# Algorithms for Big Data Analysis : Homework #5

Due on April 11, 2017 at 23:59pm

*Wen Zaiwen*

**Hu Tao**

# Problem 1

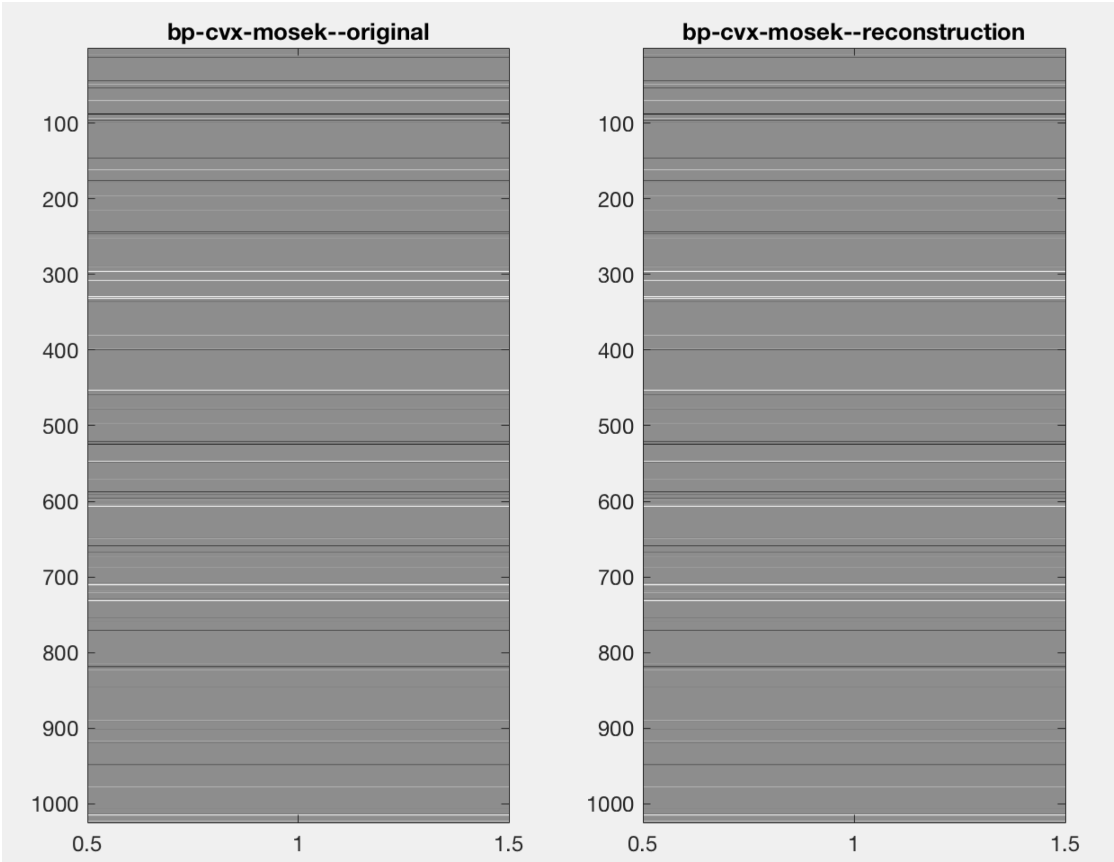
**Solution**

| Basic Persuit | Accuracy | | Run Time(s) | |
|---|---|---|---|---|
| | A=512*1024 | A=1024*2048 | A=512*1024 | A=1024*2048 |
| CVX_Mosek | 6.080e-10 | 1.469e-09 | 2.74 | 16.86 |
| Mosek | 5.075e-10 | 2.705e-08 | 2.40 | 15.26 |
| Augmented Lagragian | 7.954e-10<br>— | 9.370e-07<br>— | 3.19<br>— | 6.04<br>— |
| Dual ADMM | 8.111e-11<br>— | 3.603e-11<br>— | 1.83<br>— | 14.21<br>— |

总结:
1）在Augmented Lagragian方法中，$\tau$如果使用BB步长，结果会不收敛，但是Dual ADMM方法就没有这个问题（目前暂时不知道此bug的原因）。因此在Augmented Lagragian中使用的是固定步长,我设置$\tau = 0.001$,这样设置很显然很不好，比如我将问题的规模扩大一倍，A=1024*2048,$\tau = 0.001$就不起作用了，需要将$\tau = 0.0001$,总之$\tau$是一个很敏感的参数，最好使用BB步长动态调整。
2）mosek的准确率没有CVX Mosek的准确率高，很有可能是我fomulate的形式没有cvx帮我fomulate的形式更容易解。
3）Dual ADMM的准确率和运行时间还是比较满意的。

(a).我使用的是cvx中的mosek来求解.结果如下:

Problem 1 continued on next page. . . 

