A truncated Newton interior-point algorithm for the solution of a multicommodity spatial equilibrium model *

Luís Portugal † Luís Fernandes ‡ Joaquim Júdice §

Abstract

In this paper we introduce a Truncated Newton (TN) interior-point algorithm for solving monotone linear complementarity problems. Contrary to the usual Newton interior-point methods that use exact Newton directions, this algorithm computes at each iteration an approximate Newton direction by employing appropriate preconditioned iterative linear system solvers. The stopping criterion used in these iterative procedures is defined in terms of the norm of the residual vector associated to the linear constraints and guarantes global convergence for the TN interior-point algorithm. We describe an efficient implementation of this algorithm for solving the Linear Complementarity Problem (LCP) associated to the symmetric and nonsymmetric multicommodity spatial equilibrium problems. The Preconditioned Conjugate Gradient (PCG) or the Preconditioned Quasi Minimal Residual (PQMR) algorithms are incorporated in the implementation depending on the matrix of the LCP to be symmetric or nonsymmetric. We also describe an incomplete QR factorization preconditioning technique which greatly improves the convergence properties of the PCG and PQMR algorithms. Finally, the stopping criterion of the TN algorithm exploits the stucture of the network model and leads to an exact solution for the LCP. We report extensive computational experience that shows the great efficiency of the approach discussed in this paper.

1 Introduction

In the last decades several spatial equilibrium models have been addressed in the literature [2, 12, 14, 23, 24]. In this paper we discuss the multicommodity spatial equilibrium model studied by Güder et al. [14] and Pang [23]. This model can be formulated as a linear complementarity problem, since it is assumed that the supply and demand functions are linear. In this model there exist n separate regions which trade m commodities among them. In each region i, the price p_{ig} of commodity g and the net imports y_{ih} of commodities $h = 1, \ldots, m$ are related by

$$p_{ig} \quad = \quad a_{ig} - \sum_{h=1}^{m} b_{igh} y_{ih}, \tag{1}$$

where a_{ig} is the equilibrium price of commodity g in the absence of imports and exports and b_{igh} is a parameter related to the elasticity of demand and supply. The net import y_{ig} of

*Latest revision: October 12, 1995

†Departamento de Ciências da Terra, Universidade de Coimbra, 3000 - Coimbra, Portugal.

‡Escola Superior de Tecnologia de Tomar, 2300 - Tomar, Portugal.

§Departamento de Matemática, Universidade de Coimbra, 3000 - Coimbra, Portugal.

commodity g in region i is given by

$$y_{ig} = \sum_{j=1}^{n} x_{jig} - \sum_{j=1}^{n} x_{ijg},$$

where x_{ijg} is the trade flow of commodity g from region i to region j.

Let c_{ijg} be the unit transportation cost of commodity g from region i to region j. As stated in [14, 23], the problem is to find p_{ig}, y_{ig} and x_{ijg} satisfying

$$\begin{aligned} p_{ig} + c_{ijg} - p_{jg} &\geq 0, \\ x_{ijg} &\geq 0, \\ x_{ijg}(p_{ig} + c_{ijg} - p_{jg}) &= 0, \end{aligned} \tag{2}$$

for all $i, j = 1, \ldots, n$ and $g = 1, \ldots, m$. We assume, without loss of generality, that $x_{ijg} = 0$ for all $i = j$ and $g = 1, \ldots, m$.

This spatial equibrium problem may be represented by a directed graph $\Psi = (\mathcal{N}, \mathcal{G})$, where the set of n nodes $\mathcal{N}$ represents the regions and the set of $n(n-1)$ directed arcs $\mathcal{G}$ corresponds to directed connections between pairs of regions. In this case $\Psi = (\mathcal{N}, \mathcal{G})$ is complete, that is, $(i, j) \in \mathcal{G}$ and $(j, i) \in \mathcal{G}$ for every pair of distinct elements i and j belonging to $\mathcal{N}$. Let $G \in \mathbb{R}^{n \times n(n-1)}$ be the node-arc incidence matrix associated to the graph $\Psi = (\mathcal{N}, \mathcal{G})$ and assume, for simplicity, that the flow variables x_{ijg} are grouped according to commodities. By eliminating the variables $y_{ig}, i = 1, \ldots, n,\ g = 1, \ldots, m$ and introducing the variables z_{ijg} defined by

$$z_{ijg} = p_{ig} + c_{ijg} - p_{jg} \quad i, j = 1, \ldots n,\ g = 1, \ldots, m$$

we may state the conditions (1) and (2) as the Linear Complementarity Problem (LCP)

$$\begin{aligned} z &= J^\top B J x - J^\top a + c, \\ x &\geq 0, \\ z &\geq 0, \\ x^\top z &= 0. \end{aligned} \tag{3}$$

In this formulation, $J \in \mathbb{R}^{mn \times mn(n-1)}$ is a block diagonal matrix with m identical diagonal blocks G, B is a matrix of order mn given by

$$B = \begin{bmatrix} B_{11} & B_{12} & \cdots & B_{1g} & \cdots & B_{1m} \\ B_{21} & B_{22} & \cdots & B_{2g} & \cdots & B_{2m} \\ \vdots & \vdots & & \vdots & & \vdots \\ B_{g1} & B_{g2} & \cdots & B_{gg} & \cdots & B_{gm} \\ \vdots & \vdots & & \vdots & & \vdots \\ B_{m1} & B_{m2} & \cdots & B_{mg} & \cdots & B_{mm} \end{bmatrix}, \tag{4}$$

where each B_{gh} is an $n \times n$ diagonal matrix with diagonal entries b_{igh}, $a \in \mathbb{R}^{mn}$ is a vector with components a_{ig} and $x, z, c \in \mathbb{R}^{mn(n-1)}$ are the vectors whose elements are x_{ijg}, z_{ijg} and c_{ijg}, respectively. The matrix B is assumed to be positive definite (not necessarily symmetric) [23, 33], whence the matrix $J^\top B J$ is positive semi-definite of order $mn(n-1)$. Therefore the LCP (3) is monotone.

Single and block SOR algorithms are the most common procedures suggested in the literature for the solution of the LCP (3) [14, 23]. However, the efficiency of these iterative approaches critically depends on the choice of the value of a certain relaxation parameter. Furthermore, for large scale problems these methods are unable to compute a sufficiently accurate solution that can be used to identify an exact solution of the LCP (3). To date only instances much smaller than those we use in our experiments have been reported in the literature. On the other hand, to the best of our knowledge, there is no procedure to recover efficiently an exact solution of the LCP (3) by using information provided by a SOR algorithm.

In [7], Fernandes et al. describe the first conjugate-gradient based implementation of an interior-point method for the solution of the single commodity case of the LCP (3) ($m = 1$). Fernandes et al. show that the Mehrotra's predictor-corrector interior-point method [21] can compute approximate solutions more accurate than those computed by the SOR algorithm discussed in [14]. On the other hand, they also provide evidence that this interior-point method always requires a smaller computational effort to obtain approximate solutions with an accuracy equal to the most accurate solutions achieved by the SOR algorithm.

In a recent paper [26], Portugal and Júdice introduce an enhancement of this type of procedure for the solution of the single commodity case of the LCP (3). This new implementation takes advantage of the network structure of the model and its better efficiency is based on two main issues:

- Reduction of the $2n(n-1) \times 2n(n-1)$ system that computes the search direction at each iteration of the interior-point algorithm to a symmetric positive definite system of order n for which a nice preconditioning technique is available.

- Existence of a network based pivoting algorithm [24] that is able to recover efficiently an exact solution of the LCP (3) from a starting solution provided by the predictor-corrector method.

As stated in [29], the choice of algorithm, preconditioners and method for recovering an exact solution to the problem are the key ingredients to achieve an efficient interior-point code for network flow problems.

In this paper we study a truncated Newton interior-point method for the monotone linear complementarity problem and describe implementations of this algorithm to solve the LCP (3) when the matrix B is symmetric and nonsymmetric. The truncated Newton method has been proven to be advantageous over the other interior-point techniques [25, 27] when the structure of the problem requires an iterative procedure to compute search directions. In fact, the bulk of the interior-point algorithms is the computation at each iteration of a search direction by solving one or two systems of linear equations. In the general case, the algorithm incorporates a sparse direct solver for this purpose. However, for some special structured problems, this technique is not appropriate and an iterative linear system solver must be designed to find the search directions required by the interior-point methods [16, 29]. The idea behind the truncated optimization algorithms is to derive a stopping criterion to the iterative linear system solvers that minimizes the computational effort involved in computing search directions that guarantee global convergence [5, 6]. Recently, Portugal et al. [27] introduced a globally convergent truncated Newton interior-point algorithm for linear programming. The practical version of this algorithm for the linear minimum cost network flow problem, *PDNET*, has proven to be competitive with the sate-of-the-art network flow codes. We show that in the case of the LCP (3), it is possible to obtain a search direction at each iteration of the truncated Newton method by solving a reduced linear system of order mn. We describe efficient preconditioned iterative techniques to solve this system. When B is symmetric the matrix of the reduced system is symmetric positive definite and the Preconditioned Conjugate Gradient (PCG) algorithm is used to compute a search direction. If B is nonsymmetric the solution of the reduced system is found by employing a Preconditioned Quasi Minimal Residual (PQMR) method. We derive a stopping criterion to these iterative procedures that guarantees global convergence and a small computational effort per iteration in the truncated Newton algorithm.

The method presented in this paper attempts to find an exact solution to the LCP (3). In 1978, Glassey [12] proved that in the single commodity case there exists a solution of the LCP (3) where the trade routes of positive flow form a forest. Glassey also demonstrated that if the triangle inequality

$$c_{ijg} \leq c_{ikg} + c_{kjg}, \; i, k, j = 1, \ldots, n, \; g = 1, \ldots, m \tag{5}$$

is verified, then there exist no transshipment of flows in such a solution. It is straightforward to show by using the same arguments of Glassey [12] that, whenever the LCP (3) is feasible, there also exists a solution of the multicommodity model ($m > 1$) where the trade routes of positive flow of each commodity form a forest. Furthermore, it is also possible to demonstrate that if the condition (5) is verified, then the same conclusion about the nonexistence of transshipment of flows can be made, that is, if the triangle inequality (5) holds for $g = h$, then there exist no transshipment of flows in the forest that represents the trade flow of commodity h. In [24] it is shown that for $m = 1$ the matrix of the system that computes a basic complementary solution of the LCP (3) represented by a forest is nonsingular. We prove later that this property also holds when $m > 1$. We describe a heuristic rule which uses the information given at each iteration of the truncated Newton interior-point method to guess a set of m forests (one for each commodity) that represents a possible solution to the LCP (3). This possible solution is then computed by solving a linear system of equations with an iterative technique using a very tightened stopping criterion. As we show in the last section of the paper, this procedure, together with a perturbation scheme of the vector $c - J^\top a$ designed to reduce the degeneracy effects, recovers an exact solution for all the test problems used in the computational experience reported in this paper.

Before concluding this introduction, we present some notation and outline the remainder of the paper. I_n and $0_{m\times n}$ represent the $n\times n$ identity and the $m\times n$ null matrices respectively. We represent a vector of ones by e. Let $x \in \mathbb{R}^n$. We denote by X the $n \times n$ diagonal matrix having the components of x in the diagonal. The number of elements of the set $\mathcal{H}$ is represented by $|\mathcal{H}|$. We denote by $A_{\cdot\mathcal{H}}$ (by $A_{\mathcal{H}\cdot}$) the submatrix of A constituted by columns (by rows) with indices in the set $\mathcal{H}$. $A_{\mathcal{I}\mathcal{H}}$ represents the submatrix of A with row and column indices in the sets $\mathcal{I}$ and $\mathcal{H}$, respectively. When A is a diagonal matrix we use $A_{\mathcal{H}}$ to abbreviate $A_{\mathcal{H}\mathcal{H}}$. $x_{\mathcal{H}}$ represents the subvector of x constituted by elements with indices in the set $\mathcal{H}$. When $A \in \mathbb{R}^{mn\times mn}$ is a matrix with the same sparsity structure of matrix B defined in formula (4), we denote by a_{igh} and a_{ihg} the elements of matrix A associated to region i and commodities g and h. In this case, A_{gh} and A_{hg} are diagonal submatrices of A with diagonal elements a_{igh} and a_{ihg}, $i = 1, \ldots, n$, respectively. Let $y \in \mathbb{R}^{mn}$ and $x \in \mathbb{R}^{mn(n-1)}$. The element of y associated to region i and commodity g is denoted by y_{ig}. Similarly, we represent by x_{ijg} the element of x associated to commodity g and route (i, j). In this case, y_g and x_g denote the subvectors of y and x constituted by the elements y_{ig} and x_{ijg}, $i, j = 1, \ldots, n$, respectively. When A is a block diagonal matrix with m diagonal blocks, we denote by A_g the block associated to commodity g. In this case, a_{ijg} may have three different meanings depending on the context. If A_g is a diagonal matrix of order n, a_{iig} denotes the i-th diagonal element of A_g and is sometimes abbreviated to a_{ig}. If A_g is a nondiagonal matrix of order n, a_{ijg} represents the element of A_g with row and column indices i and j, respectively. Finally, if A_g is a diagonal matrix of order $n(n-1)$ $(n(n-1)/2)$, a_{ijg} denotes the diagonal element of A_g associated to route (i,j) (routes (i,j) and (j,i)).

The organization of this paper is as follows. Section 2 introduces the truncated Newton algorithm for the monotone LCP. The implementation of this algorithm for the LCP (3) is presented in Section 3. Finally, the computational experience and some concluding remarks are reported in sections 4 and 5, respectively.

