HW5

Carl Mueller
CSCI 5254 - Convex Optimization

April 8, 2018

6.1

Proposition 1.

$$log(x+1) \leq x \tag{1}$$
$$-log(x+1) \geq x \tag{2}$$
$$x > -1 \tag{3}$$

Proposition 2.

$$-log(1-x) \text{ is convex} \tag{4}$$

Proposition 3.

$$\phi(||u||_\infty) = -a^2 log(1 - \frac{||u||_\infty}{a^2}) \tag{5}$$

Left inequality working backwards:

$$||u||_2^2 \leq -a^2 \sum_{i=1}^{m} log(1 - \frac{u_i^2}{a^2})$$
$$\sum_{i=1}^{m} \frac{|u_i|^2}{a^2} \leq -\sum_{i=1}^{m} log(1 - \frac{u_i^2}{a^2})$$

Given proposition 1:

$$\sum_{i=1}^{m} \frac{|u_i|^2}{a^2} \leq -\sum_{i=1}^{m} log(1 - \frac{u_i^2}{a^2})$$

true when:

$$-\frac{u_i^2}{a^2} \geq -1$$

Right inequality:

$$\text{Given: } u_i^2 \leq ||u_i||_\infty^2$$

$$\sum_{i=1}^{m} -log(1-\frac{u_i^2}{a^2}) \leq \sum_{i=1}^{m} -log(1-\frac{||u_i||_\infty^2}{a^2}) \text{ given proposition 1}$$

$$\sum_{i=1}^{m} -log(1-\frac{u_i^2}{a^2}) \leq \frac{u_i^2}{||u||_2^\infty} \sum_{i=1}^{m} -log(1-\frac{||u_i||_\infty^2}{a^2}) \text{ given } \frac{u_i^2}{||u||_2^\infty} \geq 1$$

$$-a^2 \sum_{i=1}^{m} log(1-\frac{u_i^2}{a^2}) \leq -a^2 \frac{u_i^2}{||u||_2^\infty} \sum_{i=1}^{m} log(1-\frac{||u_i||_\infty^2}{a^2})$$

$$-a^2 \sum_{i=1}^{m} log(1-\frac{u_i^2}{a^2}) \leq \frac{u_i^2}{||u||_2^\infty} \phi(||u||_\infty)$$

6.9

To show convexity, the following level set must be convex:

$$S_\alpha = \{\, t_i \mid \max_{i=1,\ldots,k} |\frac{p(t_i)}{q(t_i)} - y_i| \leq \alpha \,\}$$

Due to absolute value, following inequalities must hold:

$$-\alpha q(t_i) \leq y_i q(t_i) - p(t_i) \leq \alpha q(t_i)$$

This is represent two inequalities that define a polyhedron and is therefore convex. Since the level set is convex, the original minimzation problem is at least quasiconvex.

7.3

Proposition 1.

$$P(x|y=1) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{\frac{-z^2}{2}} dz \tag{6}$$

$$P(x|y=0) = 1 - \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{\frac{-z^2}{2}} dz \tag{7}$$

Ordering probability terms in order of $y = 1$ and $y = 0$, our total probability is:

$$p(a,b) = \prod_{i=1}^{q} P_i(a^T u_i + b|y=1) \prod_{i=q+1}^{m} (1 - P_i(a^T u_i + b|y=0))$$

The negative log likelihood:

$$l(a,b) = \sum_{i=1}^{q} -log(P_i(a^T u_i + b|y=1)) + \sum_{i=q+1}^{m} -log(1 - P_i(a^T u_i + b|y=0))$$

The negative log likelihood is a convex function, so minimizing this function is a convex optimization problem.

7.4

a)

Proposition 1.

$$\text{Sample mean: } u = \frac{1}{N}\sum_{k=1}^{N} y_k \tag{8}$$

$$\text{Covariance: } Y = \frac{1}{N}\sum_{k=1}^{N}(y_k - u)(y_k - u)^T \tag{9}$$