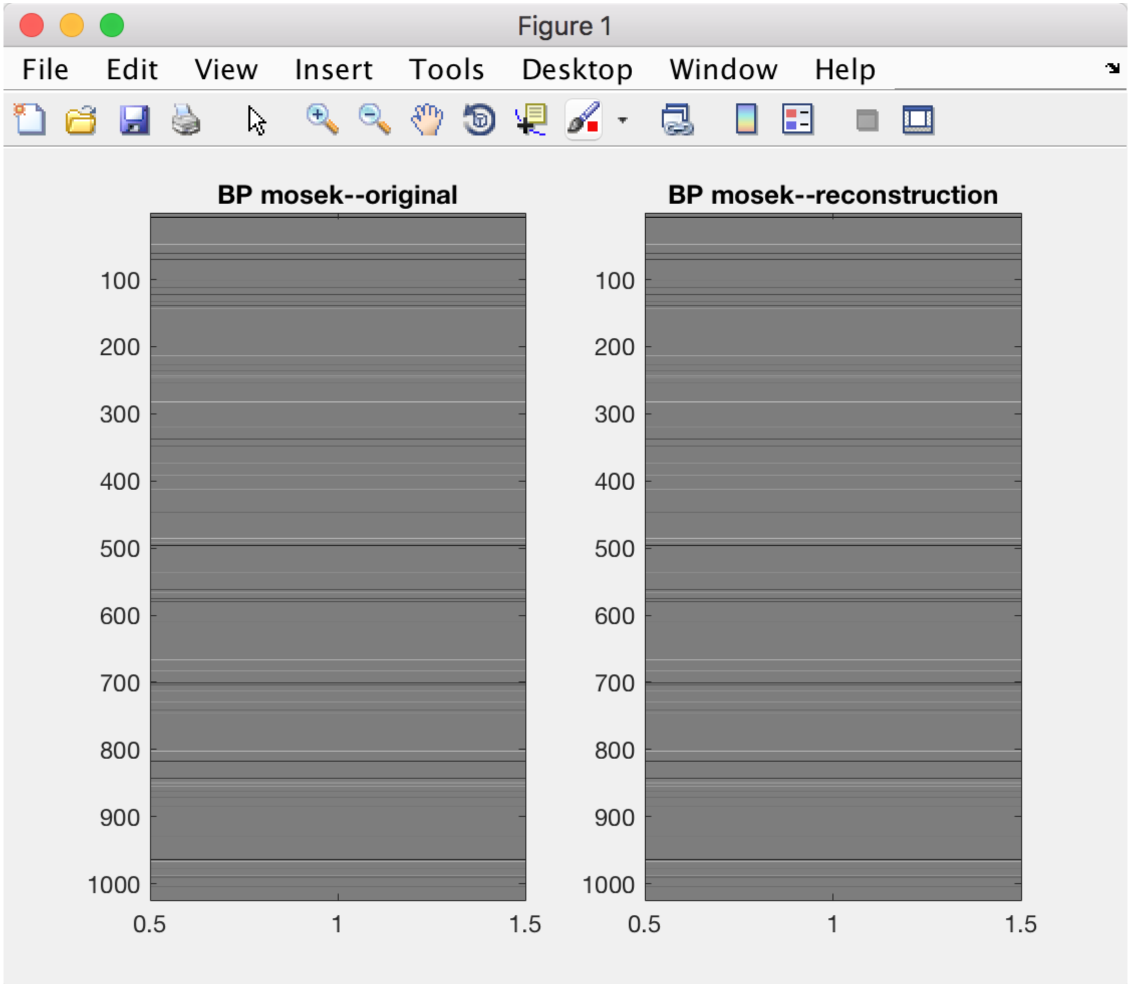



代码bp_cvx_mosek.m如下:

```
function [x0, out] = bp_cvx_mosek(A, b, opts)
n = size(A,2);
cvx_solver mosek
cvx_begin
variable x(n);
minimize(norm(x,1));
subject to
A * x == b;
cvx_end

x0 = x;
out.x = x;
end
```

(b).
对照http://docs.mosek.com/8.0/toolbox/linprog.html中描述的格式进行转换，我们将basis persuit问题按照mosek的格式进行了转换:
f为2n*1矩阵，A为$\begin{pmatrix} -E & -E \\ -E & E \end{pmatrix}$,b为2n*1矩阵,B为形式为$\begin{pmatrix} 0 & 0 \\ 0 & A \end{pmatrix}$的矩阵,c为2m*1的矩阵.
结果如下:




具体如下代码bp_mosek.m所示:

```
function [x0, out] = bp_mosek(A, b, opts)
m = size(A,1);
n = size(A,2);
x0 = rand(n,1);
mosek_f = [ones(n,1);zeros(n,1)];

mosek_A =[-eye(n),-eye(n);-eye(n),eye(n)];

mosek_b = zeros(2*n,1);

mosek_B = [zeros(m,n),zeros(m,n);zeros(m,n),A];

mosek_c = [zeros(m,1);b];

mosek_l = -ones(2*n,1)*inf;
mosek_u = ones(2*n,1)*inf;
options.Write = 'test.opf';
[x,fval,exitflag,output,lambda] =
    linprog(mosek_f,mosek_A,mosek_b,mosek_B,mosek_c,mosek_l,mosek_u,x0,options);
```

```
x0 = x(n+1:2*n);
out.x = x(n+1:2*n);

end
```

