Annals of the University of Craiova, Mathematics and Computer Science Series
Volume 37(1), 2010, Pages 35–42
ISSN: 1223-6934

# (CQ) algorithm implementation

Cristina Popirlan

Abstract. This paper presents an implementation of (CQ) algorithm of Nakajo and Takahashi. The algorithm implementation permits us to analyze the relation between the weight factor and the number of iterations and to visualize the sets C, Q and $C \bigcap Q$. From the implementation we can see the importance of the weight factor in the (CQ) algorithm (an algorithm hard to calculate, with difficult formulas and for which a lot of generalizations were given in order to analyze its convergence).

*2010 Mathematics Subject Classification.* Primary 65F10; Secondary 65Y10.
*Key words and phrases.* (CQ) algorithm, iterative methods, projection algorithm, Mann iteration, algorithm implementation, convex feasibility problem.

## 1. Introduction

Let $M_i \subset \mathcal{H},\ i = 1, ..., m$ be a family of closed convex subsets of a real Hilbert space $\mathcal{H}$ with nonempty intersection, $\bigcap M_i \neq \emptyset$. The convex feasibility problem ([7]) is:

$$Find\ a\ point\ of\ \bigcap M_i.$$

To solve the convex feasibility problem, in 2003, Nakajo and Takahashi [1] introduced the following algorithm: every Mann iteration is projected onto some conveniently chosen sets $C_n$ and $Q_n$:

$$\begin{cases} x_0 = x \in Q, \\ y_n = (1 - t_n)x_n + t_n T x_n, \\ C_n = \{z \in C : ||y_n - z|| \leq ||x_n - z||\}, \\ Q_n = \{z \in C : \langle x_0 - x_n, x_n - z\rangle \geq 0\}, \\ x_{n+1} = P_{C_n \bigcap Q_n} x_0, \end{cases}$$

where $P_M x$ is the projection of $x$ on the set M, T is the projection and $t_n \in [0, 1]$.

The name *(CQ) algorithm* comes from the sets C and Q that are calculated at every step of the algorithm. From the construction of the algorithm, the sets $C_n$ and $Q_n$ are closed and convex, so $C_n \bigcap Q_n$ is also closed and convex, and we can conclude that the sequence $\{x_n\} n \geq 0$ is well defined.

We also observe that $Fix(T) \subset C_n \bigcap Q_n$ and so the construction of the sets $C_n$ and $Q_n$ is natural; if $z \in Fix(T)$, then, if T is demicontractiv, then relation that defines $C_n$ is satisfied, and if $z \in Fix(T)$ and $x_n$ is the projection of $x_0$ on $Q_n$, then the relation that defines $Q_n$ is also satisfied. So it is natural to take $x_{n+1}$ as close to the intersection and so we take

$$x_{n+1} = P_{C_n \bigcap Q_n} x_0.$$

Received December 07, 2009. Revision received February 06, 2010.

The concrete application of the (CQ) algorithm implies the evaluation of the sets $C_n$ and $Q_n$, then explicit calculation of the projection of $x_0$ on the intersection $C_n \bigcap Q_n$, or to solve a conditioned optimization problem, at ever step of the iteration. The papers which contain data about the (CQ) algorithm implementation are limited.

All the topics about the (CQ) algorithm are about his convergence. The papers ([6], [10]) connected to this subject analyze the strong convergence ([9]) of the algorithm by studying the conditions that must be imposed to the operator so that the strong convergence ([8]) is obtained. Because the (CQ) algorithm has difficult formulas, in order to obtain strong convergence, some papers tried to improve the algorithm by modifying it ([3], [11], [12]).

The convergence of the algorithm has been extensively studied. The main result of [1] is:

**Theorem 1.1.** [1] *Let $C \subset \mathcal{H}$ be a closed convex set from a Hilbert space $\mathcal{H}$ and $T : C \to C$ a nonexpansiv operator with $Fix(T) \neq \emptyset$. Suppose that the weight sequence $\{t_n\}_{n \geq 0}$ satisfies the condition $t_n \in (0, 1]$.*

*Then the sequence $\{x_n\}_{n\geq 0}$ generated by the (CQ) algorithm converges strongly to $P_{Fix(T)}x_0$.*

**Remark 1.1.** *The (CQ) algorithm is also strong convergent if the operator is k-demicontractiv* [2] *and the following condition is satisfied: the operator I-T is demi-closed at zero.*

A partial simulation of the (CQ) algorithm was given in [4]. The program analyses the repartition of the points from a square which has the $2a$ dimensions, the square has the center in axes origin, where $a > 2$ is given by the user of the application. In the square it is built a $n^2$ points network, where $n$ is also given by the user. For every point from the network it is studied if the points belongs to one, two or none of the sets C and Q. This sets are different colored for each case in part:

(1) **blue** for the points in C that are not in Q;
(2) **green** for the points that are not in C but are in Q;
(3) **red** for the points in C and Q;
(4) **yellow** for the points that are not in C and not in Q.

