MATH 470: Numerical Optimization

Zijian Pei

December 9, 2021

Contents

1 Chapter 1

1.1 Mathematical Formulation

- x: the vector of variables
- f: the objective function, a scalar function of x needs to be maximized/minimized
- c_i: constraint function, scalar functions of x
- ε: equality constraints
- I: inequality constraints

1.2 Continuous versus Discrete Optimization

Continuous optimization: the feasible set is usually uncountably infinite; x allowed to be real numbers; easier to solve
Discrete optimization: the behavior of objective and constraints may change significantly from one point to another

1.3 Constrained and Unconstrained optimization

Constrained optimization: trivial
Unconstrained optimization: empty set for both I and ϵ

1.4 Global and Local Optimization

Global optimization: trivial
Local optimization: trivial

1.5 Convexity

1.5.1 Convex sets

A set $S \in \mathbb{R}^n$ is convex if for $x \in S$ and $y \in S$, we have $\alpha x + (1-a)y \in S$ for all $\alpha \in [0,1]$

Some key properties:

- Two disjoint convex setss have a hyperplane that separates them. $C \subseteq \{x : a^t x \leq b\}$ and $D \subseteq \{x : a^t x \geq b\}$
- Supporting hyperplane theorem:

  Theorem 1.1. *A boundary point of a convex set has a supporting hyperplane passing through it. i.e $x_0 \in BD(C)$, C convex, then there exists a such that $C \subseteq \{x : a^T x \leq a^T x_0\}$*

1.5.2 Convex functions

A function f is convex if its domain S is convex and for any two points x and y in S,
$f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha)f(y)$ (f lies below the line segment joining f(x) and f(y))

- Strict convex: $f(tx + (1-t)y) < tf(x) + (1-t)f(y)$ if $0 < t < 1$.
- Strong convex: With parameter $m > 0$: $f - \frac{m}{2}\|x\|_2^2$ is convex. (f is at least as convex as a quadratic function)

Example 1.1. *Univariate functions*

- *Exponential function e^{ax} with any a over $\mathbb{R}$*

- *Power function x^a is convex for $a \geq 1 / \leq 0$ over $\mathbb{R}_+$*
- *Logrithmic function $logx$ is concave over $\mathbb{R}_{++}$*

Example 1.2. *Affine function: $a^T x + b$ is both convex and concave*

Example 1.3. *Quadratic functions: $\frac{1}{2} x^T Q x + b^T x + c$ is convex if Q is semi-definite.*

Example 1.4. *Least square loss: $\|y - Ax\|_2^2$ is always convex.*

Example 1.5. *l_p norm:$\|x\|_p = (\sum_{i=1}^n x_i^p)^{1/p}$*

Example 1.6. *Indicator function: if C is convex, then its indicator function $I_C(x) = \begin{cases} 0 & x \in C \\ \infty & x \notin C \end{cases}$*

Example 1.7. *Support function: for any set C(convex or not), its support function $I_C^*(x) = \max_{y \in C} x^T y$ is convex.*

Example 1.8. *Max function: $f(x) = max\{x_1, ..., x_n\}$ is convex.*

proposition 1.2. *Local minimum of a convex function is a global minimum*

Proof:

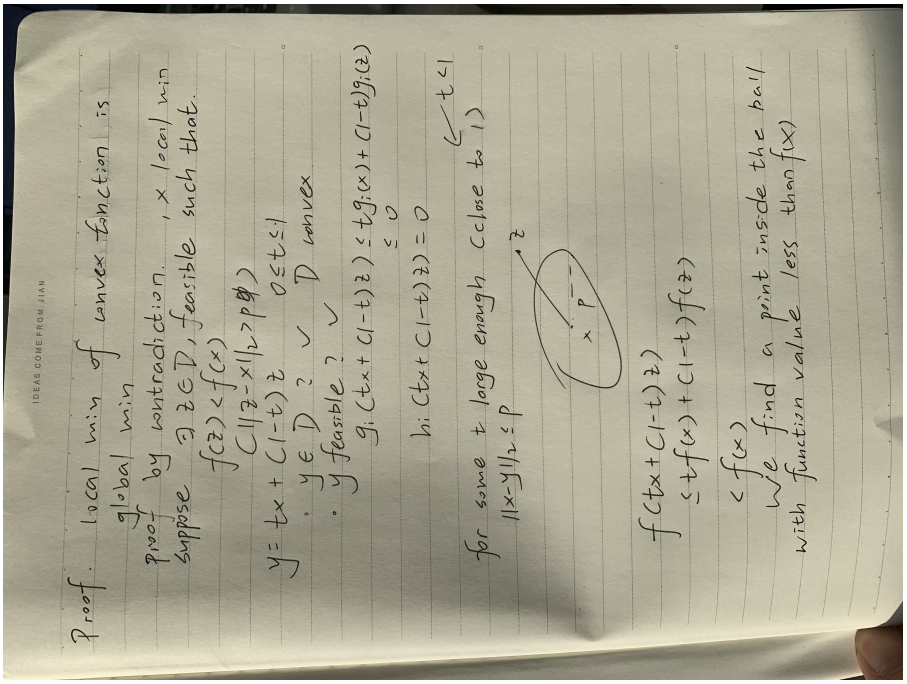

Fig. 1. Proof of proposition

2 Gradient descent

2.1 Principal of GD

Consider unconstrained smooth convex optimization

$$\min_x f(x) \tag{1}$$

Gradient descent:choose initial point $x^{(0)} \in \mathbb{R}^n$, repeat $x^{(k)} = x^{(k-1)} - t_k \bigtriangledown f(x^{(k-1)})$

