sciendo

Műszaki Tudományos Közlemények vol. 15. (2021) 11–14.
DOI: Hungarian: https://doi.org/10.33895/mtk-2021.15.03
Engliah: https://doi.org/10.33894/mtk-2021.15.03

EME MTSZ MTK

Predictor-Corrector Interior-Point Algorithm for the General Linear Complementarity Problem

Zsolt DARVAY,[1] Ágnes FÜSTÖS[2]

[1] *Babeș-Bolyai University, Faculty of Mathematics and Computer Science, Cluj-Napoca, Romania, darvay@cs.ubbcluj.ro; Transylvanian Museum Society, Department of Mathematics and Computer Science*

[2] *Babeș-Bolyai University, Faculty of Mathematics and Computer Science, Cluj-Napoca, Romania, fustosagi@yahoo.com*

Abstract

We study a predictor-corrector interior-point algorithm for solving general linear complementarity problems from the implementation point of view. We analyze the method proposed by Illés, Nagy and Terlaky [1] that extends the algorithm published by Potra and Liu [2] to general linear complementarity problems. A new method for determining the step size of the corrector direction is presented. Using the code implemented in the C++ programming language, we can solve large-scale problems based on sufficient matrices.

Keywords: *interior-point algorithm, general linear complementarity problem, predictor-corrector algorithm, numerical results, sufficient matrix, object-oriented programming.*

1. Introduction

Linear complementarity problems (*LCP*) can be encountered when solving various technical or economic problems. The problem is defined by a matrix M that appears in a linear equation, but in addition a complementarity condition must be met.

The *LCP* is an NP-complete problem, so it is very difficult to give an efficient general solution for it. There are algorithms that solve *LCP*s within polynomial time if the matrix M is positive semidefinite. As an extension of positive semidefinite matrices, the concept of sufficient matrices was introduced and it was proved that in the case of an input matrix M with this property the problem can be solved in polynomial time.

Algorithms introduced for solving the general linear complementarity problem (*GLCP*) can decide whether the matrix of the *GLCP* is $P_*(\kappa)$ or not. It should be mentioned, that the $P_*(\kappa)$ property is equivalent to sufficiency.

With the method introduced by Tibor Illés, Marianna Nagy and Tamás Terlaky [1, 3, 4] interior-point algorithms solve the GLCP in polynomial time using different built-in checks while running.

We have previously implemented a short-step interior-point algorithm to solve *GLCPs* [5]. Since predictor-corrector methods are generally more efficient, we further investigate the implementation of a variant of them.

2. The linear complementarity problem

In the case of *LCPs* we wish to find vectors $x, s \in \mathbb{R}^n$ such that:

$$\begin{cases} Mx + q = s, \\ xs = 0, \\ x \geq 0, s \geq 0, \end{cases} \tag{1}$$

where $q \in \mathbb{R}^n, M \in \mathbb{R}^{n\times n}$ and xs denotes the componentwise product of the two vectors.

3. The $P_*(\kappa)$ property

As defined by Kojima et al. [6] a matrix $M \in \mathbb{R}^{n\times n}$ has the $P_*(\kappa)$ property ($\kappa \geq 0$), if for any $x \in \mathbb{R}^n$ the following holds:

$$(1 + 4\kappa)\sum_{i\in\mathcal{J}_+(x)} x_i(Mx)_i + \sum_{i\in\mathcal{J}_-(x)} x_i(Mx)_i \geq 0, \tag{2}$$

where $\mathcal{J}_+(x) = \{1 \leq i \leq n : x_i(Mx)_i > 0\}$ and $\mathcal{J}_-(x) = \{1 \leq i \leq n : x_i(Mx)_i < 0\}$.

A matrix $M \in \mathbb{R}^{n\times n}$ is P_* if it is $P_*(\kappa)$ for some $\kappa \geq 0$. We mention that if $\kappa = 0$ then we obtain the class of positive semidefinite matrices.

