A GRADUATE COURSE

IN

OPTIMIZATION

A GRADUATE COURSE IN OPTIMIZATION

CIE6010 Notebook

Prof. Yin Zhang

The Chinese University of Hongkong, Shenzhen

香港中文大學（深圳）
The Chinese University of Hong Kong, Shenzhen

Contents

Acknowledgments

This book is from the CIE6010 in fall semester, 2018.

CUHK(SZ)

Notations and Conventions

X	Set
$\inf X \subseteq \mathbb{R}$	Infimum over the set X
$\mathbb{R}^{m \times n}$	set of all $m \times n$ real-valued matrices
$\mathbb{C}^{m \times n}$	set of all $m \times n$ complex-valued matrices
x_i	ith entry of column vector $\boldsymbol{x}$
a_{ij}	(i, j)th entry of matrix $\boldsymbol{A}$
$\boldsymbol{a}_i$	ith column of matrix $\boldsymbol{A}$
$\boldsymbol{a}_i^{\mathrm{T}}$	ith row of matrix $\boldsymbol{A}$
$\mathbb{S}^n$	set of all $n \times n$ real symmetric matrices, i.e., $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ and $a_{ij} = a_{ji}$ for all i, j
$\mathbb{H}^n$	set of all $n \times n$ complex Hermitian matrices, i.e., $\boldsymbol{A} \in \mathbb{C}^{n \times n}$ and $\bar{a}_{ij} = a_{ji}$ for all i, j
$\boldsymbol{A}^{\mathrm{T}}$	transpose of $\boldsymbol{A}$, i.e, $\boldsymbol{B} = \boldsymbol{A}^{\mathrm{T}}$ means $b_{ji} = a_{ij}$ for all i, j
$\boldsymbol{A}^{\mathrm{H}}$	Hermitian transpose of $\boldsymbol{A}$, i.e, $\boldsymbol{B} = \boldsymbol{A}^{\mathrm{H}}$ means $b_{ji} = \bar{a}_{ij}$ for all i, j
$\mathrm{trace}(\boldsymbol{A})$	sum of diagonal entries of square matrix $\boldsymbol{A}$
$\mathbf{1}$	A vector with all 1 entries
$\mathbf{0}$	either a vector of all zeros, or a matrix of all zeros
$\boldsymbol{e}_i$	a unit vector with the nonzero element at the ith entry
$\mathcal{C}(\boldsymbol{A})$	the column space of $\boldsymbol{A}$
$\mathcal{R}(\boldsymbol{A})$	the row space of $\boldsymbol{A}$
$\mathcal{N}(\boldsymbol{A})$	the null space of $\boldsymbol{A}$
$\mathrm{Proj}_{\mathcal{M}}(\boldsymbol{A})$	the projection of $\boldsymbol{A}$ onto the set $\mathcal{M}$

Chapter 4

Week4

4.1. Wednesday

You need to take care of several things during your assignment:

1. Do Not Repeat Computation! e.g., if you enter

$$|f(x^{k+1}) - f(x^k)| \leq 10^{-\varepsilon}|f(x^k)|,$$

   this is very bad, since you evaluated this function three times.

2. Arrange Computation Properly. e.g., compute

$$\boldsymbol{x}\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}$$

   is very expensive, but $\boldsymbol{x}(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A})$ is not. Be aware of the size of matrices / vectors.

3. Appreciate sparsity, e.g., to compute $\boldsymbol{A}\boldsymbol{D}\boldsymbol{A}^{\mathrm{T}}$ is bad if using $\mathrm{D} = \mathrm{diag}(\mathrm{d})$, while using $\mathrm{D} = \mathrm{sparse}(1:\mathrm{n}, 1:\mathrm{n}, \mathrm{d})$ is better.

Your code should be at least faster than the testing script.

4.1.1. Local Convergence Rate

Definition 4.1 [Q_1 Factor] Let $x^k \to x^*$ and $e_k := \|x^k - x^*\| \to 0$. The Q_1 factor of $\{x^k\}$ is given as:

$$Q_1 = \limsup_{k\to\infty} \frac{\|\boldsymbol{e}^{k+1}\|}{\|\boldsymbol{e}^k\|}$$

- The sequence $\{x^k\}$ is convergent if $Q_1 \in [0,1]$.

- $$Q_1 = \begin{cases} 0, & \text{Q-super linear convergence} \\ \rho \in (0,1), & \text{Q-linear convergence} \\ 1, & \text{Q-sublinear} \end{cases}$$

R Q-linear convergence is not always so good, e.g., $\rho = .999$ may require 20000 iterations, while $\rho = .1$ may only require 20.