(c).针对增广拉格朗日函数，我们使用PGD([1])方法进行优化。
公式推到如下:

$$
\begin{aligned}
&||x||_1 + \lambda^T(Ax-b) + \tfrac{1}{2\mu}||Ax-b||_2^2 \\
&= |x||_1 + \tfrac{1}{2\mu}||Ax-b+\mu\lambda||_2^2
\end{aligned}
$$

对右半部分求导 可以得到$g = A^T(Ax-b+\mu\lambda)$,对上面的式子进行线性化，可以得到,

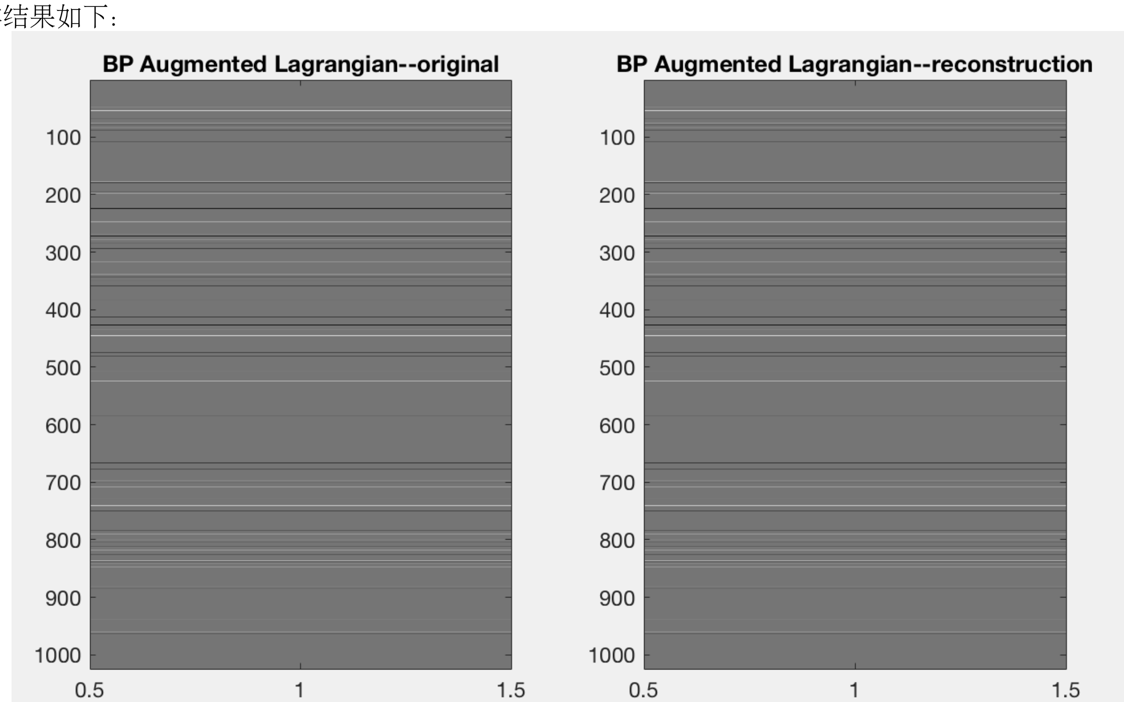
$$
\begin{aligned}
x_{k+1} &= |x||_1 + \tfrac{1}{\mu}((g_k)^T(x-x_k) + \tfrac{1}{2\tau}||x-x_k||_2^2) \\
&= |x||_1 + \tfrac{1}{2\tau\mu}||x-x_k+\tau g_k||_2^2
\end{aligned}
$$

推导到这里就可以shrinkage来求解了。最终的迭代公式如下:

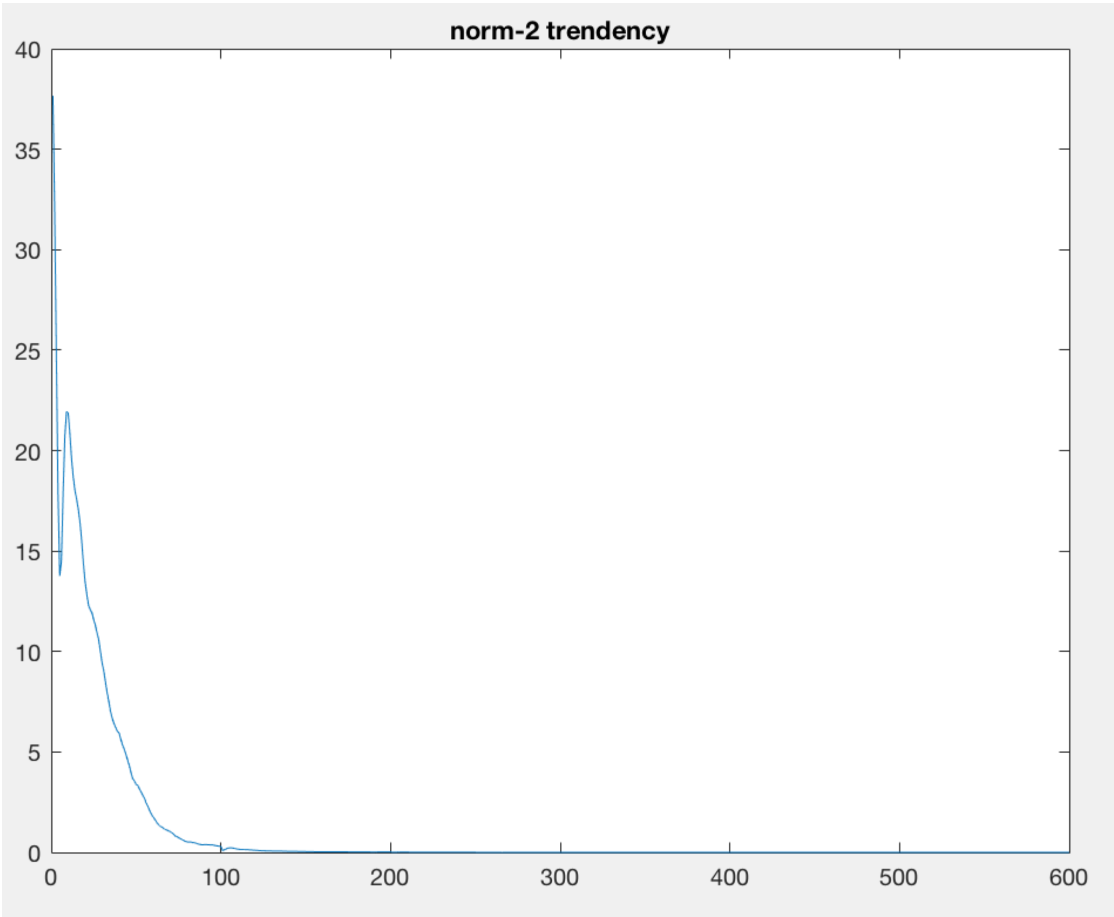
**Algorithm 1** Basic Persuit Augmented Lagragian Algorithm

**Require:** initial value $x_0, \lambda_0, \mu$,final $\mu_{final}$
  **while** $\mu > \mu_{final}$ **do**
    $\mu = \frac{\mu}{10}$
    **while** not satify the stop condition **do**
      calculate BB step:$\tau$
      $x - update : x_{k+1} = shrink(x_k - \tau g_k, \mu\tau)$
      $lambda - update : \lambda_{k+1} = \lambda_k + \frac{0.618}{\mu}(Ax_k - b)$
    **end while**
  **end while**
  Output x

代码使用了BB步长进行优化，在计算$\lambda_k$的的时候，使用了0.618的黄金神奇步长。
具体结果如下:



变量x的当前与前一次的差的3范数变化如下:




具体代码见`bp_al.m`。
（d).
求出对偶以后，最终的ADMM([2])迭代过程如下:

---

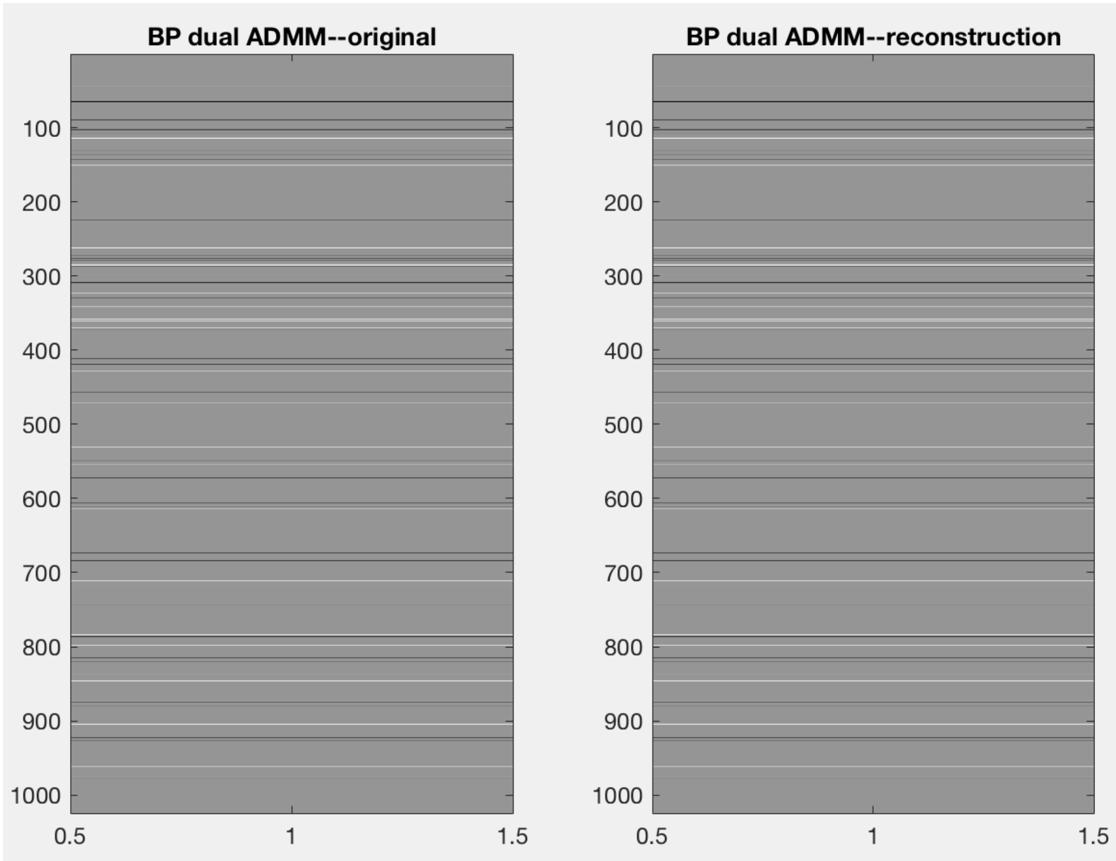
**Algorithm 2** Basic Persuit Dual ADMM Algorithm

