10. Algorithms -2 and Final Review

Gradient Descent using BLS, Newton's Method and Final Review

Sun Woo Lim

Feb 27, 2021

Descent Algorithms : General Case

We use **algorithms** to **sequentially** approach the **solution** to an optimization problem : finding the **minimum** of a function.

Here, only deal with **unconstrained minimization problem**.

Why **sequential algorithms**? Usually in **high dimensional setting!**

1. Humans : Hard to analytically find the solution! Cannot **see** the optimum point.
2. Computers : Hard to analytically find the solution! Grid Approximation? Too many Grids.

General Descent Algorithm

1. Start at $x_0 \in dom(f)$.
2. Descent Step : $x_{k+1} = x_k + s_k v_k$, s_k is called a **step size** and v_k is called a **descent direction**.

In order for v_k to be a descent direction, only need $f(x_k + s_k v_k) < f(x_k)$ for all small $s_k > 0$.

Generally, doesn't even need f : differentiable.

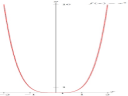
For differentiable f, **usually** set $v_k = -\nabla f(x_k)$:
"A ftn **locally decreases** in the direction of **negative gradient**.
If **gradient** is zero, I'm at an **extremum (local minimum / maximum / saddle point)**"

: called **Gradient Descent**.

Gradient Descent : Convex Function

If the objective function is **convex and differentiable** and $dom(f_0) = \mathbb{R}^n$, x^* is the global minimizer of f iff $\nabla_x f(x)|x = x^* = 0$
However, even for **convex functions**, the claim "the function decreases in the direction of its negative gradient" is only true, **locally**.

(a) $y = x^2$ works for sufficiently small fixed step size



(b) $y = x^4$ may not work even for sufficiently small fixed step size

✓ For some cases, need to decrease the **learning rate** (not necessarily every iteration). ex) $lr_{new} = lr_{old} \times 0.99$

- Advantage : less vulnerable to gradient explosion
- Downfall : ↑ of iteration to converge, more seriously, the learning may stop even if I am not at the optimal point!

Even for a convex functions, selection of **step size** is very important. Discussion about the **step size** keeps going on...

Gradient Based Exact Line Search

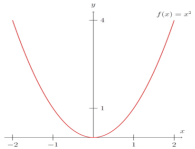
No need of differentiability of a function.

Exact Line Search Steps

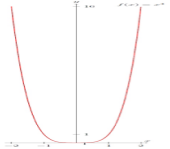
1. Set x_1 and according descent direction v_1 (negative gradient $-\nabla f(x_1)$ or something else)
2. At x_1, choose s_1^* satisfying $f(x_1 + s_1 v_1)$ is minimal.
3. $x_2 = x_1 + s_1^* v_1$ and set v_2 and choose s_2^* satisfying $f(x_2 + s_2 v_2)$ is minimal.
4. $s_k^* := argmin_{s \geq 0} f(x_k + s \times v_k)$. Solve this in every step.

Important) Finding optimal s requires solving a univariate (+ generally non-convex) optimization problem : computationally hard (Section 12.2.1.3 in Calafiore & El Ghaoui) $\rightarrow$ ELS is **not** usually used.

Cases where Gradient Based ELS work



(a) $y = x^2$: ELS works in one step



(b) $y = x^4$: ELS works in one step

ELS : Example

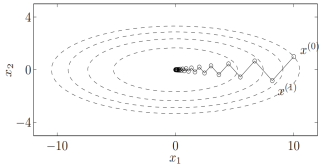


Figure 9.2 Some contour lines of the function $f(x) = (1/2)(x_1^2 + 10x_2^2)$. The condition number of the sublevel sets, which are ellipsoids, is exactly 10. The figure shows the iterates of the gradient method with exact line search, started at $x^{(0)} = (10, 1)$.

Figure: from Boyd & Vandenburghe p469

Let $\gamma \geq 1$ and $f(\vec{x}) := \frac{1}{2}(x_1^2 + \gamma x_2^2) = \frac{1}{2} x^T \begin{bmatrix} 1 & 0 \\ 0 & \gamma \end{bmatrix} x$. Let $\vec{x}_0 = (\gamma, 1)$.