4. Local κ values

In the predictor and corrector steps of each iteration, we calculate a local κ [1, 4], using the following function, and then select the maximum of all such values and determine a lower bound for the κ of the matrix M:

$$\kappa(\Delta x) = -\frac{1}{4}\frac{\Delta x^T M \Delta x}{\sum_{i\in J_+(\Delta x)} \Delta x_i (M\Delta x)_i}. \qquad (3)$$

5. The concept of the neighborhood

Let $\gamma \in (0, 1)$ and

$$\mathcal{F}^0 := \{(x, s) \in \mathbb{R}^n \times \mathbb{R}^n : -Mx + s = q,\ \ x, s > 0\}.$$

Potra and Liu [2] defined the neighborhood as follows:

$$D(\gamma) := \left\{(x, s) \in \mathcal{F}^0 : xs \geq \gamma \frac{x^T s}{n} e\right\}, \qquad (4)$$

where e denotes the *n-dimensional all-one vector*.

6. Newton's method

The algorithm starts from an initial point (x_0, s_0) and the following Newton system is solved in order to calculate the predictor and corrector directions:

$$\begin{cases} -M\Delta x + \Delta s = -r_q, \\ s\Delta x + x\Delta s = a, \end{cases} \qquad (5)$$

where $r_q = Mx + q - s$ was introduced to preserve the feasibility.

Note that in the predictor step $a = -xs$ and in the corrector one $a = \mu e - xs$. Moreover, the following notations are used in the algorithm:

$$x(\alpha) = x + \alpha\Delta x,\ s(\alpha) = s + \alpha\Delta s, \qquad (6)$$

where $\alpha > 0$ gives the size of the step.

7. The algorithm

Starting from the algorithm introduced by Illés, Nagy and Terlaky [1], the predictor-corrector algorithm solving the *GLCP* is given as follows:

Input parameters:

$\tilde{\kappa} > 0$ upper bound for κ;
$\varepsilon > 0$ accuracy parameter;
$\gamma \in (0, 1)$ neighborhood parameter;
$(x_0, s_0) \in D(\gamma)$ initial point;
$\sigma \in (0,1]$ barrier update parameter;
$\rho \in (0,1)$ step size reduction parameter;
$\beta, \beta_{max} > 0$ control parameters for the step size accuracy of the corrector step;

Output:

(x, s) – the solution of the *LCP* – or a message confirming that the $P_*(\kappa)$ property is not satisfied.

BEGIN

$x := x_0;\ s := s_0;\ \mu := \sigma\frac{(x_0)^T s_0}{n};\ \kappa := 0;$

$r_q = Mx + q - s;\ \ \alpha_p^*(\kappa) = \frac{2\sqrt{(1-\gamma)\gamma}}{(1+4\kappa)n+2};$

while $\frac{x^T s}{1 + x_0^T s_0} > \varepsilon$ **or** $\frac{\|r_q\|}{1+\|q\|} > \varepsilon$ **do begin**

Predictor step
$a = -xs;$
calculate $(\Delta x, \Delta s)$ *from (5)*
if M *is singular* **then**
  return M *is not* P_0;
end if
α = *predictorStepSize*(γ);
if $\alpha < \alpha_p^*(\kappa)$ **then**
  calculate $\kappa(\Delta x)$ *from (3) ;*
  if $\kappa(\Delta x)$ *is not defined* **then**
    return M *is not* P_* ;
  end if
  if $\kappa(\Delta x) > \tilde{\kappa}$ **then**
    return M *is not* $P_*(\tilde{\kappa})$;
  end if
  $\kappa = \kappa(\Delta x);$
  $\alpha_c^*(\kappa) = \frac{2\gamma}{(1+4\kappa)n+1};$
end if
$x = x + \rho\ \alpha\ \Delta x;$
$s = s + \rho\ \alpha\ \Delta s;$
$\mu = \sigma\ (x^T s) / n;$
$r_q = Mx + q - s;$