2 The truncated Newton interior-point algorithm

In this section, we describe the Truncated Newton (TN) interior-point algorithm for the monotone LCP. The monotone LCP is to find vectors $z \in \mathbb{R}^n$ and $x \in \mathbb{R}^n$ satisfying:

$$\begin{aligned} z &= Mx + q, \\ x &\geq 0, \\ z &\geq 0, \\ x^\top z &= 0, \end{aligned} \tag{6}$$

where $q \in \mathbb{R}^n$ and M is a Positive Semi-Definite (PSD) matrix of order n. For given $\bar{\varepsilon} > 0$, $0 < \gamma < 1$ and $\bar{\gamma} > 0$, let

$$\begin{aligned} \bar{\mathcal{N}} = \{(x,z) \in \mathbb{R}^{2n} : \ & x > 0, z > 0, \\ & x_i z_i \geq \gamma x^\top z/n \ (i = 1, 2, \ldots, n), \\ & x^\top z \geq \bar{\gamma}\|z - Mx - q\| \ or \ \|z - Mx - q\| \leq \bar{\varepsilon}\}, \end{aligned}$$

be a neighborhood of the central path [9, 20]

$$\begin{aligned} z &= Mx + q, \\ Xz &= \mu e, \\ x, z &\geq 0, \end{aligned} \tag{7}$$

defined for $\mu > 0$. The TN algorithm begins with a point $(x^0, z^0) \in \bar{\mathcal{N}}$. At iteration k, the parameter μ_k is computed by

$$\mu_k = \beta_1 \frac{(x^k)^\top z^k}{n}, \quad 0 < \beta_1 < 1. \tag{8}$$

Then, the approximate Newton direction $(\Delta x^k, \Delta z^k)$ associated to the system of nonlinear equations that defines the central path (7) is obtained as the solution of the linear system of equations

$$\begin{aligned} M\Delta x^k - \Delta z^k &= z^k - Mx^k - q + \bar{r}^k, \\ Z^k \Delta x^k + X^k \Delta z^k &= -X^k Z^k e + \mu_k e, \end{aligned}$$

where $\bar{r}^k$ is such that

$$\|\bar{r}^k\| \leq \beta_0 \|z^k - Mx^k - q\|, \quad 0 \leq \beta_0 < \beta_1.$$

Finally, a new iterate (x^{k+1}, z^{k+1}) is computed by

$$(x^{k+1}, z^{k+1}) = (x^k, z^k) + \alpha_k(\Delta x^k, \Delta z^k)$$

where α_k is a positive stepsize.

As stated in [18, 27], global convergence for the TN method is guaranteed if we choose the stepsize α_k such that $\alpha_k \in]0, 1]$ and

$$\begin{array}{rcl} (x^{k+1}, z^{k+1}) & \in & \bar{\mathcal{N}}, \\ (x^{k+1})^\top z^{k+1} & \leq & (1 - \bar{\alpha}_k(1 - \beta_3)(x^k)^\top z^k, \quad \beta_1 < \beta_3 < 1, \end{array}$$

with $\bar{\alpha}_k$ the maximum value of $\tilde{\alpha}$, $0 \leq \tilde{\alpha} \leq 1$, such that the conditions

$$\begin{array}{rcl} (x^k, z^k) + \alpha(\Delta x^k, \Delta z^k) & \in & \bar{\mathcal{N}}, \\ (x^k + \alpha \Delta x^k)^\top (z^k + \alpha \Delta z^k) & \leq & (1 - \alpha(1 - \beta_2)(x^k)^\top z^k, \quad \beta_1 < \beta_2 < \beta_3, \end{array}$$

hold for every $\alpha \in [0, \tilde{\alpha}]$. In fact, since $\|\bar{r}^k\|$ is bounded from above by $\beta_1 \|z^k - Mx^k - q\|$, it is not difficult to show that there is an iteration k such that (x^k, z^k) is either an approximate solution satisfying

$$(x^k)^\top z^k \leq \varepsilon \text{ and } \|z^k - Mx^k - q\| \leq \bar{\varepsilon}$$

for any predifined tolerances ε and $\bar{\varepsilon}$ required for complementarity and feasibility, respectively, or verifies

$$\|(x^k, z^k)\|_1 > \omega \tag{9}$$

for an arbitrarily large ω. In the first case, the algorithm achieves an approximate solution of the LCP (6). In the second case, we cannot make any conclusion. However, extensive computational experience demonstrates that the algorithm reaches an approximate solution of the LCP (6) whenever this problem has a nonempty set of solutions. We conjecture that it is possible to modify the algorithm either to derive information about the infeasibility of the LCP (6) when condition (9) holds, or to remove condition (9) when it is assumed that the LCP (6) is feasible. In fact, these two possibilities were already considered in the analysis of infeasible interior-point methods (see [35] and [18]).

The objective of this paper is to construct an efficient implementation of the TN method for the multicommodity spatial equilibrium monotone LCP (3). As discussed in the next section, we choose the stepsize α_k and the initial point (x^0, z^0) using schemes that have empirically proven to be efficient, even they do not necessarily guarantee global convergence to the TN method.

3 Implementation issues

In this section we describe the implementation of the Truncated Newton (TN) method for the multicommodity spatial equilibrium LCP (3). The main issues of this implementation are the techniques used to compute an approximate Newton direction in the symmetric and nonsymmetric cases, the stopping criterion for the TN algorihm, the initial solution and the way of choosing the stepsize α_k and the parameters β_0 and β_1.

3.1 The approximate Newton direction

As stated in the previous section the bulk of the implementation of the TN algorithm for the LCP (3) is the solution of the linear system

$$
\begin{array}{rcl}
J^\top BJ\Delta x^k - \Delta z^k & = & z^k - J^\top BJx^k + J^\top a - c + \bar{r}^k, \\
Z^k \Delta x^k + X^k \Delta z^k & = & -X^k Z^k e + \mu_k e,
\end{array} \tag{10}
$$

where $\bar{r}^k$ is some vector satisfying

$$
\|\bar{r}^k\| \quad \leq \quad \beta_0 \|z^k - J^\top BJx^k + J^\top a - c\|. \tag{11}
$$

We apply preconditioned iterative techniques to obtain the solution of (10) indirectly by solving a reduced system of order mn. When the matrix B is symmetric the reduced system is symmetric positive definite and the Preconditioned Conjugate Gradient (PCG) algorithm is employed for this purpose. If B is nonsymmetric we compute the solution of the reduced system by using a Preconditioned Quasi Minimal Residual (PQMR) method. Next we present some useful definitions, establish the equivalence between (10) and the reduced system and derive an appropriate stopping criterion to the PCG and PQMR algorithms. We note that some of the proofs of the results are immediate and thus omitted.

Proposition 1. Let $\bar{\mathcal{H}}_1 = \{(i,j) \in \mathcal{G} : i < j\}$, $\bar{\mathcal{H}}_2 = \{(i,j) \in \mathcal{G} : i > j\}$, $\mathcal{H}_1 = \{(i,j,g) : g = 1,\ldots,m \text{ and } (i,j) \in \bar{\mathcal{H}}_1\}$ and $\mathcal{H}_2 = \{(i,j,g) : g = 1,\ldots,m \text{ and } (i,j) \in \bar{\mathcal{H}}_2\}$. Then $J = [J_{\cdot\mathcal{H}_1} J_{\cdot\mathcal{H}_2}] = [H \ -H]$ where $H \in \mathbb{R}^{mn \times mn(n-1)/2}$ is a block diagonal matrix with m identical blocks $G_{\cdot\bar{\mathcal{H}}_1}$.

Proof. Obvious since J is a block diagonal matrix with m identical blocks G and $G = [G_{\cdot\bar{\mathcal{H}}_1} G_{\cdot\bar{\mathcal{H}}_2}]$ where $G_{\cdot\bar{\mathcal{H}}_1} = -G_{\cdot\bar{\mathcal{H}}_2}$. □

Proposition 2. $B = LDU$, where $L, U, D \in \mathbb{R}^{mn \times mn}$ are nonsingular matrices, D is diagonal and L and U are lower and upper triangular, respectively. Furthermore, $U = L^\top$ if B is symmetric.

Proof. The proof follows from the positive definiteness of matrix B (see for example [13]). □

Proposition 3. Let Θ^k be given by $\Theta^k = X^k (Z^k)^{-1}$. Then

$$
\left(J^\top BJ + (\Theta^k)^{-1}\right)^{-1} \quad = \quad \Theta^k - \Theta^k J^\top (B^{-1} + J\Theta^k J^\top)^{-1} J\Theta^k.
$$

Proof. This is the well-known Sherman-Morrison-Woldbury's formula and the proof follows from the nonsingularity of matrices Θ^k and B (see for example [3]). □

In the next Theorem we establish the equivalence between (10) and a linear system of order mn whose matrix can be written in terms of the matrices H, L, D, U and W^k. The matrices H, L, D, U are defined in Propositions 1 and 2, while W^k is given by $W^K = \Theta^k_{\mathcal{H}_1} + \Theta^k_{\mathcal{H}_2}$, where Θ^k is defined in Proposition 3.

Theorem 4. Let s be the solution of the linear system

$$
(D^{-1} + UHW^k H^\top L)s \quad = \quad d + r, \tag{12}
$$

where d is given by

$$\begin{aligned} d &= UH\left(\Theta^k_{\mathcal{H}_1}\left(-z^k_{\mathcal{H}_1} - \mu_k (X^k_{\mathcal{H}_2})^{-1} e + c_{\mathcal{H}_1}\right) - \Theta^k_{\mathcal{H}_2}\left(-z^k_{\mathcal{H}_2} - \mu_k (X^k_{\mathcal{H}_2})^{-1} e + c_{\mathcal{H}_2}\right)\right) + \\ & \quad D^{-1}L^{-1}a, \end{aligned} \tag{13}$$

and r is some vector satisfying

$$\|r\| \leq \beta_0 \frac{\|z^k - J^\top BJx^k + J^\top a - c\|}{\|J^\top LD\|}. \tag{14}$$

Furthermore, let

$$\Delta x^k_{\mathcal{H}_1} = \Theta^k_{\mathcal{H}_1}\left(H^T Ls + \mu_k (X^k_{\mathcal{H}_1})^{-1} e - c_{\mathcal{H}_1}\right), \tag{15}$$

$$\Delta x^k_{\mathcal{H}_2} = \Theta^k_{\mathcal{H}_2}\left(-H^T Ls + \mu_k (X^k_{\mathcal{H}_2})^{-1} e - c_{\mathcal{H}_2}\right), \tag{16}$$

$$\Delta z^k_{\mathcal{H}_1} = \mu_k (X^k_{\mathcal{H}_1})^{-1} e - z^k_{\mathcal{H}_1} - (\Theta^k_{\mathcal{H}_1})^{-1} \Delta x^k_{\mathcal{H}_1}, \tag{17}$$

$$\Delta z^k_{\mathcal{H}_2} = \mu_k (X^k_{\mathcal{H}_2})^{-1} e - z^k_{\mathcal{H}_2} - (\Theta^k_{\mathcal{H}_2})^{-1} \Delta x^k_{\mathcal{H}_2}. \tag{18}$$

Then, the direction $(\Delta x^k, \Delta z^k)$ given by $(\Delta x^k, \Delta z^k) = (\Delta x^k_{\mathcal{H}_1}, \Delta x^k_{\mathcal{H}_2}, \Delta z^k_{\mathcal{H}_1}, \Delta z^k_{\mathcal{H}_2})$ is the solution of the linear system (10) for

$$\bar{r}^k = J^\top LDr. \tag{19}$$

Moreover the condition (11) holds.

Proof. Formulas (17) and (18) can be obtained directly from the second equation in (10) by using Proposition 1. Now, by using the first equation in (10), formulas (12), (13), (17), (18) and (19), and Propositions 1, 2 and 3 we obtain

$$\begin{aligned} \Delta x^k &= \left(J^\top BJ + (\Theta^k)^{-1}\right)^{-1}\left(-J^\top BJx^k + J^\top(a + LDr) - c + \mu_k (X^k)^{-1} e\right) \\ &= \left(\Theta^k - \Theta^k J^\top (B^{-1} + J\Theta^k J^\top)^{-1} J\Theta^k\right)\left(-J^\top BJx^k + J^\top(a + LDr) - c + \right. \\ & \qquad \left. \mu_k (X^k)^{-1} e\right) \\ &= \Theta^k J^\top (B^{-1} + J\Theta^k J^\top)^{-1}\left(J\Theta^k c - \mu_k J(Z^k)^{-1} e\right) - \Theta^k c + \mu_k (Z^k)^{-1} e + \\ & \quad \Theta^k J^\top \left(I - (B^{-1} + J\Theta^k J^\top)^{-1} J\Theta^k J^\top\right) B\left(-Jx^k + B^{-1}(a + LDr)\right) \\ &= \Theta^k J^\top (B^{-1} + J\Theta^k J^\top)^{-1}\left(J\Theta^k c - \mu_k J(Z^k)^{-1} e\right) - \Theta^k c + \mu_k (Z^k)^{-1} e + \\ & \quad \Theta^k J^\top (B^{-1} + J\Theta^k J^\top)^{-1}\left(-Jx^k + B^{-1}(a + LDr)\right) \\ &= \Theta^k J^\top (B^{-1} + J\Theta^k J^\top)^{-1}\left(-Jx^k - \mu_k J(Z^k)^{-1} e + B^{-1}(a + LDr) + J\Theta^k c\right) + \\ & \quad \mu_k (Z^k)^{-1} e - \Theta^k c \\ &= \Theta^k J^\top \left((LDU)^{-1} + J\Theta^k J^\top\right)^{-1}\left(-Jx^k - \mu_k J(Z^k)^{-1} e + (LDU)^{-1} a + U^{-1} r + \right. \\ & \qquad \left. J\Theta^k c\right) + \mu_k (Z^k)^{-1} e - \Theta^k c \\ &= \Theta^k \left(J^\top L(D^{-1} + UJ\Theta^k J^\top L)^{-1}\left(UJ\Theta^k\left(-z^k - \mu_k (X^k)^{-1} e + c\right) + D^{-1}L^{-1}a + \right.\right. \end{aligned}$$

$$
\begin{aligned}
& r) + \mu_k (X^k)^{-1} e - c\Big) \\
= \;& \Theta^k \left(J^\top L (D^{-1} + UHW^k H^\top L)^{-1} (d + r) + \mu_k (X^k)^{-1} e - c \right) \\
= \;& \Theta^k \left(J^\top L s + \mu_k (X^k)^{-1} e - c \right).
\end{aligned}
$$

This shows the equalities (15) and (16). Finally, by combining condition (14) and the inequality $\|\bar{r}^k\| \leq \|J^\top LD\| \|r\|$ obtained from formula (19), it is immediate to conclude that the condition (11) is satisfied. $\Box$

Until the end of this section we discuss the factorization scheme of the matrix B and the iterative algorithms used to solve the linear system (12) when the matrix B is symmetric and nonsymmetric.