Can show $\vec{x}_k = (\gamma \rho^k, (-1)^k \rho^k)$ where $\rho := \frac{\gamma - 1}{\gamma + 1}$ and thus, $f(\vec{x}_k) = \rho^{2k} f(\vec{x}_0)$

Important) Convergence is slow if $\rho \approx 1 \leftrightarrow \gamma >> 1$. If $\gamma \in (0, 1)$, convergence is slow if $\gamma << 1$: $\kappa(\begin{bmatrix} 1 & 0 \\ 0 & \gamma \end{bmatrix})$ is big.

$\kappa(A) := \frac{\sigma_1}{\sigma_n} = ||A||_2 ||A^{-1}||_2$ derived from $\sigma_{max}(A^{-1}) = \frac{1}{\sigma_{min}(A)}$

Convergence Analysis of Gradient Based ELS in Convex Case

Assumptions

1. Assume $f : \mathbb{R}^n \to \mathbb{R}$ is twice differentiable
2. Assume $dom(f) = \mathbb{R}^n$
3. Assume $MI \succeq \nabla^2 f(x) \succeq mI$ for some $0 < m < M, \forall x \in \mathbb{R}^n$: function is quadradically upper and lower bounded.

Quadradic Bound of f thanks to the Assumptions

Given $x, y \in \mathbb{R}^n$, I have $f(y) = f(x) + \nabla f(x)^T (y - x) + \frac{1}{2}(y - x)^T \nabla^2 f(z)(y - x)$ for some z between x and y.

Thus, $f(x) + \nabla f(x)^T + \frac{m}{2}||y - x||_2^2 \leq f(y) \leq f(x) + \nabla f(x)^T + \frac{M}{2}||y - x||_2^2$.

: A good **bound** of f. Especially, quadradic lower bound $\to \exists$ unique x* $:= argmin_x f(x)$ that **achieves** $p^* = min_x f(x)$.

This assumption leads to convergence analysis about distance to x^*, speed of convergence and etc.

Convergence Analysis of Gradient Based ELS in Convex Case : Distance to Optimal Point

✓ Analysis 1 : Distance to the Optimal point

✓ Analysis 2 : Linear Convergence
When the speed of Convergence to the optimality is **exponential (geometric)**, we call **the algorithm linearly converges**.

✓ Analysis 3 : Amount of iteration needed until convergence

Backtracking Line Search : General Function

BLS attempts to find a good **step size** if the **Armijo Condition** is satisfied.

The **BLS** is another idea to choose the **step size** that makes the function decrease **locally.**
But unlike **ELS**, it **guarantees a suffient rate of decrese in** ϕ without a need to solve an optimization problem of $\phi(s)$.

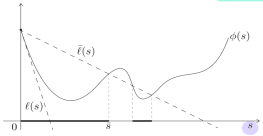
Armijo Condition

Let $\alpha \in (0, 1)$, typically $(0, 0.5)$. Assume I am not at an extremum (local min / max / saddle point).

Suppose the descent direction is v_k (which need not be $-\nabla f(x^{(k)})$

Pick $\beta \in (0, 1)$ and $\alpha \in (0, 0.5)$

The parameter α is typically chosen between 0.01 and 0.3, meaning that we accept a decrease in f between 1% and 30% of the prediction based on the linear extrapolation. The parameter β is often chosen to be between 0.1 (which corresponds to a very crude search) and 0.8 (which corresponds to a less crude search).



$$\phi(s) = f(x^{(k)} + sv_k)$$

$$\bar{l}(s) = f(x^{(k)}) + \alpha s \nabla f(x^{(k)})^T v_k$$

$$l(s) = f(x^{(k)}) + s \nabla f(x^{(k)})^T v_k$$

Again there is some $\bar{s} > 0$ such that

: Armijo Condition

$$\phi(s) < \bar{l}(s), \quad s \in (0, \bar{s}).$$

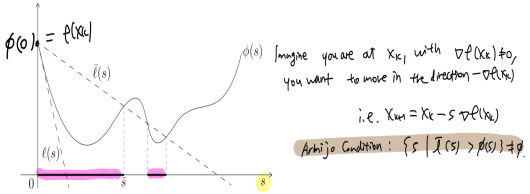
Gradient Based Backtracking Line Search : General Function

The Gradient Based **BLS** attempts to find a good **step size** if the **Armijo Condition** is satisfied.