Second-order Taylor expansion:$f(y) = f(x) + \bigtriangledown f(x)^T (y-x) + \frac{1}{2}(y-x) \bigtriangledown^2 f(x)(y-x)$

Quadratic approximation: Replace hessian by $\frac{1}{t} I$

$$f(y) = f(x) + \bigtriangledown f(x)^T (y-x) + \frac{1}{2t}\|y-x\|_2^2 \tag{2}$$

2.2 Backtracking line-search

Line search method:

First fix parameter $0 < \beta < 1$, and $0 < \alpha \leq 1/2$

Then at each iteration, start with $t = t_{init}$, and while

$$f(x - t \bigtriangledown f(x)) > f(x) - \alpha t \|\bigtriangledown f(x)\|_2^2 \tag{3}$$

shrink $t = \beta t$, else perform gradient descent update

3 Chapter 2

3.1 What is a solution

- local minimizer: x^* is called local minimizer if there exists neighborhood N of x^* such that $f(x^*) \leq f(x)$ for all $x \in N$
- global minimize: trivial
- strict minimizer: any $x \in N$, if $x \neq x^*$, $f(x) > f(x^*)$
  Note: strict does not imply isolate, but isolate implies strict

Theorem 3.1 (Taylor's Theorem). *Suppose that $f : R^n \to R$ is continuous differentiable and $p \in R^n$. Then we have:*

$$f(x+p) = f(x) + \nabla f(x+tp)^T p \qquad \text{for some } t \in (0,1) \tag{4}$$

My understanding:
This is just basic analysis stuff, to be more clear, we can consider x in one dimensional space, then

$$f(x+h) = f(x) + f'(a) * h \qquad \text{where a is some point between x and x+h} \tag{5}$$

Moreover, if f is twice continuous differentiable, we have:

$$\nabla f(x+p) = \nabla f(x) + \int_0^1 \nabla^2 f(x+tp)p\,dt. \tag{6}$$

and

$$f(x+p) = f(x) + \nabla f(x)^T p + \frac{1}{2} p^T \nabla^2 f(x+tp)p \qquad \text{for somet} \in (0,1) \tag{7}$$

Theorem 3.2 (First-Order Necessary Conditions). *If x^* is a local minimizer and f is continuous differentiable in an open neighborhood of x^*, then $\nabla f(x^*) = 0$*
My understanding:
This is kind of trivial stuff, local minimizer has the gradient of zero

Theorem 3.3 (Second-Order Necessary Conditions). *If x^* is a local minimizer of f and $\nabla^2 f$ exists and is continuous in an open neighborhood of x^*, then $\nabla f(x^*) = 0$ and $\nabla^2 f(x^2)$ is positive semidefinite.*
My understanding:
The first part is exactly Thm2.2. For the second part, I think we need first understand the definition of positive semidefinite
A square matrix A is positive definite if there is a positive scalar α such that

$$x^T Ax \geq \alpha x^T x \qquad \text{for all } x \in R^n \tag{8}$$

It's positive semidefinite if

$$x^T Ax \geq 0 \qquad \text{for all } x \in R^n \tag{9}$$

The proof in the book is assume by contradiction, which would leads to a vector p that can make

$$p^T Ap < 0 \qquad A \text{ is } \nabla^2 f(x^*) \text{ in this case} \tag{10}$$

Because the second order derivative would not turn right from negative to positive or positive to negative because of the continuous conditions. So we can find

$$p^T \nabla^2 f(x^* + tp)p \qquad \text{if } t \text{ is smaller enough I think} \tag{11}$$

Then we can get

$$f(x^* + tp) = f(x^*) + tp^T \nabla f(x^*) + \frac{1}{2}t^2 p^T \nabla^2 f(x^* + tp)p < f(x^*) \tag{12}$$

The above equation is from Theorem2.1
The third term is obviously negative, we may try to see why this above equation is less than $f(x^*)$, the trick is actually on the second term. Notice there is a first order derivative of f in the second term, this result the second term to be 0, so we are done

Theorem 3.4 (Second-Order Necessary Conditions)**.** *Suppose that ∇^2 is continuous in an open neighborhood of x^* and that $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive definite. Then x^* is a strict local minimizer of f.*
I right now can not understand the proof of this theorem, and I just try to remember this theorem

Theorem 3.5. *When f is convex, any local minimizer x^* is a global minimizer of f. If in addition f is differeitiable, then any stationary point x^* is a global minimizer of f.*

3.2 Overview of algorithms

Idea: start from x_0, use sequence of iterates to approximate the wanted value

- Line search strategy: Choose a direction p_k to search along, only take limited number of trials, and computation of exact value of $min\ f(x_k + ap_k)$ is unnecessary
- Trust Region: Construct a model function m_k near the current point x_k, and restrict the search for a minimizer of m_k to some region around x_k, approximately solve $min\ m_k(x_k + p)$. If candidate solution does not produce a sufficient decrease in f ?? quadratic form ??

