical Approximations to Newton's Method

- Practical Newton-like methods (that can be applied to large-scale problems):
  - Diagonal approximation:
    - Approximate Hessian by a diagonal matrix D (cheap to store/invert).
    - A common choice is $d_{ii} = \nabla^2_{ii} f(w^k)$.
    - This sometimes helps, often doesn't.
  - Limited-memory quasi-Newton approximation:
    - Approximates Hessian by a diagonal plus low-rank approximation B^k,
    $$B^k = D + UV^k,$$
    which supports fast multiplication/inversion.
    - Based on "quasi-Newton" equations which use differences in gradient values.
    $$(\nabla f(w^k) - \nabla f(w^{k-1})) = B^\top (w^k - w^{k-1}).$$
    - A common choice is L-BFGS.

Practical Approximations to Newton's Method

- Practical Newton-like methods (that can be applied to large-scale problems):
  - Barzilai-Borwein approximation:
    - Approximates Hessian by the identity matrix (as in gradient descent).
    - But chooses step-size based on least squares solution to quasi-Newton equations.

$$\alpha_{k+1} = -\alpha_k \frac{v^k \nabla f(w^k)}{\|v^k\|^2}, \quad \text{where} \quad v^k = \nabla f(w^k) - \nabla f(w^{k-1}).$$

    - Works better than it deserves to (*findMin*).
    - Achieves superlinear convergence for strongly-convex quadratics for $d = 2$.
    - We don't understand why it works so well for $d > 2$ (challenging math problem).
    - For non-quadratic problems, often combined with non-monotonic Armijo line-search.

(Allows function to increase on some steps.)

Practical Approximations to Newton's Method

- Practical Newton-like methods (that can be applied to large-scale problems):
  - Hessian-free Newton:
    - Uses conjugate gradient to approximately solve Newton system ($\nabla^2 f(w^k)d = \nabla f(w^k)$).
    - Requires Hessian-vector products, but these cost same as gradient.
    - If you're lazy, you can numerically approximate them using

$$\nabla^2 f(w^k)d \approx \frac{\nabla f(w^k + \delta d) - \nabla f(w^k)}{\delta}.$$

    - If f is analytic, can compute exactly by evaluating gradient with complex numbers.

      (look up "complex-step derivative")
    - You can also use forward-mode automatic differentiation to get Hessian-vector products.
- A related appraoch to the above is non-linear conjugate gradient.

Numerical Comparison with minFunc

In my experience L-BFGS performs best for many problems.

- But for some problems Hessian-free Newton or non-linear CG are better.
- Barzilai-Borwein is a great choice if you have to implement from scratch.

Result after 25 evaluations of limited-memory solvers on 2D rosenbrock:

---

x1 = 0.0000, x2 = 0.0000 (starting point)
x1 = 1.0000, x2 = 1.0000 (optimal solution)

---

x1 = 0.3654, x2 = 0.1230 (minFunc with gradient descent)
x1 = 0.8756, x2 = 0.7661 (minFunc with Barzilai-Borwein)
x1 = 0.5840, x2 = 0.3169 (minFunc with Hessian-free Newton)
x1 = 0.7478, x2 = 0.5559 (minFunc with preconditioned Hessian-free Newton)
x1 = 1.0010, x2 = 1.0020 (minFunc with non-linear conjugate gradient)
x1 = 1.0000, x2 = 1.0000 (minFunc with limited-memory BFGS - default)

Superlinear Convergence in Practice?

- You get local superlinear convergence if:
  - Gradient is Lipschitz-continuous and f is strongly-convex.
  - Function is in $\mathcal{C}^2$ and Hessian is Lipschitz continuous.
  - Oracle is second-order and method asymptotically uses Newton's direction.
- But the practical Newton-like methods don't achieve this:
  - Diagonal scaling, Barzilai-Borwein, and L-BFGS don't converge to Newton.
  - Hessian-free uses conjugate gradient which isn't superlinear in high-dimensions.
  - These methods usually outperform Nesterov's accelerated method in practice.
- Full quasi-Newton methods achieve this, but require $\Omega(d^2)$ memory/time.