Linear is always better than sublinear; super linear is always btetter than linear. e.g.,

$$\{e_1^k\} = \{\beta^k\}, \qquad \{e_2^k\} = \{\frac{1}{k^p}\} \text{ for } p = 1/2, 1, 2$$

Then $Q_1(e_1^k) = \beta \in (0,1)$; and $Q_1(e_2^k) = \lim_{k\to\infty} \frac{(k+1)^p}{k^p} = 1$. When will $e^k \leq \varepsilon$?

$$\beta^k \leq \varepsilon \implies k \geq (\frac{-1}{\ln \beta}) \ln \frac{1}{\varepsilon} = O(\ln \frac{1}{\varepsilon})$$
$$\frac{1}{k^p} \leq \varepsilon \implies k \geq (\frac{1}{\varepsilon})^{1/p}$$

If $\varepsilon = 10^{-8}$, $O(\ln \frac{1}{\varepsilon})$ is not growing very fast. For $p = 1$ case, $k \geq O(\frac{1}{\varepsilon})$

We have a faster type of convergence:

Definition 4.2 [Q_2 Factor]

$$Q_2 = \limsup_{k\to\infty} \frac{e^{k+1}}{(e^k)^2}$$

If $Q_2 = M < +\infty$,i.e., $e^{k+1} = O((e^k)^2)$, then $\{x^k\} \to x^*$ Q-quadratically.

Newton's method generally gives us the quadratic convergence.

4.1.2. Newton's Method

The newton's method requires to solve a non-linear system of equations: ($\nabla f(x) = 0$)

$$F : \mathbb{R}^n \to \mathbb{R}^n, F(x) = 0.$$

We don't know how to solve non-linear system in general. But Newton gives us the remediation: do the linearization.

$$F(x + d) \approx F(x) + \langle F(x), \boldsymbol{d} \rangle = 0$$

To get the solution, it suffices to solve

$$\boldsymbol{d} = -(F'(\boldsymbol{x}))^{-1} F(\boldsymbol{x}),$$

and hence $\boldsymbol{x} \leftarrow \boldsymbol{x} + \alpha \boldsymbol{d}$.

Algorithm. Choose $\boldsymbol{x}^0$.

For $k = 0, 1, 2, \ldots$, solve the system w.r.t. $\boldsymbol{d}$:

$$\nabla^2 f(\boldsymbol{x}^k) \boldsymbol{d} = -\nabla f(\boldsymbol{x}^k)$$

Set $\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \alpha^k \boldsymbol{d}^k$.

End.

Interpretation. In order to minimize a strictly convex function $f(x)$, we find

$$f(\boldsymbol{x} + \boldsymbol{d}) \approx f(\boldsymbol{x}) + \langle \nabla f(\boldsymbol{x}), \boldsymbol{d} \rangle + \frac{1}{2} \boldsymbol{d}^{\mathrm{T}} \nabla^2 f(\boldsymbol{x}) \boldsymbol{d} := q(\boldsymbol{d})$$

It suffices to minimize $q(\boldsymbol{d})$:

$$\nabla q(\boldsymbol{d}) = 0 \Longleftrightarrow \nabla f(\boldsymbol{x}) + \nabla^2 f(\boldsymbol{x}) \boldsymbol{d} = 0$$

How to guarntee $\boldsymbol{d}$ is the descent direction? It becomes an art.

Convergence of Rate. For $F(x) = 0$, suppose$\{\boldsymbol{x}^k\}$ is generated by Newton. As $x^k \to x^*$, $F(x^*) = 0$. From

$$x^{k+1} = x^k - [F'(x^k)]^{-1}F(x^k),$$

we find

$$\begin{aligned} x^{k+1} - x^* &= x^k - x^* - [F'(x^k)]^{-1}\left(F(x^k) - F(x^*)\right) \\ &= [F'(x^k)]^{-1}\left(F(x^*) - F(x^k) - F'(x^k)(x^* - x^k)\right) \end{aligned}$$

It follows that

$$\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^*\| \leq \|[F'(x^k)]^{-1}\|O(\|\boldsymbol{x}^k - \boldsymbol{x}^*\|^2) = O(\|\boldsymbol{x}^k - \boldsymbol{x}^*\|^2)$$

During this deviation we assume two things:

- Step-size is 1!
- The limit exists.

Hence, in practice, to implement Newton's method, 1 is the first choice; but gredient descent method is not.

Newton's method is good for nice problems, i.e., the funciton is convex, and the inverse of gredient is easy to solve.

In machine learning most time we implement the gredient descent method.

Learn and implement these things by yourself:

- Luo's note: Lecture #4, P7, Nestorov Accelerated Method

$\{a_r\}, r = 0, 1, 2, \ldots$ we have

$$a_r = \frac{1}{2}(1 + \sqrt{1 + 4a_{r-1}^2}) \text{ with } a_0 = 0$$

From a_r we generate $t^r = (a_r - 1)/a_{r+1}$

Algorithm. Set $x^0 = x^1 = 0$.

for $r = 1, 2, \ldots$

compute a^{r+1}, t^r

$\boldsymbol{y}^{r+1} = (1 + t^r)\boldsymbol{x}^r - t^r \boldsymbol{x}^{r-1}$

$\boldsymbol{x}^{r+1} = y^{r+1} - \frac{1}{L}\nabla f(\boldsymbol{y}^{r+1})$

end