Linear and Convex Optimization Homework 06

Qiu Yihang, 2021/11/08-11/10

0. Preparation

Complete gd.py. The completed gd.py (with `gd_const_ss` function) is enclosed in the zip file.

1.(a) ***Solution:***

f is L-smooth $\Leftrightarrow$ $\nabla^2 f \preceq L\boldsymbol{I}$, where $\boldsymbol{I}$ is the identity matrix, **i.e.** $\mathbf{diag}(1,1)$.

Thus, $\nabla^2 f - L\boldsymbol{I} = \mathbf{diag}(\gamma - L, 1 - L) \preceq \boldsymbol{O}$, **i.e.** $\begin{cases}\gamma - L \leq 0 \\ 1 - L \leq 0\end{cases}$.

Therefore, $L \geq \max\{\gamma, 1\}$, **i.e.** $L_{\min} = \max\{\gamma, 1\}$. ■

(b) ***Solution:***

f is m-strongly convex $\Leftrightarrow$ $\nabla^2 f \succeq m\boldsymbol{I}$, where $\boldsymbol{I}$ is the identity matrix, **i.e.** $\mathbf{diag}(1,1)$.

Thus, $\nabla^2 f - m\boldsymbol{I} = \mathbf{diag}(\gamma - m, 1 - m) \succeq \boldsymbol{O}$, **i.e.** $\begin{cases}\gamma - m \geq 0 \\ 1 - m \geq 0\end{cases}$.

Therefore, $m \leq \min\{\gamma, 1\}$, **i.e.** $m_{\max} = \min\{\gamma, 1\}$. ■

(c) ***Solution:***

I made some adjustments to the given p1.py and utils.py. The code is enclosed in the zip file.

1) Step size is $\mathbf{2.2}$.

We can see from the gradients (see Fig.01) and the gap between minimum value $f(x^\star)$ and $f(x_k)$ (see Fig.02) that the algorithm **diverges**. ■

Limit the maximum number of iterations to 20 since it diverges too fast.

The visualization of the 2D trajectory of sequence x_k and the function values $f(x_k)$ is given below (see Fig.02).

```
gamma=0.1, stepsize=2.2, number of iterations=1000
[0.1 1. ][4.02466868e-07 9.10043815e+03][1.61979580e-12 8.28179745e+07]
[6.51914144e-18 7.53679855e+11][2.62373844e-23 6.85881690e+15]
[1.05596779e-28 6.24182390e+19][4.24992051e-34 5.68033324e+23]
[1.71045220e-39 5.16935213e+27][6.88400340e-45 4.70433693e+31]
[2.77058329e-50 4.28115273e+35][1.11506798e-55 3.89603656e+39]
[4.48777918e-61 3.54556398e+43][1.80618243e-66 3.22661857e+47]
[7.26928588e-72 2.93636427e+51][2.92564672e-77 2.67222014e+55]
[1.17747588e-82 2.43183741e+59][4.73895028e-88 2.21307860e+63]
[1.90727048e-93 2.01399849e+67][7.67613177e-99 1.83282687e+71]
[3.08938872e-104 1.66795276e+075]
```

Fig.01. Gradient Sequence (Step Size = 2.2)

Note: Since the number of iterations is quite large (set as 1000),
here we only output gradients of x_k where $k = 50n, n \in \mathbb{N}^+$.

