最优化方法上机报告

吴睿林 2200010914

北京大学 数学科学学院

目录

摘要

本文是文再文老师《最优化方法》课程的上机报告，内容为求解此group LASSO问题

$$\min_{x\in\mathbb{R}^{n\times l}} \quad \frac{1}{2}\|Ax-b\|_F^2+\mu\|x\|_{1,2} \tag{1}$$

其中$A\in\mathbb{R}^{m\times n}, b\in\mathbb{R}^{m\times l}, \mu>0$

$$\|x\|_{1,2}=\sum_{i=1}^{n}\|x(i,1:l)\|_2$$

由于目标函数第一项可以将x展开写成列向量的形式，所以此问题与一般向量形式的LASSO问题的差别主要在对第二项正则项$\|x\|_{1,2}$的处理, 我们接下来对题目的具体分析也主要是围绕对此范数的处理。

本文对每题的主要分析涉及对算法的介绍，部分特殊函数代码的实现以及数值结果的对比与解释。

本文用到的技术比如连续化策略，BB步长与线搜索准则主要参考http://faculty.bicmr.pku.edu.cn/~wenzw/optbook/pages/contents/contents.html

算法性能总览

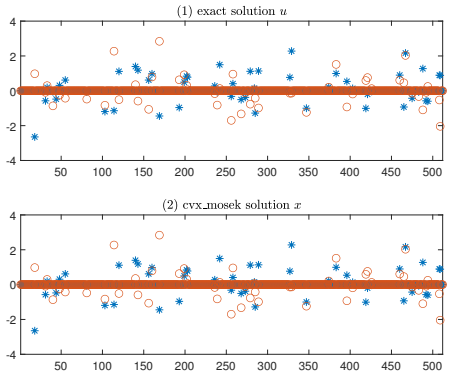
算法	CPU时间 (s)	迭代次数	目标值	稀疏度	与精确解误差	Error_M	Error_G
CVX-Mosek	7.33	-1	5.80556E-01	0.105	3.78E-05	0.00E+00	5.24E-07
CVX-Gurobi	7.11	-1	5.80556E-01	0.103	3.75E-05	5.24E-07	0.00E+00
Mosek	0.72	-1	5.80556E-01	0.103	3.75E-05	4.86E-07	8.41E-08
Gurobi	0.31	14	5.80560E-01	0.167	4.01E-05	7.00E-06	7.25E-06
SGD Primal	0.13	2276	5.80563E-01	0.100	4.83E-05	1.11E-05	1.13E-05
GD Primal	0.31	2873	5.80557E-01	0.111	4.67E-05	9.81E-06	1.02E-05
ProxGD Primal	0.08	1157	5.80556E-01	0.104	3.96E-05	3.09E-06	3.33E-06
FProxGD Primal	0.04	252	5.80556E-01	0.100	3.77E-05	2.14E-06	1.89E-06
ALM Dual	0.10	69	5.80568E-01	0.100	6.56E-05	3.15E-05	3.17E-05
ADMM Dual	0.29	597	5.80556E-01	0.101	3.99E-05	3.69E-06	3.77E-06
ADMM Primal	0.02	47	5.80600E-01	0.100	4.85E-05	1.55E-05	1.57E-05

