Proceedings of the 2nd International Conference on Computing Innovation and Applied Physics
DOI: 10.54254/2753-8818/19/20230500

Solving the smallest ball problem based on convex programming theories

Mingyuan Ji

University of California, Santa Barbara, CA, 93117

mingyuanji@ucsb.edu

Abstract. The mathematical method known to be Quadratic Programming is a branch of Convex Programming or Convex Optimization, which is then a peculiar case of Mathematical Optimization given a series of restrictions including a convex function to minimize. The Smallest Ball Problem is a problem where we seek to find the smallest enclosing ball of given points on a plane. Convex Optimization provides a solution to the Smallest Ball Problem. There are several ways to characterize a convex programming problem and its solutions, the most important of which is called the KKT conditions. By relating the Smallest Ball Problem to solving a Convex Programming Problem and using the Python packages, the radius, as well as the central point of the Smallest Ball, will be found. In addition, the underlying algorithm for solving Convex Programming Problems is studied. It can be concluded that Convex Programming, or more specifically, Quadratic Programming, gives a feasible solution to the Smallest Ball Problem.

Keywords: Convex Optimization, Quadratic Programming, Convex Programming, Mathematical Optimization, Smallest Ball Problem.

1. Introduction

Convex Programming is an area of mathematics with a broad range of applications in numerous areas of study, including estimation and signal processing, automatic control systems, communications and networks, circuit design, data analysis and modeling, and financial statistics. The mathematical aspects of Convex Optimization have been studied for about a century by mathematicians, while some newer aspects of applications are recently discovered and stimulated new interest in the topic.

The Smallest Ball Problem, or The Smallest Enclosing Ball Problem, was originally raised by mathematician James Joseph Sylvester in 1857. While it has been studied for over 100 years, new aspects and solutions to the problem are still emerging. Boris Mordukhovich et al discusses the primary aspects of the Smallest Ball Problem. The researchers start by distinguishing between the Smallest Enclosing Ball Problem and the Smallest Intersecting Ball Problem, especially focusing on proving the existence and uniqueness of their solutions by exploring the properties and features of the maximal time function, under the assumptions that it's in finite-dimensional Euclidean space [1]. In another article by Kaspar Fischer, Bernd Gärtner, and Martin Kutz, the researchers discuss a combinatorial algorithm that computes the smallest of the enclosing balls of a set of given points in Euclidean space, studied in high-dimensional situations. The algorithm resembles the simplex method in Linear Programming, and it normally requires only just a few iterations when it comes to lower dimensions and it has the capability to handle cases of very high dimensions [2].

The goal of this paper is to combine the two, namely the Convex Programming Problem and the Smallest Ball Problem, by examining the restraints, theories, and lemmas related to constructing a Convex Optimization Problem, and then the Smallest Ball Problem will be studied by solving a specific Convex Programming Problem.

2. Description of the Problem

To start with, Convex Functions are characterized in the following way.

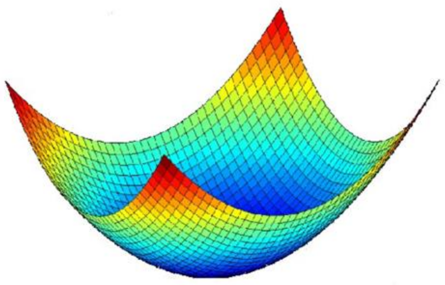


Figure 1. Convex geometric shape [1].

For a Convex Programming Problem in the 3-D space, it is essential to have both a Convex Function and a Convex Set to characterize the convexity. Figure. 1 above displays a Convex Function featuring convexity. There are three ways to characterize such a function.

