Calculation of the BI Center of a Closed Convex Polytope

V. S. Patwardhan[1]

30 September 2019

Abstract

A closed convex polytope in n dimensions is defined by the usual linear inequality constraints specified as $Ax \leq b$. The BI center of such a polytope is defined as that interior point which bisects all the line segments drawn through that point, parallel to each of the coordinate axes. The existence and uniqueness of such a center for the general case has been shown earlier. The BI center is also a maximiser of an appropriately defined multivariate function, as shown earlier. Three algorithms are proposed here for calculating the BI center iteratively. The first one is an implementation of the well-known coordinate search algorithm (CS). The second one is a generalization of this algorithm which uses simultaneous coordinate search for all coordinate directions (SCS). The third algorithm involves solution of simultaneous linear equations using matrix inversion (MI). The advantages and disadvantages of these algorithms are discussed. Any of these algorithms can be used by itself, starting from any interior point, to calculate the BI center. However, the CS and SCS algorithms approach the BI center rapidly at first, but the progress becomes slower as they get closer to the BI center. On the other hand, the MI algorithm can give the exact BI center in one step, if it is started from a point reasonably close to the BI center. These characteristics can be combined beneficially by approaching the BI center initially with the CS or SCS algorithm, and then taking the final steps with the MI algorithm, to get the BI center exactly. Preliminary computational results are presented for some simple polytopes for illustration, using these three algorithms.

1. Introduction

Defining and calculating the center of a closed convex polytope has attracted interest for a long time. Many different definitions have been used for the center, which include the center of mass of the entire polytope, the centroid of all vertices, the center of the largest inscribed sphere (or an ellipsoid), the center of the smallest sphere (or ellipsoid) which includes the polytope, the analytical center, the weighted projection center, and orthogonal projections on to polytope faces. All these definitions lead to different points as centers, whose calculation involves different degrees of computational effort. One of the motivations for these centers has been the fact that most interior point algorithms for solving linear programming problems involve a centering step at periodic intervals. Therefore, any method which leads to efficient calculation of a uniquely defined center may become useful in interior point linear programming algorithms.

In this paper, we further investigate the BI center proposed earlier by Patwardhan [2019]. The BI center of a closed convex polytope is defined as that interior point which bisects all the line segments drawn through that point, parallel to each of the coordinate axes. We present here, three different algorithms which can be used for calculating the BI center, starting from any interior point.

[1] Independent researcher. Formerly, Scientist G, National Chemical Laboratory, Pune 411008, India.
Email : vspatw@gmail.com , URL : https://www.vspatwardhan.com

Computational results obtained with all three algorithms are presented for a couple of simple polytopes and compared with each other for illustration.

2. Calculation of intersections with polytope boundary

We consider here, a closed convex polytope in n dimensions defined by the usual linear inequality constraints specified as $Ax \leq b$. It is assumed here that all the m constraints are inequality constraints, i.e. there are no equality constraints. (Any equality constraint can be removed by eliminating any one variable using the equality constraint.) The i^{th} constraint can be written as

$$\sum_{j=1}^{n} A_{ij} x_j \leq b_i \qquad for\ i = 1\ to\ m \tag{1}$$

Let P be a strictly feasible interior point so that we can write

$$\sum_{j=1}^{n} A_{ij} P_j < b_i \qquad for\ i = 1\ to\ m \tag{2}$$

The slacks at point P can be denoted as S_i, which are given by

$$S_i = b_i - \sum_{j=1}^{n} A_{ij} P_j \tag{3}$$

Since P is an interior point, $S_i > 0$ for all i. A line drawn through P, parallel to the k^{th} axis, intersects the i^{th} constraint after travelling a distance d_{ik} given by

$$d_{ik} = S_i / A_{ik} \qquad for\ k = 1\ to\ n \tag{4}$$

If $A_{ik} = 0$, then d_{ik} becomes infinite, i.e. there is no intersection with constraint i. If $A_{ik} > 0$, then $d_{ik} > 0$, i.e. the line drawn in the $+x_k$ direction intersects the i^{th} constraint. On the other hand, if $A_{ik} < 0$, then $d_{ik} < 0$, i.e. the line drawn in the $-x_k$ direction intersects the i^{th} constraint. Let d_{+k} and d_{-k} be the distances to the first intersection in the $+x_k$ and $-x_k$ directions respectively, and let Q_{+k} and Q_{-k} be the corresponding points of intersection. These are given by