---

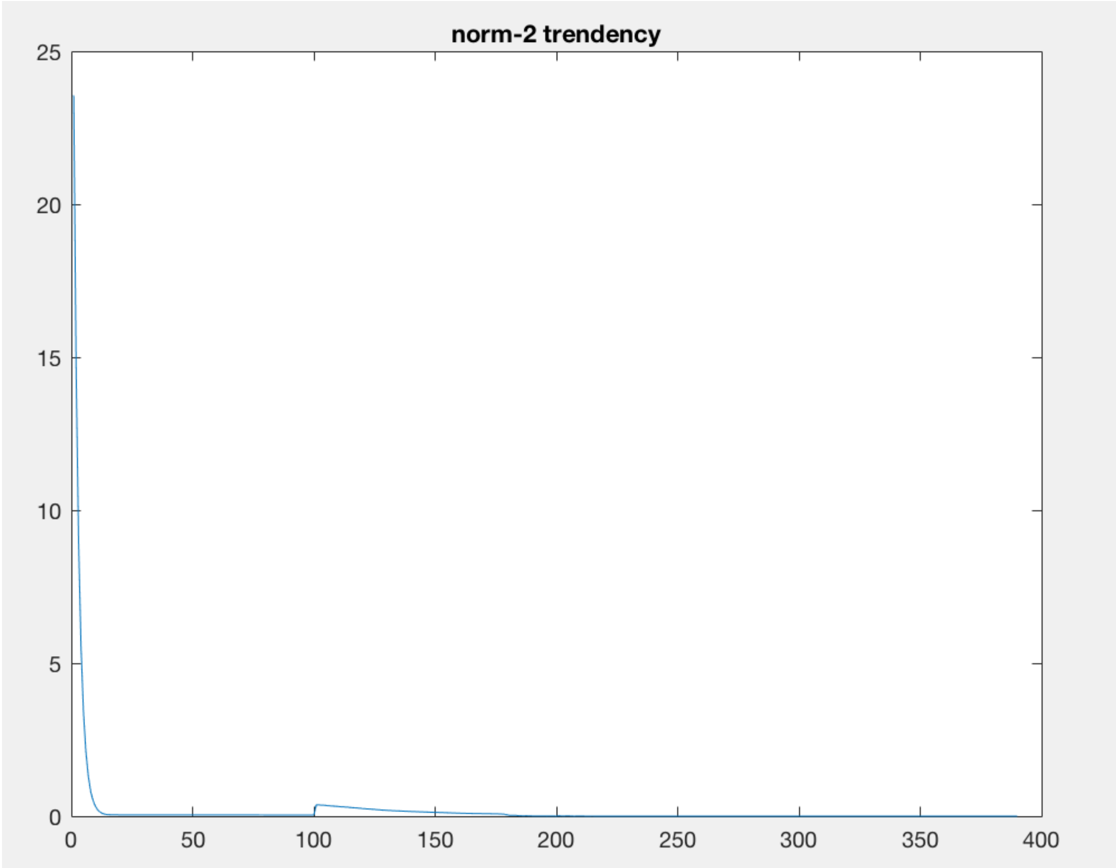
**Require:** initial value $S_0, \lambda_0, x_0, \mu$,final $\mu_{final}$
  **while** $\mu > \mu_{final}$ **do**
    $\mu = \frac{\mu}{10}$
    **while** not satify the stop condition **do**
      $\lambda^{k+1} = (AA^T)^{-1}(\mu(-Ax^k + b) + As^k)$
      $S^{k+1} = P_{[-1,1]}(A^T\lambda^{k+1} + \mu x^k)$
      $x^{k+1} = x^k + \frac{0.618}{\mu}(A^T\lambda^{k+1} - S^{k+1})$
    **end while**
  **end while**
  Output x

---

最终得到的结果如下所示:




变量x的当前与前一次的差的3范数变化如下:




最终的代码见bp_dual_admm.m.