$$\begin{aligned}
&-\frac{N}{2}nlog(2\pi) - \frac{N}{2}log(det(R)) - \frac{1}{2}R^{-1}\sum_{k=1}^{N}(y_k - a)(y_k - a)^T \\
&= -\frac{N}{2}nlog(2\pi) - \frac{N}{2}log(det(R)) - \frac{1}{2}R^{-1}\sum_{k=1}^{N}(y_k y_k^T - a y_k^T - y_k a^T + aa^T) \\
&= -\frac{N}{2}nlog(2\pi) - \frac{N}{2}log(det(R)) - \frac{1}{2}R^{-1}(\sum_{k=1}^{N} y_k y_k^T - \sum_{k=1}^{N} a y_k^T - \sum_{k=1}^{N} y_k a^T + \sum_{k=1}^{N} aa^T)) \\
&= -\frac{N}{2}nlog(2\pi) - \frac{N}{2}log(det(R)) - \frac{1}{2}R^{-1}(\sum_{k=1}^{N} y_k y_k^T - \sum_{k=1}^{N} a y_k^T - \sum_{k=1}^{N} y_k a^T + Naa^T)
\end{aligned}$$

Substitute sample mean:

$$\begin{aligned}
&= -\frac{N}{2}nlog(2\pi) - \frac{N}{2}log(det(R)) - \frac{1}{2}R^{-1}\sum_{k=1}^{N} y_k y_k^T - Nay^T - Nua^T + Naa^T \\
&= R^{-1}\sum_{k=1}^{N}(y_k - a)(y_k - a)^T - R^{-1}N(a-u)(a-u)^T \\
&= -\frac{N}{2}nlog(2\pi) - \frac{N}{2}log(det(R)) - \frac{1}{2}(NR^{-1}Y + R^{-1}N(a-u)(a-u)^T) \\
&= -\frac{N}{2}nlog(2\pi) - \frac{N}{2}log(det(R)) - \frac{1}{2}(Ntr(R^{-1}Y) + N(a-u)R^{-1}(a-u)^T)
\end{aligned}$$

Set the gradient to zero to see a and R optimal values.

$$\begin{aligned}
\nabla_a l(R,a) &= -2R^{-1}(a-u) = 0 \\
&\therefore a = u \\
\nabla_R l(R,a) &= -R^{-1} + R^{-1}(Y - (a-u)(a-u)^T)R^{-1} = 0 \\
R &= Y + (a-u)(a-u)^T \\
R &= Y + (0)(0)^T \\
&\therefore R = Y
\end{aligned}$$

7.8

Express sign function as a probability where we order values with $y > 1$ followed by $y < 0$:

$$\prod_{i=1}^{k} prob(a_i^T x + b_i + v_i > 0) \prod_{i=k+1}^{m} prob(a_i^T x + b_i + v_i < 0)$$

Since a_i and b_i are known values, the only RV is the noise term. We can epxress v_i as an expression of $a_i^T x + b_i$. P represents the cumulative density function of v_i. We can represent the probability as follows:

$$\prod_{i=1}^{k} P(-a_i^T x - b_i) \prod_{i=k+1}^{m} 1 - P(-a_i^T x - b_i)$$

Log likelihood below is concave so if we maximize, we obtain a convex problem:

$$l(x) = \sum_{i=1}^{k} log(P(-a_i^T x - b_i)) + \sum_{i=k+1}^{m} log(1 - P(-a_i^T x - b_i))$$

7.9

Given

$$y_i = f(a_i^T x + b_i + v_i), i = 1, \ldots, m$$

We know that a_i and b_i are knowns, so lets expression the random variable v_i as an expression of all other terms. We assume that f is an invertible function.

$$v_i = f^{-1}(y_i) - a_i^T x - b_i$$

The probability of observing $y_i, \ldots, y_m$ is:

$$\prod_{i=1}^{m} prob(f^{-1}(y_i) - a_i^T x - b_i)$$

$$l(x, f) = \sum_{i=1}^{m} log(prob(f^{-1}(y_i) - a_i^T x - b_i))$$

This log probability is concave w.r.t x and f. Thus maximizing generates a convex optimization problem.