Search directions for line search methods:

- Steepest descent
  My understanding of steepest descent: Things can be easily understood in one-dimensional case. Imagine we now have an point x_1, and $f(x_1)$ has the positive derivative, then if this function is continuous on this x_1, it must come from a value that has lower value than $f(x_1)$, and it must have the same vector component as the negative value of the derivative of $f(x)$. When we are not sure the actual slope between $f(x_1)$ and $f(x^*)$, we may choose the negative value of the derivative of $f(x)$ for simplicity, and take a small step each time to approach the minimum value we need.
  Interesting found when programming: When I try to use backtracking line search to get the minimum value of the function in the exercise 2.1 of this chapter, I found it seems never reach out the actual x^*, but only approaches it, I suppose because we in each iteration we assume there must be some x_2 that with $f(x_2) < f(x_1)$, actually it's right in almost all of those iterations, but we don't know the exact slope, in which I mean the slope between the minimizer and the x_k we have in each iteration. Things can be more easily understood in a parabolic equation, we may let this paraolic function be concave up, then in the

end we may have x_k in one side, and there's no way for it to approach smaller value along the negative value of its derivative because of the step length we set. Namely, it's the minimum x on the tangent line that has the slope of $\nabla f(x)$ that intersects the function restricted for the step length set in the first. But if we decrease the step length, then we can find smaller value, and begin next iteration.

- Newton direction
  The direction is derived from second-order Taylor series approximation to $f(x_k + p)$, which is

$$f(x_k + p) = f_k + p^T \nabla f_k + \frac{1}{2} p^T \nabla^2 f_k p = m_k(p), \qquad \text{and } p_k \text{ is } -(\nabla^2 f_k)^{-1} \nabla f_k \tag{13}$$

  My finding when programming:
  Newton direction is actually the negative value of Hessian times the gradient, and it actually can reach the minimum value of the function when I change the step length to different values

- Quasi-Newton
  My understanding of quasi-Newton:This method avoids computing of hessian, and the direction is defined as $-B_k^{-1} \nabla f$ as always, but in this case, B_k is not the B_k defined in Newton's method, which is exactly the hessian. In quasi-Newton, B_k is an approximation of hessian, and is not actually equals to hessian, and we actually need to manually set B_0 first, and use SRI or BFGS formula to update B_k of the next iteration. The SRI and BFGS formulas are listed here:
  SRI

$$B_{k+1} = B_k + \frac{(y_k - B_k s_k)(y_k - B_k s_k)^T}{(y_k - B_k s_k)^T s_k} \tag{14}$$

  BFGS

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k} \tag{15}$$

  Note:if in BFGS, we set B_0 to be positive definite, then the next generation updated is also positive definite
  Notice there are some new stuff here s_k and y_k, $s_k = x_{k+1} - x_k$ and is in the same dimensional space as x; $y_k = \nabla f_{k+1} - \nabla f_k$, and is a $p * 1$ vector, where p denotes the dimension of the space x belonging to

Key facts:
To produce decrease in f, the angle between the search direction and the negative value of the gradient must be less than 90 degree, this is trivial but important. If this angle is greater than 90 degree, the direction would have positive component, and increase the function

Models for trust-region methods
Set $B_k = 0$, then we only need to minimize $f_k + p^T \nabla f_k$ subject to $||p||_2 \leq \Delta_k$, and $p_k = -\frac{\Delta_k \nabla f_k}{||\nabla f_k||}$
(Steepest descent step because the direction is $-\nabla f_k$)

Scaling

- poor scaling:change of x in one direction causes big change for the function, ex: $f(x) = 10^9 x_1^2 + x_2^2$
  How to solve? Introduce new variable, order 1 ??
  Steepest descent is sensitive to poor scaling; Newton's method is unaffected.
- Exercise 2.1 The gradient in this case is equal to

$$\begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \end{bmatrix} \tag{16}$$

In general case, gradient of $x \in R^n$ should be

$$\begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \cdot \\ \cdot \\ \cdot \\ \frac{\partial f}{\partial x_n} \end{bmatrix} \tag{17}$$

4 Chapter 3

Before starting chapter3, we need to be familiar with 3 most-used directions in line-search method

- Steepest descent
$$p_k = - \nabla f(x_k) \tag{18}$$
- Newton direction
$$p_k = - \nabla^2 f^{-1}(x_k) \cdot \nabla f(x_k) \tag{19}$$
- Quasi-Newton direction
$$p_k = -B_k^{-1} \cdot \nabla f(x_k) \tag{20}$$

4.1 Step length

$$\phi(\alpha) = f(x_k + \alpha p_k) \tag{21}$$

Namely we need to identify a appropriate step length that achieves adequate reduction to f while maintains a low cost.
A simple requirement may just be
$$f(x_k + \alpha_k p_k) < f(x_k) \tag{22}$$

There are some conditions that will allow for suitable step length being created

- Wolfe conditions
$$f(x_k + \alpha_k p_k) \leq f(x_k) + c_1 \alpha_k \nabla f_k^T p_k \tag{23}$$
$$\nabla f(x_k + \alpha_k p_k)^T p_k \geq c_2 \nabla f_k^T p_k \tag{24}$$
with $0 < c_1 < c_2 < 1$

- Strong Wolfe conditions
$$f(x_k + \alpha_k p_k) \leq f(x_k) + c_1 \alpha_k \nabla f_k^T p_k \tag{25}$$
$$\nabla |f(x_k + \alpha_k p_k)^T p_k| \leq c_2 | \nabla f_k^T p_k| \tag{26}$$
with $0 < c_1 < c_2 < 1$

- Goldstein conditions
$$f(x_k) + (1 - c)\alpha_k \nabla f_k^T p_k \leq f(x_k + \alpha_k p_k) \tag{27}$$
$$f(x_k + \alpha_k p_k) \leq f(x_k) + c\alpha_k \nabla f_k^T p_k \tag{28}$$
with $0 < c < 1/2$

Lemma 4.1. *Suppose that $f : R^n \to R$ is continuously differentiable. Let p_k be a descent direction at x_k, and assume that f is bounded below along the ray $x_k + \alpha p_k | \alpha > 0$. then if $0 < c_1 < c2 < 1$, there exist intervals of step lengths satisfying the WOLFE conditions and the strong WOLFE conditions.*

The GOLDSTEIN Conditions

$$f(x_k) + (1-c)\alpha_k \bigtriangledown f_k^T p_k \leq f(x_k + \alpha_k p_k) \leq f(x_k) + c\alpha_k \bigtriangledown f_k^T p_k \tag{29}$$

Goldstein conditions are often used in Newton-type methods, but not suitable for quasi-Newton (Still need some time to figure out quasi-Newton) ????