此数据为教学网站上Test_group_lasso.m的运行结果

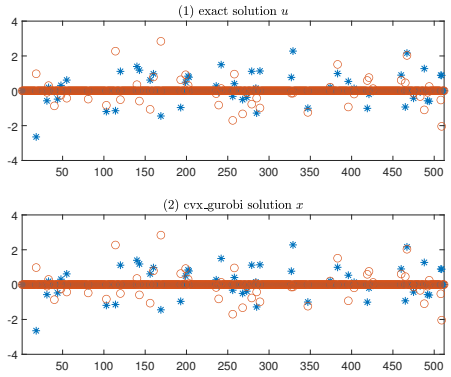
1 第一题

此题通过优化模型语言CVX调用MOSEK和Gurobi求解器对原题进行直接求解，在了解CVX的接口后直接调用即可。

可视化解的图像如下，两种不同标识的符号表示解的两列数据。可以看到利用CVX调用求解器得到的解x与u基本重合，堆叠的粗线表明解也满足稀疏性。



(a) 通过CVX调用MOSEK求解器得到的解



(b) 通过CVX调用Gurobi求解器得到的解

2 第二题

2.1 直接调用MOSEK求解

我们将原问题写成等价的二次锥规划的形式：

$$\begin{aligned} \min \quad & \frac{1}{2}t + \mu \sum_{i=1}^{n} u_i \\ \text{s.t.} \quad & \|Cx - d\|_2 \le u_0 \\ & \|A_i x\|_2 \le u_i, i = 1, \ldots, n \\ & \|u_0\|^2 \le 1 \times t \end{aligned}$$

约束条件前两行是二次锥，第三行是旋转二次锥，为符合MOSEK的接口，我们的自变量X为尺寸$(n(l+1)+2)\times 1$的列向量 $(x_1, \cdots, x_l, u, t)^\top$，其中$x_i$为$x$的第$i$列列向量。

第一行约束条件等价于$(I_{l\times l} \otimes A)X - B \in \mathcal{Q}$，其中$B$为将$b$所有列向量转为1列的向量。第二行约束条件等价于 $A_i \in \mathbb{R}^{l\times nl}$，且第$j$行的第$(j-1)n+i$个元素为1，其余元素均为0。第三个约束条件将旋转二次锥前两个元素分别设为0.5和t即可。依照此问题格式进行输入，即可直接调用MOSEK求解原问题。

2.2 直接调用Gurobi求解

原问题(1)等价于如下QCQP问题：

$$\begin{aligned} \min \quad & \frac{1}{2}\sum_{i=1}^{l}\|Ax_i - b_i\|_2^2 + \mu\sum_{j=1}^{n} u_j \\ \text{s.t.} \quad & u_j^2 \geq \sum_{i=1}^{l} x_{ij}^2 \quad j = 1, \cdots, n \\ & u_j \geq 0 \end{aligned}$$

其中x_i表示x的第i列向量，x_{ij}表示x的第i列j行的元素，这与常见的定义相反。

为了符合Gurobi的调用接口，在此QCQP问题中，我们的自变量为尺寸$n(l+1)\times 1$的列向量，即$\begin{bmatrix} x_1 \\ \vdots \\ x_l \\ u \end{bmatrix}$于是目标函数和约束均是关于此自变量的二次函数，实际上目标函数中的二次矩阵为$\begin{bmatrix} A^\top A & & & \\ & \ddots & & \\ & & A^\top A & \\ & & & 0_{n\times n} \end{bmatrix}$，第i个二次约束函数矩阵为$\begin{bmatrix} E_{ii} & & & \\ & \ddots & & \\ & & E_{ii} & \\ & & & -E_{ii} \end{bmatrix}$

其中E_{ii}为(i,i)位置的元素为1，其他位置为零0的$n\times n$矩阵。按照此QCQP问题格式进行输入，即可直接调用Gurobi求解原问题。

3 第三题

3.1 (a)原问题的次梯度法

由于原问题(1)的第一项可微，所以只用考虑正则项的次梯度。

由于$\|x\|_{1,2} = \sum_{i=1}^{n}\|x(i,1:l)\|_2$是$n$个行向量的模长之和，可以单独考虑对每个行向量的模长$\|x(i,1:l)\|_2$的次梯度。向量模长函数在非零点处可微，零点处取到最小值，所以$0_{1\times l}$是$x(i,1:l)=\vec{0}$处的一个次梯度（实际上我们在算次梯度前会对x进行截断，小于opts.thres的值均视为0），据此得到原函数的次梯度。

