Műszaki Tudományos Közlemények vol. 22. (2025) 27–31.
DOI: Hungarian: https://doi.org/10.33895/mtk-2025.22.05
English: https://doi.org/10.33894/mtk-2025.22.05

IMPLEMENTATION OF AN INTERIOR-POINT ALGORITHM BASED ON A KERNEL FUNCTION FOR WEIGHTED LINEAR COMPLEMENTARITY PROBLEMS

Zsolt DARVAY,[1,2] Edina MÁTHÉ[1,2]

[1] *Babeș-Bolyai University, Faculty of Mathematics and Computer Science, Cluj-Napoca, Romania, zsolt.darvay@ubbcluj.ro*

[2] *Transylvanian Museum Society, Department of Mathematics and Computer Science*

Abstract

We consider the solution of weighted linear complementarity problems using interior-point algorithms, where the search directions are determined by a specific kernel function. To achieve a more efficient implementation, we modify the theoretically well-performing algorithm. The resulting methods are analyzed using a code written in the Python programming language. We present numerical results for problems whose solutions, based on previous theoretical findings, would require a very high number of iterations.

Keywords: *interior-point algorithm, kernel function, weighted linear complementarity problem, search direction, Newton's method.*

1. Introduction

The concept of the weighted linear complementarity problem (WLCP) was introduced by Potra [1] in 2012. The WLCP can be defined as follows. Find a pair of vectors $(x, s) \in \mathbb{R}^n \times \mathbb{R}^n$ such that:

$$\begin{aligned} -Mx + s &= q, \\ xs &= w, \\ x,s &\geq 0, \end{aligned} \qquad (1)$$

where $q \in \mathbb{R}^n$ is a given vector, $w \geq 0$ is a given weight vector, and $M \in \mathbb{R}^{n\times n}$ is a given matrix. The WLCP is actually an extension of the linear complementarity problem (LCP), since the special case where all components of the weight vector are zero results in an LCP. The application areas of the WLCP are broad. In particular, the linear complementarity problem arises in the study of mechanical interactions, such as obstacle problems or structural design problems [2]. Moreover, the WLCP is useful in systems where variables or conditions have varying degrees of importance. It is frequently applied in economic modeling, for example in the case of Fisher's market equilibrium problem [1].

To ensure the solvability of the problem described in system (1), the matrix M must satisfy certain conditions. We analyze a variant of these conditions in which the matrix M satisfies the $P^*(\kappa)$ property. A matrix M is called a $P^*(\kappa)$-matrix if there exists a nonnegative real number κ such that the following inequality holds for all vectors $x \in \mathbb{R}^n$:

$$(1+4\kappa)\sum_{i\in I_+} x_i(Mx)_i + \sum_{i\in I_-} x_i(Mx)_i \geq 0,$$

where $I = \{1, 2, ..., n\}$,
$I_+ = \{i \in I : x_i(Mx)_i \geq 0\}$ and
$I_- = \{i \in I : x_i(Mx)_i < 0\}$.

To solve this problem, we use an interior-point algorithm. Before proceeding, we introduce some notations. Let $\mathbb{R}^n_+$ denote the set of nonnegative real n-dimensional vectors, and $\mathbb{R}^n_{++}$ the set of positive ones. We define $\mathcal{F}$ as the set of feasible solutions, and $\mathcal{F}^0$ as the set of strictly feasible pairs:

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

Furthermore, for a given $(x^0, s^0) \in \mathcal{F}^0$, we define the weighted central path, which depends on a parameter $t \in [0,1]$. For this purpose, we introduce the following notation:

$$w(t) = (1-t)w + tx^0s^0. \qquad (2)$$

If we replace the weight vector in system (1) with the expression of $w(t)$ given above, we obtain a solvable system that determines the weighted central path:

$$\begin{aligned} -Mx + s &= q, \\ xs &= w(t), \\ x,s &\geq 0. \end{aligned} \tag{3}$$

If M is a $P^*(\kappa)$-matrix and $\mathcal{F}^0 \neq \emptyset$, then the previously defined parameterized system of equations has a unique solution $(x(t), s(t))$ for each parameter $t \in [0,1]$. The set of these solutions gives the weighted central path assigned to the problem.

If we apply Newton's method to the system of equations (3), we get the following system:

$$\begin{aligned} -M\Delta x + \Delta s &= 0, \\ s\Delta x + x\Delta s &= w(t) - xs, \end{aligned} \tag{4}$$

from which the search directions can be determined.

Next, we introduce the variance vector $v = \sqrt{\frac{xs}{w(t)}}$, which can be used to define a proximity measure to the weighted central path.