Backtracking, trivial here

4.2 Convergence of line search methods

The angle θ_k between p_k and steepest descent direction $-\bigtriangledown f_k$, notice this angle must be smaller than 90 degrees to produce decrease to function f, and

$$cos\theta_k = \frac{-\bigtriangledown f_k^T p_k}{||\bigtriangledown f_k|| ||p_k||} \tag{30}$$

Theorem 4.2. *Consider any iteration of the form (3.1), where p_k is a descent direction and α_k satisfies the Wolfe conditions (3.6). Suppose that f is bounded below in R^n and that f is continuously differentiable in an open set N containing the level set $L = x : f(x) \leq f(x_0)$, where x_0 is the starting point of the iteration. Assume also that the gradient $\bigtriangledown$ is Lipschitz continuous on N, that is, there exists a constant $L > 0$ such that*

$$||\bigtriangledown f(x) - \bigtriangledown f(\bar{x})|| \leq L||x - \bar{x}|| \tag{31}$$

for all $x, \bar{x} \in N$
Then

$$\sum_{k \geq 0} cos^2\theta_k || \bigtriangledown ||^2 \leq \infty \tag{32}$$

4.3 Rate of convergence

We need to ensure line search direction P_k doe not become orthogonal to the gradient, to do this, we need to compute $cos\theta_k$ and compare it with the predecided $delta$, if cos value is smaller than $delta$, we need to fix the direction.
Concept of quadratic convex function

$$f(x) = \frac{1}{2} x^T Q x - b^T x \tag{33}$$

Q is symmetric and positive definite. The gradient is still not clear here why the answer is $Qx - b$????
Differentiate and set the direvative to 0, we get

$$\alpha_k = \frac{\bigtriangledown f_k^T \bigtriangledown f_k}{\bigtriangledown f_k^T Q \bigtriangledown f_k} \tag{34}$$

The steepest descent iterator is $x_k + a_k p_k$, where p_k in this case is $- \bigtriangledown f$

$$x_{k+1} = x_k - (\frac{\bigtriangledown f_k^T \bigtriangledown f_k}{\bigtriangledown f_k^T Q \bigtriangledown f_k}) \bigtriangledown f_k \tag{35}$$

Theorem 4.3. *When the steepest descent method with exact line searches is applied to the strongly convex quadratic function, the error norm satisfies*

$$||x_{k+1} - x^*||_Q^2 \leq (\frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1})||x_k - x^*||_Q^2 \tag{36}$$

Theorem 4.4. *Suppose that $f : R^n \rightarrow R$ is twice continuously differentiable, and that the iterates generated by the steepest-descent method with exact line search converge to a point x^* at which the Hessian matrix $\bigtriangledown^2 f(x^*)$ is positive definite. Let r be any scalar satisfying*

$$r \in (\frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1}, 1) \tag{37}$$

where $\lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$ are eigenvalues of $\bigtriangledown^2 f(x^*)$. Then for all k sufficiently large, we have:

$$f(x_{k+1}) - f(x^*) \leq r^2[f(x_k) - f(x^*)] \tag{38}$$

Theorem 4.5. *Suppose that f is twice differentiable and that the Hessian $\bigtriangledown^2 f(x^*)$ is Lipschitz continuous in a neighborhood of a solution x^* at which the sufficient conditions(Thm2.4) are satisfied. Consider the iteration $x_{k+1} = x_k + p_k$, where p_k is given by Newton's method, and is equal to $- \bigtriangledown^2 f_k^{-1} \bigtriangledown f_k$, Then we have:*
(i)if the starting point x_0 is sufficiently close to x^*, the sequence of iterates converges to x^*
(ii)the rate of convergence of x_k is quadratic
(iii)the sequence of gradient norm $||\bigtriangledown f_k||$ converges quadratically to 0

Quasi-Newton

$$p_k = -B_k^{-1} \bigtriangledown f_k \tag{39}$$

Theorem 4.6. *Suppose that $f : R^n \rightarrow R$ is twice differentiable. Consider the iteration $x_{k+1} = x_k + \alpha_k p_k$ where p_k is descent direction and α_k satisfies the WOLFE conditions with $c_1 \leq \frac{1}{2}$. If the sequence x_k converges to a point x^* such that $\bigtriangledown f(x^*) = 0$ and $\bigtriangledown^2 f(x^*)$ is positive definite, and if the search direction satisfies*

$$lim_{k \rightarrow \infty} \frac{||\bigtriangledown f_k + \bigtriangledown^2 f_k p_k||}{||p_k||} = 0 \tag{40}$$

then
(i)the step length $\alpha_k = 1$ is admissible for all k greater than a certain index k_0
(ii)if $\alpha_k = 1$ for all $k > k_0$, x_k converges to x^* superlinearly

Theorem 4.7. *Suppose that $f : R^n \to R$ is twice differentiable. Consider the iteration $x_{k+1} = x_k + p_k$, and that p_k is given by (3.34). Let us assume also that x_k converges to a point x^* such that $\nabla f(x^*) = 0$ and that $\nabla^2 f(x^*)$ is positive definite. Then x_k converges superlinearly iff (3.36) holds*

