Homework 3

陈远洋　23307130322

2025 年 3 月 22 日

Problem 1

Consider the smoothed LASSO (Least Absolute Shrinkage and Selection Operator) problem:

$$\min_x \frac{1}{2}\|Ax-b\|_2^2 + \mu L_\sigma(x),$$

where $x \in \mathbb{R}^n, A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, μ is the regularization parameter. $L_\sigma(x)$ is defined as

$$L_\sigma(x) = \sum_{i=1}^{n} l_\sigma(x_i),$$

$$l_\sigma(x_i) = \begin{cases} \dfrac{1}{2\sigma}x_i^2, & |x_i| < \sigma, \\ |x_i| - \dfrac{\sigma}{2}, & \text{otherwise.} \end{cases}$$

Let $\sigma = 0.1$. Please use a gradient descent method to numerically solve above problem with A and b provided in the zip file, where $m = 512, n = 1024$. Then write a report to illustrate the method you used and present your experiment results.

Note:

- The regularization parameter μ can be small, e.g. $\mu = 1 \times 10^{-2}$. It can be adjusted by yourself.

Solution

解存在性分析

首先，该问题一定是有解的。类似于上一次的作业，我们只讨论 $x \in A$ 的零空间的正交补情况（其他情况可以转化）。记 $t^2 = min_{\|x\|_2=1}\|Ax-b\|_2^2$，则有当 $\|x\|_2 \geq \dfrac{2\|b\|_2+1}{t}$ 时。由于 $\dfrac{\|x_i\|_2^2}{2\sigma} - \|x_i\|_2 + \dfrac{\sigma}{2} = \dfrac{1}{2\sigma}(\|x_i\|_2 - \sigma)^2 \geq 0$。且 $\|Ax-b\|_2 \geq \|Ax\|_2 - \|b\|_2 \geq t\|x\|_2 - \|b\|_2 \geq 0$，此时有：原式 $\geq \dfrac{1}{2}(\|x\|_2 t - \|b\|_2)^2 + \mu(\sum_{i=1}^n |x_i| - \dfrac{n\sigma}{2}) \geq \dfrac{1}{2}t^2\|x\|_2^2 + (-2tb+\mu)\|x\|_2 + O(1)$. 这是一个关于 $\|x\|_2$ 的二次函数，可知在 $\|x\|_2 \to \infty$ 时，原式一定也区域无穷。又由于在 $\|x_i\|_2 = \sigma$ 处，两个分段函数均趋近于 $\dfrac{\sigma}{2}$，故原式在小范围内连续。所以原式一定有全局最优解。

代码构成分析

梯度下降法的几个步骤：其中目标函数为 LASSO 问题中定义的函数，在我的代码中实现在最

Algorithm 1 Gradient Descent

Input f,x_0,$tolerance$,
$k \leftarrow 0$
while $\nabla f(x_k) \geq tolerance$ **do**
    pick a descent direction p_k such that $\nabla f(x_k)^T p_k < 0$.
    pick a step size α_k.
    update x_k : $x_{k+1} \leftarrow x_k + \alpha_k p_k$.
    $k \leftarrow k + 1$;
end while

后，名字为：$res = lasso(A, b, \sigma, \mu, x)$ 对应的梯度函数为 $dfres = dflasso(A, b, \sigma, \mu, x)$。由于问题关心的最优解是 x，所以我在代码文件的最前面将目标函数与梯度函数定义为 f,df 两个关于 x 单一变量的函数句柄（带入了预设值得参数值，可以改变）。

而下降方向的选择上，由于缺乏进一步的学习，我直接采用了负梯度方向。由于寻找下降方向采用的函数封装连起来，后续期望可以用其他下降方向更改。

对于线搜索产生的步长上，我采用了两种方法，一种是 armijo，另一种是 wolfe。原始代码如下：

```
function a = armijo(f, df, d, x, a0)
    a = a0; f0 = f(x); df0 = dot(df(x), d);
    c1 = 0.085; beta = 0.35; fx = f(x + a * d);
    while (fx > f0 + c1 * a * df0)
        a = a * beta; fx = f(x + a * d);
        if (a < 1e-10 && fx <= f0)
            break;
        end
    end
end

function a = wolfe(f, df, d, x, a0)
    a = a0; c1 = 0.09; c2 = 0.3;
    f0 = f(x); df0 = dot(df(x), d);
    secant = (f(x + a * d) - f0) / a; dfx = dot(df(x + a * d), d);
    while (secant > df0 * c1 || dfx < df0 * c2)
        % use the quadratic interpolation (f(0),f'(0),f(a)).
        % update a as the minimum point of quadratic function.
        a = df0 * a / (2 * (df0 - secant));
```

```
20                secant = (f(x + a * d) - f0) / a;
21                dfx = dot(df(x + a * d), d);
22                if (a < 1e-6 && secant <= 0)
23                    break;
24                end
25            end
26        end
```