3.1.1 The LDU factorization

The LDU factorization of the matrix B has to be computed before the first iteration of the TN algorithm. Although the LDU factorization of a positive definite matrix exists, its numerical stability and fill-in depend on other properties the matrix may enjoy. In fact, there are some classes of matrices for which it is proved that the LDU factorization is stable. In the other cases, partial pivoting or complete pivoting schemes based on column or/and row permutations are required for the purpose of stability. These pivoting schemes are inocuous when the matrix to be factorized is dense. However, when the matrix is sparse they often destroy the sparsity structure leading in many cases into almost dense matrices L and U.

The LDU factorization of matrix B can be processed without fill-in even when a pivoting procedure is required. In fact, B is a block diagonal matrix with dense diagonal blocks if its rows and columns are ordered according to regions. So, row or/and column interchanges required for stability purposes during the factorization take place inside the diagonal blocks and, at the end of the decomposition, L and U are always block diagonal matrices with lower and upper dense triangular blocks, respectively. Unfortunately, in our implementation we cannot take advantage of this nice property of matrix B. In the next section some preconditioners for the solution of the linear system (12) are presented. As we see there, these preconditioners require a factorization $B = LDU$ where the matrices L and U are triangular. If, for example, partial pivoting is carried out during the factorization, then we obtain $B = PLDU$ where P is permutation matrix. In this case, PL is not a triangular matrix and thus useless for our preconditioning purposes. In our implementation, we compute the matrices L, D and U without making use of pivoting techniques. This procedure returns L and U matrices as sparse as the lower and upper triangle of matrix B, repectively. When B is ordered according to commodities, the matrices L and U we obtain are given by

$$
L = \begin{bmatrix}
I & & & & & \\
L_{21} & I & & & & \\
\vdots & \vdots & \ddots & & & \\
L_{g1} & L_{g2} & \cdots & I & & \\
\vdots & \vdots & & \vdots & \ddots & \\
L_{m1} & L_{m2} & \cdots & L_{mg} & \cdots & I
\end{bmatrix}
\quad \text{and} \quad
U = \begin{bmatrix}
I & U_{12} & \cdots & U_{1g} & \cdots & U_{1m} \\
 & I & \cdots & U_{2g} & \cdots & U_{2m} \\
 & & \ddots & \vdots & & \vdots \\
 & & & I & \cdots & U_{gm} \\
 & & & & \ddots & \vdots \\
 & & & & & I
\end{bmatrix},
$$

where L_{gh} and U_{gh} are $n \times n$ diagonal matrices whose diagonal entries we denote by l_{igh} and u_{igh}, respectively. To the best of our knowledge, this decomposition is stable provided the norm of the skew part of B is small relative to the condition number of its symmetric part [13]. In particular this occurs at least in the following three cases:

- $B_{gh} = B_{hg},\ g, h = 1, \ldots, m$. This means that matrix B is symmetric and $U = L^\top$.
- $B_{gh} \leq 0,\ g, h = 1, \ldots, m,\ g \neq h$. In this case, B is a Z matrix.
- $|b_{igg}| \geq \sum_{h=1, h\neq g}^{m} |b_{igh}|$ or $|b_{igg}| \geq \sum_{h=1, h\neq g}^{m} |b_{ihg}|,\ \ i = 1, \ldots, n,\ g = 1, \ldots, m$. In this case, B is a Diagonally Dominant (DD) matrix.

As we show in the computational experience presented in the last section of the paper, when the matrices L, D and U are obtained without using pivoting procedures, the TN algorithm performs nicely for all the nonsymmetric randomly generated test problems. However, it is possible that the TN algorithm behaves poorly in some contrived test problems.

3.1.2 The PCG and PQMR algorithms

The conjugate gradient algorithm [15] is considered to be the most appropriate iterative method to solve linear systems with a symmetric positive definite matrix [4]. Furthermore, the quasi minimal residual method [10] is usually recommended for linear systems with a nonsymmetric matrix whose transpose is available [4]. Since the matrix $D^{-1} + UHW^kH^\top L$ always satisfies at least one of these two requirements, we have chosen preconditioned versions of these algorithms to compute the solution of the linear system (12).

The Preconditioned Conjugate-Gradient (PCG) and the Preconditioned Quasi Minimal Residual (PQMR) algorithms attempt to find an approximate solution of the linear system

$$M_L^{-1}(D^{-1} + UHW^kH^\top L)M_R^{-1}(M_R s) \quad = \quad M_L^{-1} d, \tag{20}$$

where M_L and M_R are left and right preconditioners, respectively. The aim of preconditioning is to obtain a matrix $M_L^{-1}(D^{-1} + UHW^kH^\top L)M_R^{-1}$ with more favourable spectral properties than those of $D^{-1} + UHW^kH^\top L$. Pseudo-codes for the PCG and PQMR algorithms are presented in Figures 1 and 2, respectively. The corresponding FORTRAN codes were retrieved from the ftp site **netlib.att.com** together with the BLAS and LAPACK routines they incorporate. According to the philosofy of these libraries, we supplied the routines required to solve the linear systems with the matrices M_L, M_R, $M_L^\top$, $M_R^\top$ and M and the routines required to compute the matrix-vector products of the form $\tilde{u} = (D^{-1} + UHW^kH^\top L)\hat{u}$ and $\tilde{u} = (D^{-1} + L^\top HW^kH^\top U^\top)\hat{u}$. We modified the codes in order to change the stopping criterion of the PCG and PQMR algorithms.

The main issues of the PCG and PQMR algorithms are the stopping criterion, the computation of the matrix-vector products of the form $\tilde{u} = (D^{-1} + UHW^kH^\top L)\hat{u}$ and $\tilde{u} = (D^{-1} + L^\top HW^kH^\top U^\top)\hat{u}$, the choice of the starting point s^0, the choice of the left and right preconditioners M_L and M_R, and the solution of the linear systems with the matrices M_L, M_R, $M_L^\top$, $M_R^\top$ and M. Next we discuss these five points.

According to condition (14), if global convergence is required for the TN method then the PCG and PQMR algorithms should be stopped when

$$\|r\| \quad \leq \quad \beta_0 \frac{\|z^k - J^\top BJx^k - c + J^\top a\|}{\|J^\top LD\|}.$$

Compute $r^0 = d - (D^{-1} + UHW^kH^\top L)s^0$ for some initial guess s^0
for $i = 1, 2, \ldots$
  solve $Mz^{i-1} = r^{i-1}$, where $M = M_L M_R$
  $\rho_{i-1} = (r^{i-1})^\top z^{i-1}$
  if $i = 1$
    $p^1 = z^0$
  else
    $\beta_{i-1} = \rho_{i-1}/\rho_{i-2}$
    $p^i = z^{i-1} + \beta_{i-1}p^{i-1}$
  endif
  $q^i = (D^{-1} + UHW^kH^\top L)p^i$
  $\alpha_i = \rho_{i-1}/(p^i)^\top q^i$
  $s^i = s^{i-1} + \alpha_i p^i$
  $r^i = r^{i-1} - \alpha_i q^i$
  Test stopping criterion; continue if necessary
end

Figure 1: Preconditioned conjugate gradient algorithm.

Here, r is the exact residual vector at the i-th iteration of the PCG and PQMR algorithms given by

$$r \quad = \quad d - (D^{-1} + UHW^kH^\top L)s^i$$

and (x^k, z^k) is the interior-point solution at the k-th iteration of the TN method. The vector r is expensive to compute at each PCG/PQMR iteration. However, the estimate residual r^i given by the algorithms is approximately equal to r if the iteration number i is not too big [25] and the PCG and PQMR methods usually find a good approximate solution in few iterations. So, it is reasonable to stop the iterative solvers when

$$\|r^i\|_1 \quad \leq \quad \beta_0 \frac{\|z^k - J^\top BJx^k - c + J^\top a\|_1}{\|J^\top LD\|_1} \tag{21}$$

is satisfied. The quantity $\|J^\top LD\|_1$ and the vector $J^\top a - c$ are only computed once before the first iteration of the TN algorithm. The computation of $\|z^k - J^\top BJx^k + (J^\top a - c)\|_1$ is carried out at each iteration of the TN algorithm and takes $O(mn^2 + m^2n)$ aditions and $O(m^2n)$ multiplications.

The matrix by vector products of the form $\tilde{u} = (D^{-1} + UHW^kH^\top L)\hat{u}$ and $\tilde{u} = (D^{-1} + L^\top HW^kH^\top U^\top)\hat{u}$ required by the PCG and PQMR algorithms are computed by calling **product**$(L, U, \tilde{u}, \hat{u})$ and **product**$(U^\top, L^\top, \tilde{u}, \hat{u})$, respectively (see Figure 3). In Figure 3, W_g^k denotes the the diagonal matrix of order $n(n-1)/2$ with diagonal elements w_{ijg}^k for all $i, j = 1, \ldots, n$ and $i < j$. In the procedure **product** the multiplication $G_{\cdot\bar{\mathcal{H}}_1} W_g^k G_{\cdot\bar{\mathcal{H}}_1}^\top aux_1$ is computed by employing the technique presented in [26] to compute the product of a node-arc

Compute $r^0 = d - (D^{-1} + UHW^kH^\top L)s^0$ for some initial guess s^0

$\tilde{v}^1 = r^0$; **solve** $M_L y = \tilde{v}^1$; $\rho_1 = \|y\|_2$

$\tilde{w}^1 = r^0$; **solve** $M_R^\top z = \tilde{w}^1$; $\xi_1 = \|z\|_2$

$\gamma_0 = 1$; $\eta_0 = -1$

for $i = 1, 2, \ldots$

    if $\rho_i = 0$ or $\xi_i = 0$ **method fails**

    $v^i = \tilde{v}^i/\rho_i$; $y = y/\rho_i$

    $w^i = \tilde{w}^i/\xi_i$; $z = z/\xi_i$

    $\delta_i = z^\top y$; **if** $\delta_i = 0$ **method fails**

    solve $M_R \tilde{y} = y$

    solve $M_L^\top \tilde{z} = z$

    if $i = 1$

        $p^1 = \tilde{y}$; $q^1 = \tilde{z}$

    else

        $p^i = \tilde{y} - (\xi_i \delta_i/\varepsilon_{i-1})p^{i-1}$

        $q^i = \tilde{z} - (\rho_i \delta_i/\varepsilon_{i-1})q^{i-1}$

    endif

    $\tilde{p} = (D^{-1} + UHW^kH^\top L)p^i$

    $\varepsilon_i = (q^i)^\top \tilde{p}$; **if** $\varepsilon_i = 0$ **method fails**

    $\beta_i = \varepsilon_i/\delta_i$; **if** $\beta_i = 0$ **method fails**

    $\tilde{v}^{i+1} = \tilde{p} - \beta_i v^i$

    solve $M_L y = \tilde{v}^{i+1}$

    $\rho_{i+1} = \|y\|_2$

    $\tilde{w}^{i+1} = (D^{-1} + L^\top HW^kH^\top U^\top)q^i - \beta_i w^i$

    solve $M_R^\top z = \tilde{w}^{i+1}$

    $\xi_{i+1} = \|z\|_2$

    $\theta_i = \rho_{i+1}/(\gamma_{i-1}|\beta_i|)$; $\gamma_i = 1/\sqrt{1+\theta_i^2}$; **if** $\gamma_i = 0$ **method fails**

    $\eta_i = -\eta_{i-1}\rho_i\gamma_i^2/(\beta_i\gamma_{i-1}^2)$

    if $i = 1$

        $t^1 = \eta_1 p^1$; $\tilde{t}^1 = \eta_1 \tilde{p}$

    else

        $t^i = \eta_i p^i + (\theta_{i-1}\gamma_i)^2 t^{i-1}$; $\tilde{t}^i = \eta_i \tilde{p} + (\theta_{i-1}\gamma_i)^2 \tilde{t}^{i-1}$

    endif

    $s^i = s^{i-1} + t^i$

    $r^i = r^{i-1} - \tilde{t}^i$

    Test stopping criterion; continue if necessary

end

Figure 2: Preconditioned quasi minimal residual algorithm.

Procedure **product**$(V^a, V^b, \tilde{u}, \hat{u})$
for $g = 1, \ldots, m$ **do**
$\quad aux_1 \leftarrow \hat{u}_g$, $aux_2 \leftarrow 0$
$\quad$ **for** $h = 1, \ldots, g-1$ **do** $aux_1 \leftarrow aux_1 + V^a_{gh}\hat{u}_h$
$\quad aux_2 \leftarrow G_{\cdot\bar{\mathcal{H}}_1} W^k_g G^\top_{\cdot\bar{\mathcal{H}}_1} aux_1$
$\quad \tilde{u}_g \leftarrow aux_2 + D_g^{-1}\hat{u}_g$
$\quad$ **for** $h = 1, \ldots, g-1$ **do** $\tilde{u}_h \leftarrow \tilde{u}_h + V^b_{hg} aux_2$

Figure 3: Procedure **product**.

incidence matrix by a vector. The procedure **product** uses the auxilliary vectors $aux_1, aux_2 \in \mathbb{R}^n$ and takes $O(mn^2 + m^2n)$ sums and multiplications to compute $\tilde{u}$. In each iteration of the PCG and PQMR algorithms, the procedure **product** is called once and twice respectively.