$$d_{+k} = \min(d_{ik})\ over\ all\ i, for\ which\ A_{ik} > 0 \tag{5}$$

and

$$d_{-k} = \max(d_{ik})\ over\ all\ i, for\ which\ A_{ik} < 0 \tag{6}$$

It is obvious that $d_{+k} > 0$, and $d_{-k} < 0$. Let Q_k be the midpoint of the line segment $Q_{-k}\,Q_{+k}$. Then d_k, the distance between P and Q_k, is given by

$$d_k = (d_{+k} + d_{-k})/2 \tag{7}$$

3. Characterisation of the BI center

The closeness of the interior point P to the BI center of the polytope can be characterised by the function

$$F = \prod_{k=1}^{n} F_k \tag{8}$$

where

$$F_k = (-4)\frac{(\, d_{+k}\, d_{-k})}{(d_{+k} - d_{-k})^2} \tag{9}$$

If P coincides with Q_k, then $d_{+k} = -d_{-k}$, and $F_k = 1$. This is the reason why the factor of (-4) is used in Equation (9). On the other hand, if P lies at the polytope boundary, then either $d_{+k} = 0$ or $d_{-k} = 0$, and therefore $F_k = 0$. Thus, equations (8) and (9) show that F is zero at the polytope boundary, is equal to 1 at the BI center, and has some value between 0 and 1 everywhere else (Patwardhan [1]).

The problem of finding the BI center is essentially one of maximising F (to a value of 1) using unconstrained optimisation. It may be noted that although F is continuous over the feasible region, i.e. the polytope, it's first derivative is only piecewise continuous, in view of the edges present on the polytope boundary which is defined by linear intersecting constraints. Therefore, optimisation techniques based on using the first derivative are not suitable for our purpose, although such techniques are very well developed, tested and documented (for example, one may refer to Djordjevic [2]). We need to use a method which uses only function values without any derivatives. There is a class of algorithms called as coordinate search algorithms (CS) which can be used here since they only use function values. (Since optimisation problems are often stated in a form where a function is to be minimised, these algorithms are often termed as coordinate descent algorithms. However, we use the term CS to refer to minimising or maximising a given function. Our problem here involves maximising F). A good recent introduction to coordinate search is available (Shi H. M. et al. [3]).

Let us now consider three algorithms which can be used for calculating the BI center.

4. Coordinate search algorithm (CS)

The idea here is to go from a given interior point P to a new point Q in n stages, changing one coordinate at a time using Equation (7), and repeat this step iteratively till we approach the BI center sufficiently closely. Variations of this algorithm are described by Astolfi [4].

One step of this algorithm consists of the following: Given an interior point P, we get the point Q_1, which is the midpoint of the line segment through P in the x_1 direction, which also maximises F_1 along the line segment. Then we get the point Q_2 which is the midpoint of the line segment through Q_1 in the x_2 direction, which maximises F_2 along the line segment. We continue this till we get point Q_n, which is the midpoint of the line segment through Q_{n-1} in the x_n direction. This completes the

step. It may be noted that at each step we go to a point which bisects some line segment, thereby ensuring that we do not approach the polytope boundary closely. This process is continued till we approach the BI center closely.

The details of the algorithm are given below. An arbitrary factor α is used for flexibility. This is equivalent to travelling a distance equal to αd_k towards the center of the line segment at each stage, instead of the full distance d_k.

CS Algorithm:

Given : iter = 0, and $P_{iter} = P_0$ (an interior point)
Do while $F(P_{iter}) <$ a critical value (such as 0.99)
$Q(j) = P_{iter}(j)$ for j = 1 to n
For k = 1 to n
$Q(k) = Q(k) + \alpha\, d_k$ using eq.(7)
Update S_i at Q for all i where A_{ik} is nonzero
Next k
iter = iter + 1
$P_{iter}(j) = Q(j)$ for j = 1 to n
End do
P_{iter} is the BI center of the polytope

Let us illustrate the progress of this algorithm with the following example.

Example 1: A polytope with 5 constraints and 2 nonnegative variables

$$\begin{aligned} 1.5\,x - y &\le 8 \\ 0.2\,x + y &\le 8.4 \\ -5\,x - y &\le -10 \\ -4\,x + y &\le 1 \\ 0.5\,x - y &\le 2 \\ x,\ y &\ge 0 \end{aligned} \qquad (10)$$

Let us use three starting points, i.e. (2, 7.5), (3, 0.25) and (9, 6), and three values of α, i.e. 1.0, 0.5 and 0.2. Figure 1 shows the progress of this algorithm for the three starting points, and for the three values of α. All the lines converge to the BI center. Table 1 shows the results of iterative calculations for all these cases. It is seen that the convergence is fast for α = 1, while lower values of α require more iterations for a close approach to the BI center. It is also seen from Figure 1 that for lower α, the approach to the BI center is smoother. This algorithm involves the updating of the S_i values at each step and involves extra calculations. However, its advantage lies in the fact that at any stage the calculated points remain far away from the polytope boundaries.