$$f\big((1-t)x+ty\big) \leq (1-t)f(x)+tf(y) \text{ for } \forall x,y \in R^n, 0\leq t \leq 1 \tag{1}$$

$$\forall x,z \in R^n, f(x) \geq f(z)+\nabla f(z)\cdot(x-z) \tag{2}$$

$$\forall x \in R^n, D^2 f(x) \geq 0 \tag{3}$$

For Eq. (1), one of the three inequalities above, it states that for all the arbitrary x and y in the domain of the Convex Function, every value that is on the linear connection of both points is always greater or equal to the values on the convex region between the points. Eq. (2) states that starting from a point z in the convex region, all the points in the direction of the tangent line have values smaller or equal to the corresponding points in the convex region. Finally, in Eq. (3), D represents the Hessian Matrix, obtained from taking the second derivative of the original function. It states that all the eigenvalues of the Hessian Matrix are greater or equal to 0. By satisfying one of the three conditions, a Convex Function is then constructed. We can also derive from Eq. (3) that a quadratic form of the function

$$f(x) = x^T D x \tag{4}$$

is convex if and only if $D \geq 0$ and that is all the eigenvalues of D are greater or equal to 0. Another way to put it is that D is a positive semi-definite matrix. We can see from the introduction above that convex programming closely resembles a linear programming problem. This will also be demonstrated by further exploration of how to solve a convex programming problem using computer programming as well as algorithms.

3. Method

This section will be dedicated to characterizing the solutions to the Convex Programming Problems and then it will be adapted to the Smallest Ball Problem. After that, the internal algorithm of calculation will be explained.

3.1. Theorems

There are several theorems related to characterizing the solutions to Convex Optimization Problems. Given the listed conditions that

$$C \subset R^n \text{ is a convex set of constraints}$$
$$f: R^n \rightarrow R \text{ is a convex function}$$

The following statements is true,

$$x^* \text{ minimizes } f(x) \text{ over } C$$

If and only if,

$$\nabla f(x^*) \cdot (x - x^*) \geq 0, \forall x \in C \tag{5}$$

Another important condition for characterizing the feasible solutions to a Convex Programming Problem is the famous KKT conditions, or the Karush–Kuhn–Tucker optimality conditions. Previously, the KKT conditions play a crucial role in the optimization theory. However, one important feature is that KKT conditions can be adapted to solve modified optimization problems. In a paper by Giorgio Giorgi, Bienvenido Jiménez & Vicente Novo, the researchers extend the "approximate KKT conditions" from originally a scalar optimization problem that bears equality and inequality constraints to a multiobjective optimization problem. The researchers prove that this condition suffices to say that a point to is a locally weak efficient solution free from any constraint qualification and is also sufficient to meet the assumptions of convexity [3]. In another work by Gabriel Haeser and María Laura Schuverdt, the researchers proved that a slightly modified version of the AKKT (Approximate Karush–Kuhn–Tucker) condition suffices a convex problem, either for optimization or variational inequalities [4]. KKT conditions are also able to be generalized to serve characterization of efficient solutions to optimization problems with constrained interval, according to the research paper [5]. In this paper, similarly, KKT conditions will also be used to verify solutions to the Convex Programming Problem and then adapted to solve the Smallest Ball Problem.

Given $x \geq 0$, KKT conditions state that a feasible solution x^* is optimal if and only if,

$$\exists \tilde{y} \in R^m \text{ such that } \nabla f(x^*) + A^T \tilde{y} \ \geqslant 0 \tag{6}$$

Which equality stands in between when $x_j^* > 0$

And inequality when $x_j^* = 0$

KKT conditions are then applied to characterize the solution to the Smallest Ball Problem as the solution of a certain quadratic program. Now, relating the Smallest Ball Problem to Convex Programming Problem, the following theorem stands. This is a deduction from the KKT conditions given the specific Smallest Ball Problem restraints.

Let $\{P_1, P_2, \dots P_n\}$ be points in R^d while $n \geq d$

Let Q be the $d \times n$ matrix with coordinates of the points as columns, like so

$$\begin{bmatrix} \uparrow & \uparrow & \uparrow & \\ P_1 & P_2 & P_3 & \cdots \\ \downarrow & \downarrow & \downarrow & \end{bmatrix}$$

Now for the standard setup of a Convex Programming Problem, the Convex Function to minimize would be

$$x^T Q^T Q x - \sum_{j=1}^{n} x_j P_j^T P_j \tag{7}$$

Subject to the constraints $\sum_{j=1}^{n} x_j = 1$ given $x \geq 0$

The theorem states that for this certain Convex Programming Problem, the following statements are true: The Convex Programming Problem above has an optimal solution x^* and that $\exists P^*$ such that $P^* = Qx^*$ is true for every optimal solution x^*. In relation to the Corresponding Smallest Ball Problem, it can be concluded that P^* is the center of the ball and that square of the radius of the ball is equal to $-f(x^*)$.