We use as initial estimate s^0 for the solution of the linear system (12) the vector s^i obtained in the last iteration of the previous call of the PCG/PQMR algorithm. This is done with the expectation that W^k and d suffer a small change between consecutive iterations of the TN algorithm and, consequently, the solution of the reduced system at iteration k of this interior-point method is close to the solution of the reduced system at iteration $k-1$. Extensive computational experience has indicated that this strategy is better than other alternative choices such as $y^0 = (M_L M_R)^{-1} d$ and $y^0 = 0$.

The preconditioning techniques should be chosen so that left and right preconditioners are cheap to compute, linear systems with the preconditioning matrices are easy to solve and, the most important, the condition number of the preconditioned system is smaller than the condition number of the original system. The incomplete factorization techniques attempt to find preconditioners enjoying these properties. In fact, by applying these procedures allowing an appropriate degree of fill-in during the factorization process, it is expected to find right and left preconditioners close to the matrices L and U belonging to the LU decomposition of the matrix of the original system. In the case of the reduced system (12), we must choose a preconditioning technique that is able to manipulate the matrices D, U, L, H and W^k separately, since the cost of forming $D^{-1} + UHW^kH^\top L$ is prohibitive. Furthermore, when B is symmetric the preconditioning technique must compute matrices M_L and M_R verifying $M_L = M_R^\top$, as the PCG algorithm requires a symmetric positive definite matrix $M_L^{-1}(D^{-1} + UHW^kH^\top L)M_R^{-1}$.

We employ an Incomplete QR Decomposition (IQRD) technique to compute the preconditioning matrices M_L, M_R and M. Suppose we wish to compute the QR factorization of a $m \times n$ sparse full column rank matrix A. This decomposition can be obtained by multiplying A by a set of Givens matrices of order m [11]. In this procedure, some of the elements of matrix A, together with the nonzero entries corresponding to the intermediate fill-in are turned to zero. The i-th elimination of a nonzero element corresponds to multiply matrix Q^i by $Q^{i-1}Q^{i-2}\ldots Q^2Q^1A$, where Q^i, $i = 1, 2, \ldots$ are Givens matrices. After the last element is turned to zero, we obtain

$$\overbrace{\dots Q^{i+1}Q^{i}Q^{i-1}\dots Q^{2}Q^{1}}^{Q} A = \left[\begin{array}{c} R \\ 0_{(m-n)\times n} \end{array} \right],$$

where $Q \in \mathbb{R}^{m\times m}$ and $R \in \mathbb{R}^{n\times n}$ are orthogonal and triangular matrices, respectively. In the incomplete QR decompositions involved in our preconditioning technique, we perform a set of plane rotations to eliminate the elements of matrix A outside some sparsity pattern previously chosen for the matrix R. When the multiplication $Q^{i}(Q^{i-1}Q^{i-2}\dots Q^{2}Q^{1}A)$ is carried out, the element to be eliminated is the unique element allowed to be transformed among all the elements of matrix $Q^{i-1}Q^{i-2}\dots Q^{2}Q^{1}A$ outside the sparsity structure of R. One of the consequences of this rule is that no fill-in occurs outside the nonzero pattern chosen to matrix R in every stage of the factorization. Furthermore, the product $\dots Q^{i+1}Q^{i}Q^{i-1}\dots Q^{2}Q^{1}$ corresponding to the set of plane rotations performed in the procedure is not equal to the orthogonal matrix Q associated to the exact QR factorization of matrix A. However, since Q^{i}, $i = 1, 2, \dots$ are Givens matrices this product still produces an orthogonal matrix Q.

The incomplete IQRD technique has three variants which compute different left and right preconditioners. At each iteration of the TN algorithm, we apply one of these variants to compute the matrix M and the matrices M_L and M_R respectively in the symmetric and nonsymmetric cases. The choice of a variant depends on the number of the current iteration. We consider three phases in the IQRD technique. The first two phases differ from variant to variant while Phase III is the same in all the three variants. Next a general descripton of this technique is presented and then the variants are described in detail.

Let the sets $\bar{\mathcal{H}}_1$ and $\mathcal{H}_1$ be defined as in Proposition 1. Furthermore, let $\bar{\mathcal{S}}$ denote the subset of $\mathcal{H}_1$ corresponding to the indices of the diagonal elements of W^k with a value close to zero and $\mathcal{S} = \mathcal{H}_1 - \bar{\mathcal{S}}$. In Phase I we split UHW^kHL into

$$UH_{\cdot\mathcal{S}}W^k_{\mathcal{S}}H^{\top}_{\cdot\mathcal{S}}L + UH_{\cdot\bar{\mathcal{S}}}W^k_{\bar{\mathcal{S}}}H^{\top}_{\cdot\bar{\mathcal{S}}}L,$$

and approximate UHW^kHL by $UH_{\cdot\mathcal{S}}W^k_{\mathcal{S}}H^{\top}_{\cdot\mathcal{S}}L$. As the matrices H and W^k are both block diagonal, H has m identical diagonal blocks $G_{\cdot\bar{\mathcal{H}}_1}$ and W^k has diagonal blocks W^k_g, $g = 1, \dots, m$, then UHW^kHL is a block diagonal matrix with diagonal blocks $G_{\cdot\bar{\mathcal{H}}_1}W^k_g G^{\top}_{\cdot\bar{\mathcal{H}}_1}$, $g = 1, \dots, m$. Then, we associate a subset of arcs (say $\mathcal{T}_g$) to each commodity g and set $\mathcal{S} = \cup_{g=1}^{m}\mathcal{S}_g$, where $\mathcal{S}_g = \{(i,j,g) \in \mathcal{S} : (i,j) \in \mathcal{T}_g\}$. So after Phase I is carried out we get a matrix of the form

$$H_{\cdot\mathcal{S}}W^k_{\mathcal{S}}H^{\top}_{\cdot\mathcal{S}} = \left[\begin{array}{ccccc} G_{\cdot\mathcal{T}_1}W^k_{\mathcal{S}_1}G^{\top}_{\cdot\mathcal{T}_1} & & & & \\ & \ddots & & & \\ & & G_{\cdot\mathcal{T}_g}W^k_{\mathcal{S}_g}G^{\top}_{\cdot\mathcal{T}_g} & & \\ & & & \ddots & \\ & & & & G_{\cdot\mathcal{T}_m}W^k_{\mathcal{S}_m}G^{\top}_{\cdot\mathcal{T}_m} \end{array} \right].$$

The effectiveness of incorporating this operation in the construction of the preconditioners depends on the number of the current iteration of the TN method. During the initial and intermediate iterations it is appropriate to set $\mathcal{S} = \mathcal{H}_1$ ($\mathcal{T}_g = \bar{\mathcal{H}}_1$ for all $g = 1, \dots, m$), since at these stages of the TN algorithm there are no diagonal elements of W^k whose value is reasonable to neglect. However, in the final iterations the set $\mathcal{S}$ should be chosen close to the subset of the indices of the diagonal elements of W^k converging to a positive value as $k \to \infty$. In this case, we choose $\mathcal{T}_g$ such that $\Psi_{\mathcal{T}_g} = (\mathcal{N}, \mathcal{T}_g)$ is a maximum spanning tree of the graph $\Psi_{\bar{\mathcal{H}}_1} = (\mathcal{N}, \bar{\mathcal{H}}_1)$. In this operation, $\mathcal{T}_g$ is identified using as weights of the arcs the values of the corresponding elements of the diagonal matrix W_g^k. The reason for this choice is that there exists at least one solution of the economic equilibrium linear complementarity problem where, for each commodity g, these elements of $\mathcal{H}_1$ are represented by a tree (forest) (see Theorem 8). The first choice for the set $\mathcal{S}$ is implemented in the variants 2 and 3 of the IQRD technique. The second alternative is implemented in the Variant 1. So, Variant 1 is used in the final iterations of the TN algorithm while Variants 2 and 3 are employed in the initial and intermediate iterations of this interior-point method. These three variants are discussed later in this section.

In Phase II a factorization of each diagonal block of the matrix $H_{\cdot\mathcal{S}} W_{\mathcal{S}}^k H_{\cdot\mathcal{S}}^\top$ is performed. To explain this procedure, we first note that each diagonal block $G_{\cdot\mathcal{T}_g} W_{\mathcal{S}_g}^k G_{\cdot\mathcal{T}_g}^\top$ can be written as $\bar{R}_g^\top \bar{R}_g$, where

$$\bar{R}_g = \begin{bmatrix} 0_{n\times n} \\ (W_{\mathcal{S}_g}^k)^{\frac{1}{2}} G_{\cdot\mathcal{T}_g}^\top \end{bmatrix}.$$

Hence we can apply m orthogonal decompositions to get

$$Q_g \bar{R}_g \quad \approx (=) \quad \begin{bmatrix} R_g \\ 0_{|\mathcal{T}_g|\times n} \end{bmatrix} \quad g = 1, \ldots, m. \tag{22}$$

The symbol $\approx (=)$ means that the factorizations may be exact or approximate depending on the choice of the variant of the IQRD technique. If R, $\bar{R}$ and Q are block diagonal matrices with diagonal blocks R_g, $\bar{R}_g$ and Q_g, $g = 1, \ldots, m$, respectively, then

$$D^{-1} + UH_{\cdot\mathcal{S}} W_{\mathcal{S}}^k H_{\cdot\mathcal{S}}^\top L = D^{-1} + U\bar{R}^\top Q^\top Q \bar{R} L.$$

Since Q is orthogonal, $Q^\top Q = I$ and we have

$$D^{-1} + UH_{\cdot\mathcal{S}} W_{\mathcal{S}}^k H_{\cdot\mathcal{S}}^\top L \approx (=) D^{-1} + UR^\top RL.$$

In [27, 31, 29, 30, 34] the choice of an appropriate sparsity structure for the preconditioners of a matrix of the form $(W_{\mathcal{S}_g}^k)^{\frac{1}{2}} G_{\cdot\mathcal{T}_g}^\top$ is discussed in the context of linear network flow programming. All these computational studies lead to the same conclusions. In the initial iterations of the interior-point methods a diagonal sparsity pattern is recommended for the preconditioning matrix R_g. In the itermediate and final stages, the sparsity pattern chosen to R_g should reflect the structure of the graph representing the routes of positive flow of the solution the interior-point algorithms converge to. Since there exists at least one solution of the economic equilibrium linear complementarity problem where the trade routes of positive flow are represented by a tree (forest) (see Theorem 8), the triangular sparsity pattern associated with the maximum spanning tree $\Psi_{\mathcal{T}_g} = (\mathcal{N}, \mathcal{T}_g)$ defined in Phase I is chosen for the matrix R_g in the intermediate and last iterations of the TN algorithm. The first choice for the sparsity pattern of the matrices R_g, $g = 1, \ldots, m$, is implemented in the Variant 2 of the IQRD technique. The second alternative is implemented in variants 1 and 3. Thus, Variant 1 is used in the final iterations of the TN algorithm while variants 2 and 3 are employed respectively in the initial and intermediate iterations of this interior-point method.

Phase III is described in Figure 4 and consists of transforming the matrix $D^{-1}+UR^\top RL$ as follows:

$$D^{-1}+UR^\top RL = \begin{bmatrix} UR^\top & D^{-\frac{1}{2}} \end{bmatrix} \overbrace{P^\top Q_L^\top Q_R P}^{I} \begin{bmatrix} RL \\ D^{-\frac{1}{2}} \end{bmatrix} \approx M_L M_R.$$

Here, P is a permutation matrix and Q_L and Q_R are block diagonal matrices with diagonal blocks $\bar{Q}_g$, $g=1,\ldots,m$ (see Figure 4).

If the rule described above for the incomplete QR factorizations is followed when decompositions (23) are carried out, then $Q_L^\top Q_R = I$. In fact, the incomplete decompositions (23) are computed employing the same procedure in all the three variants of the preconditioning technique. We apply Givens rotations to eliminate the diagonal elements of matrix $D_g^{-\frac{1}{2}}$ using the diagonal elements of matrix R_g. If B is symmetric Phase III is only required to be processed once as $U^\top = L$. So, it is obvious that the equalities $M_L = M_R^\top$ and $Q_L^\top Q_R = I$ hold in this case. Next we show that the equality $Q_L^\top Q_R = I$ also holds when B is nonsymmetric.

Theorem 5. $Q_L^\top Q_R = I$.

Proof. Since no fill-in is allowed in the decompositions (23) and the diagonal elements of $RU^\top$ and RL are both equal to the diagonal entries of R, the product $Q_L^\top Q_R$ can be written in the form

$$Q_L^\top Q_R \;=\; \prod_{g=1}^{m}\prod_{i=1}^{n}(Q_L^{gi})^\top \prod_{g=m}^{1}\prod_{i=n}^{1} Q_R^{gi}.$$

Here Q_L^{gi} and Q_R^{gi}, $g=1,\ldots,m$, $i=1,\ldots,n$, are Givens matrices given by

$$Q_L^{gi} = Q_R^{gi} = \begin{bmatrix} I_{(g-1)n+i-1} & & & & \\ & \frac{r_{iig}}{\sqrt{d_{ig}^{-1}+r_{iig}^2}} & & \frac{\sqrt{d_{ig}^{-1}}}{\sqrt{d_{ig}^{-1}+r_{iig}^2}} & \\ & & I_{mn-1} & & \\ & -\frac{\sqrt{d_{ig}^{-1}}}{\sqrt{d_{ig}^{-1}+r_{iig}^2}} & & \frac{r_{iig}}{\sqrt{d_{ig}^{-1}+r_{iig}^2}} & \\ & & & & I_{mn-(g-1)n-i} \end{bmatrix}$$

where d_{ig} denotes the i-th diagonal element of matrix D_g and r_{ijg} represents the element of matrix R_g with row and column indices i and j, respectively. This shows that $(Q_L^{gi})^\top Q_R^{gi} = I$ for all $g=1,\ldots,m$ and $i=1,\ldots,n$ and thus proves that the equality $Q_L^\top Q_R = I$ holds. □

It follows from the description of the implementation of the TN algorithm that the orthogonal matrices are not stored in any of three phases of the IQRD preconditioning technique. Furthermore it is possible to find explicit formulas for the elements of M_L, M_R and M in terms of the elements of the matrices $\bar{D}_g$ $\bar{F}_g$ and $\hat{F}_g$ defined in the next theorem.

