Convex Optimization Hw2

Created	@May 28, 2025 8:59 PM
Class	Convex Optiization
Type	Hw
Reviewed	☐

Convex Optimization Hw2

B10901052

聶臣璟

Problem 1

(a)

$f(x) = \log(1 + \exp(a^T x)) + \frac{1}{2}\sum_{i=1}^{5} x_i^2$

Let $z = a^T x$, and define the sigmoid function: $\sigma(z) = \frac{e^z}{1+e^z}$

Since $\frac{d}{dz}\log(1+\exp(z)) = \sigma(z)$, $\frac{d^2}{dz^2}\log(1+\exp(z)) = \sigma(z)(1-\sigma(z))$,

$\frac{d}{dx_1}\log(1+\exp(a^T x)) = \sigma(a^T x)\cdot a_1$,

$\frac{d}{dx_1}\frac{1}{2}\sum_{i=1}^{5} x_i^2 = x_i$,

which leads to the full gradient:

$\nabla f(x) = \sigma(a^T x)\cdot a + x$

$\frac{d}{dx_1}\sigma(a^T x)\cdot a_1 = \sigma(a^T x)(1-\sigma(a^T x))\cdot a_1^2$

$\frac{d}{dx_2}\sigma(a^T x)\cdot a_1 = \sigma(a^T x)(1-\sigma(a^T x))\cdot a_1 \cdot a_2$

which leads to the full hessian matrix

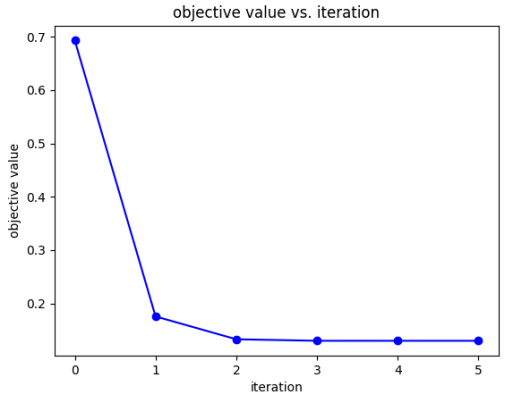
$\nabla^2 f(x) = \sigma(a^T x)(1-\sigma(a^T x))\cdot aa^T + I_5$

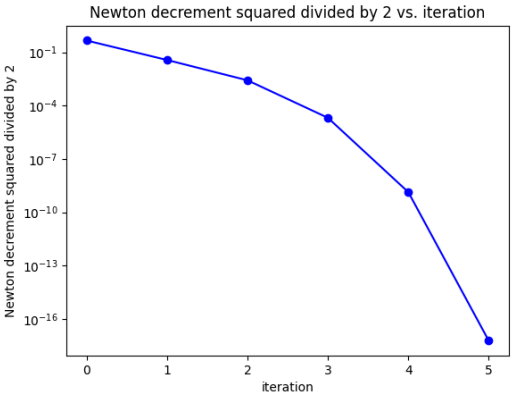
where I_5 is the 5×5 identity matrix.

(d)

This is the optimal value and x generated by my damped Newton method with backtracking line search.

```
Optimal value: 0.13001489223524 63
Optimal x: [-0.05257343 -0.10514686 -0.15772029 -0.21029372 -0.26286714]
```





(e)

The optimal value and solution derived by cvxpy are as below. The code is also implemented in “newton_solver.py”.

The difference in optimal value is about 7.42×10^{-10}, and the norm 2 difference between the solution vector x is about 1.99×10^{-5}. Both differences are small enough for application. The difference may be caused by the accuracy of floating point application or the stopping point of the algorithm.

```
cvxpy Optimal value: 0.130014892976904 44
cvxpy Optimal x: [-0.05257611 -0.10515223 -0.15772834 -0.21030446 -0.26288057]
Difference in optimal value: 7.416581515595766e-10
Difference in optimal x: 1.9918399423217382e-05
```

Problem 2

(a)

We consider the barrier formulation of a constrained optimization problem:

$\min_{x \in \mathbb{R}^n} \quad t f_0(x) + \phi(x)$

And $\phi(x)$ is defined as:

$$\phi(x) = -\sum_{i=1}^{m} \log(c_i - b_i^T x)$$

Do the gradient term-by-term,

$$\frac{d}{dx_1}(\log(c_i - b_i^T x)) = \frac{-b_{i1}}{c_i - b_i^T x} \Rightarrow \frac{d}{dx_1}(-\sum_{i=1}^{m} \log(c_i - b_i^T x)) = \sum_{i=1}^{m} \frac{b_{i1}}{c_i - b_i^T x}$$

$$\Rightarrow \nabla_x \phi(x) = \sum_{i=1}^{m} \frac{b_i}{c_i - b_i^\top x}$$

Differentiate the gradient term-by-term to get the hessian:

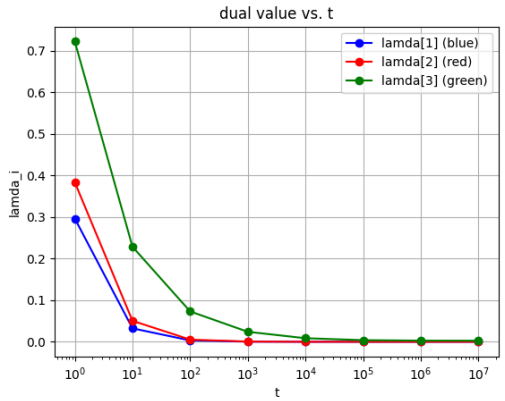
$$\frac{d}{dx_1} \sum_{i=1}^{m} \frac{b_{i1}}{c_i - b_i^T x} = \sum_{i=1}^{m} \frac{b_{i1}^2}{(c_i - b_i^T x)^2}$$

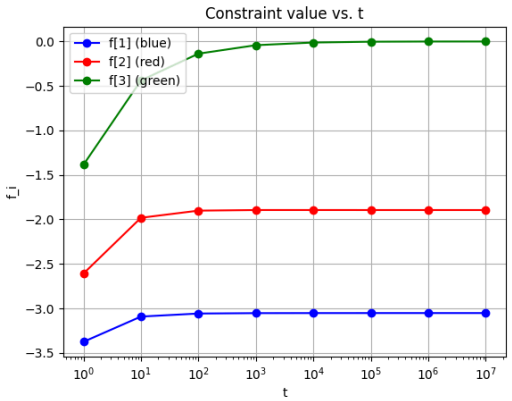
$$\frac{d}{dx_2} \sum_{i=1}^{m} \frac{b_{i1}}{c_i - b_i^T x} = \sum_{i=1}^{m} \frac{b_{i1} b_{i2}}{(c_i - b_i^T x)^2}$$

$$\Rightarrow \nabla_x^2 \phi(x) = \sum_{i=1}^{m} \frac{b_i b_i^\top}{(c_i - b_i^\top x)^2}$$

(c)

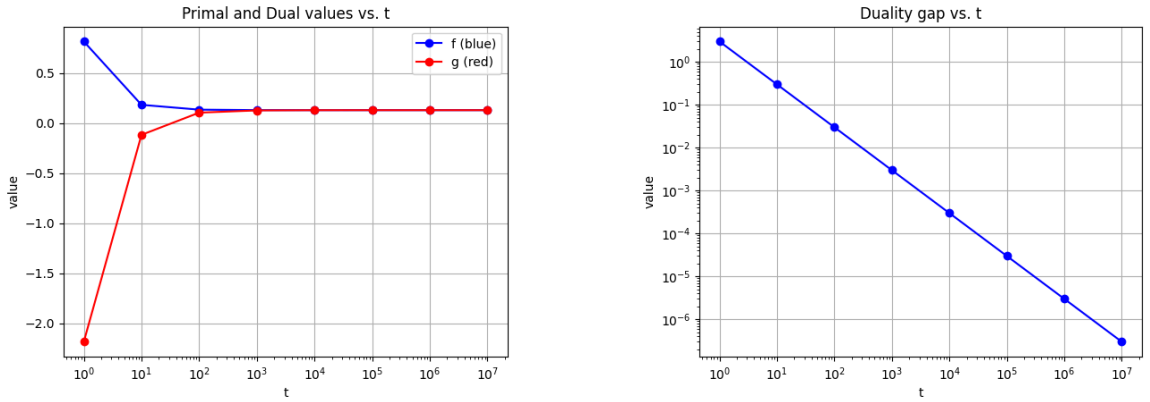
These graphs below are $\lambda_i \; vs. \; t$, and $f_i \; vs. \; t$ respectively. As shown in the figures, all the λ_i value are greater than 0 and all the f_i are smaller than 0, which satisfied the complementary slackness condition that $\lambda_i \cdot f_i \leq 0$. We can also observe that both λ_3 and f_3 are approaching to 0 from above and below respectively, and the other constraints still have a greater differences between λ_i and f_i, thus **inequality constraints 3 is activated** at the end.



