EE-588: Homework # 3

Due on Wednesday, October 23, 2019

Saket Choudhary
2170058637

Contents

4.15

a) If the relaxation LP

$$\begin{aligned} &\min c^T x \\ &\text{s.t. } Ax \preccurlyeq b \\ &\quad 0 \leq x \leq 1 \end{aligned}$$

is feasible, we get a lower bound on the solution of the original LP since $x_i \in \{0, 1\}$. the optimal value will then be lower also for the relaxed version.
b) If the optimal solution for the relaxed LP $x_i^* \in \{0, 1\}$, then it is also the optimal solution for the original LP.

4.17

Given revenue:

$$r_j(x_j) = \begin{cases} p_j x_j & 0 \leq x_j \leq q_j \\ p_j q_j + p_j^{\text{disc}}(x_j - q_j) & x_j \geq q_j \end{cases}$$

Since,

$$\begin{aligned} p_j &> 0 \\ q_j &> 0 \\ 0 < p_j^{\text{disc}} &< p_j \end{aligned}$$

,
the revenue cost function can be simplified to:

$$\min\left(p_j x_j, p_j x_j + p_j^{\text{disc}})(x_j - q_j)\right)$$

Our original optimization problem is given by:

$$\begin{aligned} \max \sum_{j=1}^{n} r_j(x_j) \\ \text{s.t. } x \succcurlyeq 0 \\ Ac \preccurlyeq c^{\text{max}} \end{aligned}$$

This is convex as $\sum_{j=1}^{n} r_j(x_j)$ is affine in concave function $r_j(x_j)$ and we are maximizing a concave function. The inequality constraints themselves are also affine.
We can reduce this to a LP as follows:

$$\begin{aligned} &\max \mathbb{1}^T s_j \\ \text{s.t}\, &x \succcurlyeq 0 \\ &Ax \preccurlyeq c^{\text{max}} \\ &p_j x_j \geq s_j, p_j^{\text{disc}}(x_j - q_j) \geq s_j \quad j = 1, 2, \ldots, N \end{aligned}$$

This LP leads to an optimal solution for the original problem as follows: for a fixed x, we have it satisfying the constraints. Also $r_j(x_j) \geq s_j$ so for a feasible x, s is in the LP and the LP objective itself is less than or equal to the revenue.

5.5

$$\min c^T x$$
$$\text{s.t. } Gx \preccurlyeq h$$
$$Ax = b$$

Consider the langragian:

$$\begin{aligned} L(x,\lambda,\nu) &= c^T x + \lambda^T(Gx-h) + \nu^T(Ax-b) \\ &= (c^T + \lambda^T G + \nu^T A)x - \lambda^T h - \nu^T b \end{aligned}$$

$$g(\lambda,\nu) = \inf_x L(x,\lambda,\nu)$$
$$\text{s.t.} \lambda \succcurlyeq 0$$

where given the linearity of the langragian function,

$$\begin{aligned} g(\lambda,\nu) &= \inf_x L(x,\lambda,\nu) \\ &= \begin{cases} -\lambda^T h - \nu^T A & c + G^T\lambda + A\nu^T = 0 \\ -\infty & \text{otherwise} \end{cases} \end{aligned}$$

The corresponding dual is given by:

$$\max g(\lambda,\nu)$$
$$\text{s.t. } c + G^T\lambda + A^T\nu = 0$$
$$\lambda \succcurlyeq 0$$

SVM and Kernels

a) Given:

$$\min_{w.\tau,v} \sum_{i=1}^{n} \tau_i + \lambda||w||_2^2$$
$$\text{s.t. } 1 - y_i(w^T x_i + v) \leq \tau_i \quad \forall i = 1, \dots, n$$
$$\tau_i \geq 0 \quad \forall i = 1, \dots, n$$

Or equivalently,

$$\min_{w.\tau,v} \sum_{i=1}^{n} \tau_i + \lambda||w||_2^2$$
$$\text{s.t. } 1 - y_i(w^T x_i + v) - \tau_i \leq 0 \quad \forall i = 1, \dots, n$$
$$-\tau_i \leq 0 \quad \forall i = 1, \dots, n$$

