Constrained Optimization

DÉPARTEMENT ARCHITECTURE DES SYSTÈMES D'INFORMATION
4^{th} year
G. Gasso

Objective
We aim to use the Python package CVXPY to solve convex optimization problems ranging from an introductory example to more elaborated problems.

1 A simple example

We want to solve the following problem

$$\begin{aligned} \min_{\boldsymbol{\theta}} \quad & \frac{1}{2}(\theta_1 - 3)^2 + \frac{1}{2}(\theta_2 - 1)^2 \\ \text{s.t.} \quad & \theta_1 + \theta_2 - 1 \leq 0 \\ & \theta_1 - \theta_2 - 1 \leq 0 \\ & -\theta_1 + \theta_2 - 1 \leq 0 \\ & -\theta_1 - \theta_2 - 1 \leq 0 \end{aligned} \tag{1}$$

1.1 Mathematical derivation…

1. Let $\boldsymbol{\theta}$ be $\boldsymbol{\theta} = \begin{bmatrix}\theta_1 \\ \theta_2\end{bmatrix}$. Show that the problem can be cast into the matrix form

$$\begin{aligned} \min_{\boldsymbol{\theta}} \quad & \frac{1}{2}\|\boldsymbol{\theta} - \mathbf{c}\|_2^2 \\ \text{s.t.} \quad & \mathbf{A}\boldsymbol{\theta} - \mathbf{b} \leq \mathbf{0} \end{aligned} \tag{2}$$

   with the vectors $\mathbf{c}$, $\mathbf{b}$ and the matrix $\mathbf{A}$ to be specified. $\mathbf{0}$ is a zeros vector of dimension 4.

2. Show that the Lagrangian function associated to Problem (2) is:

$$\mathcal{L} = \frac{1}{2}\|\boldsymbol{\theta} - \mathbf{c}\|_2^2 + \boldsymbol{\mu}^\top(\mathbf{A}\boldsymbol{\theta} - \mathbf{b})$$

   with $\boldsymbol{\mu} \in \mathbb{R}^4$ and $\boldsymbol{\mu} \geq 0$ a vector of Lagragange multipliers.

3. To get the stationary KKT condition, we need to know some useful derivatives.
   (a) Show that $\|\boldsymbol{\theta} - \mathbf{c}\|_2^2 = \boldsymbol{\theta}^\top\boldsymbol{\theta} - 2\boldsymbol{\theta}^\top\mathbf{c} + \mathbf{c}^\top\mathbf{c}$.
   (b) Knowing that $\nabla_{\boldsymbol{\theta}}(\boldsymbol{\theta}^\top\boldsymbol{\theta}) = 2\boldsymbol{\theta}$ and $\nabla_{\boldsymbol{\theta}}(\boldsymbol{\theta}^\top\mathbf{c}) = \mathbf{c}$, give the expression of $\nabla_{\boldsymbol{\theta}}\|\boldsymbol{\theta} - \mathbf{c}\|_2^2$

4. From the previous question, express the KKT stationary condition; deduce the expression of $\boldsymbol{\theta}$ as a function of $\boldsymbol{\mu}$.

5. Establish that the dual problem is

$$\begin{aligned} \min_{\boldsymbol{\lambda}} \quad & \frac{1}{2}\boldsymbol{\mu}^\top\mathbf{H}\boldsymbol{\mu} + \boldsymbol{\mu}^\top\mathbf{q} \\ \text{s.c.} \quad & \boldsymbol{\mu} \geq 0 \end{aligned} \tag{4}$$

   where the matrix $\mathbf{H}$ and the vector $\mathbf{q}$ are to be specified.
   with $\mathbf{H} = \mathbf{A}\mathbf{A}^\top$ and $\mathbf{q} = -(\mathbf{A}\mathbf{c} - \mathbf{b})$

1.2 ...and numerical implementation

We want to compute the solution of Problem (2). We will use `CVXPy` package at https://www.cvxpy.org/index.html. This package solves convex optimization problems.

1. To start under `CVXPy`, let solve the primal problem (2).

   (a) Define matrix $\mathbf{A}$ and vectors $\mathbf{b}$ and $\mathbf{c}$ as in subsection 2.1

   ```
   import numpy as np

   c = np.array([3, 1])
   b = np.ones(4)
   A = np.array([[1, 1], [1, -1], [-1, 1], [-1, -1]])
   ```

   (b) Visualize the objective function and the constraints of Problem (2).

   ```
   from utility import plot_contours_exercice_section2
   plot_contours_exercice_section2(A, b, c)
   ```

   Intuitively and using the plot, what is the solution to Problem (2)?

   (c) Define the primal problem under `CVXPy` and compute the solution.