4.4 Newton's method with Hessian Modification

Hessian matrix $\nabla^2 f(x)$ mat not be positive definite, therefore the Newton direction p_k^N.(Notice this happens when we use Newton direction only)

$$\nabla^2 f(x_k) p_k^N = -\nabla f(x_k) \tag{41}$$

$$p_k^N = -B_k^{-1} \nabla f(x_k) \qquad \text{where } B_k \text{ is the hessian} \tag{42}$$

may not be a descent direction

Given initial point x_0 for k=0,1,2,... Factorize the matrix

Theorem 4.8. *Let f be twice continuous differentiable on an open set D, and assume that the starting point x_0 of Algorithm 3.2(Line-Search Newton) is such that the level set $L = \{x \in D : f(x) \leq f(x_0)\}$ is compact. Then if the bounded modified factorization property holds, we have that*

$$lim_{k \to \infty} \nabla f(x_k) = 0 \tag{43}$$

4.5 Matrix Factorization

Every positive definite matrix A can be written as

$$A = LDL^T \tag{44}$$

Notice we only consider the symmetric case here, because hessian is a symmetric matrix.
where L is a lower triangular matrix, with entries all 0 above the diagonal. And D is a diagonal matrix with positive values along the diagonal.
Take the example of 3*3 matrix

$$\begin{bmatrix} a_{11} & a_{21} & a_{31} \\ a_{21} & a_{22} & a_{32} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} d_1 & d_1 l_{21} & d_1 l_{31} \\ d_1 l_{21} & d_1 l_{21}^2 + d_2 & d_1 l_{31} l_{21} + d_2 l_{32} \\ d_1 l_{31} & d_1 l_{31} l_{21} + d_2 l_{32} & d_1 l_{31}^2 + d_2 l_{32}^2 + d_3 \end{bmatrix} \tag{45}$$

For algorithm3.4, what I am thinking is if we find some d_i along the diagonal that is smaller than 0, then the matrix we need to factorize is not positive definite. I think this can be used to check the second order derivative of Newton's method. Because in Newton's method, we actually need hessian to be positive definite. If not, then the direction computed by Newton's method will not be decreasing. But this part is not mentioned in this book.

How to fix this Newton's method is exactly algorithm 3.2, and we need to plus an identity matrix multiplied by a positive scalar to make the matrix B_k in Newton's method positive definite, as we are using B_k to find the Newton direction.

How to compute the scalar multiplied by identity matrix is exactly algorithm 3.3. But this algorithm requires understanding of Cholesky factorization, instead of modified Cholesky factorization, but we can actually use modified Cholesky factorization. Because if the factorization process fails, the values along the diagonal of matrix D would not be all positive.

5 Chapter 4 Trust-Region Methods

The size of trust region is critical to the effectiveness of each step, because if the region is too small, it's likely to missing the opportunity to move much closer to the objective function, f in this case. If the region is too large, the minimizer of the model may be far from the objective function. For Trush-Region methods, in each step, we need to solve

$$min\, m_k(p) = f_k + g_k^T p + \frac{1}{2} p^T B_k p \qquad s.t \quad ||p|| \leq \Delta_k \tag{46}$$

The ratio of actual reduction to predicted reduction is

$$\rho_k = \frac{f(x_k) - f(x_k + p_k)}{m_k(0) - m_k(p_k)} \tag{47}$$

where the numerator is the actual reduction, and denominator is the predicted reduction

I am still trying to understand this algorithm, not yet get it.

Given $\hat{Delta} > 0, \Delta_0 \in (0, \hat{Delta}), and$

Theorem 5.1. *The vector p^* is a global solution of the trust-region problem*

$$min\, m(p) = f + g^T p + \frac{1}{2} p^T B p \qquad s.t. ||p|| \leq \Delta \tag{48}$$

iff p^* is feasible and there is a scalar $\lambda \geq 0$ such that the following condition satisfied

$$(B + \lambda I) p^* = -g \tag{49}$$

$$\lambda(\Delta - ||p^*||) = 0 \tag{50}$$

$$(B + \lambda I) is\ positive\ semidefinite \tag{51}$$

6 Background Material

Only document things that I don't know or only have little memory about
Given $x \in R^n$ and $y \in R^n$, the standard inner product is $x^T y = \sum_{i=1}^{n} x_i y_i$
A square matrix A is positive definite if there is a positive scalar α such that

$$x^T AX \geq \alpha x^T x \tag{52}$$

But where does this x come fromaa??

vector norm ????

- Semi-definite:$X \in S^n$, and $a^T Xa \geq 0$ for all $a \in R^n$, then X is semi-definite. And X can be easily checked if all eigenvalues of X are greater or equal than 0.

6.1 Some key terms

- Convex combination:$x_1, ..., x_k \in R^n$, and $\theta_1 + ... + \theta_k x_k$ with $\theta_i \geq 0$,and $\sum_{i=1}^{k} = 1$
- Convex hull:convex hull of a set C conv(C) is all convex combinations of elements.

- Cone:$C \in R^n$ such that $x \in C$ implies $tx \in C$ for all $t \geq 0$
- Convex cone:$x_1, x_2 \in C$ implies $t_1 x_1 + t_2 x_2 \in C$

7 Chapter1 New Book

Some terms and key concepts:
Sparisity:only few coefficients of $\hat{f}$ are nonzero.

7.1 Notation

7.2 Loss Functions

In general, we need to minimize the function, which has the form

$$f(w) + \lambda\Omega(w) \tag{53}$$

where $f(w)$ is the loss function, and can be RSS or some other form depending on the type of the model. And $\lambda\Omega(w)$ is the penalty function.