Geometrically, the following lemma is crucial for solving the Smallest Ball Problem.

For an array of points given by $S = \{S_1, S_2, S_3, \dots S_k\} \subseteq R^d$, k is the number of points and d is 2 which is the second dimension, the lemma characterizes the center of the smallest enclosing ball. If S^* is the center of the enclosing ball, then the following two statements are deemed as equivalent,

B is the unique smallest enclosing ball of S ↔ There is no hyperplane that separates S from S^*

Also, there are other properties to the smallest enclosing ball the center of the ball is convexly formed by all the points that are on the smallest enclosing ball.

Now that the theorems related to the uniqueness and existence of a solution are discussed, to actually solve the Smallest Ball Problem, various algorithms developed by mathematicians will be introduced. In a paper written by Kasper Fischer and Bernd Gärtner, the researchers discuss algorithms for solving the Smallest Ball Problem without utilizing Mathematical Optimization. They aimed to focus on small number cases where $n \le d + 1$ to deduct the compulsory primitive operations and validate the proof that they can be effectively realized by rational arithmetic. Their algorithm is inspired by the observation that this problem bears a combinatorial nature of Discrete Mathematics and the randomized linear-time algorithm developed by Welzl, which might not work for high numbers but worked for small instances [6]. In another paper by Shaohua Pan and Xingsi Li, the researchers developed a simple algorithm for high dimensional Smallest Ball Problems. Basically, the researchers reformulated the problem as an unconstraint Convex Optimization Problem that involves the maximum function $\max\{0, t\}$, and it can efficiently handle a problem with n up to 10,000 [7]. A similar method is also utilized by Samuel Zürcher were finding the smallest enclosing ball is defined as looking for a center c and a radius r so that to minimize the maximum distance r from a point of the point set to the center c [8].

In this paper, the following algorithms will be introduced so as to solve constraint Convex Programming Problems, which in turn, solve Smallest Ball Problems.

3.2. Active Set Method

$$X_0 \to X_1 \to X_2 \to \dots\dots \to X_k \to X_{k+1}$$
$$W_0 \to W_1 \to W_2 \to \dots \to W_k \to W_{k+1}$$

Figure 2. Active Set Method [2].

In the realm of mathematical optimization, the active-set method is a commonly used algorithm for identification of the so-called active constraints in a set of given inequality constraints. The active constraints are then written in the form of equality constraints, transforming an inequality-constrained problem and creating a more straightforward equality-constrained problem, subordinate to the inequality problem. Here, this paper defines X_0 as an arbitrary feasible point and W_0 an initial working set of constraints. Different from the Simplex Algorithm where the constraints are the same all along, in the active set method, each point will correspond to a different set of constraints shown in Figure. 2, where X and W come in pairs. X^* is defined as the local minimizer to its corresponding set of active constraints and it satisfies the following conditions.

X^* is a KKT point

${d_i}^T P = 0$ for each i such that ${d_i}^T X^* = f_i$

The procedure then generates a sequence of points $\{X_k\}$ and associated working sets W_k such that $X_{k+1} = X_k + a_k P_k$

$$\begin{pmatrix} H & {A_k}^T \\ A_k & 0 \end{pmatrix} \begin{pmatrix} P_k \\ -q_k \end{pmatrix} = \begin{pmatrix} g_k - {A_k}^T y_k \\ 0 \end{pmatrix} \tag{9}$$

Note that Eq. (9) is also a derivation of the KKT conditions regarding solutions [9].