## Problem 2

**Solution**

(a).论文([3])使用了两种方式来生成样本，如下:

model 1:

```
function [s] = generate_s()
%model 1
s = ones(30,30);
for i=1:30
for j=1:30
s(i,j) = power(0.6,abs(i-j));
end
end
end
```

model 2:

```
function [s] = generate_s_2()
%model 1
s = zeros(30,30);
```

Problem 2 continued on next page. . . 

```
sample = [0,0.5];
for i=1:30
for j=1:30
tt = discreteRnd([0.9,0.1],1);
s(i,j) = sample(tt);
end
end

for i=1:30
s(i,i) = 0;
end


for i=1:100
n_m = s+eye(30)*i/10.0;
ttt = cond(n_m);
if ttt >50 & ttt<100
fprintf('condition number: %s\n',ttt);
s =n_m;
break
end
end

end




function x = discreteRnd(p, n)
% Generate samples from a discrete distribution (multinomial).
% Input:
%   p: k dimensional probability vector
%   n: number of samples
% Ouput:
%   x: k x n generated samples x~Mul(p)
% Written by Mo Chen (sth4nth@gmail.com).
if nargin == 1
n = 1;
end
r = rand(1,n);
p = cumsum(p(:));
[~,x] = histc(r,[0;p/p(end)]);
end
```

(b). 对偶推导如下:
令Z=X,原问题可以转换成:

$$\begin{aligned} min_{X,Z}\ tr(SX) + \rho||Z||_1 - logdetX \\ S.t\ Z = X, X \succ 0 \end{aligned}$$

上述问题的Lagragian函数为:

$$L(X,Z,A,B) = tr(SX + \rho||Z||_1 - logdetX + < A, Z - X > - < B, X > +AZ + \rho||Z||_1$$

Problem 2 continued on next page... 

对于变量Z,若要$AZ+\rho||Z||_1$最小，则必须要有$||A||_\infty \le \rho$,否则就会出现unbounded.
对于变量X,对X求导可以得到:

$$X = (S^T - A - B)^{-1}$$

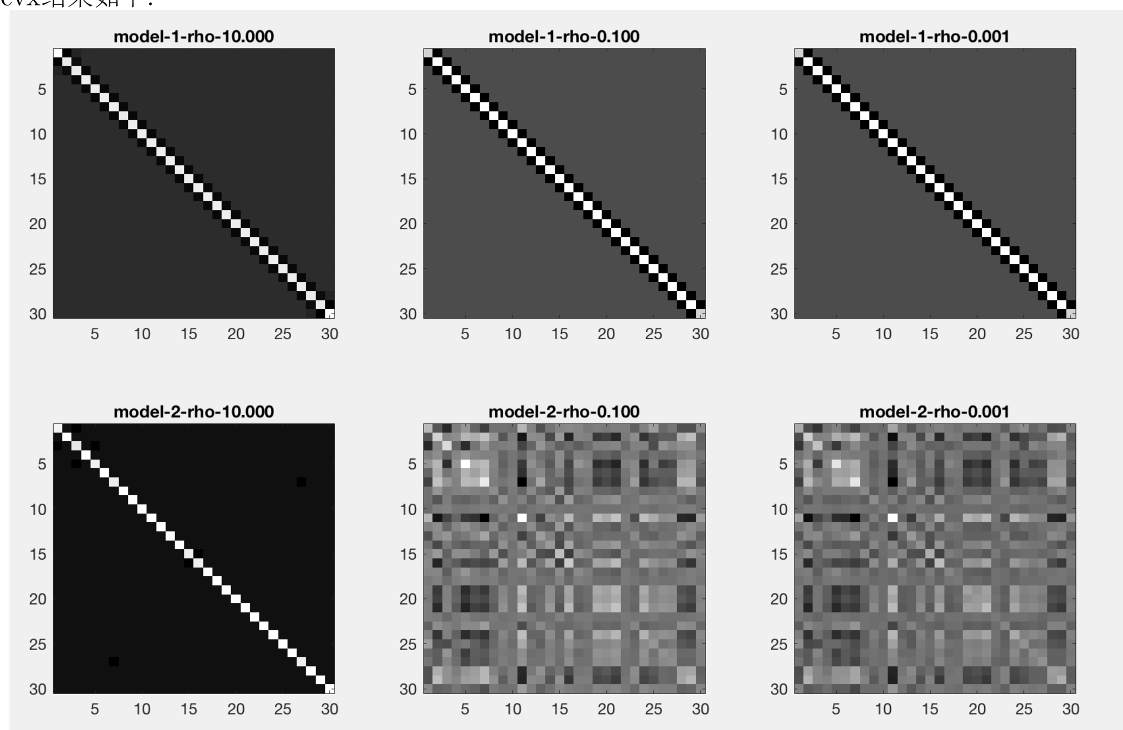
综合上述分析，原问题的对偶函数为:

$$\begin{aligned} g(A,B) &= inf_{X,Z} L(X,Z,A,B) = L(X,Z,A,B)_{X=(S^T-A-B)^{-1}} \\ &=< S^T - A - B, (S^T - A - B)^{-1} > -logdet(S^T - A - B)^{-1} \\ &\quad S.t\ ||A||_\infty \le \rho, B \succeq 0 \end{aligned}$$