There's a trick here:adding $\lambda\Omega(w)$ to the loss function is equivalent to adding a constraint of the form $\Omega(w) \leq \mu$, and w is a solution of the constraint problem iff it is a solution for a certain $\lambda \geq 0$??? Determination of these two values needs cross-validation, something I don't know yet.

7.3 Sparsity-Inducing Norms

7.3.1 Sparisity through the l_1-norm:

I believe we don't know there would be zero-coefficients for the solution w^* only if we choose proper methods in advance. The zero-coefficients are the product of the methods, for example Lasso would create zero-coefficients. And for Lasso, we need to minimize:

$$\frac{1}{2N}\sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}b_j)^2 \tag{54}$$

subject to $\sum_{j=1}^{p}|\beta_j| \leq t$, (constraint on l_1 norm).
It seems that regularzing l_1- norm(adding constraint on l_1-norm) would induce sparsity. In Statistical learning with Sparsity, it menstions that it's related to the geometry behind l_1-norm. For example, in 2-dim case, the picture within the constraint looks like a diamond, or a square rotating 45 degrees to the right/left. When the contour line hits the restriction-area, if the hit-point is at the corner, we will get one-parameter to be 0.

7.3.2 l_1/l_q-norms:

In this case, the coefficients of w^* are partitioned into different groups. Because we are using l_1 norm on groups, group sparsity would occur, i.e. the group norms are considered as coefficients and the whole norm are designed to be 0.
In the book, the l_1/l_q-norm with $q > 1$ is defined as

$$\Omega(w) = \sum_{g\in G}||w_g||_q := \sum_{g\in G} d_g\{\sum_{j\in g}|w_j|^q\}^{1/q} \tag{55}$$

Therefore, the l_1/l_2-norm should be

$$\Omega(w) = \sum_{g\in G}||w_g||_2 := \sum_{g\in G} d_g\{\sum_{j\in g}|w_j|^2\}^{1/q} \tag{56}$$

7.3.3 Norms for overlapping groups:a direct formulation

The groups span the entire set of variables, and there can be same variables in different groups. By appropriately assign different variables to different groups, and d_g which is the weight of different groups, not equal to 1, we can get sparsity in w^*.

7.3.4 Norms for overlapping groups:a latent variable formulation

$$\Omega_{union}(w) := min \sum_{g \in G} d_g ||v^g||_q \tag{57}$$

such that

$$\sum_{g \in G} v^g = w \qquad \text{and} \qquad \forall g \in G,\ v_j^g = 0 \quad \text{if } j \notin g \tag{58}$$

????Confusion

7.3.5 Multiple kernel learning

a

7.3.6 Trace norm

The trace norm is the sum of the singular values of the matrix M, and the singular values are the roots of eigenvalues of matrix MM^T.

7.3.7 Sparsity-inducing properties: A geometrical intuition

A penalty added problem can be transformed into a constrained optimization problem, which means we can instead try to minimize

$$f(w) \qquad \text{wrt} \quad \Omega(w) \leq \mu \tag{59}$$

????Confusion

7.4 Optimization Tools

7.4.1 Subgradients

Given a convex function $f : R^p \to R$, and a vector in R^n, the subgradient of at x is any $g \in \mathbb{R}^n$ such that

$$f(y) \geq f(x) + g^T (y - x) \qquad \forall y \tag{60}$$

Example 7.1. *Subgradient of* $f(x) = \|x\|_2$: $g = \begin{cases} \dfrac{x}{\|x\|_2} & x \neq 0 \\ \{z : \|z\|_2 \leq 1\} & x = 0 \end{cases}$

Example 7.2. *Subgradient of* $f(x) = \|x\|_1$: $g_i = \begin{cases} sign(x_i) & x_i \neq 0 \\ i\text{-th component between } [-1, 1] & x_i = 0 \end{cases}$

Example 7.3. *Subgradient of* $f(x) = max\{f_1(x), f_2(x)\}$, *where* f_1, f_2 *differentiable and convex.* $g_i = \begin{cases} \triangledown f_1(x) & f_1(x) > f_2(x) \\ \triangledown f_2(x) & f_1(x) > f_2(x) \\ \text{Any point on the line segment between } \triangledown f_1(x) \text{ and } \triangledown f_2(x) & f_1(x) = f_2(x) \end{cases}$

$h(v) : g(w) + z^T(v - w)$– a function of variable v is tangent to g, because it's a linear function, and only intercepts $g(v)$ at one and only one point, and is always smaller or equal than g.
In this book, it mentions subgradients are useful for nonsmooth optimization because of the following properties:

proposition 7.1. *For any convex function* $g : R^p \rightarrow R$, *a point* w *in* R^p *is a global minimum of* g *iff the condition* $0 \in \partial g(w)$ *holds*

Proof: If part is easy, if $0 \in \partial g(w)$, then $g(w) \leq g(v)$ for all v, which means w is a global minimum.
Only if: I'm still trying to undertand the proof in the book

7.4.2 Subdifferential

subdifferential is defined as the collection of subgradients:

$$\partial g(w) := \{z \in R^p \mid g(w) + z^T(v - w) \leq g(v)\} \tag{61}$$

7.4.3 Dual norm and optimality conditions

The dual norm is defined as

$$\Omega^*(z) := max\ z^T w \ \text{ such that } \Omega(w) \leq 1 \tag{62}$$

proposition 7.2. *For problem (1.1) where* Ω *is a norm on* R^p. *A vector* w *in* R^p *is optimal iff* $-\frac{1}{\lambda}\triangledown f(w) \in \partial\Omega(w)$ *with*

$$\begin{aligned} \partial\Omega(w) = &\{z \in R^p;\ \Omega^*(z) \leq 1\} && \text{if } w = 0 \\ &\{z \in R^p;\ \Omega^*(z) = 1 \text{ and } z^T w = \Omega(w)\}, && \text{otherwise} \end{aligned}$$

proposition 7.3. *A vector* w *is a solution for the Lasso problem iff*

$$\begin{aligned} \forall j = 1, ..., p \quad &\{|X_j^T(y - Xw)| \leq n\lambda && \text{if } w_j = 0 \\ &\{X_j^T(y - Xw) = n\lambda sign(w_j) && \text{if } w_j \neq 0 \end{aligned}$$