   ```
   import cvxpy as cvx
   print("----------- Solving the primal Problem ---------------")
   # define theta as the optimization problem variables
   d = 2 #dimension of theta
   theta = cvx.Variable(d)
   # define, using CVXPy format, the primal objective function
   obj = cvx.Minimize(0.5*cvx.quad_form(theta-c, np.eye(d)))
   # define the constraints
   constraints = [A@ theta - b <= 0]
   # set the primal as a CVX problem
   primal = cvx.Problem(obj, constraints)
   # Compute the solution
   primal.solve(verbose = False)

   # Print the results
   print("status of the solution = {}".format(primal.status))
   print("Primal optimal solution = {}".format(theta.value))
   obj_primal = 0.5*cvx.quad_form(theta-c, np.eye(d))
   print("primal objective function at optimality = {}".format(
       obj_primal.value))
   ```

   Compare the obtained solution to your intuitive guess.

2. Now let solve the dual problem and deduce $\boldsymbol{\theta}$ as established at question 2.1.4. Inspiring from the previous question, write the appropriate code to solve Problem (4).
   Hint: some useful matrix/vector operations under `numpy`

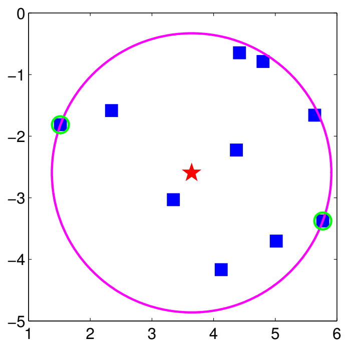
   - the matrix vector (matrix) multiplication $\mathbf{Ac}$ is: `A@c`
   - the transpose of A is either `A.T` or `np.transpose(A)`

```
print("----------- Solving the dual Problem ---------------")
# define matrix H
H = ...
# define vector q
q = ...

# define the dual variables
m = ... #dimension of dual variables mu
mu = ... #dual variables

# define, using CVXPy format, the dual objective function
obj = cvx.Minimize(...)
# define the constraints
constraints = ...
# set the dual problem and solve it
dual = cvx.Problem(...)
dual.solve(verbose = False) #compute the solution

# deduce the primal solution knowing the dual vector mu
theta_from_dual = c - np.asarray(A.T@(mu.value))
```

How both primal solutions compare?

2 Minimum enclosing ball

Assume a sub-urban area with n houses located at given coordinates $\mathbf{x}_i \in \mathbb{R}^2$, $i = 1, \cdots, n$. To build the firehouse a survey gets to the solution of settling the firehouse at position $\mathbf{z} \in \mathbb{R}^2$ so that its distance from the farthest house is minimal (see figure 2).



Mathematically, this translates into $\min_{\mathbf{z}} \max_{i=1,\cdots,n} \|\mathbf{z} - \mathbf{x}_i\|_2^2$
This problem can be equivalently expressed as

$$\begin{aligned} \min_{R \in \mathbb{R}, \mathbf{z} \in \mathbb{R}^2} \quad & R^2 \\ \text{s.c.} \quad & \|\mathbf{z} - \mathbf{x}_i\|_2^2 \leq R^2 \quad \forall\, i = 1, \cdots, n \end{aligned}$$

2.1 Again the mathematical derivation of the solution

1. What are the unknown variables of the problem? How many constraints does it involve?
2. Write the Lagrange function $\mathcal{L}$
3. Derive the stationary optimal conditions of the problem. Deduce the expression of $\mathbf{z}$ as a function of the Lagrange multipliers.
4. Using the optimality conditions show that the dual function is

$$\mathcal{L} = -\sum_{i=1}^{n}\sum_{j=1}^{n} \mu_i \mu_j \mathbf{x}_i^\top \mathbf{x}_j + \sum_{i=1}^{n} \mu_i \mathbf{x}_i^\top \mathbf{x}_i$$

   where the μ_i are the associated Lagrange multipliers to the primal.

5. Formulate the dual problem
6. From the dual solution $\boldsymbol{\mu}$, how to get the position $\mathbf{z}$ of the firehouse and the distance R to the farthest house?

2.2 Solution computation

Let the matrix

$$\mathbf{H} = \begin{pmatrix} \mathbf{x}_1^\top \mathbf{x}_1 & \mathbf{x}_1^\top \mathbf{x}_2 & \cdots & \mathbf{x}_1^\top \mathbf{x}_n \\ \vdots & \vdots & \cdots & \vdots \\ \mathbf{x}_n^\top \mathbf{x}_1 & \mathbf{x}_n^\top x_2 & \cdots & \mathbf{x}_n^\top \mathbf{x}_n \end{pmatrix} \in \mathbb{R}^{n\times n} \text{ and the vector } \mathbf{q} = \begin{pmatrix} \mathbf{x}_1^\top \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_n^\top \mathbf{x}_n \end{pmatrix}$$

The dual function can be written in the matrix form (easier to implement under CVXPy) :

$$\mathcal{L} = -\boldsymbol{\mu}^\top \mathbf{H} \boldsymbol{\mu} + \boldsymbol{\mu}^\top \mathbf{q}, \qquad \text{with} \quad \boldsymbol{\mu} = \begin{pmatrix} \mu_1 \\ \vdots \\ \mu_n \end{pmatrix} \in \mathbb{R}^n$$

The coordinates of the houses are provided in the matrix $\mathbf{X} = \begin{pmatrix} \mathbf{x}_1^\top \\ \vdots \\ \mathbf{x}_n^\top \end{pmatrix} \in \mathbb{R}^{n\times 2}$.