因此原问题的对偶问题为:

$$\begin{aligned} max_{A,B}\ & < S^T - A - B, (S^T - A - B)^{-1} > -logdet(S^T - A - B)^{-1} \\ & S.t\ ||A||_\infty \le \rho, B \succeq 0 \end{aligned}$$

(c).
使用cvx结果如下:




其中，当$\rho$=0.1或者0.001,model=2时会出现unbounded现象。
代码如下:

```
function [out] = sice_cvx_mosek(S,rho)
n = size(S,2);
cvx_solver mosek
cvx_begin
variable X(n,n);
minimize( - log_det(X) + trace(S*X) + rho*norm(X,1));
subject to
X == semidefinite(n);
```

```
cvx_end


out = X;
end
```

(d)
本问题和boyd网站上的题目类似，为了方便自己和boyd实现的结果对比，接下来采用boyd的变量命名。
原问题为:

$$\begin{aligned} max_X \ logdetX - tr(SX) - \gamma ||Z||_1 \\ S.t \ Z = X, X \succeq 0 \end{aligned}$$

原问题转换为标准的lagragian问题:

$$\begin{aligned} min_X \ -logdetX + tr(SX) + \gamma ||Z||_1 \\ S.t \ Z = X, X \succeq 0 \end{aligned}$$

相应的增广Lagragian函数为:

$$\begin{aligned} AL(X, Z, \gamma, W) &= tr(SX) - logdetX + \gamma ||Z||_1 + < W, -Z + X > + \frac{1}{2\rho} ||X - Z||_F^2 \\ &= tr(SX) - logdetX + \gamma ||Z||_1 + \frac{1}{2\rho} ||X - Z + \rho W||_F^2 \end{aligned}$$

使用ADMM来求解该问题:

$$\begin{aligned} X &= argmin_X (tr(SX) - logdetX + \frac{1}{2\rho} ||X - Z + \rho W||_F^2) \\ Z &= shrinkage(x + \rho W, \rho\gamma) \\ W &= W + \frac{1}{\rho}(X - Z) \end{aligned}$$

主要是第一步中的X-update比较麻烦，对右边求导，并且右乘矩阵X,化简以后可以得到:

$$X^2 + (\rho W + \rho S - Z)X - \rho E = 0$$

首先考虑对$\rho W + \rho S - Z$做谱分解，得到:

$$\begin{aligned} \rho W + \rho S - Z &= Q \wedge Q^T \\ \tilde{\wedge_{ii}} &= \frac{-\wedge_{ii} + \sqrt{\wedge_{ii}^2 + 4\rho}}{2} \\ X_{new} &= Q \tilde{X} Q^T \end{aligned}$$

这样就解决了x-update的问题，最终的算法流程如下:
具体代码参考`covsel.m`.

**Algorithm 3** Sparse Inverse Covariance Estimation Algorithm

**Require:** initial value $X_0, \gamma, \rho, W_0, Z_0$,final $\rho_{final}$

**while** $rho > \rho_{final}$ **do**

$\rho = \frac{\rho}{10}$

**while** not satisfy the stop condition **do**

$spectraldecomposition: \ \rho W + \rho S - Z = Q \wedge Q^T$

$\tilde{\wedge_{ii}} = \frac{-\wedge_{ii} + \sqrt{\wedge_{ii}^2 + 4\rho}}{2}$

$X - update: \ X_{new} = Q\tilde{X}Q^T$

$Z - update: Z_{new} = shrinkage(x + \rho W, \rho\gamma)$

$W_{new} = W + \frac{1}{\rho}(X - Z)$

**end while**

**end while**

Output x

# Problem 3

**Solution**

(a). 做变量替换以后如下:

$$\begin{aligned} min_{X,Y} \ ||SX - I||_F^2 \leq \sigma^2 \\ S.tY = X, X \succeq 0 \end{aligned}$$

对应的Lagragian函数如下:

$$\begin{aligned} L(X, Y, t, A, B) = ||Y||_1 + t(||SX - I||_F^2 - \sigma^2) + < A, Y - X > - < B, X > \\ \frac{\partial L}{\partial X} = -2tS^T(SX - I) - A - B = 0 \\ X = (S^TS)^+[S^T - \frac{1}{2t}(A + B)] \end{aligned}$$

下面再以Y为研究对象:

$$min_Y L(X, Y, t, A, B) = ||Y||_1 + < A, Y >$$

当$||A||_\infty \leq 1$时，最小值为0，否则unbounded.