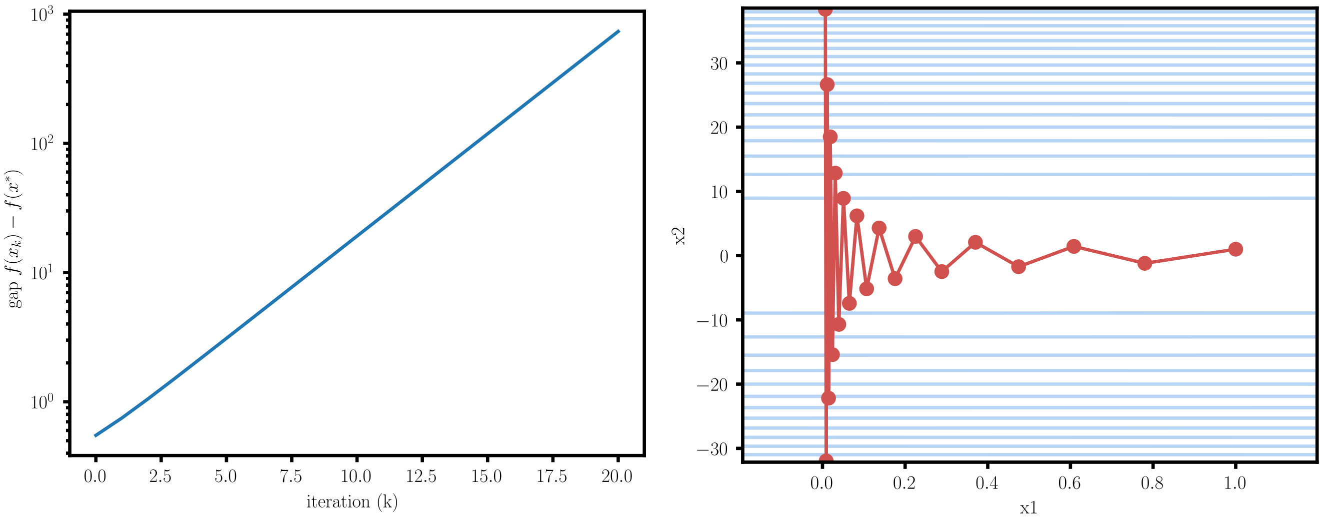


Fig.02. The Divergence of Gap and the Trajectory of x_k and $f(x_k)$ (Step Size = 2.2)

In the following cases, we set the maximum number of iterations as 100000.

2) Step size is 1.

The total number of iterations is **<u>88</u>** (see Fig.03). ■

Since it is far less than maximum number of iterations we set, the algorithm obviously **<u>converges</u>** in this case. Also, we know it converges from the change of the gap between $f(x^{\star})$ and $f(x_k)$ (see Fig.04). ■

```
In [1]: runfile('D:/Textbooks/2021-2022-1/Linear and Convex
Optimization/hw6/p1.py', wdir='D:/Textbooks/2021-2022-1/Linear and
Convex Optimization/hw6')
gamma=0.1, stepsize=1, number of iterations=88
x_k = [9.40461087e-05 0.00000000e+00]
```

Fig.03. Program Outputs (Step Size = 1)

The visualization of the 2D trajectory of sequence x_k and the function values $f(x_k)$ is given below (see Fig.04).

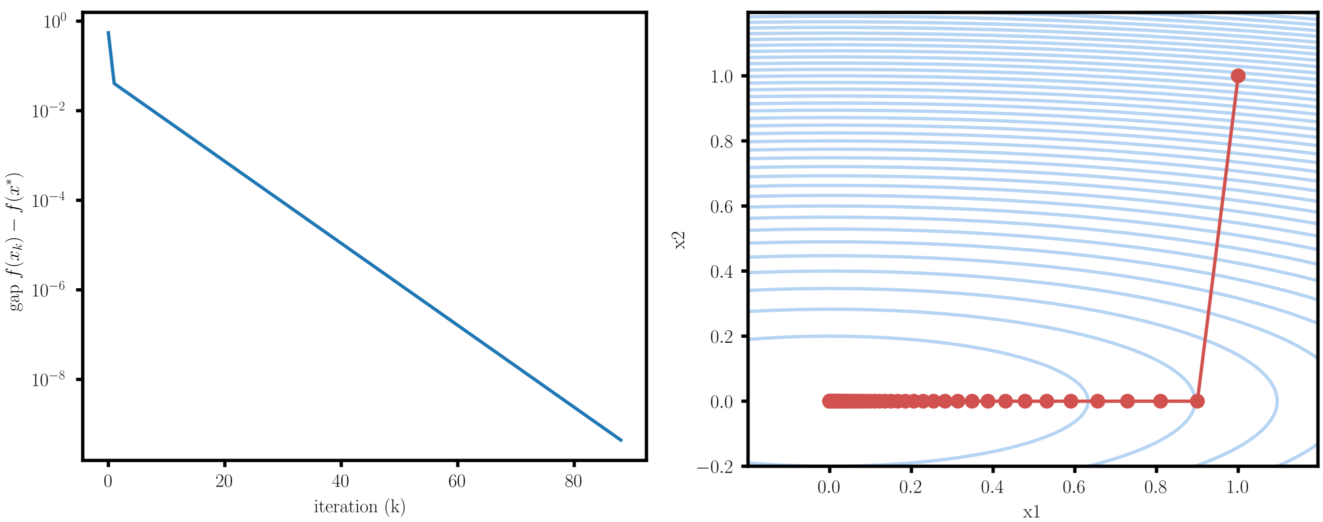


Fig.04. The Divergence of Gap and the Trajectory of x_k and $f(x_k)$ (Step Size = 1)

3) Step size is $\mathbf{0.1}$.