To solve the problem, we will use the method of kernel-based interior-point algorithms. A function ψ is called a kernel function if it satisfies the following conditions:

$\psi: \mathbb{R}_{++} \to \mathbb{R}_+$ is twice continuously differentiable, $\psi(1) = \psi'(1) = 0$ and $\psi''(t) > 0$ for all $t > 0$. For a given kernel function, the associated barrier function $\Psi: \mathbb{R}^n_{++} \to \mathbb{R}_+$ can be defined as follows:

$$\Psi(v) = \sum_{i=1}^{n} \psi(v_i).$$

The system of search directions can be specified using kernel functions:

$$\begin{aligned} -M\Delta x + \Delta s &= 0, \\ s\Delta x + x\Delta s &= -w(t)v\nabla\Psi(v). \end{aligned} \tag{5}$$

It can be shown that using the logarithmic kernel function $\psi(t) = \frac{t^2-1}{2} - \log(t)$ in system (5) yields the original system of equations (4), which is obtained by applying Newton's method. In what follows, we use the kernel function

$$\psi(t) = \frac{t^2-1}{2} + \frac{t^{-1}-1}{2} - \frac{t-1}{2}$$

to compute the search directions. A detailed theoretical analysis of this kernel function is provided in the paper of Chi, Wang, and Lesaja [3].

2. Theoretical algorithm

We present the theoretical algorithm introduced in [3] and then discuss the modifications made for the implementation. Some steps of the algorithm are given by separate functions. To determine the search directions, we substitute the kernel function ψ into the expression on the right-hand side of the system (5), solve the resulting system and return the computed directions. This is done with the following function:

function ***searchDirection*** (x, s, t)
begin
  Solve the system of equations (5).
  return $(\Delta x, \Delta s)$;
end.

To perform the full-Newton step, we add the search directions to the current iterates:

function ***fullNewtonStep*** $(x, s, \Delta x, \Delta s)$
begin
  return $(x, s) + (\Delta x, \Delta s)$;
end.

The parameter defining the weighted central path is updated using the following function:

function ***updateUsingTheta*** (t, θ)
begin
  return $(1 - \theta)t$;
end.

Based on these functions, the theoretical algorithm is presented as follows.

Algorithm 1. (Chi, Wang és Lesaja [3])

Assume that for the pair of vectors $(x^0, s^0) \in \mathcal{F}^0$ we have $x^0 s^0 \geq w$.

Let $\varepsilon > 0$ be the accuracy parameter.

Let $\theta = \bar{\theta}_{min}$ be the barrier update parameter, where

$$\bar{\theta}_{min} = \frac{\gamma}{3\gamma + (18\bar{\kappa} + 10)\beta},$$

$$\bar{\kappa} = \frac{(1 + 4\kappa)max(x^0 s^0)}{4\,min(w)} - 0.25,$$

$\gamma = min(w)$, $\beta = \|x^0 s^0 - w\|$.

Let $\tau > 0$ be the threshold parameter.

Let $t^0 = 1$, such that $\delta(x^0, s^0; t^0) \leq \tau$, where

$$\delta(x, s; t) = \left\| v\left(\frac{\Delta x}{x} + \frac{\Delta s}{s}\right) \right\| = \left\| \frac{e}{2} + \frac{e}{2v^2} - v \right\|.$$

$x = x^0$, $s = s^0$, $t = t^0$;

while $\|xs - w\| > \varepsilon$ **do**
begin
  $(\Delta x, \Delta s) =$ ***searchDirection*** (x, s, t);
  $(x, s) =$ ***fullNewtonStep*** $(x, s, \Delta x, \Delta s)$;
  $t =$ ***updateUsingTheta*** (t, θ);
end.

To implement it efficiently, we made several modifications to Algorithm 1.

3. Modified algorithm

Since it is possible during the steps of the algorithm to reach an infeasible point, we introduce the residual vector

$r_q = -Mx + s - q$.

Instead of system (5) we determine the search directions using the following modified system:

$$\begin{aligned} -M\Delta x + \Delta s &= -r_q, \\ s\Delta x + x\Delta s &= -w(t)v\nabla\Psi(v). \end{aligned} \qquad (6)$$

The function assigned to system (6) is the following:

function ***modifiedSearchDirection***(x, s, t)
begin
  $r_q = -Mx + s - q$;
  Solve the system of equations (6).
  return $(\Delta x, \Delta s)$;
end.

Instead of moving to the next point with a full-Newton step, we calculate the maximum step-size to the boundary in a variable α, and then reduce it by a factor $\rho \in (0, 1)$:

function ***NewtonStep***$(x, s, \Delta x, \Delta s, \rho)$
begin
  $\alpha_x = min\left\{\frac{-x_i}{\Delta x_i} \middle| \Delta x_i < 0\right\}$;
  $\alpha_s = min\left\{\frac{-s_i}{\Delta s_i} \middle| \Delta s_i < 0\right\}$;
  $\alpha = min\{\alpha_x, \alpha_s\}$;
  return $(x, s) + \rho\,\alpha(\Delta x, \Delta s)$;
end.