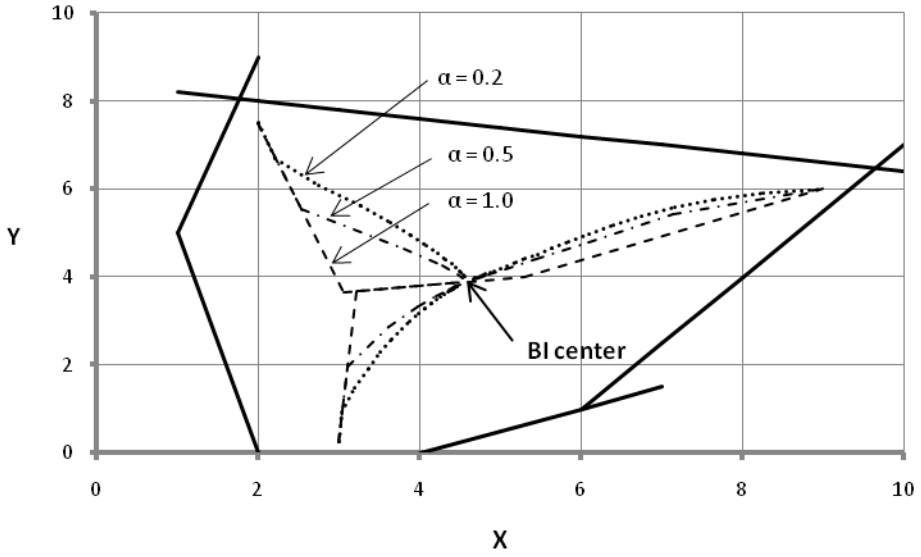

Figure 1: Convergence to the BI center for Example 1 (CS algorithm)

No.	P1, init	P2, init	α	No. of iter. for F=0.99	No. of iter. for F=0.999
1	2	7.5	1	3	4
2	2	7.5	0.5	5	6
3	2	7.5	0.2	12	16
4	3	0.25	1	3	4
5	3	0.25	0.5	6	8
6	3	0.25	0.2	14	21
7	9	6	1	4	4
8	9	6	0.5	7	9
9	9	6	0.2	16	23

Table 1: Convergence to the BI center for Example 1 (CS algorithm)

5. Simultaneous coordinate search algorithm (SCS)

The idea here is to go from a given interior point P to a new point Q using Equation (7), changing all the coordinates simultaneously, and repeat this step iteratively till we approach the BI center sufficiently closely.

One step of this algorithm consists of going from the current point P to a new point Q , using

$$Q_j = P_j + \alpha\, d_j \qquad for\ j = 1\ to\ n \tag{11}$$

The multiplying factor α has been used for flexibility, as before. The details of the SCS algorithm are given below:

SCS algorithm:

Given : iter = 0, and $P_{iter} = P_0$ (an interior point)
Do while $F(P_{iter})$ < a critical value (such as 0.99)
$Q_j = P_{iter,j} + \alpha\, d_j$ *for j = 1 to n*
iter = iter+1
$P_{iter}(j) = Q(j)$ *for j = 1 to n*
End Do
P_{iter} *is the BI center of the polytope*