The data are provided in the file `maisons.csv` and can be retrieved using.

```python
import numpy as np
X = np.genfromtxt("maisons.csv", delimiter=",")
print(X)
```

Given the matrix $\mathbf{X}$, we can note that $\mathbf{H} = \mathbf{X}\mathbf{X}^\top$.

1. From these elements, solve the dual problem with CVXPy..

```python
import numpy as np
import matplotlib.pyplot as plt
import cvxpy as cvx

# matrix and vectors of the dual problem
H = X.dot(X.T)
q = np.diag(H)
n = H.shape[0]
e = np.ones(n)
zer = np.zeros(n)

# mu is defined as the optimization variable
mu = cvx.Variable(n)
# Fill in the blank to define the objective function
obj = cvx.Minimize(...)
# Fill in the blank to define the constraints
constr = [...]

# Solve the constrained optimization problem
prob = cvx.Problem(obj, constr)
prob.solve(verbose = False)

# The dual solution mu is squeezed to get a vector (ndarray)
mu = np.squeeze(np.asarray(mu.value))

#Print the dual solution
print('Dual solution mu = ', mu)
#Print the dual objective value
print("Objective function value = ', -1.0*prob.value)
```

2. From the dual solution $\boldsymbol{\mu}$, compute the primal solutions $\mathbf{z}$ and R.

```python
# Computation of z : uncomment the lines below to use t
z = np.zeros(2)
for i in range(n):
  z += mu[i]*X[i,:]

# A matrix way to compute z: uncomment the line below to use this
   computation
#z = np.multiply(X, np.outer(mu, np.ones(X.shape[1]))).sum(axis=0)

print("The center z = ", z)

# compute the ray R
threshold = 1e-3
pos = np.where(mu >= seuil) #index des coeff mu non nuls
Distancequad = z.dot(z) - 2*(X.dot(z)) + q #% distance d(z, xi)^2
R2 = np.mean(Distancequad[pos])
```

3. Plot the points $\mathbf{x}_i$, the location $\mathbf{z}$. and the enclosing ball.

```python
# plot the samples x_i and z
fig = plt.figure()
plt.plot(X[:,0], X[:,1], "s", color="b", markerfacecolor="b", markersize
     = 10)
```

```
plt.plot(z[0], z[1], "rp", markersize=15)

# plot of minimum enclosing ball
t = np.arange(0, 2*np.pi+0.02, 0.01)
plt.figure(fig.number)
plt.plot(Z[0] + np.sqrt(R2)*np.cos(t), Z[1] + np.sqrt(R2)*np.sin(t), "m"
    )
# plot of the farthest house (those located on the circle centered on z
    and of ray R)
plt.plot(X[pos,0], X[pos,1], "og", alpha=0.5, markersize = 15, linewidth
    =2)
```

4. Now assurme we are given two new coordinates as follows

```
outliers = [
 [3.0000, -3.0000],
 [2.5000, 2.5000]
 ]

Xoutliers = np.array(outliers)
```

Are these samples inside the minimal enclsoing ball? You may illustrate on a figure the enclosing ball and the new samples.

5. To allow these new samples to be more closer to the sought center $\mathbf{z}$, the optimisation problem is augmented by including the constraints $\|\mathbf{z}-\mathbf{x}_j\|_2^2 \leq R^2 \quad \forall j = 1, \cdots, m$ (here $m = 2$) related to the new samples $\mathbf{x}_j$ in `Xoutliers`. Show that the new optimization problem admist a dual function of the form

$$\mathcal{L} = -\tilde{\boldsymbol{\mu}}^\top \tilde{\mathbf{H}} \tilde{\boldsymbol{\mu}} + \tilde{\boldsymbol{\mu}}^\top \tilde{\mathbf{q}},$$

with

$$\tilde{\boldsymbol{\mu}} = \begin{pmatrix} \mu_1 \\ \vdots \\ \mu_n + m \end{pmatrix} \in \mathbb{R}^{n+m}, \quad \tilde{\mathbf{X}} = \begin{pmatrix} \mathbf{X} \\ \mathbf{X}_{\text{outliers}} \end{pmatrix} \in \mathbb{R}^{(n+m)\times 2} \quad \text{and} \quad \tilde{\mathbf{H}} = \tilde{\mathbf{X}}\tilde{\mathbf{X}}^\top$$

6. Using CVXPY, solve the new dual problem and infer the new solution location $\tilde{z}$. Plot in a graphics the new enclosing ball, $\tilde{z}$ along with all samples. Compare to solution obtained at question 3. What do you remark? Which approach can you propose to alleviate the issue related to far located houses (outliers)?