The **BLS** is another idea to choose the **step size** to **ensure** $f(x_{new}) \leq f(x_{old})$: function is decreasing **locally.**

Armijo Condition

Let $\alpha \in (0,1)$, typically $(0, 0.5)$. Assume $\nabla f(x_k) \neq 0$: I am not at an extremum (local min / max / saddle point).



$$l(s) := f(x^{(k)}) - s\underbrace{\|\nabla f(x^{(k)})\|_2^2}_{= \nabla f(x_k)^T \nabla f(x_k)}$$

$$\bar{l}(s) := f(x^{(k)}) - \alpha s\|\nabla f(x^{(k)})\|_2^2 \quad \text{Slope here} = \alpha \nabla f(x_k)$$

is less steep than the slope of $\nabla f(x_k)$

$$\phi(s) := f(x^{(k)} - s\nabla f(x^{(k)}))$$

Gradient Based Backtracking Line Search : General Function

Pick $\alpha \in (0, 0.5)$ and $\beta \in (0, 1)$

At $x^{(k)}$ to determine $x^{(k+1)}$

- Set $s := 1$

- While

$\phi(s)$

$f(x^{(k)} - s\nabla f(x^{(k)})) >$

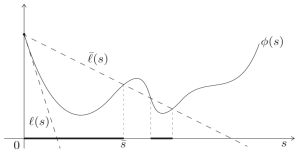
$f(x^{(k)}) - \alpha s \|\nabla f(x^{(k)})\|_2^2$ $\bar{l}(s)$

(i.e. the Armijo condition fails)

replace s by βs and repeat.

Step by step decrease the step size : $s \to \beta s$

- Return $x^{(k)} - s\nabla f(x^{(k)})$ as $x^{(k+1)}$.



The algorithm will terminate in a finite number of iterations when $\nabla f(x^{(k)}) \neq 0$ since

There is an open interval $(0, \bar{s})$ (i.e. $\bar{s} > 0$) such that

$\phi(s)$

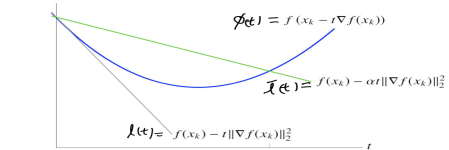
$f(x^{(k)} - s\nabla f(x^{(k)})) <$

$f(x^{(k)}) - \alpha s \|\nabla f(x^{(k)})\|_2^2$

$\bar{l}(s)$

for all $s \in (0, \bar{s})$.

Gradient Based Backtracking Line Search : Convex Function



Exercise) Let $f : \mathbb{R}^2 \to \mathbb{R}$ with $dom(f) = \mathbb{R}^2$, $f(x) = x_1^2 + 3x_2^2$. Let the hyperparameters $\alpha = 0.25, \beta = 0.5$. Now, $x_k = (2, 2)^T$. Use the Gradient Based **BLS** to find the next position of x : x_{k+1}.

Convergence Analysis of Gradient Based BLS in Convex Case

Same assumption as in ELS case :

1. Assume $f : \mathbb{R}^n \to \mathbb{R}$ is twice differentiable
2. Assume $dom(f) = \mathbb{R}^n$
3. Assume $MI \succeq \nabla^2 f(x) \succeq mI$ for some $0 < m < M, \forall x \in \mathbb{R}^n$: function is quadradically upper and lower bounded.

Convergence of ELS vs BLS : Visualization

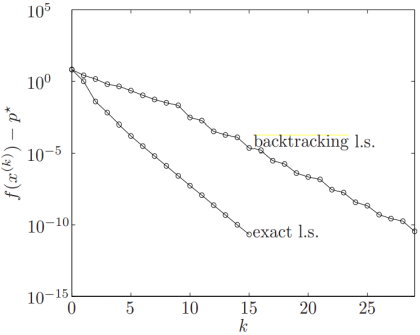


Figure 9.4 Error $f(x^{(k)}) - p^\star$ versus iteration k of the gradient method with backtracking and exact line search, for the problem in $\mathbf{R}^2$ with objective f given in (9.20). The plot shows nearly linear convergence, with the error reduced approximately by the factor 0.4 in each iteration of the gradient method with backtracking line search, and by the factor 0.2 in each iteration of the gradient method with exact line search.