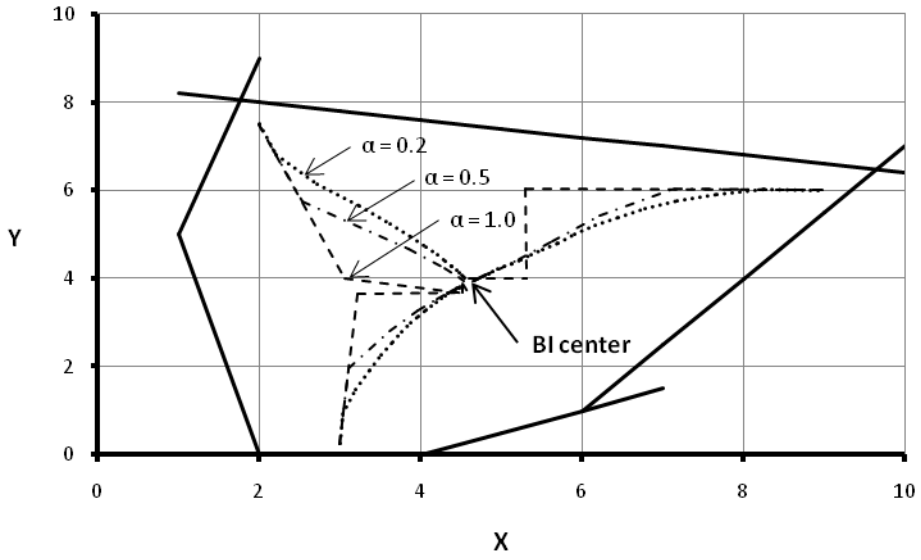

Figure 2: Convergence to the BI center for Example 1 (SCS algorithm)

No.	P1, init	P2, init	α	No. of iter. for F=0.99	No. of iter. for F=0.999
1	2	7.5	1	3	3
2	2	7.5	0.5	5	6
3	2	7.5	0.2	12	17
4	3	0.25	1	3	3
5	3	0.25	0.5	6	8
6	3	0.25	0.2	13	20
7	9	6	1	4	4
8	9	6	0.5	7	9
9	9	6	0.2	16	23

Table 2. Convergence to the BI center for Ex. 1 (SCS algorithm)

The performance of this algorithm for example 1 is shown in Figure 2 and Table 2. The performance of this algorithm is almost the same as that of the CS algorithm. This algorithm does not involve any updating of the S_i values at each step and therefore, involves less calculations. However, at times, the calculated points may be close to the polytope boundaries.

6. Matrix Inversion algorithm (MI)

Let us first characterise the BI center by assuming that point P is the BI center, as shown in Figure 3. Let U_k be the index of the first constraint hit by a line drawn through P, in the $+x_k$ direction, and let L_k be the index of the first constraint hit by a line drawn through P, in the $-x_k$ direction. Let d_{+k} and d_{-k} be the corresponding distances. Since P is the BI center, it bisects the line, and d_{+k} and d_{-k} are equal in magnitude. Using equations (3) and (4), this condition can be written as

$$\frac{\left(b_{U_k} - \sum_{j=1}^{n} A_{U_k j} P_j\right)}{A_{U_k k}} + \frac{\left(b_{L_k} - \sum_{j=1}^{n} A_{L_k j} P_j\right)}{A_{L_k k}} = 0 \qquad (12)$$

This equation is valid for all k = 1 to n. Algebraic manipulation gives

$$\sum_{j=1}^{n} \left(A_{L_k k} A_{U_k j} + A_{U_k k} A_{L_k j}\right) P_j = \left(A_{L_k k} b_{U_k} + A_{U_k k} b_{L_k}\right) \quad for\ k = 1\ to\ n \qquad (13)$$

Equation (13) is a set of (n x n) linear equations which can be solved for P_j, j = 1 to n, using matrix inversion or any iterative method. It should be noted here that the coefficients appearing in Equation (13) correspond to intersections of lines drawn through the BI center, which is not known beforehand! One way around this difficulty is to start with an initial interior point, P_0, and draw lines through this point to determine the set of constraints intersected, $\psi_0 = [(U_k, L_k), k = 1 \text{ to } n)]$. Now generate the coefficients appearing in Equation (13), calculate P_1 by matrix inversion or any iterative method, and get the corresponding set ψ_1. If ψ_0 and ψ_1 are identical, then P_1 is the BI center. If ψ_0 and ψ_1 are different, then calculate P_2 and ψ_2. We can continue this until ψ_{iter-1} and ψ_{iter} are identical, to get P_{iter} as the BI center. This algorithm is described below.

MI algorithm:

Given : iter = 0, $P_{iter} = P_0$ (an interior point)
Calculate the set of intersections $\psi_0 = [(U_k, L_k), k = 1 \text{ to } n)]$ using Eqns (4) to (6)
Do
Iter = iter + 1
Calculate coefficients in Equation (12) using ψ_{iter-1}
Calculate point Q by solving Equation (12)
If Q is a feasible point then
$P_{iter} = Q$
else
Calculate Q_1 and Q_2 as the two intersections of line QP_{iter-1} and the polytope boundary

P_{iter} = the midpoint of Q_1 and Q_2
End If
Calculate the set of intersections ψ_{iter} from P_{iter}
If ψ_{iter} and ψ_{iter-1} are identical then
exit do
End If
End do
P_{iter} is the BI center of the polytope

The performance of this algorithm for example 1 is shown in Figure 3. For this example, the BI center is reached in just a few steps, starting from any of the five different points considered.