For a fixed $\tau_i \geq 0$, we have $1 - y_i(w^T x_i + v) \leq \tau_i \implies \max(0, 1 - y_i(w^T x_i + v)) = 1 - y_i(w^T x_i + v)$ and hence we obtain the original optimization problem $\min_{w,v} \max(0, 1 - y_i(w^T x_i + v)) + \lambda||w||_2^2$
b) Dual problem:

$$\max \tilde{L}(\tau) L(\alpha, \beta w, v, \tau) \quad = \lambda||w||_2^2 + \sum_{i=1}^{n} \tau_i + \sum_{i=1}^{n} \alpha_i(1 - y_i(w^T x_i + v) - \tau_i) - \sum_{i=1}^{n} \beta_i \tau_i$$
$$\nabla_w L = 0$$
$$\implies w - \sum_{i=1}^{n} \alpha_i y_i x_i = 0$$
$$\implies w = \sum_{i=1}^{n} \alpha_i y_i x_i$$
$$\nabla_v L = 0$$
$$\implies \sum_{i=1}^{n} \alpha_i = 0$$
$$\nabla_{\tau_i} L = 0$$
$$\implies 1 - \alpha_i - \beta_i = 0$$
$$\implies \alpha_i + \beta_i = 1 \quad \forall i = 1, \dots, n$$

Substituting for w, we get:

$$\max \quad \tilde{L} = \lambda(\sum_{i=1}^{n} \alpha_i y_i x_i)^2 + \sum_{i=1}^{n} \alpha_i + \sum_{i=1}^{n} \tau_i(1 - \sum_{i=1}^{n} \alpha_i - \sum_{i=1}^{n} \beta_i) - \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i x_j$$
$$= \sum_{i=1}^{n} \alpha_i + \lambda(\sum_{i=1}^{n} \alpha_i^2 y_i^2 x_i^2) + (2\lambda - 1) \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i x_j$$
$$\text{s.t. } 0 \leq \alpha_i \leq 1$$
$$\sum_{i=1}^{n} \alpha_i = 0$$

SVM and Kernels continued on next page...

c) KKT conditions:

$$\begin{aligned}
\alpha_i &\geq 0 \\
\beta_i &\geq 0 \\
1 - y_i(w^T x_i + v) - \tau_i &\leq 0 \\
\alpha_i(1 - y_i(w^T x_i + v) - \tau_i) &\leq 0 \\
\beta_i &\geq 0 \\
\beta_i \tau_i &\geq 0 \\
w &= \sum_{i=1}^{n} \alpha_i y_i x_i \\
\sum_{i=1}^{n} \alpha_i &= 0 \\
\alpha_i + \beta_i &= 1
\end{aligned}$$

d) If the training samples themselves are not known and we have access to the kernel matrix $K_{i,j} = \langle x_i, x_j \rangle$, we can still compute the dual problem by substituting $K_{i,j}$ as the inner product:

$$\max L(\alpha) = \sum_{i=1}^{n} \alpha_i + \lambda(\sum_{i=1}^{n} \alpha_i^2 y_i^2 K_{i,i}) + (2\lambda - 1) \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K_{i,j}$$

s.t. $0 \leq \alpha_i \leq 1$

$$\sum_{i=1}^{n} \alpha_i = 0$$

Reformulating constraints in CVX

Using the corrected constraints,

(a) `norm` cannot be used with an equality constraint as it is not affine and is redundant here.
Change to: `x+2*y ==0; x-y == 0`

(b) The outermost `square` requires affine inputs.
Change to: `square_pos(square(x+y)) <= x-y`

(c) `1/x` is not convex unless the domain is restricted to $\mathbb{R}_+$ and similarly `1/y` requires $\mathbb{R}_+$ for it to be convex. `inv_pos` function inherently uses $\mathbb{R}_+$ as the domain for any variable inside it.
Change to: `inv_pos(x) + inv_pos(y) <=1`

(d) `norm` can only take affine inputs.
Change to:
```
norm([u,v]] <= 3*x +y
max(x,1) <= u
max(y,2) <= v
```

(e) `x*y` is not concave. we use the trick from previous part.
Change to: `x >= inv_pos(y)`

(f) $(x+y)^2$ is convex while `sqrt(y)` is concave and a convex function cannot be divided by concave.
Change to: `quad_over_lin(x+y,y)`

(g) x^3 is convex over $\mathbb{R}_+$ and hence we need to change to `pow_pos` so that the constraints of $\mathbb{R}_+$.
Change to: `pow_pos(x,3) + pow_pos(y,3) <=1`