Not until a subspace minimizer is found can a constraint be removed. Therefor, the algorithm basically initializes by putting constraints into the working set, and once the first subspace minimizer or local minimizer is found, each group will start with a constraint removal and end up with a step to a minimizer. Figure. 2 illustrates this procedure from step 1 to step k, the algorithm identifies the correct constraints and then ends up with one extra step, which is k+1, to the final minimizer.

3.3. Steepest Descent Method

The inspiration for this method comes from the obvious notion that if one takes a step towards the direction of the negative gradient, a continuous function should initially decrease. Therefore, the only problem is to determine how to choose the length of the step [10].

$$f(x+v) \approx f(x) + \nabla f(x)^T \tag{10}$$

This algorithm starts with the Taylor approximation of the function value that is sought to be minimized. Now, the question of how to choose the value of v so that the directional derivative becomes as small as possible is addressed. Provided that $\nabla f(x)^T v < 0$, it is imperative to take v as large as possible. Define $\Delta X_{sd} = \|\nabla f(x)\|_* \, \Delta X_{nsd}$ in which ΔX_{nsd} is an infinitely small step of the unit norm that generates the greatest decrease.

Then, repeat the following procedures until the optimal feasible solution is achieved.

Start with a point X,

1. Compute ΔX_{sd} ;
2. Conduct Line Search in the direction of the steepest descent;
3. Compute $X = X + t\Delta X_{sd}$

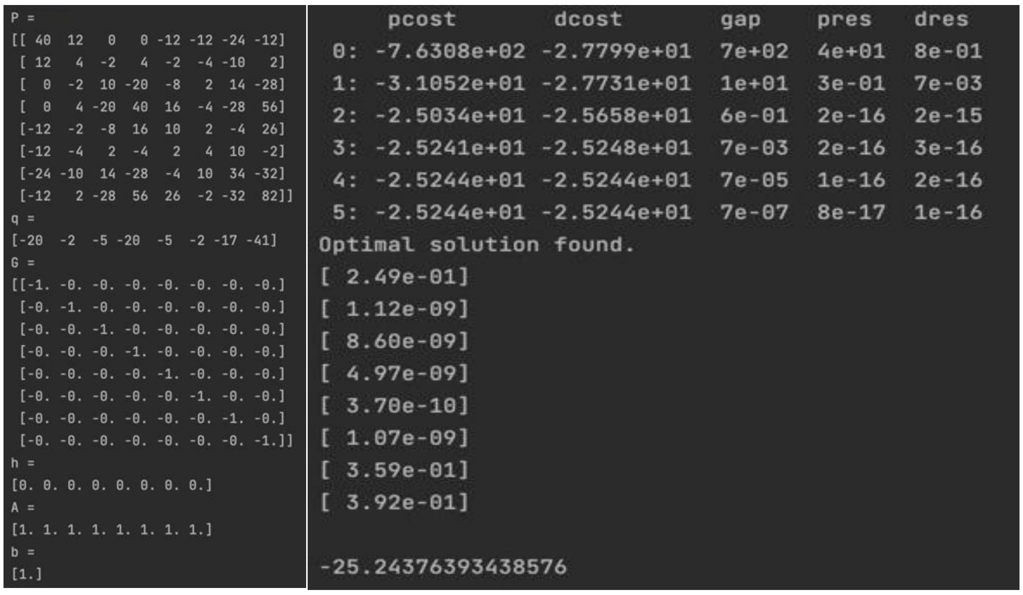
4. Computational Experiment

The two algorithms above both give a result to the Convex Optimization Problem. However, to solve a Convex Programming Problem most efficiently, the CVXOPT package in Python programming is extremely useful. With a little modification in its setup, the Smallest Ball Problem can be solved. The following is a demonstration of how to use programming methods to an actual example of the Smallest Ball Problem.