In extension of that paper which implements a single step of the algorithm, in this paper a complete implementation of the algorithm is given. All the conclusion, from [5] and [4], regarding the distribution of a points network in the sets C, Q and $C_n \bigcap Q_n$, are extended in this paper which also analyze the influence of the weight factor onto the number of iterations calculated until the solution if found.

## 2. The (CQ) algorithm - implementation and numerical experiments

The main idea of the algorithm is to project each Mann iteration on an intersection of sets that are built at every step $C_n$ and $Q_n$. The application simulates the algorithm and represents the sets $C_n$, $Q_n$ and $C_n \bigcap Q_n$. For different input parameters, the weight factor t, the initial approximation (starting point) and the network that is analyzed, we study how the network is organized between the sets C, Q and $C \bigcap Q$, how the weight factor influences the number of iterations until the solution is found.

Let us consider a family of 4 sets: $M_i$, $i = \overline{1, \ldots, 4}$. Each set contains the points $(x, y)$ that verify the inequation $a_i x + b_i y + c_i \geq 0$ where $a_i, b_i, c_i \in \mathcal{R}$ are given. This way we obtain a 4 inequations system with the solution given by the intersection of those 4 sets. Thus, the solution of the system represents the solution for the following convex feasibility problem: *Find a point of* $M = \bigcap_{i=1}^{4} M_i$.

In the paper it is implemented the (CQ) algorithm for solving the following convex feasibility problem:

$$\begin{cases} M_i = \{(x_1, x_2) \in \mathcal{R} \times \mathcal{R}, a_i x_1 + b_i x_2 + c_i \geq 0\}, i = \overline{1, \ldots, 4} \\ find\ (x_1, x_2) \in M = \bigcap_{i=1}^{4} M_i \end{cases}$$

The sets $M_i$ are half-spaces determinate by $a_i x_1 + b_i x_2 + c_i = 0$, $a_i, b_i, c_i \in \mathcal{R}$, $i = \overline{1, \ldots, 4}$. So the sets are convex subsets of the space $\mathcal{R}^2$.

The application is made in MathCad and implements the algorithm (CQ) in the form proposed by Nakajo and Takahashi [1]. To simulate the (CQ) algorithm need projection operator which we will build with following four lines that will make projections and that, in our case, will lead to a square of side 2.

$$\begin{cases} d_1(x_1, x_2) = x_1 + 1 \geq 0 \\ d_2(x_1, x_2) = -x_1 + 1 \geq 0 \\ d_3(x_1, x_2) = x_2 + 1 \geq 0 \\ d_4(x_1, x_2) = -x_2 + 1 \geq 0 \end{cases}$$

The sets $M_i = \{(x_1, y_1) \in \mathcal{R} \times \mathcal{R}, d_i(x_1, x_2) \geq 0\}$ are convex and the intersection is represented by a square of side 2.

In the algorithm, the square is build with vectors a and b:

$$\text{a=}\begin{pmatrix} -1 \\ 1 \\ 0 \\ 0 \end{pmatrix} \text{ and b=}\begin{pmatrix} 0 \\ 0 \\ 1 \\ -1 \end{pmatrix}$$

The application is build on the following algorithm: having a starting point, at every step determinate the side square on which the projection is made on, then the sets C, Q and $C \bigcap Q$ are calculated by browsing the points from the network in order to have a visual representation of the sets and, in the end of the application, the line from the starting point to the final point (the point from the square of side 2, the projection of the starting point onto this square) is drawn.

The starting point is set in the algorithm, as a weight factor $t$, but can be modified easily so studying how these parameters influence the algorithm.