Theorem 6. Let $\bar{D}$ be a $mn \times mn$ diagonal matrix with diagonal elements given by $\bar{d}_{ig} = d_{ig}^{-1} + r_{iig}^2$, $g=1,\ldots,m$, $i=1,\ldots,n$. Furthermore, let $\bar{F}_g$ and $\hat{F}_g$ be two matrices having the same off-diagonal sparsity structure of R_g and elements in row i and column j given by

$$\bar{f}_{ijg} = r_{iig}r_{ijg}/\bar{d}_{ig} \quad \text{and} \quad \hat{f}_{ijg} = \begin{cases} 1 & \text{if} \quad i=j \\ r_{iig}r_{ijg}/\bar{d}_{ig} & \text{if} \quad i \neq j, \end{cases}$$

For $V = L$ (and $V = U^\top$ when B is nonsymmetric) **do**

1. Let P be a permutation matrix such that

$$P \begin{bmatrix} RV \\ D^{-\frac{1}{2}} \end{bmatrix} = \begin{bmatrix} R_1 & & & & & \\ D_1^{-\frac{1}{2}} & & & & & \\ R_2V_{21} & R_2 & & & & \\ 0 & D_2^{-\frac{1}{2}} & & & & \\ \vdots & \vdots & \ddots & & & \\ R_gV_{g1} & R_gV_{g2} & \cdots & R_g & & \\ 0 & 0 & \cdots & D_g^{-\frac{1}{2}} & & \\ \vdots & \vdots & & \vdots & \ddots & \\ R_mV_{m1} & R_mV_{m2} & \cdots & R_mV_{mg} & \cdots & R_m \\ 0 & 0 & \cdots & 0 & \cdots & D_m^{-\frac{1}{2}} \end{bmatrix}.$$

   Compute the matrices $\hat{R}_{gh}$, $g = 1, \ldots, m$, $h = 1, \ldots, g$, from the set of m incomplete orthogonal factorizations

$$\bar{Q}_g \begin{bmatrix} R_gV_{g1} & \cdots & R_gV_{gh} & \cdots & R_g & 0 & \cdots & 0 \\ 0 & \cdots & 0 & \cdots & D_g^{-\frac{1}{2}} & 0 & \cdots & 0 \end{bmatrix} = \begin{bmatrix} \hat{R}_{g1} & \cdots & \hat{R}_{gh} & \cdots & \hat{R}_{gg} & 0 & \cdots & 0 \\ 0 & \cdots & 0 & \cdots & 0 & 0 & \cdots & 0 \end{bmatrix} \quad g = 1, \ldots, m. \tag{23}$$

   Here, $\hat{R}_{gh}$, $h = 1, \ldots, g-1$, have the same sparsity structure of R_g and $\hat{R}_{gg}$ has nonzero diagonal elements and the same off-diagonal sparsity structure of R_g.

2. Let $\hat{R}$ be a matrix given by

$$\hat{R} = \begin{bmatrix} \hat{R}_{11} & & & & & \\ \hat{R}_{21} & \hat{R}_{22} & & & & \\ \vdots & \vdots & \ddots & & & \\ \hat{R}_{g1} & \hat{R}_{g2} & \cdots & \hat{R}_{gg} & & \\ \vdots & \vdots & & \vdots & \ddots & \\ \hat{R}_{m1} & \hat{R}_{m2} & \cdots & \hat{R}_{mg} & \cdots & \hat{R}_{mm} \end{bmatrix}.$$

   If B is symmetric set $M_L = \hat{R}^\top$ and $M_R = \hat{R}$.
   If B is nonsymmetric set $M_L = \hat{R}^\top$ if $V = U^\top$ and $M_R = \hat{R}$ if $V = L$.

Figure 4: Phase III of the IQRD preconditioning technique.

respectively. Then $M_L = F_L^\top \bar{D}^{\frac{1}{2}}$, $M_R = \bar{D}^{\frac{1}{2}} F_R$ and $M = F_L^\top \bar{D} F_R$, where F_L and F_R are equal to

$$\begin{bmatrix} \hat{F}_1 & & & & & \\ \bar{F}_2 V_{21} & \hat{F}_2 & & & & \\ \vdots & \vdots & \ddots & & & \\ \bar{F}_g V_{g1} & \bar{F}_g V_{g2} & \cdots & \hat{F}_g & & \\ \vdots & \vdots & & \vdots & \ddots & \\ \bar{F}_m V_{m1} & \bar{F}_m V_{m2} & \cdots & \bar{F}_m V_{mg} & \cdots & \hat{F}_m \end{bmatrix}$$

respectively for $V = U^\top$ and $V = L$.

Proof. The proof follows from the observation that in factorizations (23) the element (i,j) of matrix $\hat{R}_{gh}$ is the first element of the vector that results from the multiplication

$$\begin{bmatrix} \frac{r_{iig}}{\sqrt{d_{ig}^{-1} + r_{iig}^2}} & \frac{\sqrt{d_{ig}^{-1}}}{\sqrt{d_{ig}^{-1} + r_{iig}^2}} \\ -\frac{\sqrt{d_{ig}^{-1}}}{\sqrt{d_{ig}^{-1} + r_{iig}^2}} & \frac{r_{iig}}{\sqrt{d_{ig}^{-1} + r_{iig}^2}} \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix}$$

with

$$\lambda_1 = \begin{cases} r_{ijg} & \text{if} \quad g = h \\ r_{ijg} v_{jgh} & \text{if} \quad g > h \\ 0 & \text{if} \quad g < h \end{cases} \quad \text{and} \quad \lambda_2 = \begin{cases} \sqrt{d_{ig}^{-1}} & \text{if} \quad g = h \quad \text{and} \quad i = j \\ 0 & \text{otherwise.} \end{cases}$$

Here, v_{igh} is the i-th diagonal element of the block V_{gh} of the triangular matrix V. □

Next we show that the IQRD preconditioning technique computes nonsingular matrices M_L and M_R.

Corollary 7. M_L and M_R are nonsigular matrices.

Proof. The proof follows immediately from the fact that $\bar{D}_g^{\frac{1}{2}}$ and $\hat{F}_g$, $g = 1, \ldots, m$, are diagonal or triangular matrices with nonzero diagonal elements. □

As stated before, the two first phases of the IQRD preconditioning technique can be performed by using one of the three possible variants. These procedures differ on the choice of the sets $\mathcal{T}_g$, $g = 1, \ldots, m$ that define the splitting carried out in Phase I and on the choice of the sparsity structure for the matrices R_g, $g = 1, \ldots, m$ in Phase II. By using Theorem 6 it is also possible to give explicit formulas for the elements of the matrices F and $\bar{D}$ that are used in the definition of the preconditioners M_L, M_R and M. Next we discuss the variants and present these formulas in each one of the three cases. To simplify the explanation we ommit the index k and denote by w_{ijg} the diagonal element of matrix W^k associated to arc (i,j) and commodity g.

Variant 1. This variant is employed in the final iterations of the TN algorithm. So, $\mathcal{T}_g$ is chosen such that $\Psi_{\mathcal{T}_g} = (\mathcal{N}, \mathcal{T}_g)$, $g = 1, \ldots, m$, are maximum spanning trees of the graph

$\Psi_{\bar{\mathcal{H}}_1} = (\mathcal{N}, \bar{\mathcal{H}}_1)$. Furthermore, for each $g = 1, \dots, m$, R_g has a triangular sparsity pattern associated to the spanning tree $\Psi_{\mathcal{T}_g} = (\mathcal{N}, \mathcal{T}_g)$. The set $\mathcal{T}_g$ is identified using as weights of the arcs the values of the corresponding elements of the diagonal matrix W_g^k. Suppose that $\Psi_{\mathcal{T}_g} = (\mathcal{N}, \mathcal{T}_g)$ is rooted at node ρ_g and, for each $i \in \mathcal{N} - \{\rho_g\}$, let α_{ig} and β_{ig} be two nodes defined by $\alpha_{ig} = min\{i, pred_{ig}\}$ and $\beta_{ig} = max\{i, pred_{ig}\}$, where $pred_{ig}$ represents the predecessor node of node i in the spanning tree $\Psi_{\mathcal{T}_g} = (\mathcal{N}, \mathcal{T}_g)$. Furthermore, suppose that the column associated to the arc $(\alpha_{ig}, \beta_{ig})$ is the γ_{ig}-th column of matrix $G_{\cdot\mathcal{T}_g}$. In Phase II, factorizations (22) are exact and

$$Q_g = \begin{bmatrix} 0_{n\times(n-1)} & \hat{Q}_g\hat{P}_g \\ I_{n-1} & 0_{(n-1)\times n} \end{bmatrix},$$

where $\hat{Q}_g$ is a $n \times n$ diagonal matrix with diagonal elements given by

$$\hat{Q}_{ig} = \begin{cases} 1 & \text{if } i > pred_{ig} \text{ or } i = \rho_g \\ -1 & \text{if } i < pred_{ig}, \end{cases} \tag{24}$$

and $\hat{P}_g$ is a $n \times n$ permutation matrix with unitary elements in positions $(\rho_g, 1)$ and $(i, \gamma_{ig}+1)$, $i = 1, \dots, n$, $i \neq \rho_g$. The elements of matrix R_g are in this case given by

$$r_{ijg} = \begin{cases} \sqrt{w_{\alpha_{ig}\beta_{ig}g}} & \text{if } i = j \text{ and } i \neq \rho_g \\ -\sqrt{w_{\alpha_{ig}\beta_{ig}g}} & \text{if } j = pred_{ig} \\ 0 & \text{otherwise.} \end{cases}$$

So, by using Theorem 6 we may derive the following formulas to compute the elements of matrices $\bar{D}_g$, $\bar{F}_g$ and $\hat{F}_g$:

$$\bar{d}_{ig} = \begin{cases} d_{ig}^{-1} & \text{if } i = \rho_g \\ d_{ig}^{-1} + w_{\alpha_{ig}\beta_{ig}g} & \text{if } i \neq \rho_g, \end{cases}$$

$$\bar{f}_{ijg} = \begin{cases} w_{\alpha_{ig}\beta_{ig}g}/\bar{d}_{ig} & \text{if } i = j \text{ and } i \neq \rho_g \\ -w_{\alpha_{ig}\beta_{ig}g}/\bar{d}_{ig} & \text{if } j = pred_{ig} \\ 0 & \text{otherwise,} \end{cases}$$

and

$$\hat{f}_{ijg} = \begin{cases} 1 & \text{if } i = j \\ -w_{\alpha_{ig}\beta_{ig}g}/\bar{d}_{ig} & \text{if } j = pred_{ig} \\ 0 & \text{otherwise.} \end{cases}$$

This variant of the IQRD preconditioning technique requires two vectors of order mn and $O(mn)$ aditions and divisions respectively to store and compute the elements of matrices $\bar{D}_g$, $\bar{F}_g$ and $\hat{F}_g$, $g = 1, \dots, m$, different from one and zero. Additionally, Variant 1 requires the computation of m spanning trees. This task is carried out using the Fibonacci heap variant of Prim's algorithm [28] described in [1]. This implementation of Prim's algorithm finds a maximum spanning tree in $O(n(n-1)/2 + nlog(n))$ time.