Cubic Regularization of Newton's Method

- Gradient descent ($\alpha_k = 1/L$) uses upper-bound on second-order term,

$$w^{k+1} \in \operatorname*{argmin}_{w} \left\{ f(w) + \nabla f(w)^T (w - w^k) + \frac{L}{2}\|w - w^k\|^2 \right\}.$$

- Cubic regularization of Newton's method upper bounds third-order term,

$$w^{k+1} \in \operatorname*{argmin}_{w} \left\{ f(w) + \nabla f(w)^T (w - w^k) + \frac{1}{2}(w - w^k)\nabla^2 f(w)(w - w^k) + \frac{M}{6}\|w - w^k\|^3 \right\}.$$

- An alternative to line-search (or "trust-region") methods.
  - Leads to global (non-asymptotic) convergence rates.
  - Guarantees decrease if M is Lipschitz constant of Hessian.
    - Though this might gives steps that are smaller than needed.
  - Can be combined with acceleration to give faster rates than Newton.
  - Requires iterative solution to compute w^{k+1}.
- Recent work shows "quartic regularization" is feasible for convex functions.
  - Uses iterative solver for w^{k+1} with tensor-vector products.

Summary

- Polyak-Łojasiewicz inequality leads to linear convergence of gradient descent.
  - Only needs $O(\log(1/\epsilon))$ iterations to get within ϵ of global optimum.
- Strongly-convex differentiable functions functions satisfy PL-inequality.
  - Adding L2-regularization makes gradient descent go faster.
- Newton's method uses second-derivatives to converge faster.
  - Expensive in pure form, but practical approximations exist.

- Next time: why does L1-regularization set variables to 0?

Why is $\mu \leq L$?

- The descent lemma for functions with L-Lipschitz ∇f is that

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

- Minimizing both sides in terms of v (by taking the gradient and setting to 0 and observing that it's convex) gives

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

- So with PL and Lipschitz we have

$$\frac{1}{2\mu}\|\nabla f(w)\|^2 \geq f(w) - f^* \geq \frac{1}{2L}\|\nabla f(w)\|^2,$$

which implies $\mu \leq L$.

C^1 Strongly-Convex Functions satisfy PL

- If $g(x) = f(x) - \frac{\mu}{2}\|x\|^2$ is convex then from C^1 definition of convexity

$$g(y) \geq g(x) + \nabla g(x)^\top (y - x)$$

or that

$$f(y) - \frac{\mu}{2}\|y\|^2 \geq f(x) - \frac{\mu}{2}\|x\|^2 + (\nabla f(x) - \mu x)^\top (y - x),$$

which gives

$$\begin{aligned} f(y) &\geq f(x) + \nabla f(x)^\top (y - x) + \frac{\mu}{2}\|y\|^2 - \mu x^\top y + \frac{\mu}{2}\|x\|^2 \\ &= f(x) + \nabla f(x)^\top (y - x) + \frac{\mu}{2}\|y - x\|^2, \quad \text{(complete square)} \end{aligned}$$

the inequality we used to show C^2 strongly-convex function f satisfies PL.

Linear Convergence without Strong-Convexity

- The least squares problem is convex but not strongly convex.
  - We could add a regularizer to make it strongly-convex.
  - But if we really want the MLE, are we stuck with sub-linear rates?

- Many conditions give linear rates that are weaker than strong-convexity:
  - 1963: Polyak-Łojasiewicz (PL).
  - 1993: Error bounds.
  - 2000: Quadratic growth.
  - 2013-2015: essential strong-convexity, weak strong convexity, restricted secant inequality, restricted strong convexity, optimal strong convexity, semi-strong convexity.
- Least squares satisfies all of the above.

- Do we need to study any of the newer ones?
  - No! All of the above imply PL except for QG.
  - But with only QG gradient descent may not find optimal solution.