Additional Exercises:

3.9

a)

Given:

$$z = [\Re x, \Im x]$$

Setup a system of equations using the vector breakdown of x for its $\Re$ and $\Im$ components:

$$||x||_2^2 = ||z||_2^2$$

$$\begin{bmatrix} \Re A & -\Im A \\ \Im A & \Re A \end{bmatrix} \begin{bmatrix} \Re x \\ \Im x \end{bmatrix} = \begin{bmatrix} \Re b \\ \Im b \end{bmatrix}$$

This becomes the optimization problem:

$$\begin{aligned} \underset{z}{\text{minimize}} \quad & ||z||_2 \\ \text{subject to} \quad & \begin{bmatrix} \Re A & -\Im A \\ \Im A & \Re A \end{bmatrix} \begin{bmatrix} \Re x \\ \Im x \end{bmatrix} = \begin{bmatrix} \Re b \\ \Im b \end{bmatrix} \end{aligned}$$

b)

Define the second order cone:

$$K_i = \{ \, (z,t) \mid ||z||_2 \leq t \, \}$$

The SOCP:

$$\begin{aligned} \text{minimize} \quad & t \\ \text{subject to} \quad & |z||_2 \\ & \begin{bmatrix} \Re A & -\Im A \\ \Im A & \Re A \end{bmatrix} \begin{bmatrix} \Re x \\ \Im x \end{bmatrix} = \begin{bmatrix} \Re b \\ \Im b \end{bmatrix} \end{aligned}$$

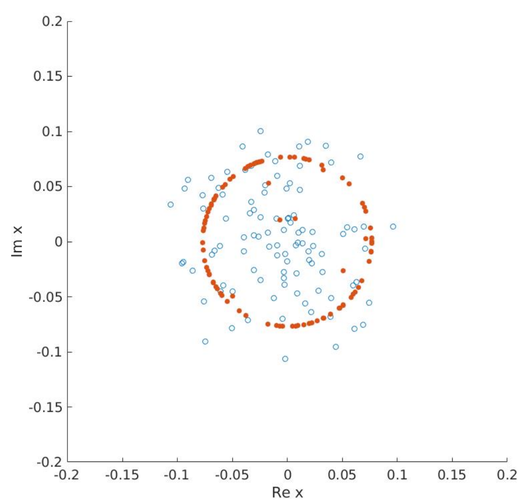
c)

Code:

```
randn('state',0);
m = 30; n = 100;
Are = randn(m,n); Aim = randn(m,n);
bre = randn(m,1); bim = randn(m,1);
A = Are + i*Aim;
b = bre + i*bim;

Atot = [Are -Aim; Aim Are];
btot = [bre; bim];
z_2 = Atot'*inv(Atot*Atot')*btot;
x_2 = z_2(1:100) + i*z_2(101:200);
```

```
cvx_begin
    variable x(n) complex
    minimize( norm(x) )
    subject to
    A*x == b;
cvx_end

cvx_begin
    variable xinf(n) complex
    minimize( norm(xinf,Inf) )
    subject to
    A*xinf == b;
cvx_end

figure(1)
scatter(real(x),imag(x)), hold on,
scatter(real(xinf),imag(xinf),[],'filled'), hold off,
axis([-0.2 0.2 -0.2 0.2]), axis square,
xlabel('Re x'); ylabel('Im x');
```

Results: The red dots represent the infinity norm.



4.1

a)

Code:

```
M = [1 -1/2; -1/2 2];
m = [-1 0]';
A = [1 2; 1 -4; 5 76];
```

```
b = [-2 -3 1]';
delta = .1

cvx_begin
    variable x(2)
    dual variable y
    minimize(quad_form(x, M)+m'*x)
    subject to
        y: A*x <= b;
cvx_end
p_star = cvx_optval
y
x
```

Results:
p_star = 8.2222
y =
1.8994
3.4684
0.0931
x =
-2.3333
0.1667

KKT Conditions
Primal:

$$x_1^* + 2x_2^* \leq u_1$$
$$x_1^* + -4x_2^* \leq u_2$$
$$5x_1^* + 76x_2^* \leq 1$$

Dual:

$$\lambda_1^*, \lambda_2^*, \lambda_3^* \geq 0$$

Complementary Slackness:

$$\lambda_1^*(x_1^* + 2x_2^* - u_1) = 0$$
$$\lambda_2^*(x_1^* + -4x_2^* - u_2) = 0$$
$$\lambda_3^*(5x_1^* + 76x_2^* - 1) = 0$$

First Order Conditions:

$$4x_2^* - x_1^* + 2\lambda_1^* - 4\lambda_2^* + 76\lambda_3^* = 0$$
$$2x_1^* - x_2^* - 1 + \lambda_1^* + \lambda_2^* + 5\lambda_2^* = 0$$

b)