于是应用次梯度法的下降方向取为 $d^k = -\left(A^\top(Ax-b) + \mu\cdot \mathrm{SGD}(x)\right)$。

其中求取正则项次梯度的SGD函数如下：

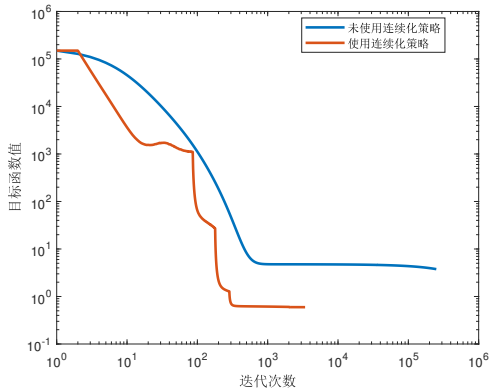
求正则项次梯度

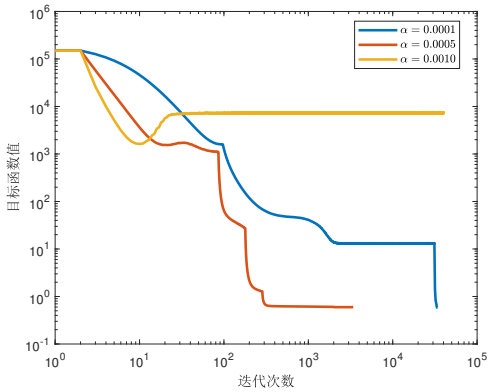
```
function result = SGD(x)
    row_vector_norm = vecnorm(x')';
    row_vector_norm(row_vector_norm == 0) = 1;
    result = x ./ row_vector_norm;
end
```

下图为各参数变化时的函数值随迭代次数变化图像。

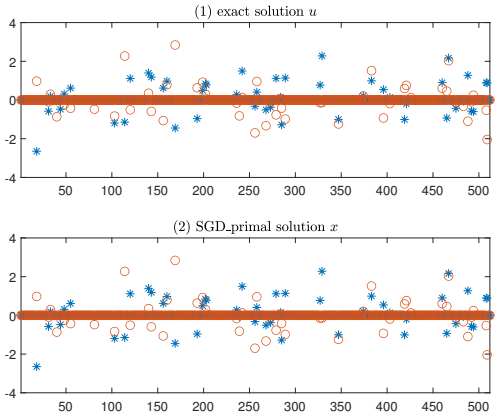
左图在原题添加连续化策略后，下降速度得到显著增长，因为在原问题中μ更大会让次梯度随之增大，此时收敛速度加快，能更快逼近原问题的解。

对于步长的选取，由于下降趋势稳定，问题求解时的特点是下降速度缓慢，所以我没有采用消失步长，而是采用固定步长，各步长跑出来的程序结果如右图，对于步长选取过大($\alpha = 0.001$)的情况，目标函数值可能并不会收敛到最优值，步长选取过小($\alpha = 0.0001$)，收敛速度会大幅降低。





如下是使用连续化策略+次梯度下降法得到的解x与原稀疏解u的对比，基本一致。



3.2 (b)原问题光滑化后的梯度法

由于原问题(1)的第一项光滑，所以只用考虑正则项的光滑化。

而正则项$\|x\|_{1,2}$是每个列向量$x(i,1:l)$的模长之和，所以只用考虑对单个列向量的模长进行光滑化，与低维情形类似，我们在不可微点也就是原点处用二次多项式近似原函数，即

$$l_\sigma(x(i,1:2)) = \begin{cases} \sqrt{x_{i1}^2 + x_{i2}^2} & \|x(i,1:2)\|_2 \geq \sigma \\ \frac{1}{2\sigma}(x_{i1}^2 + x_{i2}^2) + \frac{\sigma}{2} & \|x(i,1:2)\|_2 < \sigma \end{cases}$$