We implemented two methods to reduce the parameter t. The first is the same as the version given in the theoretical algorithm, which uses θ to reduce the value of t:

function ***updateParameter***(t, θ, x, s, ζ)
begin
  return ***updateUsingTheta*** (t, θ);
end.

The second variant updates the parameter t using the method introduced in [4] applying a scaling factor $\zeta \in (0,1]$:

function ***updateParameter***(t, θ, x, s, ζ)
begin
  return ***updateUsingZeta***(x, s, ζ);
end
function ***updateUsingZeta***(x, s, ζ)
begin
  return $t = \zeta \left| \frac{x^T s - w^T e}{(x^0)^T s^0 - w^T e} \right|$;
end.

Assuming that $x^0 s^0 \neq w$, we obtain that the denominator of the above expression cannot be 0. The modified algorithm is presented below.

Algorithm 2. Suppose that for the pair of vectors $(x^0, s^0) \in \mathcal{F}^0$ we have $x^0 s^0 \geq w$, $x^0 s^0 \neq w$.

Let $\varepsilon > 0$ be the accuracy parameter. Furthermore, let $\theta \in (0, 1)$, and $\zeta \in (0, 1]$ be the update parameters.

Let $\rho \in (0, 1)$ be the scaling parameter of the step.
$t^0 = 1$, $x = x^0$, $s = s^0$, $t = t^0$;

while $\|xs\text{-}w\| > \varepsilon$ **do**
begin
  $(\Delta x, \Delta s) =$ ***modifiedSearchDirection***(x, s, t);
  $(x, s) =$ ***NewtonStep***$(x, s, \Delta x, \Delta s, \rho)$;
  $t =$ ***updateParameter***(t, θ, x, s, ζ);
end.

We present several numerical results obtained using the above algorithm.

4. Numerical results

The algorithm presented above was implemented in the Python programming language, using the NumPy library for linear algebra operations. Each time we executed the program we used the matrix proposed by Csizmadia [5]:

$$M = \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ -1 & 1 & 0 & \dots & 0 \\ -1 & -1 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -1 & -1 & -1 & \dots & 1 \end{pmatrix}.$$

For this matrix, it was proven in [6] that for $n \geq 4$ we have $\kappa = 2^{2n-8} - 0.25$. This matrix plays a significant role in the study of $P^*(\kappa)$-WLCP, as the value of κ increases exponentially with the problem size. We also assumed that $q = [0, 1, ..., n-1]^T$. Furthermore, we used the initial values $x^0 = s^0 = e$, where e is the n-dimensional all-one vector. Note that in this case, the condition $(x^0, s^0) \in \mathcal{F}^0$ holds.

To satisfy the inequality $x^0 s^0 \geq w$, the values of the weight vector w were chosen as randomly generated positive real numbers less than one. Since we analyzed the algorithm from several perspectives, the elements of w were generated under different constraints. Furthermore, we used the values $\varepsilon = 10^{-6}$, $\rho = 0.9$. As for the parameter θ, we also investigated values greater than the theoretically determined value $\bar{\theta}_{min}$, from the following set:

$\theta \in \{\bar{\theta}_{min}, 10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 0.01, 0.02, ..., 0.1, 0.2, ..., 0.9\}$.

We found that, in general, increasing θ still yields correct results, and the algorithm converges to the solution more quickly. To illustrate this, we consider the number of iterations and CPU times (in seconds) obtained for the Csizmadia ma-

trix of the size 4×4. We used a computer with an Intel Core i5-1035G1 processor and 8 GB of RAM, under the Windows 10 operating system, in a Python 3.11.6 environment for the tests (**Table 1**).

The following table presents the minimum number of iterations obtained for different values of θ as a function of the dimension. It can be observed that the minimum number of iterations increases linearly with the dimension. In the last row we indicated from which value of θ we get this minimum number of iterations. It can be concluded that in case of higher dimensions, the number of iterations cannot always be reduced by increasing the value of θ (**Table 2**).

Further results were obtained with the implementation in which the parameter t is reduced using a scalar ζ. With this method, we achieved similar results in terms of the number of iterations of Algorithm 2 (**Figure 1**).

In case of $n=10$, $\zeta=0.1$, **Figure 2** is intended to show how the categorization of the weight vector into different intervals affects the values of s in the solution. It is known that if the weight vector is zero, then we get back in the vector s the increasing natural numbers represented by the vector q. When the values of w fall into an interval close to zero, the values of s approximate those of q. If we select an interval for the values of the weight vector, the elements of which are farther from zero, then the values of s will also shift accordingly.