(h) `xy` is not concave. $xy - z^2 = x(y - z^2/y)$
Change to: `x+z <= 1 + geo_mean([x-quad_over_lin(z,y),y])`

combining all the reforumated constraints in CVX results in an intractable problem:

```
cvx_begin
variables x y u z v
x == 0;
y == 0;
square_pos( square( x + y ) ) <= x - y
inv_pos(x)+inv_pos(y)<=1
norm( [ u ; v ] ) <= 3*x + y;
max( x , 1 ) <= u;
max( y , 2 ) <= v;
x >= inv_pos(y);
x >= 0;
y >= 0;
quad_over_lin(x + y , sqrt(y)) <= x - y + 5;
pow_pos(x,3) + pow_pos(y,3) <= 1;
x+z <= 1+geo_mean([x-quad_over_lin(z,y),y])
cvx_end
```

solution:

```
Calling SDPT3 4.0: 64 variables, 26 equality constraints
   For improved efficiency, SDPT3 is solving the dual problem.
------------------------------------------------------------

 num. of constraints = 26
 dim. of sdp    var  = 26,   num. of sdp  blk  = 13
 dim. of socp   var  =  3,   num. of socp blk  =  1
 dim. of linear var  = 22
 2 linear variables from unrestricted variable.
 *** convert ublk to lblk
*******************************************************************
   SDPT3: Infeasible path-following algorithms
*******************************************************************

  sqlp stop: dual problem is suspected of being infeasible
-------------------------------------------------------------------
 number of iterations   = 12
 residual of dual infeasibility
 certificate X          = 3.13e-12
 reldist to infeas.    <= 3.56e-13
 Total CPU time (secs)  = 0.44
 CPU time per iteration = 0.04
 termination code       =  2
 DIMACS: 1.9e-05  0.0e+00  7.3e-01  0.0e+00  -1.0e+00  5.2e-03
-------------------------------------------------------------------

------------------------------------------------------------
Status: Infeasible
Optimal value (cvx_optval): +Inf
```

Optimal Activity Levels

Code:

```
    A = [1 2 0 1;
     0 0 3 1;
     0 3 1 1;
     2 1 2 5;
     1 0 3 2];
cmax = [100; 100; 100; 100; 100];

p = [3; 2; 7; 6];
pdisc = [2; 1; 4; 2];
q = [4; 10 ;5; 10];

cvx_begin
variable x(4)
```

```
maximize( sum(min(p.*x,p.*q+pdisc.*(x-q))) )
subject to
x >= 0;
A*x <= cmax
cvx_end

x
r = min(p.*x,p.*q+pdisc.*(x-q))
revenue = sum(r)
avg_price_per_unit = r./x
```

Output:

```
Calling SDPT3 4.0: 17 variables, 8 equality constraints
   For improved efficiency, SDPT3 is solving the dual problem.
------------------------------------------------------------

 num. of constraints =  8
 dim. of linear var  = 17
*******************************************************************
   SDPT3: Infeasible path-following algorithms
*******************************************************************

 number of iterations   = 10
 primal objective value =  1.92500000e+02
 dual   objective value =  1.92500000e+02
 gap := trace(XZ)       = 4.00e-07
 relative gap           = 1.04e-09
 actual relative gap    = 1.03e-09
 rel. primal infeas (scaled problem)   = 2.53e-13
 rel. dual     "        "       "      = 4.78e-12
 rel. primal infeas (unscaled problem) = 0.00e+00
 rel. dual     "        "       "      = 0.00e+00
 norm(X), norm(y), norm(Z) = 2.2e+00, 8.8e+01, 1.1e+02
 norm(A), norm(b), norm(C) = 1.2e+01, 1.1e+01, 2.3e+02
 Total CPU time (secs)  = 0.40
 CPU time per iteration = 0.04
 termination code       =  0
 DIMACS: 3.5e-13  0.0e+00  1.1e-11  0.0e+00  1.0e-09  1.0e-09
-------------------------------------------------------------------

------------------------------------------------------------
Status: Solved
Optimal value (cvx\_optval): +192.5


x =
```

```
    4.0000
   22.5000
   31.0000
    1.5000


r =

   12.0000
   32.5000
  139.0000
    9.0000


revenue =

  192.5000


avg_price_per_unit =

    3.0000
    1.4444
    4.4839
    6.0000
```