于是$l_\sigma(x)$为行向量模长的一个光滑近似，且梯度为：

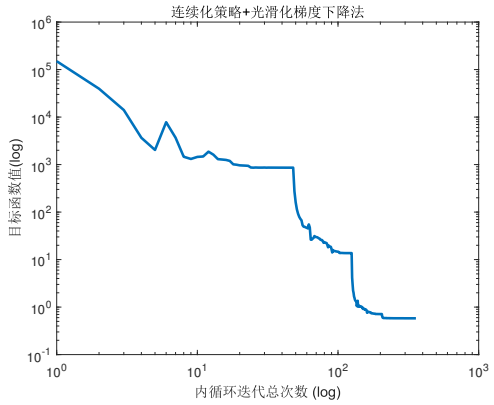
$$\nabla l_\sigma(x) = \begin{cases} x(i,1:2) & \|x(i,1:2)\|_2 \geq \sigma \\ \frac{1}{\sigma}x(i,1:2) & \|x(i,1:2)\|_2 < \sigma \end{cases}$$

于是函数的正则项$\|x\|_{1,2} \approx L_\sigma(x) = \sum_{i=1}^n l_\sigma(x(i,1:l))$，其中$L_\sigma$是可微函数

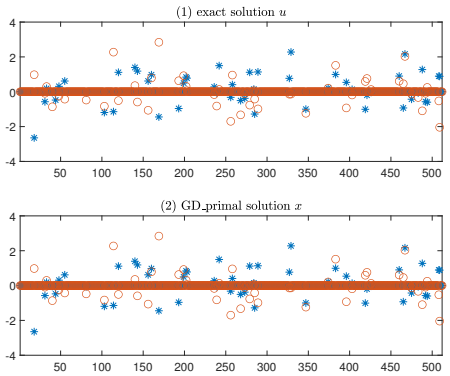
于是应用梯度法的下降方向取为 $d^k = -\left(A^\top(Ax-b) + \mu \cdot \nabla L_\sigma(x)\right)$

对于步长的选取我采用了BB步长+线搜索检索。

最后再在外层套上连续化策略，我们得到如下目标函数值-迭代次数曲线与解的对比，可以看到在每次减小μ之后，目标函数值都会有显著的下降。



(a) 目标函数值-迭代次数曲线



(b) 光滑化梯度法得到的解

3.3 (d)原问题的近似点梯度法

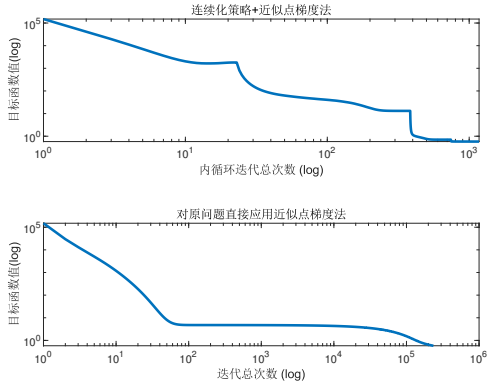
对于目标函数(1)，近似点梯度法考虑令 $\phi(x) = \frac{1}{2}\|Ax-b\|_F^2, h(x) = \mu\|x\|_{1,2}$,对光滑部分做梯度下降，并对非光滑部分使用近似点算子，得到近似点梯度法迭代格式 $x^{k+1} = \text{prox}_{t_k\mu\|\cdot\|_{1,2}}\left(x^k - t_k\nabla\phi\left(x^k\right)\right)$，其中近似点算子prox的计算对$x$每个行向量逐行考虑即可，实现代码如下：

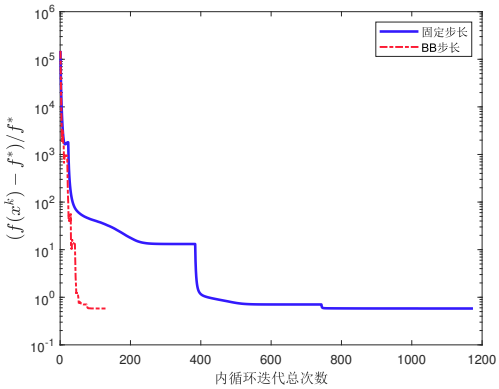
正则项的邻近算子

```
function result = prox(x, mu)
    row_norms = vecnorm(x')';
    scaling_factor = max(1 - mu ./ row_norms, 0);
    result = bsxfun(@times, x, scaling_factor);
end
```