Corrector step
$a = \mu e - xs;$
calculate $(\Delta x, \Delta s)$ *from (5);*
if M *is singular* **then**
  return M *is not* P_0;
end if
if $(x(\alpha_c^*(\kappa)), s(\alpha_c^*(\kappa)) \notin D(\gamma)$ **then**
  calculate $\kappa(\Delta x)$ *from (3);*
  if $\kappa(\Delta x)$ *is not defined* **then**
   return M *is not* P_* ;
  end if
  if $\kappa(\Delta x) > \tilde{\kappa}$ **then**
    return M *is not* $P_*(\tilde{\kappa})$;
  end if
  $\kappa = \kappa(\Delta x);$
  $\alpha_p^*(\kappa) = \frac{2\sqrt{(1-\gamma)\gamma}}{(1+4\kappa)n+2};$

end if
$\alpha = correctorStepSize(\gamma, \beta, \beta_{max})$;
$x = x + \rho\,\alpha\,\Delta x$;
$s = s + \rho\,\alpha\,\Delta s$;
$\mu = \sigma\,(x^T s)/n$;
$r_q = Mx + q - s$;
end
END

7.1. Determining the step sizes

During the implementation of the algorithm, specific methods for calculating the step sizes were given.

7.1.1. Calculation of the predictor step size

In the predictor step, we calculated the step size α using the method given by Potra and Liu [2], but we also took into account that the step length should preserve the conditions $x > 0, s > 0$. The following function illustrates this:

function *predictorStepSize(γ):*

$$\alpha_1 = min\left\{-\frac{x_i}{\Delta x_i} \;\middle|\; \Delta x_i < 0\right\};$$

$$\alpha_2 = min\left\{-\frac{s_i}{\Delta s_i} \;\middle|\; \Delta s_i < 0\right\};$$

$$\alpha = min\{\alpha_1, \alpha_2\};$$

$$u = \frac{xs}{\mu};\quad v = \frac{\Delta x \Delta s}{\mu};$$

$$t = \frac{1-\gamma}{(1+4\kappa)n+1};$$

$$\alpha = min\left\{\alpha, \frac{2}{1+\sqrt{1-4e^T v/n}}\right\};$$

for $i = 1$ **to** n **do begin**
$b = v_i - (1-t)\gamma e^T v/n$;
$c = u_i - (1-t)\gamma$;
$\Delta = c^2 - 4bc$;
if $\Delta > 0$ **and** $b \neq 0$ **then**

$$\alpha = min\left\{\alpha, \frac{2(u_i - (1-t)\gamma)}{u_i - (1-t)\gamma + \sqrt{\Delta}}\right\};$$

end if
end
return α;
end

7.1.2. Calculation of the corrector step size

We propose a new method for determining the size of the corrector step, which is different from the one presented in [2]. The function gives the maximum possible step length α, which maintains the condition $x, s > 0$. Then we divide the interval $[0, \alpha]$ into β equal parts and verify whether the step size taken for the values at the endpoints of the subintervals is in the $D(\gamma)$ neighborhood or not. If so, we calculate

$$\mu(\alpha) = \frac{x(\alpha)^T s(\alpha)}{n}$$

for the new point (x, s) and then select the minimum of all these $\mu(\alpha)$ values as the final step size.

If no such point is found, which is in the neighborhood, the value of β is multiplied by 2 and the above procedure is repeated. For the value of β we have introduced an upper limit β_{max}, and if it is reached, the algorithm is stopped with an error. The method can be described using the following function:

function *correctorStepSize(γ, β, β_{max}):*

$$\alpha_1 = min\left\{-\frac{x_i}{\Delta x_i} \;\middle|\; \Delta x_i < 0\right\};$$

$$\alpha_2 = min\left\{-\frac{s_i}{\Delta s_i} \;\middle|\; \Delta s_i < 0\right\};$$

$\alpha = min\{\alpha_1, \alpha_2\}$;
notfound = true;
while $\beta < \beta_{max}$ *and notfound* **do begin**
for $i = 1$ **to** n **do begin**

$$\bar{\alpha} = \frac{\alpha i}{\beta};$$

if $\left(x(\bar{\alpha}), s(\bar{\alpha})\right) \in D(\gamma)$ **and** $\mu(\bar{\alpha}) > \mu(\alpha)$ **then**
$\alpha = \bar{\alpha}$;
notfound = false;
end if
end
$\beta = 2\beta$;
end
if $\beta \geq \beta_{max}$ **then return** *"error"*;
return α;
end

8. Numerical results

The algorithm was implemented in the C++ programming language, in the Visual Studio integrated development environment under Windows operating system on a computer with a 1.9 GHz processor.