其中 armijo 函数采用 armijo 准则判断步长是否合适，即：$f(x+\alpha d) \le f(x) + c_1\alpha\nabla f(x)^T d$。当步长不满足条件时，指数衰减步长，每次步长乘以 β。直到满足条件，同时为了保证在步长较小时舍入误差导致 $f(x+\alpha d)$ 与 $f(x)$ 相等，增加了判断条件 $\alpha < 1e-10 \bigwedge f(x+\alpha d) \le f(x)$ 满足时直接跳出。

wolfe 函数采用 wolfe 准则判断步长是否合适，即：$f(x+\alpha d) \le f(x) + c_1\alpha\nabla f(x)^T d$（1）并且有 $\nabla f(x+\alpha d) \ge c_2\nabla f(x)$。当步长不满足条件时，采用二次插值法更新步长。记 $f(x+\alpha d) = \phi(\alpha)$ 即利用 $\phi(0), \phi'(0), \phi(a)$ 三点确定二次函数的最小值，更新步长为插值函数的最小值点。让我们来进行计算差分：

$$\phi[0] = \phi(0), \phi[\alpha] = \phi(\alpha)$$

$$\phi[0,0] = \phi'(0),\ \phi[0,\alpha] = \frac{\phi(\alpha) - \phi(0)}{\alpha}$$

$$\phi[0,0,\alpha] = \frac{\phi[0,\alpha] - \phi[0,0]}{\alpha}$$

令 $secant = \dfrac{f(x+\alpha d) - f(x)}{\alpha}$，则有：（1）$\Leftrightarrow secant \le c_1\nabla f(x)^T d$ 且二次函数最低点 α_{min} 满足 $\alpha_{min} = \dfrac{-\phi[0,0]}{2\phi[0,0,\alpha]} = \dfrac{-\nabla f(x)^T d\alpha}{2(secant - \nabla f(x)^T d)}$。进一步地，为了防止舍入误差导致更新步长与函数值相差过小，增加了判断条件 $\alpha < 1e-6 \bigwedge secant \le 0$，满足时直接跳出。

整体上,将线搜索、方向选择函数实现封装为 ls 与 dir 句柄,供主要的梯度下降函数 $gradient_descent$ 调用。通过改变句柄对应的函数可以更换线搜索以及方向选择方法。设置梯度精度要求以及最大迭代次数用作停机条件。整个 $gradient_descent$ 函数的实现逻辑与算法一一致，这里不再赘述。

求解过程

首先，我们导入数据 A, b 并设置参数 $\sigma = 0.1, \mu = 1\times10^{-2}$。并设置初始值步长 α_0，解 $x_0 = \vec{0}$。根据需求，设置线搜索方法以及方向选择方法（更改 ls, dir 句柄即可）。然后设置最大迭代次数与梯度精度要求，静待结果。

实际调整参数的过程是非常漫长的，但需要调整的主要是线搜索方法内部的参数 c_1, c_2, β 以及步长初始值 α_0。经过多次尝试，我最终选择了如下参数：

- armijo, $c_1 = 0.085, beta = 0.35$
- wolfe, $c_1 = 0.09, c_2 = 0.3$

步长初始值 $\alpha_0 = 1 \times 10^{-3}$。

虽然二次插值的方法任然不能保证一定能找到满足 $wolfe$ 准则的步长，但在我的实验中，这种情况没有发生。而对于那些找到后步长已经非常小甚至使得函数值几乎不变的情况，我则直接跳出循环，认为已经找到了最优解（这种情况在设置梯度模长要求为 1e-6 时发生过，如果不直接跳出，则会陷入无限循环）。

实验结果

以题设 $\mu = 1 \times 10^{-2}$，为标准情况。最优解 x 存放在 solution.mat 文件中. $f(x) = 0.818310$，$\|\nabla f(x)\| = 1.5805 \times 10^{-6}$。