Table 1. *The number of iterations obtained for different values of θ.*

Θ	Iteration	CPU
0.1	139	0.0156
0.01	1445	0.0625
10^{-3}	14498	0.7031
10^{-4}	145029	6.8281
10^{-5}	1450342	107.3906
10^{-6}	14503470	1035.1250
$\bar{\theta}_{min}=3.86\cdot10^{-7}$	37573477	3006.2031

Table 2. *Minimum iteration number as a function of the dimension, where D denotes the dimension, and I the minimum number of iterations.*

D	20	30	40	50	100	200	300	400	500
I	18	22	26	29	48	84	121	158	195
Θ	0.7	0.6	0.5	0.5	0.3	0.2	0.2	0.2	0.09

This trend is also observed in the case of x, where the elements of this vector converge to zero as the values of the weight vector are chosen closer to zero (**Figure 3**).

5. Conclusions

We modified the kernel function-based interior-point algorithm published by Chi, Wang, and Lesaja [3] to efficiently solve the $P^*(\kappa)$-WLCP. We investigated two variants to reduce the parameter that determines the weighted central path. We presented various numerical results using the Python code we developed.

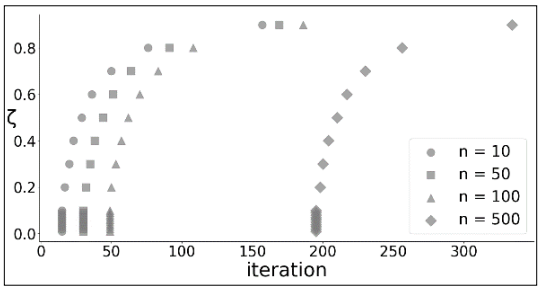

Fig. 1. *Number of iterations of the algorithm as a function of the parameter ζ.*

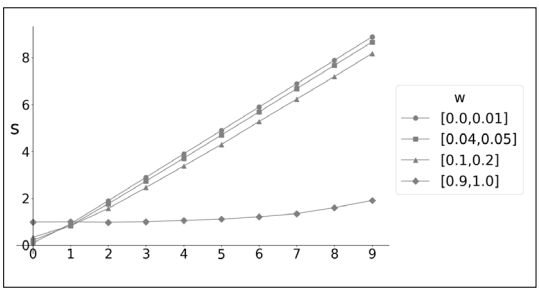

Fig. 2. *Values of the vector s as a function of the weight vector.*

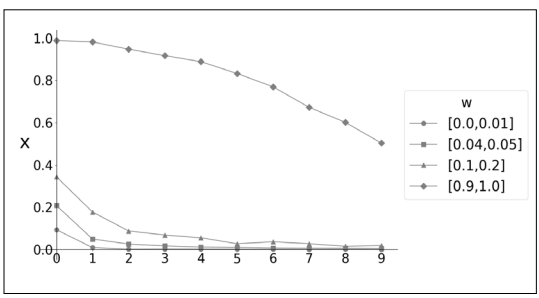

Fig. 3. *Values of the vector x as a function of the weight vector.*

Acknowledgements

The authors express their gratitude to the Transylvanian Museum Society for the support provided for the research work.

References

[1] Potra F. A.: *Weighted Complementarity Problems – a New Paradigm for Computing Equilibria*. SIAM Journal on Optimization, 22/4. (2012) 1634–1654. https://doi.org/10.1137/110837310

[2] Ferris M. C., Pang J. S.: Engineering and Economic Applications of Complementarity Problems. SIAM Review, 39/4. (1997) 669–713. https://doi.org/10.1137/S0036144595285963

[3] Chi X., Wang G., Lesaja G.: *Kernel-Based Full-Newton Step Feasible Interior-Point Algorithm for $P^*(\kappa)$-Weighted Linear Complementarity Problem.* Journal of Optimization Theory and Applications, 202/1. (2024) 108–132. https://doi.org/10.1007/s10957-023-02327-9

[4] Darvay Zs., Orbán A.-Sz.: *Implementation of the Full-Newton Step Algorithm for Weighted Linear Complementarity Problems*. Papers on Technical Science, 15. (2021) 15–18. https://doi.org/10.33894/mtk-2021.15.04

[5] 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

[6] E.-Nagy M., Illés T., Povh J., Varga A., Zerovnik J.: *Sufficient Matrices: Properties, Generating and Testing*. Journal of Optimization Theory and Applications, 202/1. (2024) 204–236. https://doi.org/10.1007/s10957-023-02280-7