(a) Boyd & Vandenburghe, p 471

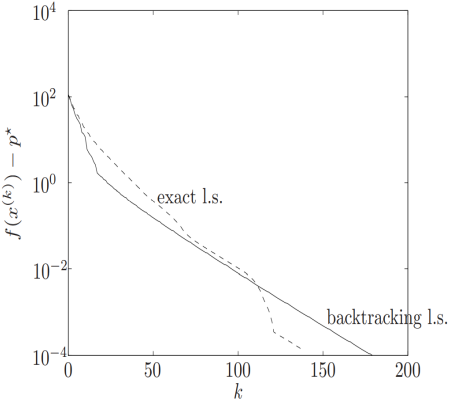


Figure 9.6 Error $f(x^{(k)}) - p^\star$ versus iteration k for the gradient method with backtracking and exact line search, for a problem in $\mathbf{R}^{100}$.

(b) Boyd & Vandenburghe, p 473

Newton's Method : Overview

✓ **Newton's Method** idea : In every iteration, approximate the function locally by its 2nd order approximation
: Uses both gradient and hessian.

✓ Newton's Method is a 2nd order method while Gradient Descent is a 1st order method.

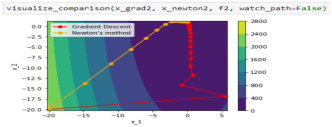
Assumpions of f :
1) Assume $f : \mathbb{R}^n \to \mathbb{R}$ is twice differentiable and convex, $dom(f) = \mathbb{R}^n$
2) $MI \succeq \nabla^2 f(x) \succeq mI$ for some $0 < m < M, \forall x \in \mathbb{R}^n$: used in BLS based Newton's Method.
3) $||\nabla^2 f(y) - \nabla^2 f(x)||_2 \leq L||y - x||_2 \ \forall x, y \in \mathbb{R}^n$: used in BLS based Newton's Method.

Best Second Order Taylor Series Approximation of f **at** x : $f(y) \approx f(x) + \nabla f(x)^T(y - x) + \frac{1}{2}(y - x)^T \nabla^2 f(x)(y - x)$

Take gradient of RHS w.r.t. y and set it zero $\to \nabla f(x) + \nabla^2 f(x)(y - x) = 0$. Minimized if $y = x - \nabla^2 f(x)^{-1} \nabla f(x)$

Newton's Method compared to Gradient Descent :

1. Convergence to optimality is quadradic (speed is doubly exponential) :)
2. Calculation of the Hessian :(



Pure Newton's Method

1. Start at $x_0 \in dom(f)$.
2. At x_k, if $\nabla f(x_k) \neq 0$, $x_{k+1} = x_k - \nabla^2 f(x_k)^{-1} \nabla f(x_k)$
   : **Descent Direction** : $-\nabla^2 f(x_k)^{-1} \nabla f(x_k)$ while **step size** : always 1.

✓ Ex1) $f(x) = x^2 - sin(x)$. Show f is strictly convex and set the update equation using the pure Newton's Method.

✓ Ex2) $f(\vec{x}) = \frac{1}{2}x_1^2 + \frac{7}{20}x_2^2$. $x_0 = [0, 1]^T$. Show that the pure Newton's method gets to x^* at **one step**. Why?

Note that taylor approximation of a quadradic function is itself at every position!

Newton's Method Using BLS

✓ $\phi(s) := f(x_k - s\nabla^2 f(x_k)^{-1} f(x_k))$
✓ $\bar{l}(s) := f(x_k) - \alpha s \nabla f(x_k)^T \nabla^2 f(x_k)^{-1} \nabla f(x_k)$

1. Start at $x_0 \in dom(f)$ and set $\beta \in (0,1), \alpha \in (0, 0.5)$ and $s = 1$: initial step size = 1.
2. At x_k, if $\nabla f(x_k) \neq 0$, Carry out a Backtracking Line Search :
   - While $\phi(s) > \bar{l}(s) : s \leftarrow \beta s$
   - Else (: when Armijo Condition is satisfied), $x_{k+1} = x_k - s\nabla^2 f(x_k)^{-1} \nabla f(x_k)$.