而不同的方法选择，精度对实验结果有着显著影响。结果如下：

表 1: 不同方法选择，精度对结果的影响

方法	精度	耗时	迭代次数	函数值	梯度模长
wolfe	1e-4	70.97s	251571	0.818310	9.9998e-5
wolfe	1e-5	371.49s	333266	0.818310	9.9971e-6
wolfe	1e-6	429.27s	409115	0.818310	1.6687e-6
armijo	1e-3	49.21s	255715	0.818329	9.9889e-4
armijo	1e-4	248.27s	361530	0.818310	9.9930e-5
armijo	1e-5	94.55s	476582	0.818310	9.9984e-6

与此同时，当我们改变 μ 时，结果、收敛速度会显著变化。具体来说，当我们统一要求梯度模长在 1e-5 以下时，我们有以下结果：

表 2: 不同 μ 选择对结果的影响

μ	耗时	迭代次数	函数值	梯度模长
1e0	1.44s	4733	81.643264	4.6859e-6
1e-1	11.09s	41476	8.181363	9.9220e-6
1e-2	98.46s	333266	0.818310	9.9971e-6
1e-3	327.30s	1000000(超界未停)	0.082180	0.003342
0	0.11s	141	3.1684e-13	8.0594e-6

可以看到，一定范围内随着 μ 的减小，算法收敛速度变慢，但是当 $\mu \to 0$ 时，算法将飞速收敛。

值得注意的是，耗时受到硬件以及设备运行状态等多方面影响，比如在上面两个表格中，wolfe 准则在 $\mu = 1e-5$ 的运行时间相差较大。故其仅具有相对的参考价值。