Let's consider $x = (x_1, x_2) \in \mathcal{R}^2$, check for each of the four inequalities if $d_i(x) < 0$, $i = \overline{1, 4}$ and take $k$ so that

$$d_k(x) = \max_{i=\overline{1,4}, d_i(x)<0} |d_i(x)|.$$

The projection operator T is calculated with the formula:

$$T\begin{pmatrix} x \\ y \end{pmatrix} = -\begin{pmatrix} a_k \\ b_k \end{pmatrix} t + \begin{pmatrix} x \\ y \end{pmatrix}$$

where $\begin{pmatrix} a_k \\ b_k \end{pmatrix}$ represents the projection line and $t$ is given by the following formula

$$t = \frac{a_k x + b_k y + 1}{a_k^2 + b_k^2}.$$

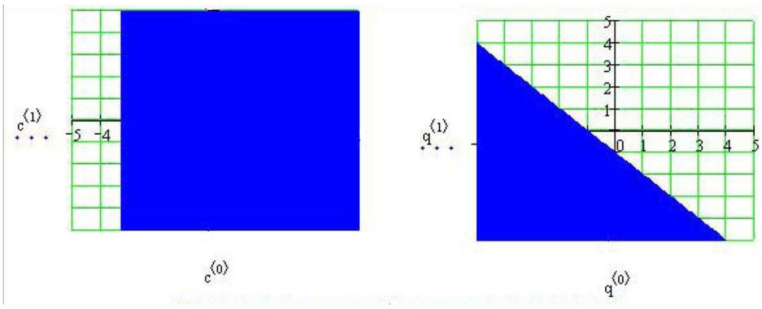

FIGURE 1. The construction of sets C and Q

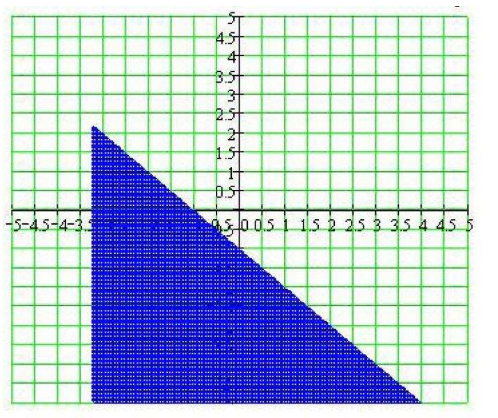

FIGURE 2. The construction of set $C \bigcap Q$

In order to obtain the set $C \bigcap Q$ the condition that must be satisfied is obtain by unifying the conditions from the sets C and Q: $nr_1 < nr_2$ and $ps > 0$, where $nr_1$, $nr_2$ and $ps$ are calculated in the algorithm.

The visual representations from figures 1 and 2 where obtain for the initial point $(4, -2)$. For a different starting point the representation of sets C and Q is different and so the intersection set $C \bigcap Q$, the set where the projection is made in order to obtain the next iteration, is also different.

In the application presented in [4] the obtained conclusion is that the weight factor influence the representation of sets C and Q, and consequently of set $C \bigcap Q$. The results obtained with this application reinforce previous results, and extend them from a partial simulation to the entire algorithm.

Next the application build the iterative sequence, from the initial point with the (CQ) algorithm the next iterations are calculated until the final point is the projection of the starting point onto the square of side 2. And so the algorithm is used to solve a convex feasibility (the inequalities that gives the square sides represents the sets and the square is the intersection of those sets). The algorithm permits us to visualize the projections until a point from the intersection is determined.

The function that permits us to construct the sequence of iterations is described in the following algorithm:

**Algorithm 1** Set C and set Q

i=0
ii=0
$p = T(x_1, x_2)$
$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = (1-t)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + t\begin{pmatrix} p_0 \\ p_1 \end{pmatrix}$
For $i = \overline{0, n}$
    $z_1 = -aa + \frac{2 \cdot aa}{n} \cdot i$
    For $j = \overline{0, n}$
        $z_2 = -aa + \frac{2 \cdot aa}{n} \cdot j$
        $nr_1 = (y_1 - z_1)^2 + (y_2 - z_2)^2$
        $nr_2 = (x_1 - z_1)^2 + (x_2 - z_2)^2$
        $ps = (x0_1 - x_1)(x_1 - z_1) + (x0_2 - x_2)(x_2 - z_2)$
        If $nr_1 < nr_2$ then
            $c1_i = z_1$
            $c2_i = z_2$
            i=i+1
        If $ps > 0$ then
            $q1_{ii} = z_1$
            $q2_{ii} = z_2$
            ii=ii+1
$c\_pct_0 = c1$
$c\_pct_1 = c2$
$q\_pct_0 = q1$
$q\_pct_1 = q2$

**Algorithm 2** (CQ) Algorithm - construction of the iterative sequence

$s_{0,0} = x_1$
$s_{0,1} = x_2$
i=0
While $i < nr_{max}$
    $v = cq(s_{i,0}, s_{i,1})$
    $i = i + 1$
    $s_{i,0} = min(v_{0,0}, v)_0$
    $s_{i,1} = min(v_{0,0}, v)_1$

In the sequence construction algorithm the *min* function determines the point with minimum norm (it practical helps to determinate the projection point on the set $C \bigcap Q$). In the figure 4 it can be seen the iterations sequence build with the algorithm.