7.4.4 Fenchel Conjugate and Duality Gaps

f^*, the conjugate of f, is defined as

$$f^*(z) := sup[z^T w - f(w)] \tag{63}$$

proposition 7.4. *Let* Ω *be a norm on* R^p. *The following equality holds for any* $z \in R^p$.

$$\begin{aligned} sup[z^T w - \Omega(w)] = l_{\Omega^*}(w) = &0 \ if\ \Omega^*(z) \leq 1 \\ &+\infty \quad otherwise \end{aligned}$$

where the dual norm $\Omega^*(z)$ is defined as

$$\Omega^(z) = max < w, z > \tag{64}$$

subject to the norm of w is less than or equal to 1

proposition 7.5. *Let w be a vector in R^p, and z be a vector in the domain of f^*. We have then the following inequality*

$$f(w) + f^*(z) \geq w^T z \tag{65}$$

proposition 7.6. *If f^* and Ω^* are respectively, the Fenchel conjugate of a convex and differentiable function f and the dual norm of Ω, then we have*

$$max \ -f^*(z) \leq min \ f(w) + \lambda\Omega(w) \tag{66}$$

8 Proximal methods

8.1 Proximal operator

The proximal operator is:

$$\underset{z}{argmin}\ \frac{1}{2t}\|x-z\|_2^2 + h(z) \tag{67}$$

Proximal method is applied to minimize the function:

$$f(x) = g(x) + h(x) \tag{68}$$

where g is differentiable and convex and h non-smooth but convex. And sum of two convex functions is convex.

8.2 Pricipal of proximal methods

My understanding: A smooth term combined with a non-smooth term makes the problem impossible to be solved by just taking the gradient and iteratively update the value of x, since the whole function we need to examine is not always differentiable. Therefore, we try to separately consider the smooth term and the non-smooth term. For the smooth term, we incorporate the idea of gradient descent; for the combination part, we induce a concept called smooth operator, which would find the minimizer of the add-up value of the smooth and non-smooth term

Step of proximal methods:

- Step 1:Quadratically approximate the smooth term g, and use the method of gradient descent to update x, and we may call the updated value x^+
- Step 2:The above x^+ actually servers as intermediate, we need to further compute the proximal operator of x^+, we may denote the vector get for the next iteration as x^{++}

Some knowledge needed before doing proximal methods:

- Gradient descent:choose initial point $x^{(0)} \in \mathbb{R}^n$, repeat $x^{(k)} = x^{(k-1)} - t_k \bigtriangledown f(x^{(k-1)})$
- Second-order Taylor expansion:$f(y) = f(x) + \bigtriangledown f(x)^T (y-x) + \frac{1}{2}(y-x) \bigtriangledown^2 f(x)(y-x)$
- Quadratic approximation: Replace hessian by $\frac{1}{t}I$

$$f(y) = f(x) + \bigtriangledown f(x)^T (y-x) + \frac{1}{2t}\|y-x\|_2^2 \tag{69}$$

Whole process illustrated by equation:

$$\begin{aligned} x^{++} &= \underset{z}{argmin}\, \overline{g}(z) + h(z) \\ &= \underset{z}{argmin}\, g(x) + \nabla g(x)^T (z - x) + \frac{1}{2t}||z - x||_2^2 + h(z) \\ &= \underset{z}{argmin}\, \frac{1}{2t}||z - (x - t \bigtriangledown g(x))||_2^2 + h(z) \end{aligned}$$

where $\overline{g}(z)$ is just the quadratic approximation of the smooth term.

Split process illustrated by equation:

- Update the vector of the smooth term

$$x^+ = x^{(k-1)} - t_k \bigtriangledown g(x^{(k-1)}) \tag{70}$$

- Then, according to the idea of proximal operator, combine the smooth term and the non-smooth term

$$prox_t(x) = \underset{z}{argmin}\, \frac{1}{2t}\|x - z\|_2^2 + h(z) \tag{71}$$

$$\begin{aligned} x^{(k)} &= \underset{z}{argmin}\, \frac{1}{2t}||z - (x^{(k-1)} - t_k \bigtriangledown g(x^{(k-1)}))||_2^2 + h(z) \\ &= \underset{z}{argmin}\, \frac{1}{2t}\|z - x^+\|_2^2 + h(z) \\ &= \underset{z}{argmin}\, \frac{1}{2t}\|x^+ - z\|_2^2 + h(z) \\ &= prox_{t_k}(x^+) \end{aligned}$$

Reformation of proximal methods:

- We may try to set proximal mathods to have the form of

$$x^{(k)} = x^{(k-1)} - t_k G(x^{(k-1)}) \quad \text{for some } G \tag{72}$$

- We may induce $G_{t_k}(x)$ as

$$G_{t_k}(x) = \frac{x - prox_{t_k}(x - t_k \bigtriangledown g(x))}{t_k} \tag{73}$$

- Proof:

$$\begin{aligned} x^{(k-1)} - t_k G_{t_k}(x^{(k-1)}) &= x^{(k-1)} - t_k \frac{x^{(k-1)} - prox(x^{(k-1)} - t_k \bigtriangledown g(x^{(k-1)}))}{t_k} \\ &= prox(x^{(k-1)} - t_k \bigtriangledown g(x^{(k-1)})) \\ &= prox(x^+) \\ &= x^{(k)} \end{aligned}$$

Step length determination:We may use backtracking line-search.