We used the following parameters: $\rho = 0.95$, $\sigma = 0.1$, $\varepsilon = 10^{-5}$, $\gamma = 0.9$, $\beta = 100$, $\beta_{max} = 1000$, $\tilde{\kappa} = 10^{40}$ and chose $x_0 = e$ and $s_0 = e$ as the initial point.

8.1. Sufficient matrices

For the problems based on sufficient matrices generated by the method described in [7], our algorithm found the solution in at most 6 iterations. Table 1 shows the average running time (in seconds) for problems of the same size.

Table 1. *Average running time for LCPs presented in* **[7]** *depending on the size of the matrix*

n	10	20	50
CPU	0.1209	0.1941	0.5876
n	100	200	500
CPU	2.0456	9.9068	131.9007

Table 2. *Variation of maximum local κ values depending on the size of the matrix*

n	10	50	100
κ	347.53	$4.65 \cdot 10^{16}$	$1.89 \cdot 10^{34}$

8.2. Csizmadia's matrix

In case of the matrices introduced by Zsolt Csizmadia, it has been theoretically proven **[8]**, that the maximal κ increases exponentially depending on the size of the problem. The obtained local κ values confirm this theoretical result (see **Table 2**).

The obtained κ values vary similar to the results published in **[9]**.

9. Conclusions

In this paper, we presented our implementation of the predictor-corrector algorithm, which can solve the *GLCP* introduced by Illés, Nagy and Terlaky **[1].** We introduced a new method for calculating the step size of the corrector step in the implemented version of the algorithm. We summarized the obtained numerical results for inputs based on sufficient matrices. In case of the matrices introduced by Zsolt Csizmadia, the exponential growth of κ depending on the size of the matrix was observed.

Acknowledgement
The authors acknowledge the research support of the Transylvanian Museum Society (EME).

References

[1] Illés T., Nagy M., Terlaky T.: *Polynomial Interior Point Algorithms for General Linear Complementarity Problems*. Algorithmic Operations Research, 5/1. (2010) 1–12.

[2] Potra F. A., Liu X.: *Predictor-Corrector Methods for Sufficient Linear Complementarity Problems in a Wide Neighborhood of the Central Path*. Optimization Methods and Software, 20/1. (2005) 145–168. https://doi.org/10.1080/10556780512331318038

[3] Illés T., Nagy M., Terlaky T.: *Ep Theorem for Dual Linear Complementarity Problems*. Journal of Optimization Theory and Applications, 140. (2009) 233–238. https://doi.org/10.1007/s10957-008-9440-0

[4] Illés T., Nagy M., Terlaky T.: *A Polynomial Path-Following Interior Point Algorithm for General Linear Complementarity Problems*. Journal of Global Optimization, 47/3. (2010) 329–342. https://doi.org/10.1007/s10898-008-9348-0

[5] Darvay Zs., Füstös Á.: *Numerical Results for the General Linear Complementarity Problem.* Műszaki Tudományos Közlemények, 9. (2019) 43–46. https://doi.org/10.33894/mtk-2019.11.07

[6] Kojima M., Megiddo N., Noma T., Yoshise A.: *A Unified Approach to Interior Point Algorithms for Linear Complementarity Problems*. Springer Verlang, Berlin, 1991.

[7] Illés T., Morapitiye S.: *Generating Sufficient Matrices*. In: *Short Papers of the 8th VOCAL Optimization Conference: Advanced Algorithms held in Esztergom, Hungary*. (Ed.: Friedler F.) Pázmány Péter Katolikus Egyetem, Budapest, 2018. 56–61.

[8] de Klerk E., E.-Nagy M.: *On the Complexity of Computing the Handicap of a Sufficient Matrix*. Mathematical Programming, 129. (2011) 383–402. https://doi.org/10.1007/s10107-011-0465-z

[9] Darvay Zs., Illés T., Povh J., Rigó P. R.: *Feasible Corrector-Predictor Interior-Point Algorithm for $P_*(\kappa)$-Linear Complementarity Problems Based on a New Search Direction*. SIAM Journal on Optimization, 30/3. (2020) 2628–2658. https://doi.org/10.1137/19M1248972