As it could be seen from the figure 4, the value of weight factor significantly influences the number of iterations calculated until the solution of the convex feasibility problem is found. This conclusion holds for a large number of examples, so it can be generalized.

If the weight factor is small (and that mean that it is between 0 and 1.0) the iterations number is really large. This number become smaller when the weight factor is getting closer to 2.0.

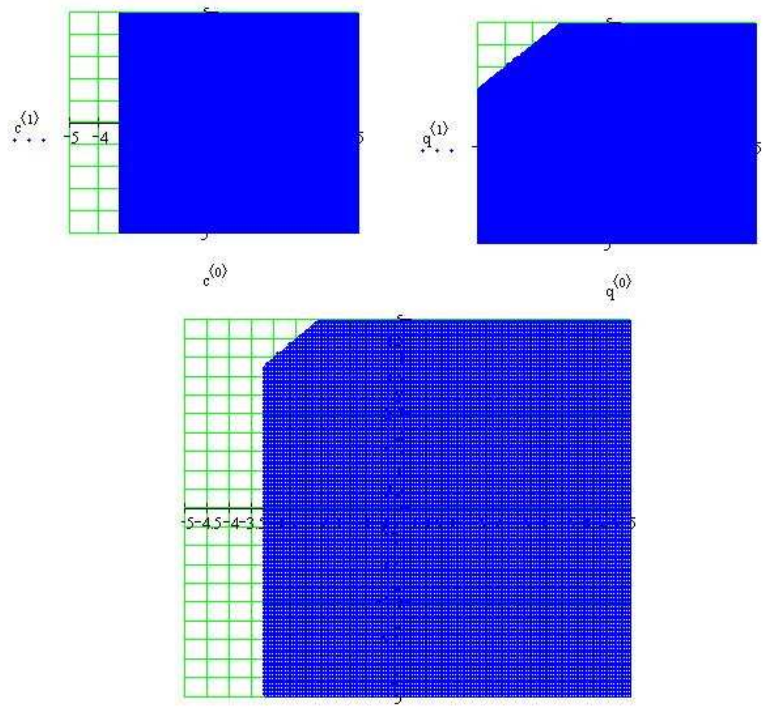


FIGURE 3. The representation of sets C, Q and $C \bigcap Q$

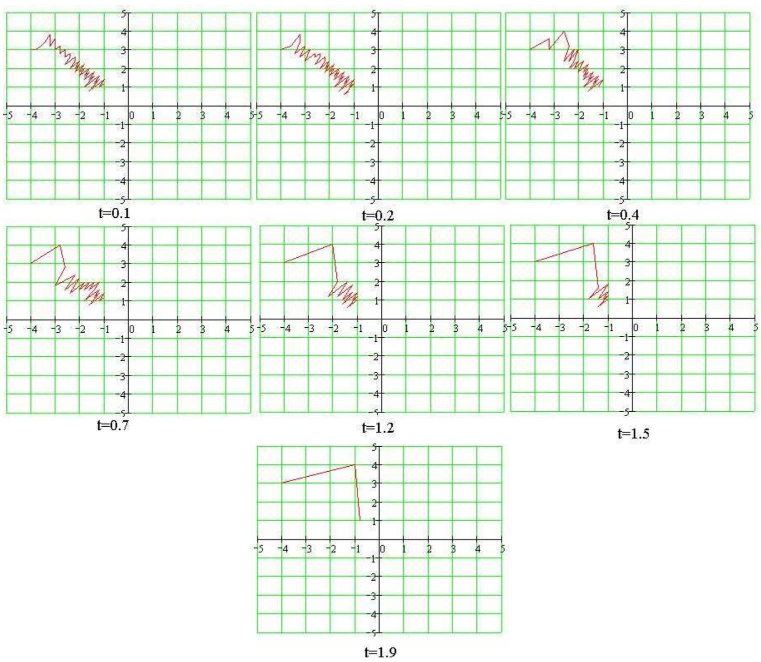


FIGURE 4. Construction of (CQ) sequence

TABLE 1. Relation between the weight factor and the iterations number

| weight factor | iterations number | iterations number |
|---|---|---|
| 0.1 | 257 | 372 |
| 0.2 | 200 | 368 |
| 0.3 | 169 | 331 |
| 0.4 | 129 | 304 |
| 0.5 | 93 | 304 |
| 0.6 | 80 | 304 |
| 0.7 | 57 | 275 |
| 0.8 | 35 | 227 |
| 0.9 | 18 | 201 |
| 1.0 | 17 | 183 |
| 1.1 | 17 | 178 |
| 1.2 | 17 | 157 |
| 1.3 | 13 | 140 |
| 1.4 | 7 | 122 |
| 1.5 | 5 | 68 |
| 1.6 | 3 | 49 |
| 1.7 | 2 | 28 |
| 1.8 | 2 | 3 |
| 1.9 | 2 | 3 |

From numerical experiments we can draw the conclusion that for (CQ) algorithm the optimal value for the weight factor is a value closer to 2 (this analyze was made for different examples and it was explained in tabel 1).

## 3. Conclusions and Future Work

The application represents a complete simulation of the (CQ) algorithm and allows us to analyze the way weight factor influences the number of iterations calculated until the solution of the convex feasibility problem is computed (until a point from the set $C \bigcap Q$ is found). It can easily be observed how the sets C, Q and $C \bigcap Q$ are constructed. The road from the starting point (the initial approximation) to the finish point (the projection of the initial point onto the square of side 2) can be visualize.

The (CQ) algorithm was implemented in this paper in order to study the connection between the weight factor and the number of iterations calculated until a solution of the convex feasibility problem is found.

The results are as expected. The weight factor influences the iterations number and this has nothing to do with the initial point. This means that for any initial point for small weight factor the iterations number has a large value and when the weight factor increases the number of iterations decreases.