Idea:Check if the smooth term produce smaller value

- Fix parameter β between 0 and 1
- Check if $g(x^{(k)}) > g(x^{(k-1)}) - t_k \bigtriangledown g(x^{(k-1)})G_{t_k}(x^{(k-1)}) + \frac{t}{2}\|G_{t_k}(x^{(k-1)})\|_2^2$
- If yes, shrink t_k, else do nothing.

Example 8.1. *Lasso:*

$$\begin{aligned} f(\beta) &= g(\beta) + h(\beta) \\ &= \frac{1}{2}\|y - X\beta\|_2^2 + \lambda||\beta||_1 \end{aligned}$$

Steps:

- *We first take the gradient of g wrt to β, which is the first term. And from this gradient, we can update β^+ using gradient descent, and get a vector, we may call it β^+*
- *Gradient of the smooth term:*
$$\bigtriangledown g(\beta) = -X^T(y - X\beta) \tag{74}$$
- *Update the vector β^+*
$$\beta^+ = \beta - t \bigtriangledown g(\beta) \tag{75}$$
- *Then we use the formula*

$$\begin{aligned} prox(\beta^+) &= \underset{z}{argmin}\ \frac{1}{2t}\|\beta^+ - z\|_2^2 + h(z) \\ &= \underset{z}{argmin}\ \frac{1}{2t}\|\beta^+ - z\|_2^2 + \lambda\|z\|_1 \end{aligned}$$

- *Now we observe that the above optimization problem is actually soft-thresholding problem, therefore we have*

$$z_i = \begin{cases} \beta_i^+ - \lambda & \beta_i > \lambda \\ 0 & -\lambda \le \beta_i^+ \le \lambda \\ \beta_i^+ + \lambda & \beta_i < -\lambda \end{cases}$$

Soft-thresholding:Is actually Lasso with $X = I$

$$\min_{\beta} \frac{1}{2}\|y - \beta\|_2^2 + \lambda\|\beta\|_1 \tag{76}$$

- Note this function is convex but not always differentiable, therefore it has some sharp coners, and we must consider different cases separately.
- We may use optimal condition:if x^* is a minimizer if and only if 0 is a subgradient of f at x^*

$$\begin{aligned} & 0 \in \partial(\frac{1}{2}\|y - X\beta\| + \lambda\|\beta\|) \\ \implies & 0 \in -X^T(y - X\beta) + \lambda\partial\|\beta\|_1 \\ \implies & X^T(y - X\beta) = \lambda v \quad \text{for some v in } \partial\|\beta\|_1 \end{aligned}$$

- Note that in one-dimensional case, $\partial\|\beta\|_1$ has the shape of V, therefore, we can easily find that

$$v_i = \begin{cases} \{1\} & \beta_i > 0 \\ [-1, 1] & \beta_i = 0 \\ \{-1\} & \beta_i < 0 \end{cases}$$

- Now we substitute X with I

$$y - X\beta = \lambda v \tag{77}$$

Example 8.2. *Ridge regression:*

$$\begin{aligned} f(\beta) &= g(\beta) + h(\beta) \\ &= \frac{1}{2}\|y - X\beta\|_2^2 + \lambda||\beta||_2 \end{aligned}$$

Steps:

- *We first take the gradient of g wrt to β, which is the first term. And from this gradient, we can update β^+ using gradient descent, and get a vector, we may call it β^+*
- *Gradient of the smooth term:*

$$\triangledown\, g(\beta) = -X^T(y - X\beta) \tag{78}$$

- *Update the vector β^+*

$$\beta^+ = \beta - t \triangledown\, g(\beta) \tag{79}$$

- *Then we use the formula*

$$prox(\beta^+) = \frac{1}{1+\lambda}\beta^+ \tag{80}$$

9 Coordinate Descent

In coordinate descent, an initial $x^{(0)}$ is setted, and iteratively update $x^{(k)}$ for k=1,2,3...

$$
\begin{aligned}
&x_1^{(k)} \in argmin_{x_1} f(x_1, x_2^{(k-1)}, x_3^{(k-1)}, ..., x_n^{(k-1)}) \\
&x_2^{(k)} \in argmin_{x_2} f(x_1^{(k)}, x_2, x_3^{(k-1)}, ..., x_n^{(k-1)}) \\
&x_3^{(k)} \in argmin_{x_3} f(x_1^{(k)}, x_2^{(k)}, x_3, ..., x_n^{(k-1)}) \\
&..... \\
&x_n^{(k)} \in argmin_{x_n} f(x_1^{(k)}, x_2^{(k)}, x_3^{(k)}, ..., x_n)
\end{aligned}
$$

Example 9.1. *Linear regression:*

$$min_\beta \frac{1}{2}||y - X\beta||_2^2 \tag{81}$$

Then we need to minimize for each β_i

$$0 = \triangledown_i f(\beta) = X_i^T (X\beta - y) = X_i^T (X_i\beta_i + X_{-i}\beta_{-i} - y) \tag{82}$$

Reform the equation, we can get

$$\beta_i = \frac{X_i^T (y - X_{-i}\beta_{-i})}{X_i^T X_i} \tag{83}$$

Example 9.2. *Laaso:*