The algorithm **<u>converges</u>** in this case. The total number of iterations is **<u>918</u>** (see Fig.05).

```
In [2]: runfile('D:/Textbooks/2021-2022-1/Linear and Convex
Optimization/hw6/p1.py', wdir='D:/Textbooks/2021-2022-1/Linear and
Convex Optimization/hw6')
Reloaded modules: gd, utils
gamma=0.1, stepsize=0.1, number of iterations=917
x_k = [9.94199284e-05 1.09744094e-42]
```

Fig.05. Program Outputs (Step Size = 0.1)

The visualization of the change of the change of the gap between $f(x^{\star})$ and $f(x_k)$ and the 2D trajectory of sequence x_k and the function values $f(x_k)$ is given below (see Fig.06).

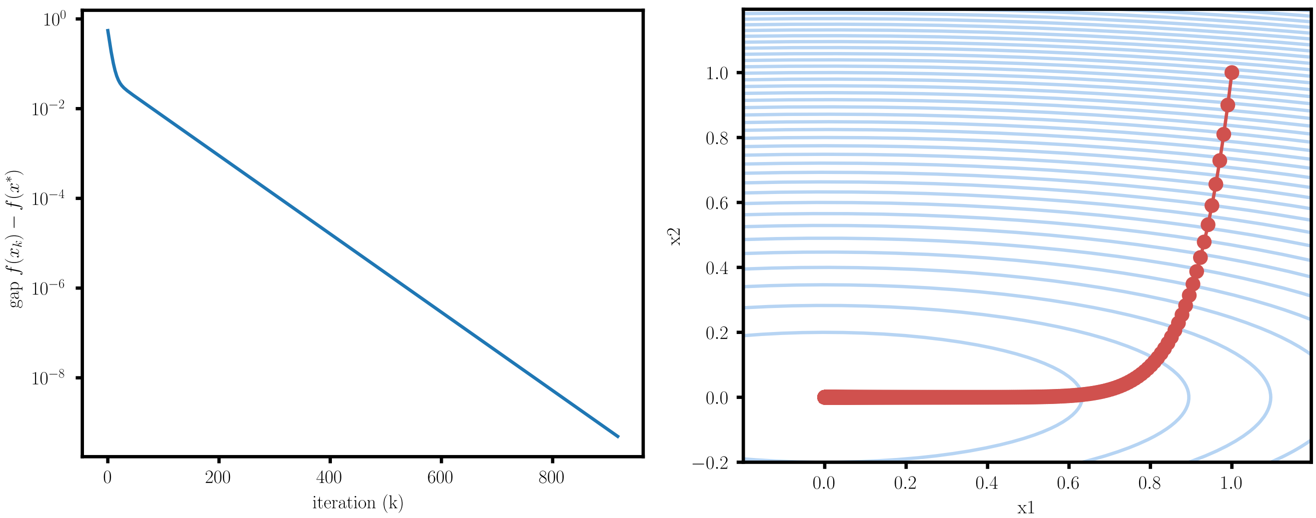


Fig.06. The Divergence of Gap and the Trajectory of x_k and $f(x_k)$ (Step Size = 0.1)

4) Step size is $\mathbf{0.01}$.

The algorithm **<u>converges</u>** in this case.

The total number of iterations is **<u>9206</u>** (see Fig.08 on the next page).