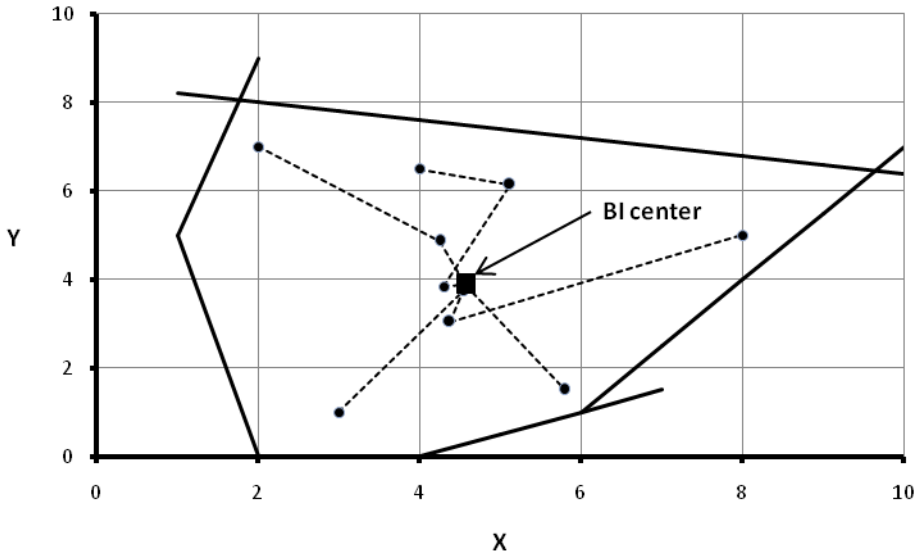

Figure 3: Convergence to the BI center for Example 1 (MI algorithm)

If the MI algorithm is started from a point away from the BI center, i.e. close to the polytope boundary, then the next calculated point may be an infeasible point. This possibility has been incorporated in the MI algorithm description. Even then, this algorithm may not converge to the BI center in some rare cases.

7. A combined approach to the calculation of the BI center

It has been stated earlier that the CS and SCS algorithms converge to the BI center iteratively. However, as they approach the BI center, further approach becomes slower and slower as the steps become shorter. On the other hand, the MI algorithm may give infeasible iterates when far away from the BI center but can get to the exact BI center quickly when it is not too far away from it. These two approaches can be combined to get to the BI center quickly.

Consider the polytope defined by Example 1, shown in Figure 4, which also shows the BI center. If two lines are drawn through the BI center, they intersect constraints (1, 2, 3, 5). A grey rectangle surrounding the BI center has also been shown in the same figure. If we take any point within the grey rectangle and draw two lines in the x and y direction, then they intersect the same set of constraints (1, 2, 3, 5). This implies that if we start from any point within the grey rectangle, and use

the MI algorithm, then we reach the BI center in one step! This gives us a way of combining algorithms in a beneficial manner. We can start from any interior point, apply the CS or the SCS algorithm till we get an F value more than some critical value such as 0.99, and then apply the MI algorithm to get the BI center exactly.

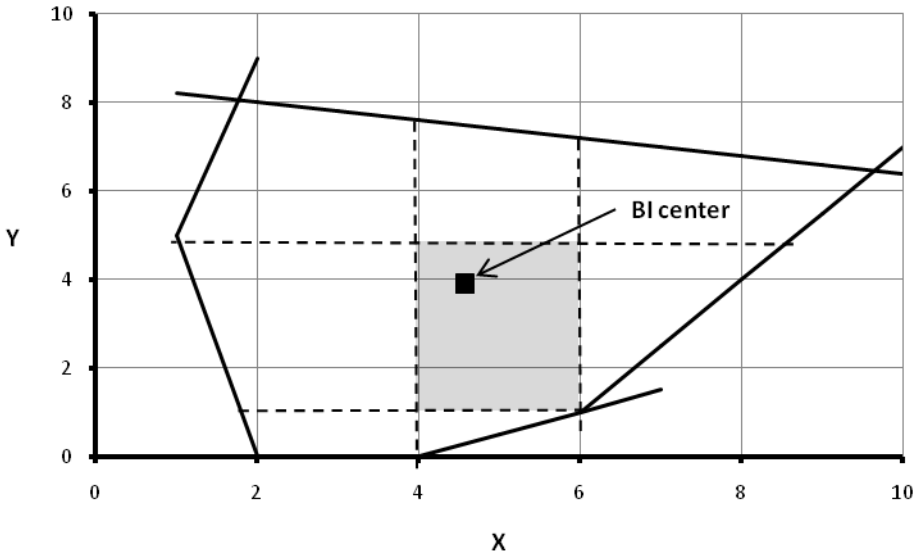

Figure 4. Interior region which gives the BI center in one step

8. Some results with another example

Let us consider a larger problem in four dimensions as given below.