因此原问题的对偶函数为:

$$g(t, A, B) = t(||SX - I||_F^2 - \sigma^2) - (A + B, X)$$

其中$X = (S^TS)^+[S^T - \frac{1}{2t}(A + B)]$

最后原问题的对偶问题为:

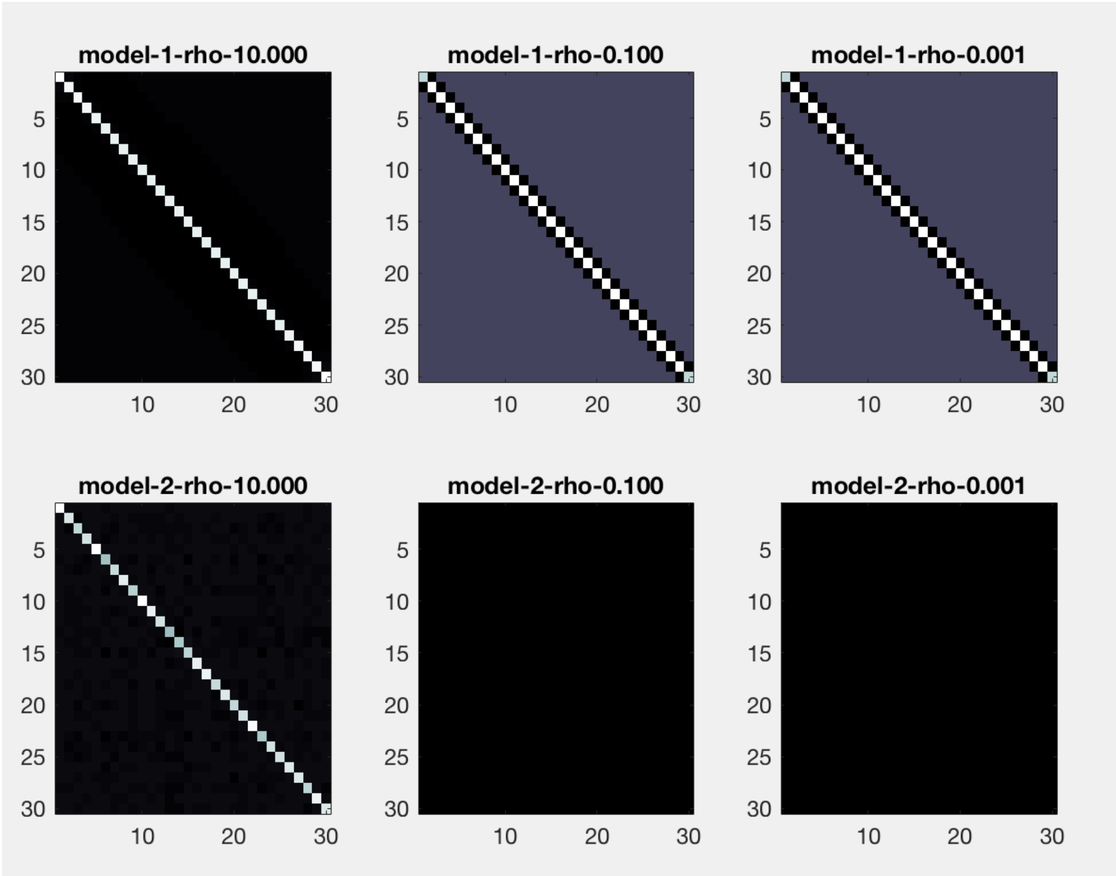
$$\begin{aligned} max_{t,A,B} \ t(||SX - I||_F^2 - \sigma^2) - (A + B, X) \\ S.t||A||_\infty \leq 1, B \succeq 0, t > 0 \end{aligned}$$

其中$X = (S^TS)^+[S^T - \frac{1}{2t}(A + B)]$

(b).

使用cvx结果如下:

Problem 3 continued on next page. . . 




当$\rho$=0.1或者0.001,model=2的时候，会出现infeasible的现象。
代码如下:

```
function [out] = f_cvx_mosek(S,rho)
n = size(S,2);
I = eye(n);
cvx_solver mosek
cvx_begin
variable X(n,n);
minimize(norm(X,1));
subject to
X == semidefinite(n);
norm(S*X-I,'fro') <= rho;
cvx_end


out = X;
end
```

(c)
使用SDPT3代码如下:

```
content...
```

(d).

Problem 3 continued on next page...

//TODO

# References

[1] W. Yin, S. Osher, D. Goldfarb, and J. Darbon, "Bregman iterative algorithms for \ell_1-minimization with applications to compressed sensing," *SIAM Journal on Imaging sciences*, vol. 1, no. 1, pp. 143–168, 2008.

[2] J. Yang and Y. Zhang, "Alternating direction algorithms for \ell_1-problems in compressive sensing," *SIAM journal on scientific computing*, vol. 33, no. 1, pp. 250–278, 2011.

[3] T. Cai, W. Liu, and X. Luo, "A constrained 1 minimization approach to sparse precision matrix estimation," *Journal of the American Statistical Association*, vol. 106, no. 494, pp. 594–607, 2011.