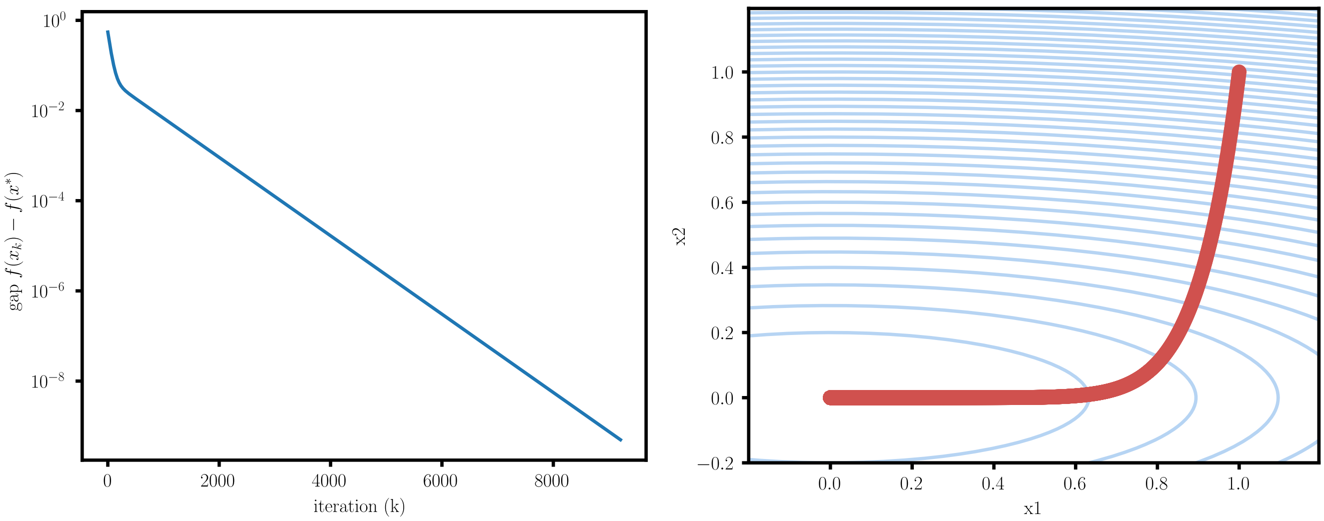


Fig.07. The Divergence of Gap and the Trajectory of x_k and $f(x_k)$ (Step Size = 0.01)

```
In [3]: runfile('D:/Textbooks/2021-2022-1/Linear and Convex
Optimization/hw6/p1.py', wdir='D:/Textbooks/2021-2022-1/Linear and
Convex Optimization/hw6')
Reloaded modules: gd, utils
gamma=0.1, stepsize=0.01, number of iterations=9206
x_k = [9.99734336e-05 6.57054607e-41]
```

Fig.08. Program Outputs (Step Size = 0.01)

The visualization of the change of the gap between $f(x^{*})$ and $f(x_k)$ and the 2D trajectory of sequence x_k and the function values $f(x_k)$ is given above (see Fig.07 on the previous page).

(d) ***Solution:***

The number of iterations we need corresponding to different γ is given in the following table and figure (See Table 01 and Fig.09). From the outputs we find 0.1 is the most appropriate γ.

$\boldsymbol{\gamma}$	1	0.1	0.01	0.001
#(Iterations)	1	88	688	4603

Table 01. The Number of Iterations in Cases with Different γ

```
In [3]: runfile('D:/Textbooks/2021-2022-1/Linear and Convex
Optimization/hw6/p1.py', wdir='D:/Textbooks/2021-2022-1/Linear and
Convex Optimization/hw6')
Reloaded modules: gd, utils
gamma=1, stepsize=1, number of iterations=1
x_k = [0. 0.]
gamma=0.1, stepsize=1, number of iterations=88
x_k = [9.40461087e-05 0.00000000e+00]
gamma=0.01, stepsize=1, number of iterations=688
x_k = [0.00099315 0.        ]
gamma=0.001, stepsize=1, number of iterations=4603
x_k = [0.00999867 0.        ]
```

Fig.09. Program Outputs (In Cases with Different γ)

It is self-evident that the number of iterations **<u>increases</u>** as γ decreases. ■

2. ***Solution:***

The source code is given in the zip-file with the name `p2.py`.

Reference: the original `p1.py` given;

https://numpy.org/doc/stable/reference/generated/numpy.linalg.solve.html

We set step size as 0.1. The result is given below.

```
In [13]: runfile('D:/Textbooks/2021-2022-1/Linear and Convex
Optimization/hw6/p2.py', wdir='D:/Textbooks/2021-2022-1/Linear and
Convex Optimization/hw6')
Reloaded modules: gd
stepsize=0.1, number of iterations=55
w_k = [1.5        1.99999532]
f(w_k) = 4.000000000021872
```

Fig.10. Program Outputs (Problem 2, Part. 01)

The solution we acquired in Problem 4 in HW5 is given in the following table.