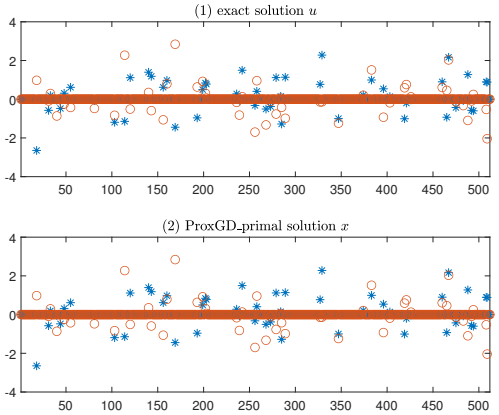
对于步长的选取我采用了BB步长+线搜索准则（Zhang&Hager），外层调用用连续化策略，两者均能提高下降速度，但是连续化策略效果更为显著.

如下是添加连续化策略与添加步长选取各自的对比图左图中可以看到连续化策略每次减小μ之后都能对目标函数值显著下降，大约在内循环迭代900次左右达到0.5806，达到小数点后四位精度。没有使用连续化策略的近似点梯度法则下降缓慢，最终循环迭代230000次才进入0.5806。对于右图中步长选取的对比，使用BB步长+线搜索准则可以更快收敛，但在连续化策略的应用下，两者速度均不慢。





连续化策略+近似点梯度法得到的解与精确解u几乎一致：



3.4 (e)加速近似点梯度法

本题利用 Nesterov 加速的近似点梯度法进行优化。

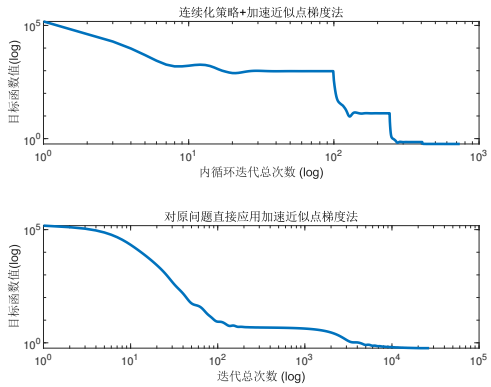
该算法同样被外层连续化策略调用，在连续化策略下完成某一固定正则化系数的内层迭代优化。

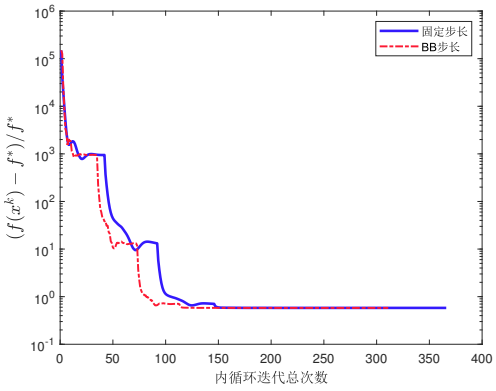
在每一步迭代时，算法首先在之前两步迭代点的基础上进行一步“Nesterov加速” $y^k = x^{k-1} + \frac{k-2}{k+1}\left(x^{k-1} - x^{k-2}\right)$，然后再在 y^k 处进行一步近似点梯度法，$x^k = \mathrm{prox}_{t_k\mu\|\cdot\|_1}\left(y^k - t_k A^\top\left(Ay^k - b\right)\right)$。近似点算子prox的计算与上题相同。