To start with, eight points on a 2-D plane are randomly picked as the points we seek to enclose. Using a random number generator, the points are as follows, and the goal is to look for the center as well as the radius of the smallest enclosing ball.

$$[-4,2]$$
$$[-1,1]$$
$$[-1,-2]$$
$$[2,4]$$
$$[2,1]$$
$$[1,-1]$$
$$[1,-4]$$
$$[4,5]$$

The CVXOPT package is imported and then the eight different points are entered as the columns of the Q matrix. P is the quadratic portion of the convex function and q is the linear modification. G and h are representations of the inequality constraints while A and b identify the equality constraints. All of the letters mentioned above symbolize matrices and any Convex Programming Problem bears such six factors. Referring to the standard setup of the Convex Programming Problem, Q and P would be the matrices in Eq. (7) which is the Convex Function and A and b are saying that all the variables add up to 1, according to Eq. (8). See Appendix for the complete code.



```
P =
[[ 40  12   0   0 -12 -12 -24 -12]
 [ 12   4  -2   4  -2  -4 -10   2]
 [  0  -2  10 -20  -8   2  14 -28]
 [  0   4 -20  40  16  -4 -28  56]
 [-12  -2  -8  16  10   2  -4  26]
 [-12  -4   2  -4   2   4  10  -2]
 [-24 -10  14 -28  -4  10  34 -32]
 [-12   2 -28  56  26  -2 -32  82]]
q =
[-20  -2  -5 -20  -5  -2 -17 -41]
G =
[[-1. -0. -0. -0. -0. -0. -0. -0.]
 [-0. -1. -0. -0. -0. -0. -0. -0.]
 [-0. -0. -1. -0. -0. -0. -0. -0.]
 [-0. -0. -0. -1. -0. -0. -0. -0.]
 [-0. -0. -0. -0. -1. -0. -0. -0.]
 [-0. -0. -0. -0. -0. -1. -0. -0.]
 [-0. -0. -0. -0. -0. -0. -1. -0.]
 [-0. -0. -0. -0. -0. -0. -0. -1.]]
h =
[0. 0. 0. 0. 0. 0. 0. 0.]
A =
[1. 1. 1. 1. 1. 1. 1. 1.]
b =
[1.]
```

```
     pcost       dcost       gap    pres   dres
 0: -7.6308e+02 -2.7799e+01  7e+02  4e+01  8e-01
 1: -3.1052e+01 -2.7731e+01  1e+01  3e-01  7e-03
 2: -2.5034e+01 -2.5658e+01  6e-01  2e-16  2e-15
 3: -2.5241e+01 -2.5248e+01  7e-03  2e-16  3e-16
 4: -2.5244e+01 -2.5244e+01  7e-05  1e-16  2e-16
 5: -2.5244e+01 -2.5244e+01  7e-07  8e-17  1e-16
Optimal solution found.
[ 2.49e-01]
[ 1.12e-09]
[ 8.60e-09]
[ 4.97e-09]
[ 3.70e-10]
[ 1.07e-09]
[ 3.59e-01]
[ 3.92e-01]

-25.24376393438576
```

Figure 3. Program result.

As illustrated above in Figure. 3, on the left, the program generates the matrices corresponding to the set-up of the problem. On the right, the result is displayed with the middle eight entries being the optimal solution X and at the bottom, the negative value would be the minimized convex function value. Some further calculations are required. Here, the square root of the negative of the minimized function is taken to be the radius of the ball. Also, in the solution matrix, all the significantly slow entries will be taken as 0, and then the matrix is multiplied by the original Q matrix. Therefore, a 2x1 matrix is obtained and that would be the coordinates of the center of the smallest ball. Thus, the program successfully finds the Smallest Enclosing Ball of the eight given points in the plane.

5. Discussion

The validity of our solution can be verified by looking for a plane that separates the points that are on the ball and the central point of the enclosing ball. This attempt will result in a failure which means there is no smaller ball than the current one to find. Also, the center of the ball is within the convex formed by the points on the ball. Therefore, it can be concluded that the findings are accurate.

The two algorithms for solving the Convex Programming problems are largely different, in a way that the Active Set Method is purely computational while the Steepest Descent Method approaches the problem in a geometric sense. Similarly, they both embody the general mindset of optimization algorithms and that is, executing a series of procedures under given conditions until the optimal solution is found. For the Smallest Ball Problem, it is usually true that the number count of balls is bigger than the number of dimensions. In this case, there are eight points in 2-D. For problems that involve even greater dimensions and number of balls, whether the Convex Programming method would still apply remains unanswered.