Code:

```
M = [1 -1/2; -1/2 2];
m = [-1 0]';
A = [1 2; 1 -4; 5 76];
b = [-2 -3 1]';

cvx_begin
    variable x(2)
    dual variable y
    minimize(quad_form(x, M)+m'*x)
    subject to
        y: A*x <= b;
cvx_end
p_star = cvx_optval

array = [0 -1 1];
table = [];
delta = 0.1;

for i = array
    for j = array
        p_pred = p_star - [y(1) y(2)]*[i; j]*delta;
        cvx_begin
            variable x(2)
            minimize(quad_form(x,M)+m'*x)
            subject to
                A*x <= b+[i;j;0]*delta
        cvx_end
        p_exact = cvx_optval;
        table = [table; i*delta j*delta p_pred p_exact]
    end
end
```

Results:

d_1	d_2	p^*_{pred}	p^*_{exact}
0	0	8.2222	8.2222
0	-0.1000	8.5691	8.7064
0	0.1000	7.8754	7.9800
-0.1000	0	8.4122	8.5650
-0.1000	-0.1000	8.7590	8.8156
-0.1000	0.1000	8.0653	8.3189
0.1000	0	8.0323	8.2222
0.1000	-0.1000	8.3791	8.7064
0.1000	0.1000	7.6854	7.7515

We can see that $p^*_{pred} \leq p^*_{exact}$ for all pertubations.

5.2

The objective function $\max_{i=1,\ldots,k} |f(t_i) - y_i|$ is not convex, however it is quasiconvex:

$$\{\, t, y, \alpha \mid \max_{i=1,\ldots,k} |f(t_i) - y_i| \leq \alpha \,\}$$

as it is a linear inequality.

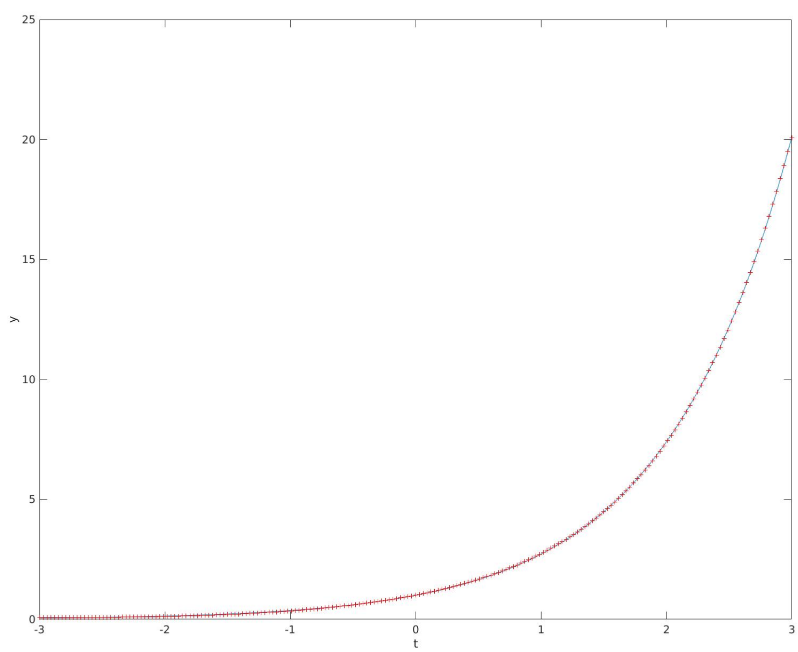


Figure 1: Data and optimal function fit.