Variant 2. This variant of the IQRD technique is used in the initial iterations of the TN method. In this variant we set $\mathcal{T}_g = \bar{\mathcal{H}}_1$ and $\bar{\mathcal{T}}_g = \emptyset$ for all $g = 1, \ldots, m$. Furthermore, in Phase II the matrices R_g, $g = 1, \ldots, m$, are chosen to be diagonal. Suppose that the column associated to the arc (i, j) is the γ_{ij}-th column of matrix $G_{\bar{\mathcal{H}}_1}$. Factorizations (22) are incomplete and Q_g is given by

$$Q_g = \prod_{(i,j)\in\bar{\mathcal{H}}_1} Q_g^{ija} Q_g^{ijb} = \prod_{i=1}^{n} \prod_{j=i+1}^{n} Q_g^{ija} Q_g^{ijb}$$

where

$$Q_g^{ija} = \begin{bmatrix} I_{i-1} & & & & \\ & \frac{\lambda_1}{\sqrt{w_{ijg}+\lambda_1^2}} & & \frac{-\sqrt{w_{ijg}}}{\sqrt{w_{ijg}+\lambda_1^2}} & \\ & & I_{n-i+\gamma_{ij}-1} & & \\ & \frac{\sqrt{w_{ijg}}}{\sqrt{w_{ijg}+\lambda_1^2}} & & \frac{\lambda_1}{\sqrt{w_{ijg}+\lambda_1^2}} & \\ & & & & I_{n(n-1)/2-\gamma_{ij}} \end{bmatrix} \tag{25}$$

and

$$Q_g^{ijb} = \begin{bmatrix} I_{j-1} & & & & \\ & \frac{\lambda_2}{\sqrt{w_{ijg}+\lambda_2^2}} & & \frac{\sqrt{w_{ijg}}}{\sqrt{w_{ijg}+\lambda_2^2}} & \\ & & I_{n-j+\gamma_{ij}-1} & & \\ & \frac{-\sqrt{w_{ijg}}}{\sqrt{w_{ijg}+\lambda_2^2}} & & \frac{\lambda_2}{\sqrt{w_{ijg}+\lambda_2^2}} & \\ & & & & I_{n(n-1)/2-\gamma_{ij}} \end{bmatrix}, \tag{26}$$

with λ_1 and λ_2 denoting respectively the elements (i, i) and (j, j) of the matrix

$$Q_g^{i(j-1)a} Q_g^{i(j-1)b} \ldots Q_g^{i(i+1)a} Q_g^{i(i+1)b} \ldots\ldots\ldots Q_g^{1na} Q_g^{1nb} \ldots Q_g^{12a} Q_g^{12b} \bar{R}_g.$$

After carrying out the incomplete factorizations (22), we obtain diagonal matrices R_g, $g = 1, \ldots, m$, with diagonal elements given by

$$r_{iig} \;=\; \sqrt{\sum_{j=1}^{i-1} w_{jig} + \sum_{j=i+1}^{n} w_{ijg}}.$$

In this case, the formulas derived from Theorem 6 to compute the diagonal entries of the diagonal matrices $\bar{D}_g$, $\bar{F}_g$ and $\hat{F}_g$ are

$$\bar{d}_{ig} = d_{ig}^{-1} + \sum_{j=1}^{i-1} w_{jig} + \sum_{j=i+1}^{n} w_{ijg}, \quad \bar{f}_{iig} = (\textstyle\sum_{j=1}^{i-1} w_{jig} + \sum_{j=i+1}^{n} w_{ijg})/\bar{d}_{ig} \quad \text{and} \quad \hat{f}_{iig} = 1.$$

This variant of the IQRD technique requires two vectors of order mn to store the entries of matrices $\bar{D}_g$, $\bar{F}_g$ and $\hat{F}_g$, $g = 1, \ldots, m$ different from one and zero. Furthermore, these matrices are computed by performing $O(mn)$ divisions and $O(mn^2)$ additions.

Variant 3. This variant is employed in the intermediate iterations of the TN algorithm. As in Variant 2, we set $\mathcal{T}_g = \bar{\mathcal{H}}_1$ and $\bar{\mathcal{T}}_g = \emptyset$ for all $g = 1, \ldots, m$. Furthermore, R is chosen to be a matrix with nonzero diagonal elements and the same off-diagonal sparsity pattern of the matrix R presented in Variant 1. Since the sets $\mathcal{T}_g$, $g = 1, \ldots, m$ are already used to define the approximation made in Phase I, the maximum spanning tree that represents the sparsity pattern of matrix R_g is denoted in this case by $\Psi_{\hat{\mathcal{T}}_g} = (\mathcal{N}, \hat{\mathcal{T}}_g)$, where $\hat{\mathcal{T}}_g$ is equal to the set of arcs $\mathcal{T}_g$ defined in Variant 1. Again, suppose that $\Psi_{\hat{\mathcal{T}}_g} = (\mathcal{N}, \hat{\mathcal{T}}_g)$ is rooted at node ρ_g and, for each $i \in \mathcal{N} - \{\rho_g\}$, let α_{ig} and β_{ig} be two nodes defined by $\alpha_{ig} = min\{i, pred_{ig}\}$ and $\beta_{ig} = max\{i, pred_{ig}\}$, where $pred_{ig}$ represents the predecessor node of node i in $\Psi_{\hat{\mathcal{T}}_g} = (\mathcal{N}, \hat{\mathcal{T}}_g)$. Furthermore, let $Succ_{ig}$ represent the set of succesor nodes of node i in the tree $\Psi_{\hat{\mathcal{T}}_g} = (\mathcal{N}, \hat{\mathcal{T}}_g)$ and suppose that the column associated to arc (i,j) is the γ_{ij}-th column of matrix of matrix $G_{\cdot\bar{\mathcal{H}}_1}$. In Phase II the orthogonal matrix Q_g is given by

$$Q_g = \left(\prod_{(i,j) \in \{\bar{\mathcal{H}}_1 - \hat{\mathcal{T}}_g\}} Q_g^{ija} Q_g^{ijb} \right) \begin{bmatrix} \hat{Q}_g & \\ & I_{n(n-1)/2} \end{bmatrix} \hat{P}_g,$$

where $\hat{P}_g$ is a permutation matrix of order $n(n+1)/2$, $\hat{Q}_g$ is the diagonal matrix defined in (24) and Q_g^{ija} and Q_g^{ijb} are Givens matrices. The product

$$\begin{bmatrix} \hat{Q}_g & \\ & I_{n(n-1)} \end{bmatrix} \hat{P}_g \bar{R}_g$$

is computed exactly and the permutation matrix $\hat{P}_g$ has the unitary elements in positions

- (ρ_g, ρ_g)
- $(i, n + \gamma_{\alpha_{ig}\beta_{ig}}), i \in \mathcal{N} - \{\rho_g\}$
- $(n + \gamma_{ij}, n + \gamma_{ij}), (i,j) \in \{\bar{\mathcal{H}}_1 - \hat{\mathcal{T}}_g\}$
- $(n + \gamma_{\alpha_{ig}\beta_{ig}}, i), i \in \mathcal{N} - \{\rho_g\}$.

Furthermore, Q_g^{ija} and Q_g^{ijb} are given by (25)-(26) where λ_1 and λ_2 denote respectively the elements (i,i) and (j,j) of the matrix obtained after the previous stage of the incomplete factorization. By processing the incomplete factorizations (22) following the rule described above we obtain a matrix R_g with elements r_{ijg} given by

$$r_{ijg} = \begin{cases} \sqrt{\sum_{k \in \mathcal{A}_{ig}^a} w_{ikg} + \sum_{k \in \mathcal{A}_{ig}^b} w_{kig}} & \text{if } i = j \\ \dfrac{-w_{\alpha_{ig}\beta_{ig}g}}{\sqrt{\sum_{k \in \mathcal{A}_{ig}^a} w_{ikg} + \sum_{k \in \mathcal{A}_{ig}^b} w_{kig}}} & \text{if } j = pred_{ig} \\ 0 & \text{otherwise.} \end{cases}$$

where $\mathcal{A}_{ig}^a = \{k \in \mathcal{N} : k \notin Succ_{ig} \text{ and } i < k\}$ and $\mathcal{A}_{ig}^b = \{k \in \mathcal{N} : k \notin Succ_{ig} \text{ and } i > k\}$. The formulas derived from Theorem 6 to compute the elements of matrices $\bar{D}_g$, $\bar{F}_g$ and $\hat{F}_g$ are in this case

$$\bar{d}_{ig} = d_{ig}^{-1} + \sum_{k \in \mathcal{A}_{ig}^a} w_{ikg} + \sum_{k \in \mathcal{A}_{ig}^b} w_{kig},$$

$$\bar{f}_{ijg} = \begin{cases} (\sum_{k \in \mathcal{A}_{ig}^a} w_{ikg} + \sum_{k \in \mathcal{A}_{ig}^b} w_{kig}) / \bar{d}_{ig} & \text{if } i = j \\ -w_{\alpha_{ig}\beta_{ig}g} / \bar{d}_{ig} & \text{if } j = pred_{ig} \\ 0 & \text{otherwise,} \end{cases}$$

Procedure **solve1**$(V, \hat{D}, \tilde{u}, \hat{u})$
for $g = 1, \ldots, m$ **do**
  $aux_2 \leftarrow 0$
  for $h = 1, \ldots, g-1$ **do** $aux_2 \leftarrow aux_2 + V_{gh}\tilde{u}_h$
  $aux_2 \leftarrow \hat{D}_g \hat{u}_g - \bar{F}_g aux_2$
  Solve $\hat{F}_g \tilde{u}_g = aux_2$

Figure 5: Procedure **solve1**.

and

$$\hat{f}_{ijg} = \begin{cases} 1 & \text{if } \; i = j \\ -w_{\alpha_{ig}\beta_{ig}g}/\bar{d}_{ig} & \text{if } \; j = pred_{ig} \\ 0 & \text{otherwise.} \end{cases}$$

Variant 3 requires three vectors of order mn and $O(mn^2)$ aditions and $O(mn)$ divisions respectively to store and compute the elements of matrices $\bar{D}_g$, $\bar{F}_g$ and $\hat{F}_g$, $g = 1, \ldots, m$, different from one and zero. Variant 3 also requires the computation of m spanning trees. The algorithm employed to compute these trees is the same algorithm used in Variant 1.

We note that in the nonsymmetric case we have to compute $\bar{D}^{\frac{1}{2}}$ since $M_L = F_L^\top \bar{D}^{\frac{1}{2}}$ and $M_R = \bar{D}^{\frac{1}{2}} F_R$. This operation involves mn square roots. In the symmetric case this additional computational effort is avoided because the PCG algorithm uses as preconditioner the matrix $M = F_L^\top \bar{D} F_R$ with $F_R = F_L$.

We use a heuristic rule to select which variant of the IQRD preconditioning technique is employed in each iteration of the TN interior-point method. To derive this rule, we conducted several computational experiments using symmetric and nonsymmetric test problems with different numbers of commodities and regions. The selected variant does not depend on the symmetry of the problem since the relative performance of the variants is the same in symmetric and nonsymmetric problems. Variant 2 is chosen in the initial iterations of the TN algorithm. Since $R_g^\top R_g = diag(G_{\cdot\bar{\mathcal{H}}_1} W_g^k G_{\cdot\bar{\mathcal{H}}_1}^\top)$ for all $g = 1, \ldots, m$ in Variant 2, this choice is consistent with the observation made in [27, 31, 29, 30, 34] that in practice the Jacobi preconditioners perform well in the first iterations of the interior-point methods. The number of PCG/PQMR iterations taken at each interior-point move is monitored. If at iteration k of the TN algorithm the number of PCG/PQMR iterations exceeds $\sqrt{mn}$, then at iteration $k+1$ Variant 3 is chosen and used until the number of inner iterations exceeds again $\sqrt{mn}$. At this point, Variant 1 is chosen and used until the end. This is also consistent with the observation made in [25, 27, 26, 31, 29, 30] that a matrix R_g with a triangular structure associated to a maximum spanning tree can be a good preconditioner for the matrix $G_{\cdot\bar{\mathcal{H}}_1} W_g^k G_{\cdot\bar{\mathcal{H}}_1}^\top$ in the intermediate and last stages of the interior-point methods.

The linear systems of the form $M_R\tilde{u} = \hat{u}$, $M_L^\top \tilde{u} = \hat{u}$, $M_R^\top \tilde{u} = \hat{u}$ and $M_L \tilde{u} = \hat{u}$ required by the PQMR algorithm are solved by calling respectively **solve1**$(L, \bar{D}^{-\frac{1}{2}}, \tilde{u}, \hat{u})$, **solve1**$(U^\top, \bar{D}^{-\frac{1}{2}}, \tilde{u}, \hat{u})$, **solve2**$(L, \bar{D}^{-\frac{1}{2}}, \tilde{u}, \hat{u})$ and **solve2**$(U^\top, \bar{D}^{-\frac{1}{2}}, \tilde{u}, \hat{u})$ (see Figures 5 and 6). In the PCG algorithm, the solution of linear system $M\tilde{u} = \hat{u}$ is computed by calling

Procedure **solve2**$(V, \hat{D}, \tilde{u}, \hat{u})$
$aux_1 \leftarrow \hat{u}$
for $g = m, \ldots, 1$ **do**
Solve $\hat{F}_g^\top \tilde{u}_g = aux_{1g}$
$aux_2 \leftarrow \bar{F}_g^\top \tilde{u}_g$
for $h = 1, \ldots, g-1$ **do** $aux_{1h} \leftarrow aux_{1h} - V_{gh} aux_2$
$\tilde{u}_g \leftarrow \hat{D}_g \tilde{u}_g$

Figure 6: Procedure **solve2**.

solve2$(U^\top, \bar{D}^{-1}, \bar{u}, \hat{u})$ and then **solve1**$(L, I, \tilde{u}, \bar{u})$. In the procedures **solve1** and **solve2** the linear systems with the matrices $\hat{F}_g$ and $\hat{F}_g^\top$ are solved by employing the technique presented in [26]. The procedures **solve1** and **solve2** use the auxilliary vectors $aux_1 \in \mathbb{R}^{mn}$ and $aux_2 \in \mathbb{R}^n$ and take $O(m^2 n)$ sums(subtractions) and multiplications(divisions) to compute $\tilde{u}$. In each iteration of the PCG and PQMR algorithms, each one of the procedures **solve1** and **solve2** is called once and twice respectively.