PL Inequality for Least Squares

- Least squares can be written as $f(x) = g(Ax)$ for a σ-strongly-convex g and matrix A, we'll show that the PL inequality is satisfied for this type of function.
- The function is minimized at some $f(y^*)$ with $y^* = Ax$ for some x, let's use $\mathcal{X}^* = \{x | Ax = y^*\}$ as the set of minimizers. We'll use x_p as the "projection" (defined next lecture) of x onto $\mathcal{X}^*$.

$$
\begin{aligned}
f^* = f(x_p) &\geq f(x) + \langle \nabla f(x), x_p - x \rangle + \frac{\sigma}{2} ||A(x_p - x)||^2 \\
&\geq f(x) + \langle \nabla f(x), x_p - x \rangle + \frac{\sigma\theta(A)}{2} ||x_p - x||^2 \\
&\geq f(x) + \min_y \left[\langle \nabla f(x), y - x \rangle + \frac{\sigma\theta(A)}{2} ||y - x||^2 \right] \\
&= f(x) - \frac{1}{2\theta(A)\sigma} ||\nabla f(x)||^2.
\end{aligned}
$$

- The first line uses strong-convexity of g, the second line uses the "Hoffman bound" which relies on $\mathcal{X}^*$ being a polyhedral set defined in this particular way to give a constant $\theta(A)$ depending on A that holds for all x (in this case it's the smallest non-zero singular value of A), and the third line uses that x_p is a particular y in the min.

Linear Convergence for "Locally-Nice" Functions

- For linear convergence it's sufficient to have

$$L[f(x^{t+1}) - f(x^t)] \geq \frac{1}{2}\|\nabla f(x^t)\|^2 \geq \mu[f(x^t) - f^*],$$

for all x^t for some L and μ with $L \geq \mu > 0$.

(technically, we could even get rid of the connection to the gradient)

- Notice that this only needs to hold for all x^t, not for all possible x.
  - We could get linear rate for "nasty" function if the iterations stay in a "nice" region.
  - We can get lucky and converge faster than the global L/μ would suggest.

- Arguments like this give linear rates for some non-convex problems like PCA.

Convergence of Iterates

- Under strong-convexity, you can also show that the iterations converge linearly.

- With a step-size of $1/L$ you can show that

$$\|w^{k+1} - w^*\| \leq \left(1 - \frac{\mu}{L}\right)\|w^k - w^*\|.$$

- If you use a step-size of $2/(\mu + L)$ this improves to

$$\|w^{k+1} - w^*\| \leq \left(\frac{L - \mu}{L + \mu}\right)\|w^k - w^*\|.$$

- Under PL, the solution w^* is not unique.
  - You can show linear convergence of $\|w^k - w_p^k\|$, where w_p^k is closest solution.

Improved Rates on Non-Convex Functions

- We showed that we require $O(1/\epsilon)$ iterations for gradient descent to get norm of gradient below ϵ in the non-convex setting.
- Is it possible to improve on this with a gradient-based method?
- Yes, in 2016 it was shown that a gradient method can improve this to $O(1/\epsilon^{3/4})$:
  - Combination of acceleration and trying to estimate a "local" μ value.

Complexity of Minimizing Strongly-Convex Functions

- For strongly-convex functions:
  - Sub-gradient methods achieve optimal rate of $O(1/\epsilon)$.
  - If ∇f is Lipschitz continuous, we've shown that gradient descent has $O(\log(1/\epsilon))$.

- Nesterov's algorithms improves this from $O(\frac{L}{\mu}\log(1/\epsilon))$ to $O(\sqrt{\frac{L}{\mu}}\log(1/\epsilon))$.
  - Corresponding to linear convergence rate with $\rho = (1 - \sqrt{\frac{\mu}{L}})$.
  - This is close to the optimal dimension-independent rate of $\rho = \left(\frac{\sqrt{L}-\sqrt{\mu}}{\sqrt{L}+\sqrt{\mu}}\right)^2$.