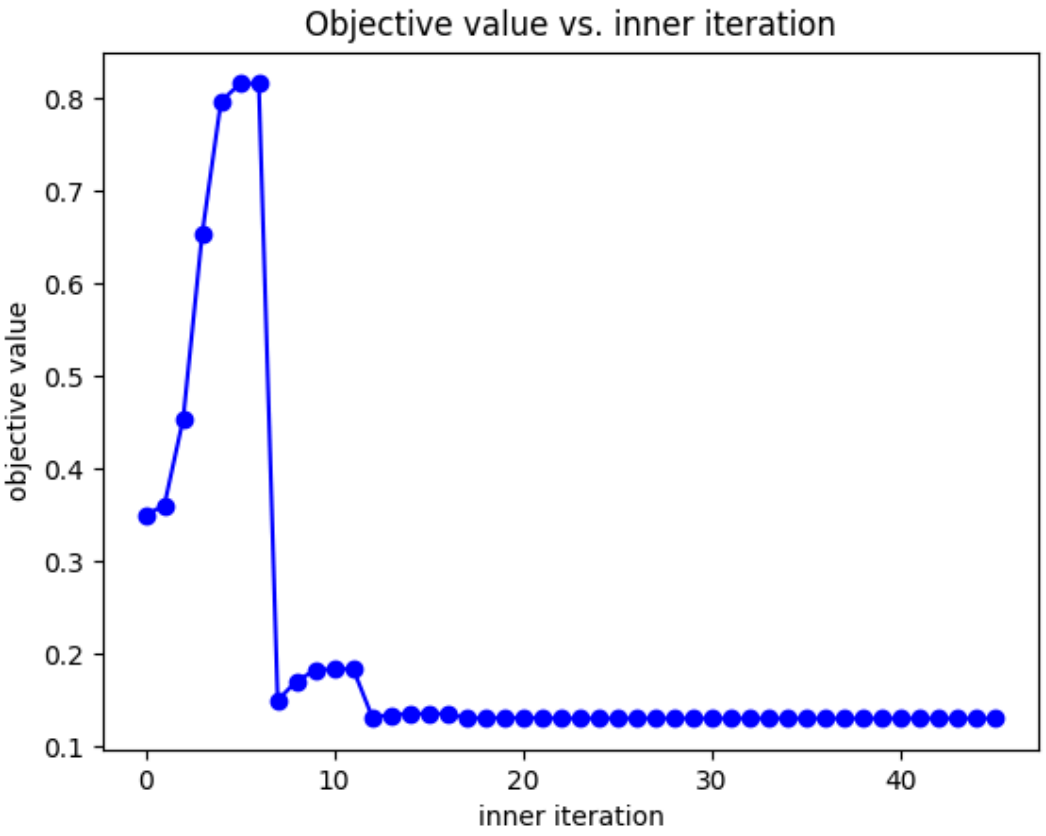


The figure of primal value f and dual value g are shown as the graph below. As both value approaching 0, the difference between f and g is shown as the

graph in the right. We can observe that the duality gap is getting small and closer to 0.



This is the graph of objective value $f_0(x)\ vs.\ inner\ iteration.$



There are **8** centering steps and **46** newton steps in total before the program terminated.

```
Optimal x: [-0.05264442 -0.10528883 -0.16059632 -0.2105775  -0.26055877]
Optimal value: 0.1300217956803816
Inner iterations per step: 46
Total center steps: 8
```

(d)

The optimal value and solution derived by cvxpy are as below. The code is also implemented in “barrier_method.py”.

The difference in optimal value is about 9.74×10^{-8}, and the norm 2 difference between the solution vector x is about 2.60×10^{-5}. Both differences are small enough for application. The difference may be caused by the accuracy of floating point application or the stopping point of the algorithm.

```
cvxpy Optimal value: 0.13002169823078316
cvxpy Optimal x: [-0.05264451 -0.10528901 -0.16057843 -0.21057802 -0.26057761]
Difference in optimal value: 9.744959844626422e-08
Difference in optimal x: 2.599083315100353e-05
```