Example 2: A polytope with 5 constraints and 4 nonnegative variables

$$\begin{aligned} x_1 + x_2 - x_3 + x_4 &\le 8 \\ x_1 + 0.5\,x_2 - x_3 &\le 3 \\ 0.5\,x_1 - 2\,x_2 + x_3 &\le 2 \\ -x_1 + 0.5\,x_2 - 0.5\,x_4 &\le 3 \\ x_1 + 3\,x_2 + 1.5\,x_3 + 2\,x_4 &\le 25 \\ x_1,\ x_2,\ x_3,\ x_4 &\ge 0 \end{aligned} \qquad (14)$$

This example has five constraints defining the polytope, and in addition, all four variables are nonnegative, which is equivalent to four more constraints.

CS algorithm: If we apply the CS algorithm to this polytope, starting from the arbitrary point (1, 2, 2.5, 1.3) we approach the BI center iteratively. Figure 5 shows the approach of the x_1 coordinate to the BI center for three values of α. Similar figures can be shown for other coordinates, but are not presented here, for the sake of brevity. Table 3 shows the approach to the BI center in terms of the number of iterations required for getting F of 0.99 and 0.999, for three values of α. It is seen that for

α = 1, the approach to the BI center is fast, but x_1 oscillates to some extent. For lower values of α, the approach to the BI center is smoother, but takes a greater number of iterations.

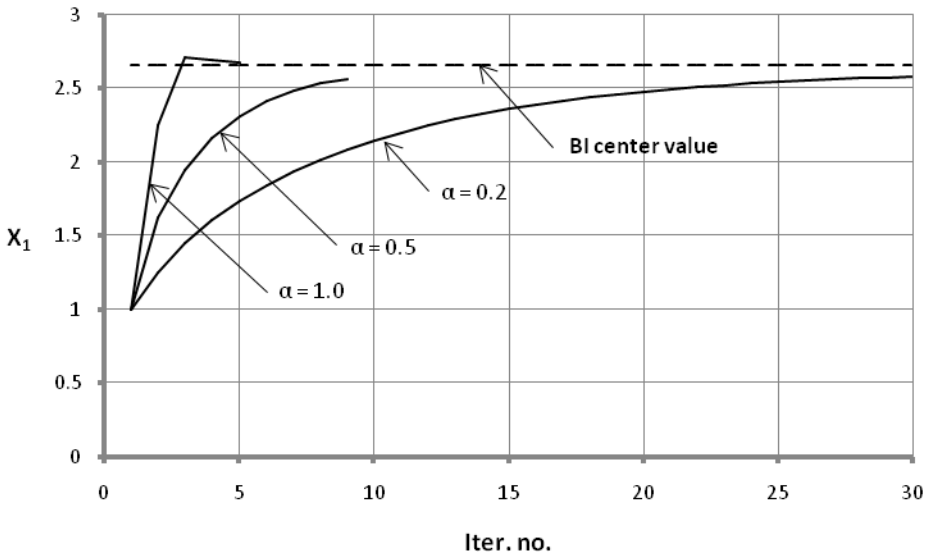

Figure 5. Convergence of the x_1 values (CS algorithm, three values of α)

No.	A	No. of iter. for F=0.99	No. of iter. for F=0.999
1	1	3	3
2	0.5	5	9
3	0.2	13	30

Table 3: Iterations required for convergence for Example 2 (CS algorithm)

SCS algorithm: If we apply the SCS algorithm to this polytope, starting from the same arbitrary point used in the CS algorithm, i.e. (1, 2, 2.5, 1.3), we approach the BI center iteratively. Figure 6 shows the approach of the x_1 coordinate to the BI center for three values of α. Table 4 shows the approach to the BI center in terms of the number of iterations required for getting F of 0.99 and 0.999, for three values of α. It is seen that for α = 1, the approach to the BI center is fast, but x_1 oscillates to some extent. For lower values of α, the approach to the BI center is smoother, but takes more iterations. It is also seen that the SCS algorithm is slower than the CS algorithm in approaching the BI center.