以迭代次数为横轴，函数值与最优值的差的以 10 为底的对数为纵轴，绘制出二者关系曲线。如下图所示：

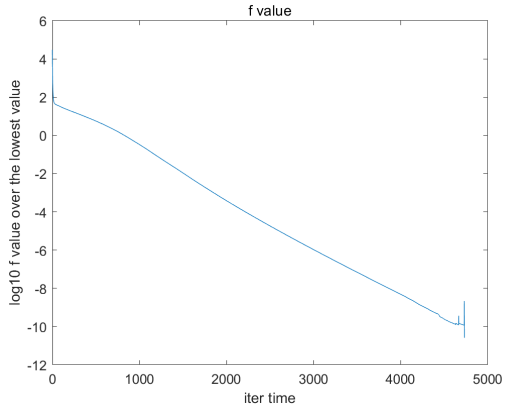

图 1: $\mu = 1$ 时的收敛历史

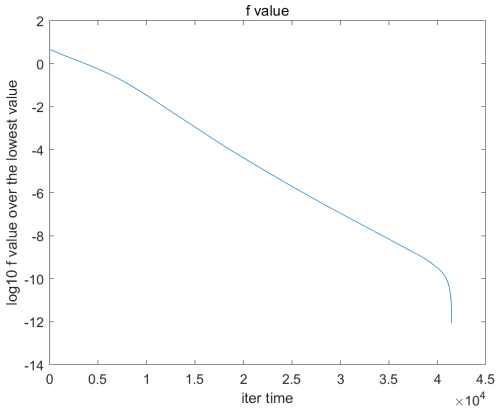

图 2: $\mu = 1e-1$ 时的收敛历史

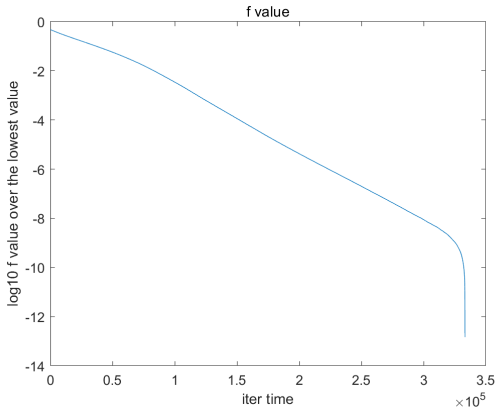

图 3: $\mu = 1e-2$ 时的收敛历史

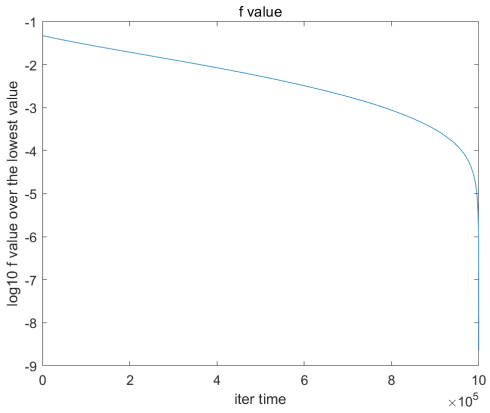

图 4: $\mu = 1e-3$ 时的收敛历史

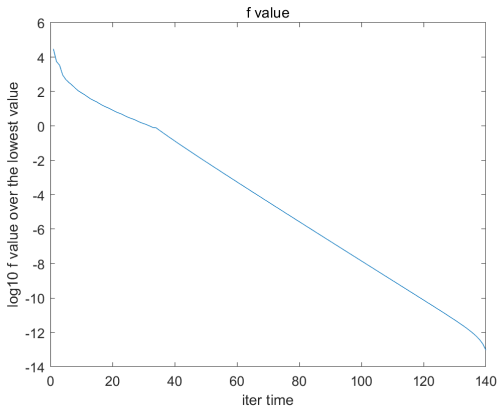

图 5: $\mu = 0$ 时的收敛历史

更新

在上过第二周的课后，我对此次作业有如下更新：

1. 增加了对收敛性的分析。

2. 对 armijo 准则的参数选择进行了优化，使得迭代次数进一步减小。

首先对本问题。对梯度 $\nabla f(x) = A^T(Ax-b) + \mu h(x)$，其中 $h(x)_i = \begin{cases} \frac{x_i}{\sigma}, & |x_i| < \sigma, \\ sign(x_i), & \text{otherwise.} \end{cases}$ 在迭代格式 $x_{k+1} = x_k - \alpha_k \nabla f(x_k)$ 下。有：

$$\begin{aligned} x_{k+1} - x_* &= x_k - x_* - \alpha_k(A^T(Ax_k - b) - \mu h(x_k)) \\ &= x_k - x_* - \alpha_k(\nabla f(x_*) + A^T A(x_k - x_*) + \mu(h(x_k) - h(x_*))) \\ &= (I - \alpha_k A^T A)(x_k - x_*) - \alpha\mu(h(x_k) - h(x_*)) \end{aligned}$$

并且在 $\|x_k - x_*\|$ 很小时，我们可以近似的认为 $h(x_k) \approx h(x_*)$，其中 x_* 指代最优解。因此,$\|x_{k+1} - x_*\| \le max\{|1-\alpha\lambda_n|, |1-\alpha\lambda_1|\}\|x_k - x_*\|$，其中 λ_n, λ_1 是矩阵 $A^T A$ 的两个最大、最小特征值。在本问题中由于 A 是一个行数为列数一半的矩阵，所以 $A^T A$ 至少有一半的为 0 的特征值，而经过一番计算可知 $A^T A$ 的特征值中恰有 512 个零，且 $\lambda_{513} = 89.8163$, $\lambda_{1024} = 2937.7$. 故问题本身较为病态。

在步长的选择上，对我上面的 $wolfe$ 准则下搜索的步长取对数下来看，可以发现较多情况下步长均在 10^{-3} 左右。带入上式，由于有一半的 0 特征值，故很大概率下前面的二范数项几乎不起作用，迭代前进主要由后面的“一范数”惩罚项决定。由于在 $|x_i| \ge \sigma$ 时,$h(x_i)$ 对相同符号下的值恒定。故我们考虑 $|x_i| < \sigma$ 的情况能让 $\Delta h(x)$ 更大，下降更多，而此时有 $x_{k+1} - x_* = (1 - \frac{\alpha\mu}{\sigma})(x_k - x_*)$ 可以看到此时下降的速度 $1 - \frac{\alpha\mu}{\sigma}$ 量级为 $1 - 10^{-4}$。所以下降速度十分糟糕。估计下 $(1-10^{-4})^{10^4} \approx \frac{1}{e}$, 而 $log(10^5) \approx 11.5$。故需要 11.5×10^4 次迭代才能达到 10^{-5} 精度。