如下是添加连续化策略与添加步长选取各自的对比图。

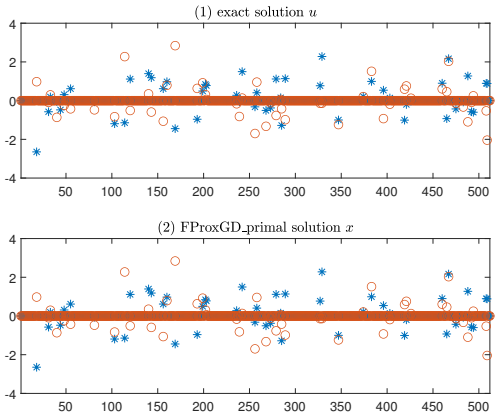
在左边的图中，上下两组算法的唯一区别为是否添加连续化策略，均采用固定步长，可以看到结果与上题类似，连续化策略依旧发挥着显著作用，但是相比而言两者的速度都更快了，这也体现了“Nesterov加速算法”确实加速了。

右边的图中唯一区别为步长选取，两者均采用连续化策略，可以看到BB步长表现情况好于固定步长。





连续化策略+加速近似点梯度法得到的解与精确解u几乎一致：



3.5 (f)对偶问题的增广拉格朗日函数法

原问题(1)的对偶问题为

$$\begin{aligned} \min \quad & \frac{1}{2}\|z\|_F^2 + \langle z, b\rangle \\ \text{s.t.} \quad & A^\top z + w = 0 \\ & \|w\|_{\infty,2} < \mu \end{aligned} \tag{2}$$

于是可以写出对偶问题的增广拉格朗日函数：

$$L_\sigma(z,w,x) = \langle b,z\rangle + \frac{1}{2}\|z\|_F^2 + I_{\|w\|_{\infty,2}\le\mu}(w) - \left\langle x, A^\top z + w\right\rangle + \frac{\sigma}{2}\left\|A^\top z + w\right\|_F^2$$

由此得到算法的迭代格式为：

$$(z^{k+1}, w^{k+1}) = \arg\min\left\{\langle b,z\rangle + \frac{1}{2}\|z\|_F^2 + I_{\|w\|_{\infty,2}\le\mu}(w) - \left\langle x, A^\top z + w\right\rangle + \frac{\sigma}{2}\left\|A^\top z + w\right\|_F^2\right\}$$

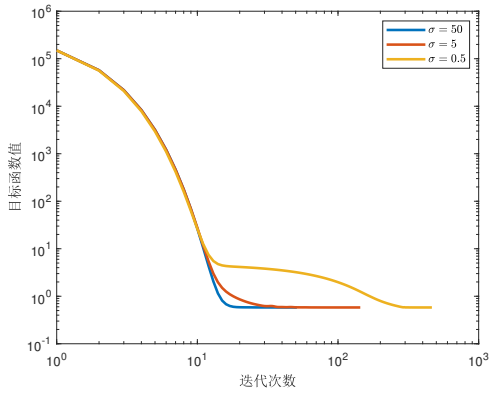
$$x^{k+1} = x^k + \sigma\left(A^\top z^{k+1} + w\right).$$

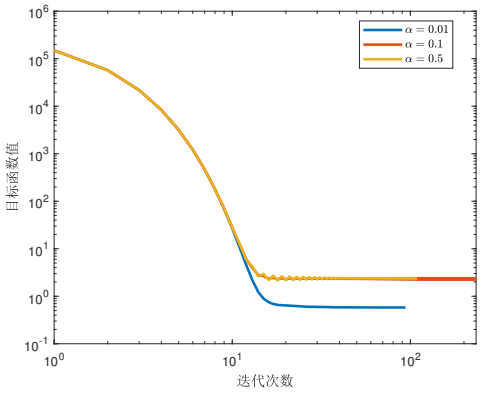
对于第一个子问题，由于右边是关于z^{k+1}的二次函数，可以直接求极值点 $z = (I + \sigma AA^\top)^{-1}(A(x - \sigma w))$ 代入增广拉格朗日函数可以将z^{k+1}消元，子问题化成求解w决定的函数的最小值点，由于子问题为一个复合优化问题，为一个示性函数和二次函数的组合，所以我们考虑用近似点梯度法求取最小值点，即$w_{i+1}^{k+1} = \mathrm{proj}_{\alpha I_{\|\cdot\|_{\infty,2}\le\mu}}(w_i^{k+1} - \alpha g_i^k)$ 其中梯度的光滑项梯度g_i^k的计算式利用复合函数求导法则可以得到为

$$g_i^k = -x^k + \sigma w_i^{k+1} - \sigma^2 A^\top (I + \sigma A^\top A)^{-1}(Aw_i^{k+1}) - \sigma A^\top (I + \sigma A^\top A)^{-1}(b - Ax^k)$$