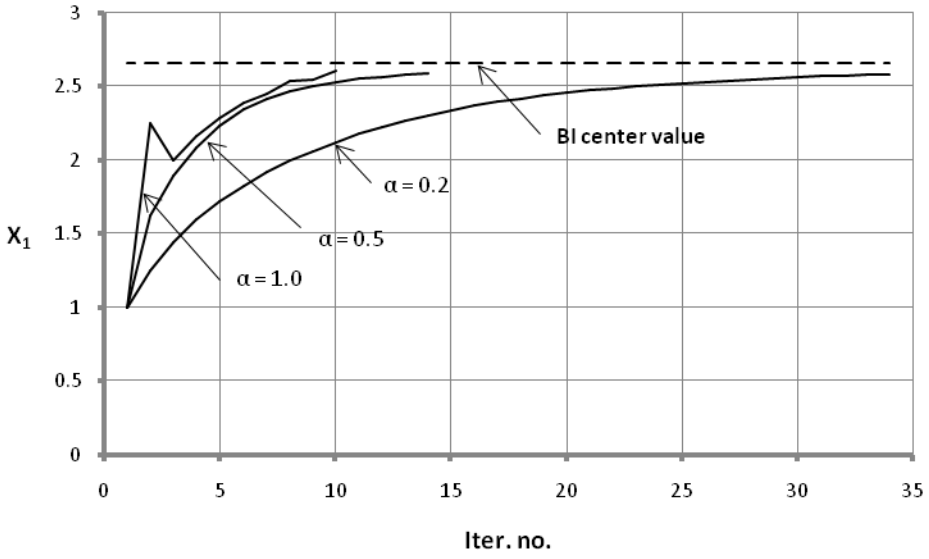

Figure 6. Convergence of the x_1 values (SCS algorithm, three values of α)

No.	A	No. of iter. for F=0.99	No. of iter. for F=0.999
1	1	7	10
2	0.5	6	14
3	0.2	15	34

Table 4: Iterations required for convergence for Ex. 2 (SCS algorithm)

MI algorithm: If we apply the MI algorithm to this polytope, starting from the same arbitrary point used in the CS and SCS algorithms, i.e. (1, 2, 2.5, 1.3), we approach the BI center in three steps as shown in Table 5.

No.	X1	X2	X3	X4	F value
1	1	2	2.5	1.3	0.329
2	2.394	2.773	3.175	2.381041	0.980
3	2.667	2.889	3.778	2	1

Table 5: Iterations required for convergence for Example 2 (MI algorithm)

9. Some results with the combined approach

Let us illustrate the combined approach using both the example problems. Since the CS and SCS algorithms are comparable, we use only the SCS algorithm for the initial approach to the BI center.

Example 1: Let us first consider Example 1 with the SCS algorithm, and use α = 0.2 . Let us use the interior starting point (9, 6). Table 6 shows the progress of the algorithm with iterations, i.e. the point coordinates and the F value. It shows that we reach F value of 0.95, 0.99 and 0.999 after 12, 16

and 23 iterations respectively. It also shows that as we approach the BI center, further progress becomes slower. Let us now consider three cases as follows:

Iter. No.	X	Y	F value
1	9.000	6.000	0.156
2	8.258	6.010	0.398
3	7.665	5.921	0.518
4	7.183	5.773	0.610
5	6.784	5.592	0.686
6	6.448	5.396	0.747
7	6.162	5.195	0.797
8	5.914	4.997	0.842
9	5.697	4.815	0.888
10	5.516	4.663	0.921
11	5.364	4.536	0.944
12	5.236	4.429	0.960
13	5.129	4.340	0.972
14	5.039	4.266	0.980
15	4.963	4.204	0.986
16	4.900	4.152	0.990
17	4.847	4.109	0.993
18	4.803	4.072	0.995
19	4.765	4.042	0.996
20	4.734	4.016	0.997
21	4.708	3.995	0.998
22	4.686	3.977	0.998
23	4.668	3.962	0.999

Table 6. Approach to the BI center (Example 1, SCS algorithm, $\alpha = 0.2$)

Case 1: After 16 iterations, we reach the point (4.9, 4.152) and an F value of 0.99. If we now switch to the MI algorithm (which used matrix inversion) at this point, we get the BI center (4.573, 3.886) in just one step, with F = 1.

Case 2: Let us switch to the MI algorithm earlier than in Case 1. After 12 iterations, we reach the point (5.236, 4.429) and an F value of just 0.96. If we now switch to the MI algorithm at this point, we still get the BI center in just one step, with F = 1.

Case 3: Let us switch to the MI algorithm even earlier than in Case 2. After 9 iterations, we reach the point (5.697, 4.815) and an F value of just 0.888. If we now switch to the MI algorithm at this point, we still get the BI center in just one step, with F = 1 !