Problem in HW5	Solving Method	Solution ($x^\star$)	Solution ($f(x^\star)$)
4(a)	Calculation by hand	$(1.5, 2)^T$	4
4(b)	Lasso	$(1.49999883, 1.99999744)^T$	4.0
4(c)	Ridge Regression	$(1.5, 2)^T$	4.0

Table 02. Solutions of Problem 4 in HW5

Now we find the solution by solving the normal equation $\boldsymbol{X^T(Xw - y) = X^TXw - X^Ty = 0}$ using `np.linalg.solve`. The result is as follows.

```
The solution calculated using numpy.linalg.solve:
w_k = [1.5 2. ]
f(w_k) = 4.0
```

Fig.11. Program Outputs (Problem 2, Part. 02)

Ignore numerical errors (all errors in this problem are of 10^{-6} or even smaller).

We find that solutions we acquire by hand, by Lasso, by Ridge Regression, by Gradient Descent and by solving the normal equation using `np.linalg.solve` is **exactly the same**.

■

3. *Solution:*

The source code is given in the zip-file with the name `p3.py`. I made some minor adjustments to the original `p3.py`.

Reference: the original `p1.py` and `p3.py` given.

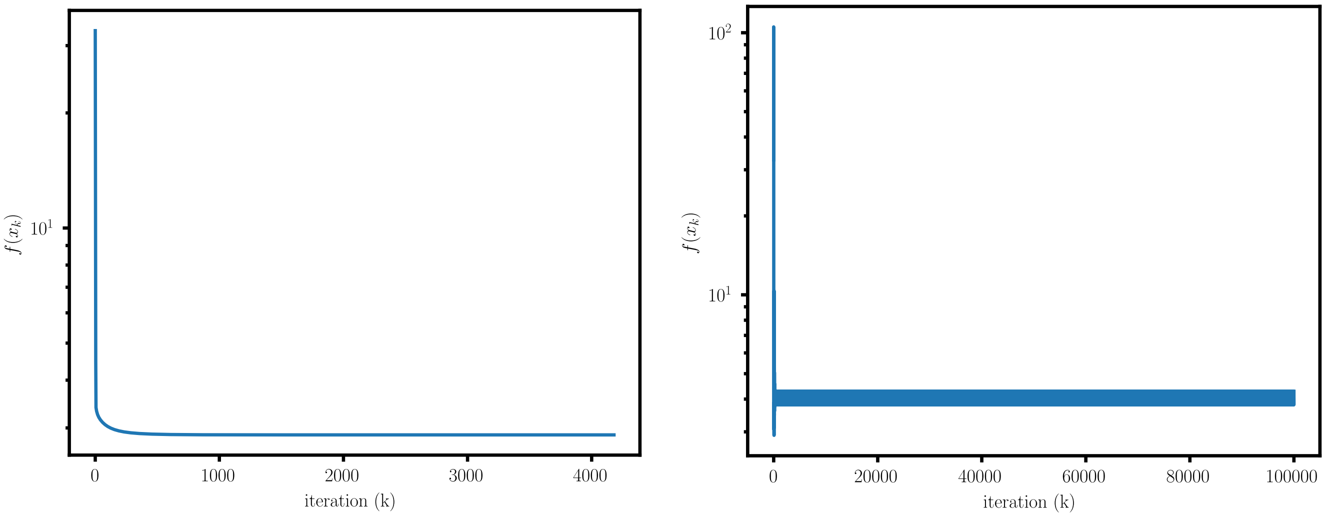
Several different step sizes are tried. Here I'd like to discuss the results when step sizes are 0.1 and 0.3 respectively. Results are shown in the figure below.

```
In [3]: runfile('D:/Textbooks/2021-2022-1/Linear and Convex
Optimization/hw6/p3.py', wdir='D:/Textbooks/2021-2022-1/Linear and
Convex Optimization/hw6')
Reloaded modules: gd, utils
stepsize = 0.1 :
#(iterations) = 4181, accuracy = 0.8666666666666667
w = [-1.4702005   4.44377554 -4.37548189]
stepsize = 0.3 :
#(iterations) = 100001, accuracy = 0.9333333333333333
w = [-2.65236875  5.2432428  -4.89998979]
```

Fig.12. Program Outputs (Problem 3, step size=0.1)