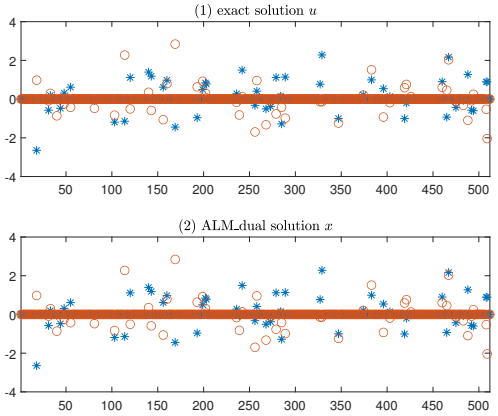
proj为到范数球的投影算子，在(g)小节中有具体实现的代码。

运行结果可视化如下图，左边是目标函数值随迭代次数的变化图像，变量是罚因子σ，可以看到罚因子σ越大时，函数值下降地越快，但随着σ越大，$(I+\sigma A^\top A)$的Cholesky分解的计算代价就越大，并且会有目标函数发散的可能。右边是目标函数值随着迭代次数的变化图像，变量是应用在子问题的近似点梯度法的步长α，可以看到在α过大时，原目标函数值不会收敛到最优值。





可以看到利用增广拉格朗日函数法得到的解和精确解u几乎一致



3.6 (g)对偶问题的交替方向乘子法

本题使用交替方向乘子法对原问题的对偶问题(2)进行优化，对偶问题还是同上一题：

$$\begin{aligned} \min \quad & \frac{1}{2}\|z\|_F^2 + \langle z, b\rangle \\ \text{s.t.} \quad & A^\top z + w = 0 \\ & \|w\|_{\infty,2} < \mu \end{aligned}$$

增广拉格朗日函数同理：

$$L_\sigma(z,w,x) = \langle b, z\rangle + \frac{1}{2}\|z\|_F^2 + I_{\|w\|_{\infty,2}\le\mu}(w) - \left\langle x, A^\top z + w\right\rangle + \frac{\sigma}{2}\left\|A^\top z + w\right\|_F^2$$

相比于上一题的增广拉格朗日函数法，此时迭代格式发生变化，不需要对z,w进行联合求解增广拉格朗日函数的极小值，而是分别交替求z,w关于增广函数$L_\sigma(z,w,x)$的极小值点即可，由于关于z的极小值点并没有显式解，我们对其用到$\|\cdot\|_{1,2}$范数球的投影算子proj来计算其迭代点，最终迭代格式如下：

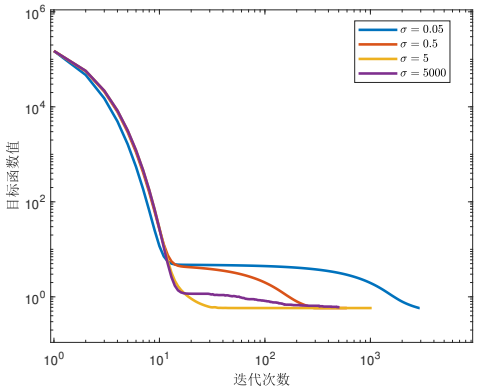
$$\begin{aligned} w^{k+1} &= \mathrm{proj}_\mu(-A^\top z^k + \frac{x^k}{\sigma}) \\ z^{k+1} &= \left(I + \sigma AA^\top\right)^{-1} A\left(x^k - \sigma w^{k+1}\right) - b \\ x^{k+1} &= x^k - \gamma\sigma\left(A^\top z^{k+1} + w^{k+1}\right) \end{aligned}$$