Example 2: Let us now consider Example 2 with the SCS algorithm, and use $\alpha = 0.5$. Let us use the interior starting point (1, 2, 2.5, 1.3) as before. Table 7 shows the progress of the algorithm with iterations, i.e. the point coordinates and the F value. It shows that we reach F value of 0.95, 0.99 and 0.999 after 3, 6 and 14 iterations respectively. It also shows that as we approach the BI center, further progress becomes slower. Let us now consider two cases as follows:

Iter. No.	X_1	X_2	X_3	X_4	F value
1	1.000	2.000	2.500	1.300	0.328
2	1.625	2.595	2.625	2.431	0.876
3	1.894	2.692	2.907	2.671	0.952
4	2.087	2.694	3.123	2.669	0.973
5	2.237	2.692	3.251	2.602	0.984
6	2.345	2.699	3.338	2.527	0.990
7	2.419	2.712	3.399	2.457	0.993
8	2.470	2.729	3.447	2.396	0.995
9	2.505	2.747	3.488	2.344	0.996
10	2.531	2.764	3.524	2.299	0.997
11	2.551	2.779	3.556	2.260	0.998
12	2.567	2.793	3.584	2.227	0.998
13	2.580	2.805	3.608	2.198	0.998
14	2.591	2.815	3.630	2.173	0.999

Table 7. Approach to the BI center (Example 2, SCS algorithm, $\alpha = 0.5$)

Case 1: After 6 iterations, we reach the point (2.345, 2.699, 3.338, 2.527) and an F value of 0.99. If we now switch to the MI algorithm (which used matrix inversion) at this point, we get the BI center (2.667, 2.889, 3.778, 2) in just one step, with F = 1.

Case 2: Let us switch to the MI algorithm earlier than in Case 1. After 3 iterations, we reach the point (1.894, 2.692, 2.907, 2.671) and an F value of just 0.952. If we now switch to the MI algorithm at this point, we still get the BI center in just one step, with F = 1 !

These examples and the various cases considered, clearly show that once we reach a reasonably high F value (0.99, 0.95 and 0.88 in the cases considered) we can get the BI center quickly using the MI algorithm which uses matrix inversion.

10. Conclusions

Three algorithms are proposed for calculating the BI center of a closed convex polytope. The CS algorithm involves sequential centering of all coordinates, while the SCS algorithm involves simultaneous centering of all coordinates. These algorithms were tested with two examples: the first one with five constraints and two variables, and the second one with five constraints and four variables. It was found that the computational characteristics of both these algorithms are comparable for the two small polytopes considered. The CS algorithm is faster than the SCS algorithm in its approach to the BI center, but involves some extra computations in each iteration

(for updating the S_i values). It also has the advantage that iterates are never too close to the polytope boundary. For both these algorithms, it was found that a step size multiplier less than 1 gives a smoother approach to the BI center, though it increases the number of iterations. It was also found that both these algorithms become progressively slower as they approach the BI center.

The MI algorithm is based on the solution of simultaneous linear equations. It was shown to converge well to the BI center for both the examples. It was also shown to jump to the exact BI center in one step, if the starting point is close enough to the BI center. The region which corresponds to this single step solution to the BI center was shown graphically for one of the examples. It was found to generate infeasible iterates if it is started too close to the boundary. The algorithm described here includes a feature which deals with this possibility.

A combined approach was tried which combines desirable characteristics of these algorithms. This approach is based on the use of the CS or SCS algorithm to get close to the BI center (as indicated by the F value of 0.9 or higher), and then use the MI algorithm to take the final steps to the exact BI center. The approach was shown to apply successfully for the two examples considered, where just one step of the MI algorithm was enough for reaching the BI center. Further work is needed for testing these algorithms with larger and more complex polytopes.

11. References

[1] Patwardhan V. S., *Center Of A Closed Convex Polytope: A New Definition*, DOI: 10.13140/RG.2.2.35394.53446 (2019)
(Can also be downloaded from www.vspatwardhan.com)

[2] Djordjevic S., *Some Unconstrained Optimization Methods* DOI: 10.5772/intechopen.83679 (2019)

[3] Shi H. M., Tu S., Xu Y. and Yin W., *A Primer on Coordinate Descent Methods*, arXiv:1610.00040 [math.OC] , (2017)

[4] Astolfi A., *Optimisation: An Introduction,* http://www3.imperial.ac.uk/pls/portallive/docs/1/7288263.PDF , (2006)