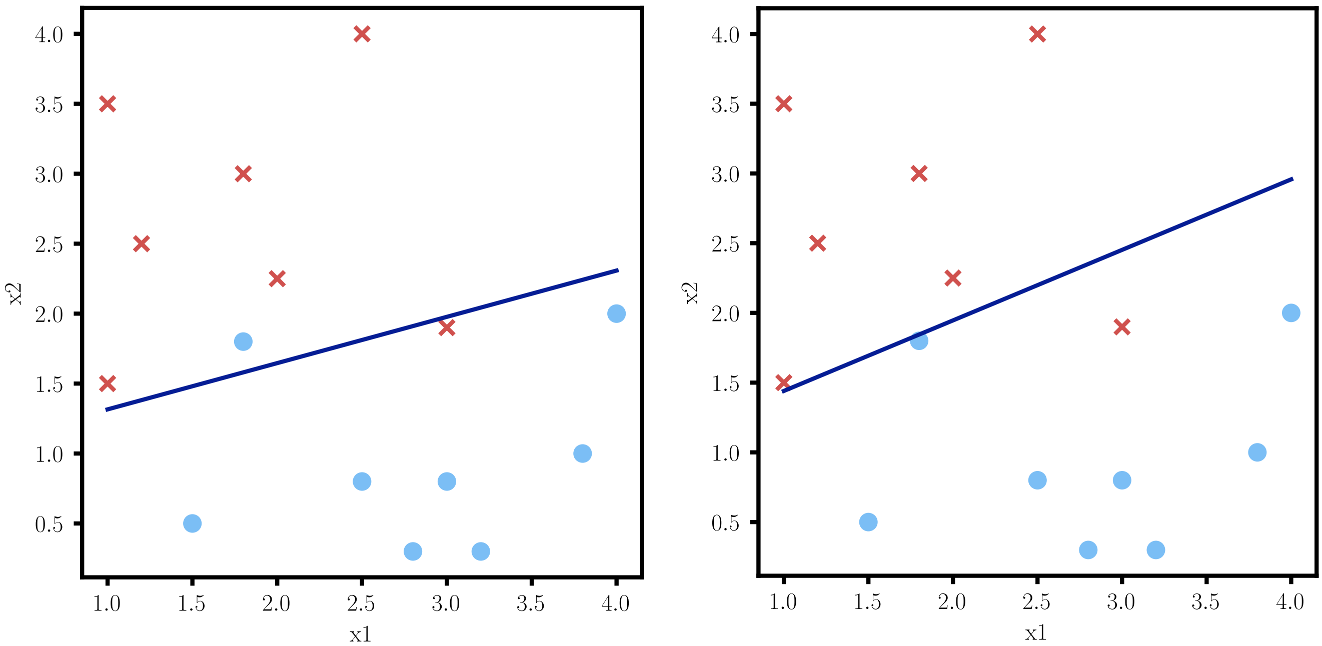
Though it seems the accuracy when $step\ size = 0.3$ is better, it can be seen from the Fig.13 (see next page) that when $step\ size = 0.3$, the model is likely to be overfitting.

The visualization of the results is also given below (see Fig.14 on the next page). We find that the result when $step\ size = 0.3$ seems to be overfitting on the training dataset. It might be weaker in prediction.



(a) *step size* = 0.1 **(b)** *step size* = 0.3

Fig.13. $f(x_k)$'s Convergence Curve (Step Size = 0.1 and 0.3)



(a) *step size* = 0.1 **(b)** *step size* = 0.3

Fig.14. The Visualization of Logistic Regression Result and Training Dataset

From the analysis above, 0.1 seems to be a more appropriate step size.

Thus, the optimal $\boldsymbol{w}^\star$ is $(-1.4702005, 4.44377554, -4.37548189)^T$. The accuracy of the classifier on the training dataset is 86.7%. ■

4. ***Solution:***

$f(x)$ is α-strongly convex $\Rightarrow$ $\nabla^2 f \succeq \alpha \boldsymbol{I}$.

$g(x)$ is β-strongly convex $\Rightarrow$ $\nabla^2 g \preceq \beta \boldsymbol{I}$ $\Rightarrow$ $-\nabla^2 g \succeq -\beta \boldsymbol{I}$.

$$\nabla^2 h = \nabla^2 (f - g) = \nabla^2 f - \nabla^2 g \succeq \alpha \boldsymbol{I} - \beta \boldsymbol{I} = (\alpha - \beta)\boldsymbol{I}.$$

Since $\alpha - \beta \geq 0$, $(\alpha - \beta)\boldsymbol{I} \succeq \boldsymbol{O}$.

Thus,

$$\nabla^2 h \succeq (\alpha - \beta)\boldsymbol{I} \succeq \boldsymbol{O}.$$

By the second-order condition for convexity, we know h is convex. *Qed.* ■