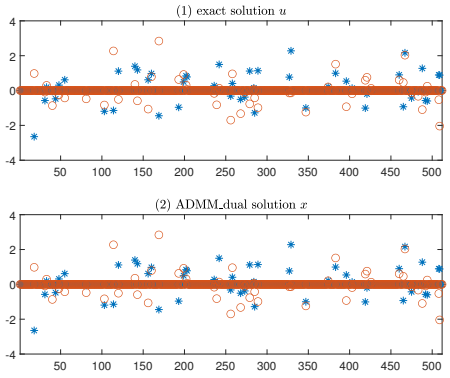
对于矩阵求逆，由于需要求逆的矩阵不变，我们在进行迭代前先对 $I+\sigma AA^\top$ 的Cholesky分解进行缓存加速后续迭代过程，对于其中的投影算子proj，其实现代码如下：

范数球的投影算子

```
function result = proj(x, mu)
    result = zeros(size(x));
    row_norms = vecnorm(x')';
    result(row_norms<mu,:) = x(row_norms<mu,:);
    normlize = x./row_norms;
    result(row_norms>=mu,:) = mu*normlize(row_norms>=mu,:);
end
```

此题最终运行的结果如下图，左边为原问题目标函数值随迭代次数的下降情况，分别画出了不同罚因子σ下的下降图像，可以看到与上题类似，σ越大下降越快，但是实际上当σ过大时，如图中$\sigma = 5000$，此时不仅矩阵分解的计算量大，最终也无法收敛到最优值。右图为解的情况，可以看到对偶问题的交替方向乘子法得到的解与精确解u几乎一致





3.7 (h)原问题的线性化交替方向乘子法

本题使用线性化交替方向乘子法对原问题(1)进行优化，我们考虑原问题的一个等价形式：

$$\begin{aligned} \min_{x,z} \quad & \frac{1}{2}\|Ax-b\|_F^2 + \mu\|z\|_{1,2} \\ \text{s.t.} \quad & x = z \end{aligned}$$

对于此问题引入拉格朗日乘子 y, 得到增广拉格朗日函数

$$L_\sigma(x,z,y) = \frac{1}{2}\|Ax-b\|_F^2 + \mu\|z\|_{1,2} + y^\top(x-z) + \frac{\sigma}{2}\|x-z\|_F^2$$

在 ADMM 的每一步迭代中，交替更新 x, z与乘子y ，但是在一般的交替方法乘子法中 $z^{k+1} = \arg\min_z \left(\mu\|z\|_{1,2} + \frac{\sigma}{2}\left\|x^{k+1} - z + y^k/\sigma\right\|_2^2\right)$ 没有显式解，我们考虑利用线性化加

之对求邻近算子来进行迭代z^{k+1}，具体来说是对于子问题目标函数的二次项用一阶展开对其近似，即： $z^{k+1} = \arg\min_z \left(\mu\|z\|_{1,2} + \sigma(z^k - x^{k+1} - y^k/\sigma)z + \frac{1}{2\eta}\|z - z^k\|_F^2\right)$ 这等价于作步长为η的近似点梯度下降，最终得到的迭代格式如下：

$$
\begin{aligned}
x^{k+1} &= \left(A^\top A + \sigma I\right)^{-1}\left(A^\top b + \sigma z^k - y^k\right)\\
z^{k+1} &= \mathrm{prox}_{\mu\eta\|\cdot\|_1}\left(\eta\sigma x^{k+1} + \eta y^k + (1-\eta\sigma)z^k\right)\\
y^{k+1} &= y^k + \gamma\sigma(x^{k+1} - z^{k+1})
\end{aligned}
$$

最终程序在步长η取不同值跑出来的结果如左图，可以看到在步长过大时目标函数不会收敛到最优值（$\eta = 160$的情形），右图为$\eta = 100$时ADMM_primal的解与精确解的对比，基本一致。