3.2 Stopping criterion

In this section we discuss the stopping criterion for the TN interior-point algorithm. A point (x, z) is said to be a complementary solution of the LCP (3) if satisfies

$$\begin{aligned} z &= J^\top BJx - J^\top a + c \\ x_{ijg} z_{ijg} &= 0, \; i, j = 1, \ldots, n, \; g = 1, \ldots, m. \end{aligned} \tag{27}$$

Let $\mathcal{B}$ and $\bar{\mathcal{B}}$ be two subsets forming a partition of $\mathcal{H}_1 \cup \mathcal{H}_2$. Based on this partition, we can rewrite (27) as

$$\begin{bmatrix} z_{\mathcal{B}} \\ z_{\bar{\mathcal{B}}} \end{bmatrix} = \begin{bmatrix} J_{\cdot\mathcal{B}}^\top BJ_{\cdot\mathcal{B}} & J_{\cdot\mathcal{B}}^\top BJ_{\cdot\bar{\mathcal{B}}} \\ J_{\cdot\bar{\mathcal{B}}}^\top BJ_{\cdot\mathcal{B}} & J_{\cdot\bar{\mathcal{B}}}^\top BJ_{\cdot\bar{\mathcal{B}}} \end{bmatrix} \begin{bmatrix} x_{\mathcal{B}} \\ x_{\bar{\mathcal{B}}} \end{bmatrix} - \begin{bmatrix} c_{\mathcal{B}} - J_{\cdot\mathcal{B}} a \\ c_{\bar{\mathcal{B}}} - J_{\cdot\bar{\mathcal{B}}} a \end{bmatrix}. \tag{28}$$

A complementary basic solution is obtained by setting $x_{\bar{\mathcal{B}}} = 0$ and $z_{\mathcal{B}} = 0$ in the conditions (28). The null variables x_{ijg}, $(i, j, g) \in \bar{\mathcal{B}}$ and z_{ijg}, $(i, j, g) \in \mathcal{B}$ are called nonbasic, while x_{ijg}, $(i, j, g) \in \mathcal{B}$ and z_{ijg}, $(i, j, g) \in \bar{\mathcal{B}}$ are said to be basic variables. A complementary basic solution is said to be feasible if $x_{\mathcal{B}} \geq 0$ and $z_{\bar{\mathcal{B}}} \geq 0$. In this case it is a solution of the LCP (3). This solution is called nondegenerate if the values of all the basic variables are nonzero. Otherwise it is said to be degenerate.

We stop the TN method as soon as we are able to identify a solution of the LCP (3). When the value of the merit function $\|z - J^\top BJx - c + J^\top a\|_1 + \|Xz\|_1$ becomes sufficiently small, we use the information given at the current iteration of the TN algorithm (say iteration k) to guess an optimal basis $\mathcal{B}$. Then, we compute and test the feasibility of the corresponding basic complementary solution. If feasibility is satisfied we stop with a solution of the LCP (3). Otherwise we return to the TN algorithm (iteration $k+1$). A formal description of this procedure is presented in Figure 7. The two most important issues in the procedure **stopping criterion** are the implementation of the procedure **guess basis** and the solution of the linear

Procedure **stopping criterion**(W^k,$\mathcal{T}_g(g=1,\ldots,m)$)

1. Call **guess basis**(W^k,$\mathcal{T}_g(g=1,\ldots,m)$,$\mathcal{B}$,$\bar{\mathcal{B}}$)

2. Solve $J_{\cdot\mathcal{B}}^\top BJ_{\cdot\mathcal{B}}x_\mathcal{B} = J_{\cdot\mathcal{B}}^\top a - c_\mathcal{B}$

3. If $x_\mathcal{B} \geq 0$ **then** goto 4 **else** return to the TN algorithm (iteration $k+1$)

4. Compute $z_{\bar{\mathcal{B}}} = J_{\cdot\bar{\mathcal{B}}}^\top BJ_{\cdot\mathcal{B}}x_\mathcal{B} + c_{\bar{\mathcal{B}}} - J_{\cdot\bar{\mathcal{B}}}^\top a$

5. If $z_{\bar{\mathcal{B}}} \geq 0$ **then stop** with $(x_\mathcal{B}, 0)$ and $(0, z_{\bar{\mathcal{B}}})$ a solution of the LCP (3)
else return to the TN algorithm (iteration $k+1$)

Figure 7: Procedure **stopping criterion**.

system $J_{\cdot\mathcal{B}}^\top BJ_{\cdot\mathcal{B}}x_\mathcal{B} = J_{\cdot\mathcal{B}}^\top a - c_\mathcal{B}$. Next it is shown that whenever the LCP (3) is feasible, this problem has a solution with a forest structure. Furthermore, it is also demonstrated that the matrix $J_{\cdot\mathcal{B}}^\top BJ_{\cdot\mathcal{B}}$ associated with a basic complementary solution with a forest structure is positive definite. These results are used to guess the bases and to solve the associated systems.

Theorem 8. If the LCP (3) is feasible, then this problem has a solution where the trade routes of positive flow of each commodity form a forest. Furthermore, if condition (5) holds for $g = h$, then transhipment of flows does not occur in the forest that represents the trade flow of commodity h.

Proof. In [12] this result is proved for the single commodity case ($m = 1$). The generalization for the multicommodity case is straightforward and thus ommited. □

The sets $\mathcal{B}$ and $\bar{\mathcal{B}}$ associated to a basic complementary solution of the LCP (3) may be defined by $\mathcal{B} = \{(i,j,g) : g = 1,\ldots,m \text{ and } (i,j) \in \mathcal{B}_g\}$ and $\bar{\mathcal{B}} = \{\mathcal{H}_1 \cup \mathcal{H}_2\} - \mathcal{B}$ respectively, with $\mathcal{B}_g$ denoting the trade routes of positive flow of commodity g. Let the sets $\bar{\mathcal{B}}_g$, $\mathcal{V}_g$ and $\bar{\mathcal{V}}_g$ be defined by $\bar{\mathcal{B}}_g = \mathcal{G} - \mathcal{B}_g$, $\mathcal{V}_g = \{i \in \mathcal{N} : \exists(i,j) \in \mathcal{B}_g \text{ or } \exists(j,i) \in \mathcal{B}_g\}$ and $\bar{\mathcal{V}}_g = \mathcal{N} - \mathcal{V}_g$, respectively. For each commodity g, the matrix G can be reordered into

$$G = \begin{bmatrix} G_{\cdot\mathcal{B}_g} & G_{\cdot\bar{\mathcal{B}}_g} \end{bmatrix} = \begin{bmatrix} G_{\mathcal{V}_g\mathcal{B}_g} & G_{\mathcal{V}_g\bar{\mathcal{B}}_g} \\ G_{\bar{\mathcal{V}}_g\mathcal{B}_g} & G_{\bar{\mathcal{V}}_g\bar{\mathcal{B}}_g} \end{bmatrix},$$

where $G_{\bar{\mathcal{V}}_g\mathcal{B}_g}$ is equal to the null matrix. Now, suppose that $\Psi_{\mathcal{B}_g} = (\mathcal{V}_g, \mathcal{B}_g)$ is a forest constituted by γ_g trees rooted at nodes ρ_{kg}, $k = 1,\ldots,\gamma_g$. Furthermore, let $\mathcal{B}_{kg}$ represent the set of arcs belonging to the k-th tree of forest $\Psi_{\mathcal{B}_g} = (\mathcal{V}_g, \mathcal{B}_g)$ and let the set of nodes $\mathcal{V}_{kg}$ be defined by $\mathcal{V}_{kg} = \{i \in \mathcal{N} : \exists(i,j) \in \mathcal{B}_{kg} \text{ or } \exists(j,i) \in \mathcal{B}_{kg}\}$. In this case, $G_{\mathcal{V}_g\mathcal{B}_g}$ is a block diagonal matrix with diagonal blocks $G_{\mathcal{V}_{kg}\mathcal{B}_{kg}}$, $k = 1,\ldots,\gamma_g$. Furthermore, $G_{\mathcal{V}_{kg}\mathcal{B}_{kg}}$ is a $|\mathcal{V}_{kg}| \times |\mathcal{B}_{kg}|$ matrix that can be reordered into

$$G_{\mathcal{V}_{kg}\mathcal{B}_{kg}} = \begin{bmatrix} G_{(\mathcal{V}_{kg}-\{\rho_{kg}\})\mathcal{B}_{kg}} \\ G_{\rho_{kg}\mathcal{B}_{kg}} \end{bmatrix},$$

where $G_{\rho_{kg}\mathcal{B}_{kg}} \in \mathbb{R}^{1\times|\mathcal{B}_{kg}|}$ and $G_{(\mathcal{V}_{kg}-\{\rho_{kg}\})\mathcal{B}_{kg}}$ is a $|\mathcal{B}_{kg}|\times|\mathcal{B}_{kg}|$ triangular matrix with diagonal elements equal to 1 or -1.

Theorem 9. Let $\mathcal{B}$ and $\bar{\mathcal{B}}$ be the sets associated with a basic complementary solution of the LCP (3). $J_{\cdot\mathcal{B}}^\top BJ_{\cdot\mathcal{B}}$ is nonsingular iff $\Psi_{\mathcal{B}_g} = (\mathcal{N}, \mathcal{B}_g)$, $g = 1, \ldots, m$ are forests.

Proof. If there exits g such that $\Psi_{\mathcal{B}_g} = (\mathcal{N}, \mathcal{B}_g)$ contains a cycle, then there exists $v_g \neq 0$ such that $G_{\cdot\mathcal{B}_g} v_g = 0$ [24]. In this case $J_{\cdot\mathcal{B}}^\top BJ_{\cdot\mathcal{B}}$ is singular since $J_{\cdot\mathcal{B}}^\top BJ_{\cdot\mathcal{B}} v = 0$ for $v = (0, 0, \ldots, v_g, \ldots, 0, 0) \neq 0$. On the other hand, let $\Psi_{\mathcal{B}_g} = (\mathcal{N}, \mathcal{B}_g)$ be a forest. Then $G_{\mathcal{V}_{kg}\mathcal{B}_{kg}} v_{kg} \neq 0 \ \forall v_{kg} \in \mathrm{I\!R}^{|\mathcal{B}_{kg}|} - \{0\}$ because $G_{(\mathcal{V}_{kg} - \{\rho_{kg}\})\mathcal{B}_{kg}}$ is a triangular matrix with nonzero diagonal elements. Since B is positive definite, this proves that $J_{\cdot\mathcal{B}}^\top BJ_{\cdot\mathcal{B}}$ is nonsingular when $\Psi_{\mathcal{B}_g} = (\mathcal{N}, \mathcal{B}_g)$ is a forest for all $g = 1, \ldots, m$. □

The procedure **guess basis** uses Theorem 8 to guess the sets $\mathcal{B}$ and $\bar{\mathcal{B}}$ associated to a solution of the LCP (3) (see Figure 8). The procedure requires as input the maximum spanning trees $\Psi_{\mathcal{T}_g} = (\mathcal{N}, \mathcal{T}_g)$, $g = 1, \ldots, m$. Since either Variant 1 or Variant 3 are already active the first time procedure **guess basis** is called, $\Psi_{\mathcal{T}_g} = (\mathcal{N}, \mathcal{T}_g)$, $g = 1, \ldots, m$ are the maximum spanning trees used by these variants of the IQRD preconditioning technique. In procedure **guess basis** the arcs belonging to $\mathcal{T}_g$ and associated to an element of the matrix W_g^k less than $wcut$ are removed from the tree $\Psi_{\mathcal{T}_g} = (\mathcal{N}, \mathcal{T}_g)$. After this operation the forest $\Psi_{\tilde{\mathcal{T}}_g} = (\mathcal{V}_g, \tilde{\mathcal{T}}_g)$ is obtained. Then each node i in $\mathcal{V}_g$ is marked either as importer or exporter of commodity g depending on the direction of the arc having the maximum weight among all the arcs in the tree containing node i. In this last operation it is assumed that the triangular inequality (5) holds for all $g = 1, \ldots, m$. Finally, if node i is marked as importer(as exporter) then we put $(pred_i, i, g)$ $((i, pred_i, g))$ in set $\mathcal{B}$.

We conducted several computational experiments to test the ability of the procedure **guess basis** to recover the set $\mathcal{B}$ corresponding to a solution of the LCP (3). We observed that, when

$$\|z - J^\top BJx - c + J^\top a\|_1 + \|Xz\|_1 < 0.1,$$

the procedure can find an optimal set $\mathcal{B}$ for all the test problems except for the highly degenerate ones. As suggested in [31], an ε-perturbation technique helps to overcome such difficulty. In this procedure we use the TN algorithm to solve a perturbed LCP where the elements of the cost coefficients vector $\bar{c}$ are defined by $\bar{c}_{ijg} = c_{ijg}(1 + \delta_{ijg}\varepsilon)$, where δ_{ijg} is a uniform random number in the interval $[-1, 1]$ and ε is a small perturbation. This technique is far from being perfect, despite being, as much as we know, the most effective device for preventing degeneracy effects in practice. If ε is too small, then usually the perturbed problem is still degenerate. If ε is too big, then two cases may be possible. If we use the perturbed data in the termination criterion, the TN interior-point algorithm may stop with a point that is not a solution of the LCP (3). On the other hand, if we use the termination criterion with the original vector c, the TN interior-point algorithm may never stop. In order to avoid stopping with a point that is not a solution of the LCP (3), we have used the second alternative as suggested in Kaliski and Ye [17]. The implementation of this procedure in our code has provided a solution for all the test problems described in this paper.