6. Conclusion

Given a reasonable setup, the Convex Programming Problems can be just as easy to solve as a regular Linear Programming Problem. For solving a Convex Programming Problem, it’s important to note the prerequisites and characterization of the solutions before conducting the algorithm. Also, the problem requires slight modification on the KKT conditions to be applicable to finding the smallest ball.

Examining the original thesis of this paper, Convex programming can be used as a perfect tool to solve the Smallest Ball Problem and it implies that Convex Programming is vastly applicable in numerous geometric problems as well as other sorts of realms. It can be reasoned that among many possible solutions to the Smallest Ball Problem, Convex Programming holds a special place for its adaptability, feasibility, and convenience. This paper is not only to show that Convex Programming can be used to solve classical geometric problems but also to demonstrate the potential of Convex Programming. The inner workings and algorithms of Convex Programming have been thoroughly studied by mathematicians, but its newer applications are yet to be discovered.

References

[1] Mordukhovich, B., Nguyen, M.N., Villalobos, C.: The smallest enclosing ball problem and the smallest intersecting ball problem: existence and uniqueness of solutions (2012)
[2] Fischer, K., Gartner, B., Kutz, M.: Fast Smallest-Enclosing-Ball Computation in High Dimensions (2003)
[3] Giorgi, G., Jiménez, B., Novo, V.: Approximate Karush–Kuhn–Tucker Condition in Multiobjective Optimization (2016)
[4] Haeser, G., Schuverdt, M.L.: On Approximate KKT Condition and its Extension to Continuous Variational Inequalities (2011)
[5] Ghosh, D., Singh, A., Shukla, K.K., Manchanda, K.: Extended Karush-Kuhn-Tucker condition for constrained interval optimization problems and its application in support vector machines (2019)
[6] Fischer, K., Gartner, B.: The Smallest Enclosing Ball of Balls: Combinatorial Structure and Algorithms (2003)
[7] Pan, S., Li, X.: An efficient algorithm for the smallest enclosing ball problem in high dimensions (2006)
[8] Samuel Zürcher. Smallest Enclosing Ball for a Point Set with Strictly Convex Level Sets. Institute of Theoretical Computer Science, 2007.
[9] Elizabeth Wong. Active-Set Methods for Quadratic Programming. University of California, San Diego, 2011.
[10] Juan C. Meza. Steepest Descent. Lawrence Berkeley National Laboratory, 2017.

Appendix

```
  import numpy as np
import cvxopt
p1 = [-4,2]
p2 = [-1,1]
p3 = [-1,-2]
p4 = [2,4]
p5 = [2,1]
p6 = [1,-1]
p7 = [1,-4]
p8 = [4,5]
Q = np.array([p1,p2,p3,p4,p5,p6,p7,p8])
Q = np.ndarray.transpose(Q)
P = 2 * np.matmul(np.ndarray.transpose(Q),Q)
q = -np.array([np.dot(p1,p1),
np.dot(p2,p2),np.dot(p3,p3),np.dot(p4,p4),np.dot(p5,p5),
               np.dot(p6,p6),np.dot(p7,p7),np.dot(p8,p8)])
G = -np.identity(8)
h = np.zeros(8)
A = np.ones(8)
b = np.ones(1)
print('P =');print(P)
print('q =');print(q)
print('G =');print(G)
print('h =');print(h)
print('A =');print(A)
print('b =');print(b)
P = cvxopt.matrix(P, tc = 'd')
q = cvxopt.matrix(q, tc = 'd')
G = cvxopt.matrix(G, tc = 'd')
h = cvxopt.matrix(h, tc = 'd')
A = cvxopt.matrix(A, (1,8), tc = 'd')
b = cvxopt.matrix(b, tc = 'd')
sol = cvxopt.solvers.qp(P,q,G,h,A,b)
print(sol['x'])
print(sol['primal objective'])
```