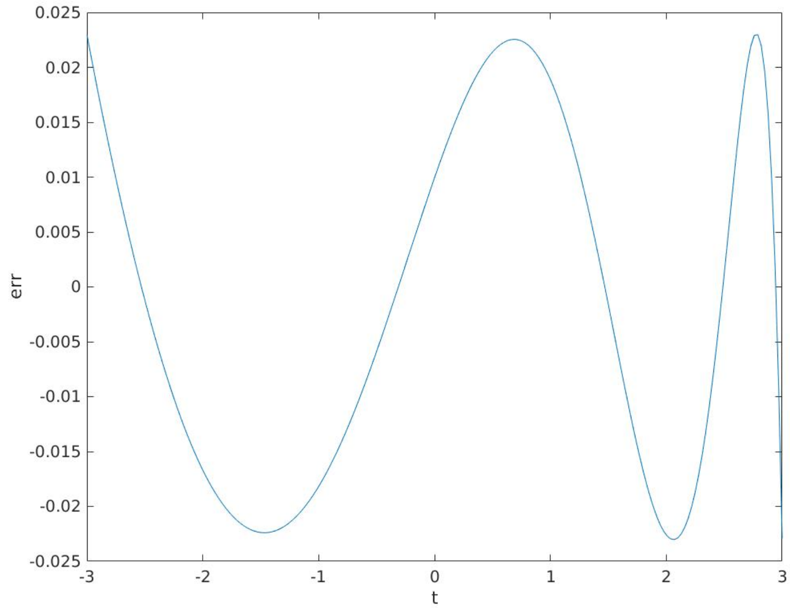


Figure 2: Error for the given t value.

To solve we can use the bisection method:

Code:

```
upper = exp(5);
lower = 0;
tolerance = .001
k = 201
t=(-3:6/(k-1):3)';
y=exp(t);
% 1 + t + t^2
T=[ones(k,1) t t.^2];

while upper - lower >= tolerance
    midpoint = (lower + upper)/2
    cvx_begin
    % a_0, a_1, a_2
    variable a(3)
    % b_0, b_1
    variable b(2)
    subject to
        abs(T*a-y.*(T*[1;b])) <= midpoint*T*[1;b]
    cvx_end
    if strcmp(cvx_status,'Solved')
        a_star = a;
        b_star = b;
```

```
        upper = midpoint;
        value = midpoint;
    else
        lower = midpoint
    end
end

y_star = T*a_star./(T*[1;b_star]);
y_star
a_star
b_star

figure(1);
plot(t,y,'g', t,y_star,'r');
xlabel('t');
ylabel('y');

figure(2);
plot(t, y_star-y);
xlabel('t');
ylabel('err');
```

Results:
a$_s tar$ =
1.0099
0.6115
0.1133
b_star =
− 0.4147
0.0485

5.6

Code:

```
% tv_img_interp.m
% Total variation image interpolation.
% Defines m, n, Uorig, Known.
% Load original image.
pwd()
Uorig = double(imread('/home/carl/CUBoulder/coursework/5254/HW5/tv_img_interp
[m, n] = size(Uorig);
% Create 50% mask of known pixels.
rand('state', 1029);
Known = rand(m,n) > 0.5;
```

```
%%%%% Put your solution code here
% Calculate and define Ul2 and Utv.
% Placeholder:
cvx_begin
variable Ul2(m, n);
Ul2(Known) == Uorig(Known);
Ux = Ul2(2:end,2:end) - Ul2(2:end,1:end-1);
Uy = Ul2(2:end,2:end) - Ul2(1:end-1,2:end);
% Squared / l2 norm
minimize(norm([Ux(:); Uy(:)], 2));
cvx_end
cvx_begin
variable Utv(m, n);
Utv(Known) == Uorig(Known);
Ux = Utv(2:end,2:end) - Utv(2:end,1:end-1);
Uy = Utv(2:end,2:end) - Utv(1:end-1,2:end);
% abs or l1 norm
minimize(norm([Ux(:); Uy(:)], 1)); % tv roughness measure
cvx_end
%%%%%
% Graph everything.
figure(1); cla;
colormap gray;
subplot(221);
imagesc(Uorig)
title('Original image');
axis image;
subplot(222);
imagesc(Known.*Uorig + 256-150*Known);
title('Obscured image');
axis image;
subplot(223);
imagesc(Ul2);
title('l_2 reconstructed image');
axis image;
subplot(224);
imagesc(Utv);
title('Total variation reconstructed image');
axis image;
```

Results:

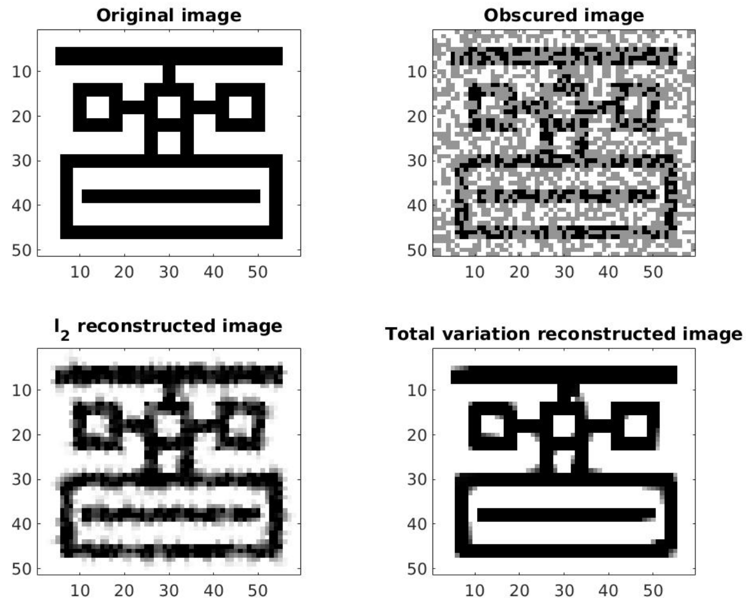


Figure 3: Interpolation results.

5.13

a)

We constrain the problem such that $c^T x_i$ for all censored data points ($i = M+1, \ldots, K$) must be greater than the lower bound D while minimizing the uncensored data $i = 1, \ldots, M$.

$$\begin{array}{ll} \underset{c}{\text{minimize}} & \sum_{i=1}^{M} (y_i - c^T x_i)^2 \\ \text{subject to} & c^T x_i \geq D, \text{ for } i = M+1, \ldots, K \end{array}$$

b)

Code:

```
% data for censored fitting problem.
randn('state',0);
n = 20; % dimension of x's
M = 25; % number of non-censored data points
K = 100; % total number of points
c_true = randn(n,1);
X = randn(n,K);
y = X'*c_true + 0.1*(sqrt(n))*randn(K,1);
% Reorder measurements, then censor
[y, sort_ind] = sort(y);
sort_ind
```

```
X = X(:,sort_ind);
D = (y(M)+y(M+1))/2;
y = y(1:M);
X_uncen = X(:,1:M)
X_cen = X(:,M+1:K)
cvx_begin
    variable c(n)
    minimize(sum_square(y - X_uncen'*c))
    subject to
        X_cen'*c >= D
cvx_end
cvx_begin
    variable c_ls(n)
    minimize(sum_square(y - X_uncen'*c_ls))
cvx_end

norm(c - c_true, 2) / norm(c_true,2)

norm(c_ls - c_true, 2) / norm(c_true,2)
```

Results:
Errors:
$\hat{c} = 0.1538$
$c_{ls} = 0.3907$

5.15

a)

We can optimize the following:

$$\begin{array}{ll} \underset{P}{\text{minimize}} & \frac{1}{N}\sum_{i=1}^{N}(d_i - (x_i - y_i)^T P (x_i - y_i))^2 \\ \text{subject to} & P \succeq 0 \end{array}$$

Another approach would be to maxmize $i = 1, \ldots, M$ dissimilar points for the P-metric while keep $i = M+1, \ldots, N$ similar points less then some arbitrarily small value α:

$$\begin{array}{ll} \underset{P}{\text{maxmize}} & \sum_{i=1}^{M}((x_i - y_i)^T P (x_i - y_i))^{\frac{1}{2}} \\ \text{subject to} & P \succeq 0 \\ & \sum_{i=M+1}^{N}(x_i - y_i)^T P (x_i - y_i) \leq \alpha \end{array}$$

b)

Code:

```
%% data for learning a quadratic metric
% provides X, Y, d, X_test , Y_test , d_test
rand('seed',0);
randn('seed',0);
n = 5; % dimension
N = 100; % number of distance samples
N_test = 10;
X = randn(n,N);
Y = randn(n,N);
X_test = randn(n,N_test);
Y_test = randn(n,N_test);
P =randn(n,n);
P = P*P'+eye(n);
sqrtP = sqrtm(P);
d = norms(sqrtP*(X-Y)); % exact distances
d = pos(d+randn(1,N)); % add noise and make nonnegative
d_test = norms(sqrtP*(X_test-Y_test));
d_test = pos(d_test+randn(1,N_test));
P
alpha = 5;
[d_test, sort_ind] = sort(d_test);
X_test = X_test(:,sort_ind);
Y_test = Y_test(:,sort_ind);
diff = X_test-Y_test

clear P sqrtP;
cvx_begin
    variable P(n,n)
    minimize((1/N_test)*pow_pos(sum(d_test' - sqrt(diag(diff'*P*diff))),2)),
    subject to
    P>0
cvx_end
```

Result: Mean Squared Error = +1.24901e-10