: **Descent Direction** : $-\nabla^2 f(x_k)^{-1} \nabla f(x_k)$ while **step size** : starting from 1, possibly decrease.

✓ Convergence Analyis under suitable conditions stated in the previous slide.

- Initially, **damped Newton's Phase** where $|f(x_k) - p^*| = f(x_k) - p^* \downarrow$ by at least a fixed amount at each iteration.

  : ✓ $\exists \eta \in (0, \frac{m^2}{L})$ and $\gamma > 0$ s.t. if $||\nabla f(x_k)||_2 \geq \eta, f(x_k) - f(x_{k+1}) \geq \gamma$: **Damped Newton Phase**

- Later, a **pure Newton's Phase** where in m number of steps, $|f(x_k) - p^*| = f(x_k) - p^* \downarrow$ by a **factor of** $(0.5)^{2^m}$

  : ✓ If $||\nabla f(x_k)||_2 < \eta$, no backtracking needed and $\frac{L}{2m^2}||\nabla f(x_{k+1})||_2 \leq (\frac{L}{2m^2}||\nabla f(x_k)||_2)^2$: **Pure Newton Phase**

The speed of convergence to the optimal point is **doubly exponential** : called **quadradic convergence**.

Total Review

1. Linear Algebra
:**Linear Algebra** : basis for this whole course + some optimization problems solely can be solved using **Linear Algebra** knowledge

Keywords : Vector, Norm, Matrix, Matrix Norm, Symmetric Matrix, PSD, PD, EVD, SVD, Projection, Regression

2. Convex Optimization Problem
: Start discussing optimization problem, especially convex problem. Learn how to form a **dual** of the original problem.

Keywords : Convex Set, Convex Function, Lagrangian, KKT, Slater's, Strong Dual, Weak Dual

3. Types of Convex Optimization Problem
Although **convex problem** is a broad subject, can be formulated as **LP, QP, QCQP, SOCP, SDP**. Additionally handle **GP**.

Keywords: LP, QP, QCQP, GP, SOCP, SDP

4. Algorithms
Learn Algorithms to solve optimization problems **without constraints**. Not merely learning ideas of **Gradient Descent, Newton's Method**, analyze how to adjust the **step size** to make the algorithm converge or learn properly.

Keywords: Descent Algorithm, Step Size, ELS, Armijo Condition, BLS, Gradient Descent, Newton's Method

Application of this Course

1. Linear Algebra

✓ L1, L2, L_∞ norms used so many times (loss functions, regularization, ...)

✓ Projection, Regression with several situations (including RIDGE / LASSO)

✓ Applications of SVD : PCA, FA (Factor Analysis), ...

2. Convex Optimization Problem

✓ Statistics : MLE, logistic regression, Maximum a posteriori Estimation, Robust Learning, distribution fitting, ...

✓ Bump into many papers addressing **Dual, KKT, Slaters, log-sum-exp,...**.

3. Types of Convex Optimization Problem

✓ Utility Maximization, Facility Optimization, Logistic Regression, SVM (Hard Margin, Soft Margin), Derivation of Entropy, Mimimum Time Path problem, LASSO, projecting a point onto a hyperplane

✓ If you can analyze which problem class you are handling with $\rightarrow$ Use the formula for the solution or get the dual.

4. Algorithms

✓ Analyze the objective function : convex?

✓ Be careful about the step size even for convex functions (Initialization would be important for non-convex functions)

✓ A high portion of ML, DL is optimizing certain **objective function** using sgd / adam / other...

Further

✓ Hard course $\rightarrow$ start from broad concepts, narrow down.

✓ **Studying only crucial parts (for you) is not a bad idea : you major different study areas**

Zillions of things you may do in further studies or at life:

Data Analysis, Data Science, Data Engineer, Multi Dimensional Statistics, Differential Equations, Math Analysis, Bayesian Stat, Supervised Learning, Unsupervised Learning, Reinforcement Learning, Optimization itself...

The importance of this course? Important, but importance vary by person.

✓ This is only the beginning of the course. Without painful training, you end up with nothing.

Take a Course / Life Experience / Solve problems in Boyd & Vandenburghe or Calafiore & El Ghaoui / ...

✓ For section 2 and 3, we learned how mainly **humans** solve optimization problems. How to make **computers** solve them : go onto the next course.