那么 10^{-3} 的步长有多糟糕呢？我们来计算一下全局的 Lipschitz 常数 L，容易得到 $L = \lambda_{1024} + \frac{\mu}{\sigma}$。对应的全局收敛区间为 $[0, \frac{2}{L}]$ 约为 $[0, 6.8 \times 10^{-4}]$ 可以看到，这个步长已经十分糟糕了。

所以我们希望尽量选取较大的步长，那么经过我又几个晚上的调参，最终我能够使得在 $\mu = 1 \times 10^{-2}$ 时，平均步长上升至 $10^{-2.5}$ 左右，这个时候的收敛次数就较少了。当然，我觉得这个问题差不多就是这样了，因为它实在是一个较为病态的问题。

最终代码放在“solution1.m”中，求解结果如下图所示：

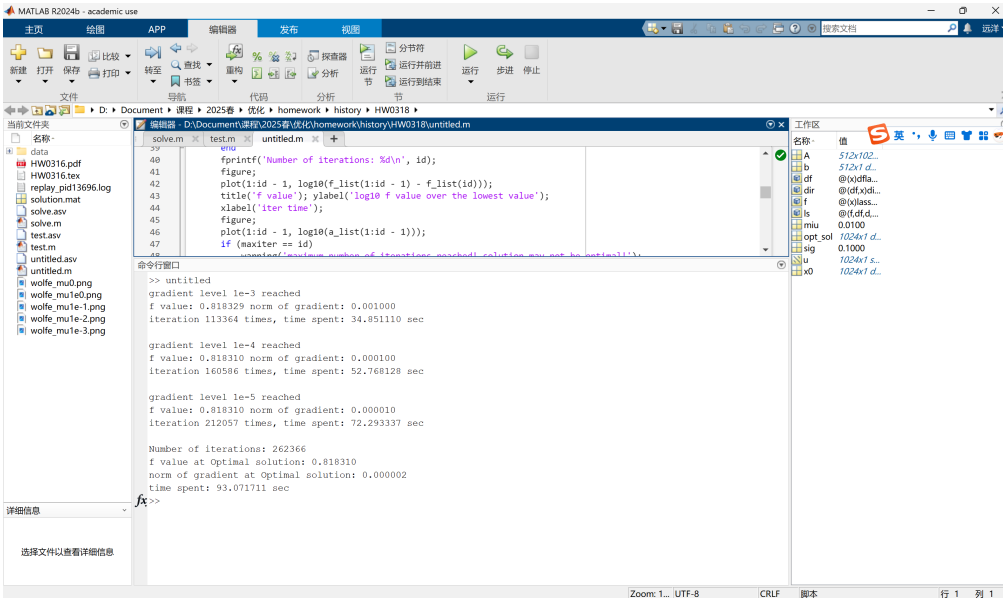

图 8: $\mu = 1 \times 10^{-2}$，梯度模长达到不同阶段时的效果

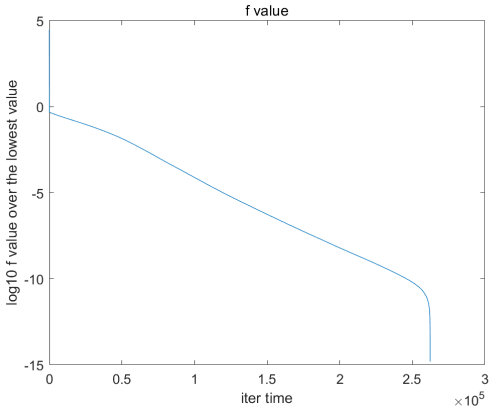

图 6: $\mu = 1 \times 10^{-2}$，函数值减去最小值取对数的收敛历史

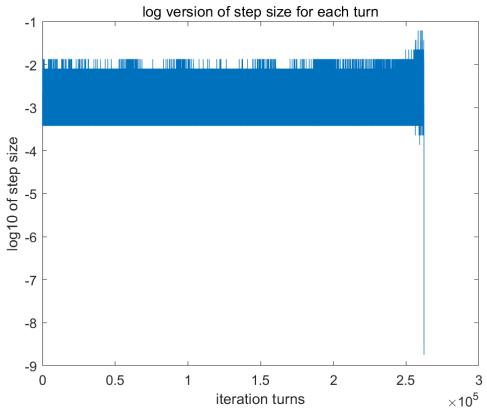

图 7: $\mu = 1 \times 10^{-3}$，步长取对数的收敛历史