All results from this paper are obtain by numerical experiments and by applying the (CQ) algorithm implementation to a lot of numerical data.

In the future work, theoretical results must be obtain in order to sustain the numerical results obtained in this paper.

## References

[1] K. Nakajo and W. Takahashi, Strong convergence theorems for nonexpansive mappings and non-expansive semigroups, *Journal of Mathematical Analysis and Applications* **279** (2003), 372-379.

[2] G. Marino and H.K. Xu, Weak and strong convergence theorems for strictly pseudocontraction in Hilbert spaces, *Journal of Mathematical Analysis and Applications* **329** (2007), 336-349.

[3] W. Takahashi, Y. Takeuchi and R. Kubota, Strong convergence theorems by hybrid methods for families of nonexpansive mappings in Hilbert spaces, *Journal of Mathematical Analysis and Applications* **341** (2008), 276-286.

[4] C. Popîrlan, Partial simulation of (cq) algorithm, *Annals of the University of Craiova - Mathematics and Computer Science Series* **36** (2009), no. 2, 63–68.

[5] C. Popîrlan, The relation between the weight factor and the number of steps in a projection algorithm, *Annals of the University of Craiova - Mathematics and Computer Science Series* **36** (2009), no. 1, 55–64.

[6] C. Martinez-Yanesa and H.K. Xu, Strong convergence of the CQ method for fixed point iteration processes, *Nonlinear Anal.* **64** (2006), 2400–2411.

[7] H. Bauschke and J.M. Borwein, On projection algorithms for solving convex feasibility problems, *SIAM REVIEW* **38** (1996), no. 3, 367-426.

[8] Y. I. Alber, Metric and generalized projection operators in Banach spaces: properties and applications, *Theory and Applications of Nonlinear Operators of Accretive and Monotone Type* (1996), 15–50.

[9] W. Takahashi, *Nonlinear Functional Analysis*, Yokohama-Publishers, 2000.

[10] Y. Su and X. Qina, Monotone CQ iteration processes for nonexpansive semigroups and maximal monotone operators, *Nonlinear Analysis: Theory, Methods & Applications* **68** (2008), no. 12, 3657–3664.

[11] Q. Yang, The relaxed CQ algorithm solving the split feasibility problem, *Inverse Problems* **20** (2004), 1261–1266.

[12] B. Qu and N. Xiu, A note on the CQ algorithm for the split feasibility problem, *Inverse Problems* **21** (2005), no. 5, 1655–1665.

(Cristina Popirlan) University of Craiova, Faculty of Mathematics and Computer Science, Department of Computer Science, 13 Alexandru Ioan Cuza Street, Craiova, 200585, Romania

*E-mail address*: cpopirlan@inf.ucv.ro