We solve the linear system $J_{\cdot\mathcal{B}}^\top BJ_{\cdot\mathcal{B}} x_{\mathcal{B}} = J_{\cdot\mathcal{B}}^\top a - c_{\mathcal{B}}$ using the PCG and the PQMR algorithms when matrix B is symmetric and nonsymmetric, respectively. Let $\mathcal{V} = \{(i, g) : g = 1, \ldots, m \text{ and } i \in \mathcal{V}_g\}$, $\hat{\mathcal{V}} = \{(i, g) \in \mathcal{V} : i \neq \rho_{kg}, k = 1, \ldots, \gamma_g\}$ and $\bar{\mathcal{V}} = \{(i, g) : i = 1, \ldots, n, \ g = 1, \ldots, m\} - \hat{\mathcal{V}}$. Then, $J_{\cdot\mathcal{B}}^\top BJ_{\cdot\mathcal{B}} = J_{\cdot\mathcal{B}}^\top LUJ_{\cdot\mathcal{B}} = J_{\mathcal{V}\mathcal{B}}^\top L_{\mathcal{V}\cdot} U_{\cdot\mathcal{V}} J_{\mathcal{V}\mathcal{B}}$. The left and right preconditioners we use in the iterative solvers are $J_{\hat{\mathcal{V}}\mathcal{B}}^\top L_{\hat{\mathcal{V}}\hat{\mathcal{V}}}$ and $U_{\hat{\mathcal{V}}\hat{\mathcal{V}}} J_{\hat{\mathcal{V}}\mathcal{B}}$, respectively, where $J_{\hat{\mathcal{V}}\mathcal{B}}$ is a block diagonal matrix with diagonal blocks $G_{(\mathcal{V}_{kg} - \{\rho_{kg}\})\mathcal{B}_{kg}}$, $g = 1, \ldots, m$, $k = 1, \ldots, \gamma_g$ and, $L_{\hat{\mathcal{V}}\hat{\mathcal{V}}}$ and $U_{\hat{\mathcal{V}}\hat{\mathcal{V}}}$ are matrices obtained after removing the rows and columns of matrices L and U with indices in the set $\bar{\mathcal{V}}$. The starting point we use in the PCG and

```
Procedure guess basis(W^k, T_g (g = 1,...,m), B, B̄)
for g = 1,...,m do
    Set T̃_g = {(i,j) ∈ T_g : w^k_ijg > wcut} and V_g = {i ∈ N : ∃(i,j) ∈ T̃_g or ∃(j,i) ∈ T̃_g}
    Set γ_g equal to the number of trees that constitute the forest Ψ_T̃g = (V_g, T̃_g)
    Set B_g = ∅
    for k = 1,...,γ_g do
        Set T̃_kg equal to the set of arcs belonging to the k-th tree of forest Ψ_T̃g = (V_g, T̃_g)
        Set V_kg = {i ∈ N : ∃(i,j) ∈ T̃_kg or ∃(j,i) ∈ T̃_kg}
        Find (α,β) = argmax{w^k_ijg : (i,j) ∈ T̃_kg} and use the direction of arc (α,β) to
        mark each node in V_kg either as importer or exporter of commodity g.
        Set B_kg = ∅
        for all i ∈ V_kg do
            if i is marked as importer set B_kg = B_kg ∪ {(pred_i, i)}
            if i is marked as exporter set B_kg = B_kg ∪ {(i, pred_i)}
        Set B_g = B_g ∪ B_kg
Set B = {(i,j,g) : g = 1,...,m and (i,j) ∈ B_g} and B̄ = {H_1 ∪ H_2} − B
```

Figure 8: Procedure **guess basis**.

PQMR algorithms is the subvector of the current interior-point x^k constituted by the elements x^k_{ijg}, $(i,j,g) \in \mathcal{B}$. Finally, we stop at i-th iteration of the iterative solvers if the norm of the exact residual vector given by $r = J_{\cdot\mathcal{B}}^\top (a - BJ_{\cdot\mathcal{B}} x^i_{\mathcal{B}}) - c_{\mathcal{B}}$ is smaller than 10^{-14}. As we will show in the last section, this procedure has recovered an "exact" solution for all the test problems used in our computational experiments.

3.3 Other implementation issues

We conclude this section by discussing the choice of the starting point, the stepsize α_k and the value of the parameters β_0 and β_1 in the TN interior-point algorithm.

To compute an initial point for the TN algorithm, we consider the equivalence between the LCP (3) and the optimization problem

$$\begin{array}{rl} \min & \|z - J^\top BJx - c + J^\top a\| + \|Xz\| \\ \text{subject to:} & x \geq 0, z \geq 0. \end{array} \tag{29}$$

Then we obtain an initial point by solving (29) in the subregion of the positive orthant defined by $x = \alpha e$, $z = \beta e$ and $\alpha, \beta \geq \varepsilon$, where ε is a small tolerance. If the l_2 norm is used in the objective function of (29), then the initial point (x^0, z^0) is given by

$$(x^0, z^0) = \left(\varepsilon e, \max\{ \frac{e^\top (c - J^\top a)}{mn(n-1)}, \varepsilon \} e \right),$$

since $J^\top BJe = 0$. We have conducted several computational experiments for comparing this initialization with other choices for the vector (x^0, z^0) reported in [8, 32]. The results support the use of this starting point with $\varepsilon = 0.1$ in the particular case of the LCP (3).

In the implementation of the TN algorithm, the stepsize α_k is given by

$$\alpha_k = 0.99995 \max\{\alpha : x^k + \alpha\Delta x^k \geq 0,\ z^k + \alpha\Delta z^k \geq 0\}.$$

This scheme was initially proposed in [19] for linear programming and have proven to be effective in the solution of linear complementarity problems (see [8]).

Finally we follow the recommendation in [27] and set $\beta_0 = 0.09$ and $\beta_1 = 0.1$ in formulas (8) and (21).

4 Computational experience

In this section we report our computational experience with the TN algorithm for solving symmetric and nonsymmetric problems with randomly generated data. All the experiments were performed on a SUN SPARCstation 10-52 with 64 Mbytes of RAM. The codes are written in FORTRAN and were compiled with the Sun Fortran Compiler (f77) using flags **-O4 -cg89 -libmil -native**. All the CPU times reported are excluded from input and output time and were measured with the internal function **etime**. We use a similar technique to that proposed in [23] to generate the test problems. The elements of the vector of costs c_{ijg} are set equal to $d_{ig} + d_{jg}$ where 60% of the d_{ig}'s (chosen randomly) are equal to 10 and the other 40% are random numbers between 0 and 10. We note that the triangular inequality (5) holds for all $i, k, j = 1, \ldots, n$ and $g = 1, \ldots, m$ with this choice of the vector c. The elements of the vector a are random numbers in the interval $[0, 10]$. Let $\bar{B} = \Sigma + \Gamma\Gamma^\top$, where $\Sigma \in \mathbb{R}^{mn \times mn}$ is a diagonal matrix with randomly generated diagonal elements in the interval $]0, 1]$ and $\Gamma \in \mathbb{R}^{mn \times mn}$ is a matrix with the same sparsity structure of matrix B and randomly generated nonzero elements in the interval $]-1, 1[$. We note that $\bar{B}$ is symmetric positive definite. So, we set $B = \bar{B}$ if we wish B to be a symmetric matrix. Otherwise, we obtain a nonsymmetric positive definite matrix B by setting

$$b_{igh} = \begin{cases} \delta_{igh}\bar{b}_{igh} & \text{if } g < h \\ 0.5\bar{b}_{igh} & \text{if } g = h \\ \bar{\delta}_{igh}\bar{b}_{igh} & \text{if } g > h, \end{cases}$$

where δ_{igh} is a random number in the interval $]0, 1[$ and $\bar{\delta}_{igh}$ is given by $\bar{\delta}_{igh} = 1 - \delta_{ihg}$ for all $i = 1, \ldots, n$, $g = 1, \ldots, m$ and $h = 1, \ldots, g-1$. In this case, B is positive definite because the equality $\bar{B} = B + B^\top$ holds and $\bar{B}$ is symmetric positive definite.

The results of the experiments are illustrated in Tables 1 and 2, where we use the following notations:

lcpdim dimension of the LCP,
rsd dimension of the reduced system,
ite number of TN iterations,
pcgit number of PCG iterations,
pqmrit number of PQMR iterations,
scc number of calls of the procedure **stopping criterion**,
tncpu CPU time in seconds taken by the TN algorithm,

cpcpu CPU time in seconds to construct the preconditioners,
sccpu CPU time in seconds required by the procedure **stopping criterion**,
cpu total CPU time in seconds (*tncpu*+*sccpu*).

Ther results presented in Tables 1 and 2 lead to the following conclusions:

- The IQRD preconditioning technique together with the transformation of the system of equations and stopping criterion proposed in Theorem 4, constitute an efficient technique to compute the search directions in the TN method. In fact, the PCG and the PQMR algorithms have always achieved an approximate solution of the reduced system (12) verifying condition (21). On the other hand, by looking at the column of Table 1 corresponding to the number $\frac{pcgit}{rsd*ite}$, we can conclude that in the symmetric problems the average number of inner iterations required to compute the search directions are in the best and worst cases $0.01mn$ and $0.08mn$, respectively. In the nonsymmetric problems, these numbers are $0.07mn$ and $0.26mn$. This is quite impressive since the order of the linear system (10) is $2mn(n-1)$. Furthermore, the cost of computing the preconditioners is not meaningful. In the worst case (the single commodity problems), this task requires less than 2% of the total CPU time spent by the TN algorithm (see the column of Tables 1 and 2 corresponding to the number $\frac{100*cpcpu}{tncpu}$).

- Nonsymmetric problems require more CPU time to solve than symmetric problems despite the number of TN iterations is about the same in both cases. In fact, the number of inner iterations is in general superior in the nonsymmetric problems. Furthermore, the cost of a PQMR iteration is higher than the cost of a PCG iteration. We recall that the bulk of the PCG and PQMR algorithms are the matrix-vector multiplications and the solution of the linear systems with the preconditioning matrices. The computational effort involved in these tasks for the PQMR algorithm is twice that of the PCG method.

- The TN algorithm together with the procedure **stopping criterion** constitute a robust technique to compute a solution of the LCP (3). The TN algorithm achieves in all the test problems a point verifying $\|z - J^\top BJx - c + J^\top a\|_1 + \|Xz\|_1 < 0.1$ in a reasonable number of iterations both in the symmetric and nonsymmetric cases. Since the cost per iteration is small in the TN algorithm, this represents an aceptable total cost to find an approximate solution. Furthermore, it is necessary no more than 10 calls of the procedure **stopping criterion** to recover a solution for all but for 2 symmetric test problems. Finally, the additional CPU time required by the procedure **stopping criterion** to recover a solution of the LCP (3) is in almost the test problems a small fraction of the total cpu time (see the column of Tables 1 and 2 corresponding to the number $\frac{100*sccpu}{cpu}$).

Table 1: Symmetric problems.

Test Problems				TN algorithm				St. criterion		
m	n	$lcpdim$	rsd	ite	$\frac{pcgit}{rsd*ite}$	$tncpu$	$\frac{100*cpcpu}{tncpu}$	scc	$\frac{100*sccpu}{cpu}$	cpu
1	100	9900	100	20	0.07	2.72	1.47	1	0.37	2.73
1	300	89700	300	29	0.03	45.10	1.17	1	0.31	45.24
1	500	249500	500	44	0.02	198.32	1.12	1	0.19	198.70
1	700	489300	700	49	0.02	464.63	1.00	3	0.57	467.30
1	900	809100	900	49	0.01	920.20	1.19	1	0.17	921.80
1	1100	1208900	1100	58	0.01	1754.67	1.18	1	0.13	1757.00
10	50	24500	500	22	0.07	13.95	0.21	1	0.99	14.09
10	100	99000	1000	33	0.04	89.03	0.16	7	2.91	91.70
10	150	223500	1500	60	0.03	348.81	0.13	28	5.40	368.90
10	200	398000	2000	36	0.03	428.31	0.13	4	0.88	432.10
10	250	622500	2500	49	0.03	1005.81	0.16	6	0.81	1014.00
10	300	897000	3000	53	0.02	1634.20	0.17	6	0.54	1643.00
10	350	1221500	3500	59	0.02	2522.37	0.17	10	0.97	2547.00
10	400	1596000	4000	55	0.02	3261.17	0.17	4	0.30	3271.00
20	25	12000	500	18	0.08	8.90	0.22	2	4.00	9.25
20	50	49000	1000	24	0.06	45.73	0.21	4	4.53	47.90
20	75	111000	1500	32	0.04	121.97	0.17	8	6.25	130.10
20	100	198000	2000	34	0.04	268.23	0.07	5	2.89	276.20
20	125	310000	2500	35	0.03	405.03	0.12	5	2.43	415.10
20	150	447000	3000	34	0.03	560.55	0.10	2	1.07	566.60
20	175	609000	3500	39	0.02	926.06	0.08	4	1.59	941.00
20	200	796000	4000	40	0.02	1291.59	0.07	3	1.03	1305.00
20	250	1245000	5000	46	0.02	2556.83	0.12	4	0.90	2580.00
30	25	18000	750	24	0.06	24.40	0.18	8	15.66	28.93
30	50	73500	1500	111	0.02	292.00	0.13	89	32.27	431.10
30	75	166500	2250	29	0.03	242.08	0.10	5	4.81	254.30
30	100	297000	3000	31	0.03	468.23	0.07	3	2.59	480.70
30	150	670500	4500	40	0.02	1465.18	0.06	7	3.67	1521.00
30	175	913500	5250	43	0.02	2268.70	0.10	7	3.12	2032.00
30	200	1194000	6000	48	0.02	2933.69	0.09	9	3.11	3028.00
40	25	24000	1000	23	0.05	42.64	0.15	5	10.89	47.85
40	50	98000	2000	28	0.04	183.21	0.10	6	8.35	199.90
40	75	222000	3000	33	0.03	444.58	0.07	9	9.31	490.20
40	100	396000	4000	38	0.02	943.67	0.06	8	6.66	1011.00
50	25	30000	1250	21	0.04	59.17	0.09	5	11.91	67.17
50	50	122500	2500	29	0.03	302.83	0.07	6	9.87	336.00
50	75	227500	3750	36	0.03	805.53	0.06	8	8.54	880.70
50	100	495000	5000	40	0.02	1561.90	0.05	10	10.10	1737.00
50	125	775000	6250	37	0.02	2322.80	0.04	5	4.69	2437.00

Table 2: Nonsymmetric problems.

Test Problems				TN algorithm				St. criterion		
m	n	$lcpdim$	rsd	ite	$\frac{pqmrit}{rsd*ite}$	$tncpu$	$\frac{100*cpcpu}{tncpu}$	scc	$\frac{100*sccpu}{cpu}$	cpu
10	20	3800	200	11	0.13	2.24	0.83	2	7.05	2.41
10	40	15600	400	15	0.13	15.99	0.23	4	9.61	17.69
10	60	35400	600	17	0.14	49.98	0.09	2	5.05	52.64
10	80	63200	800	20	0.14	124.22	0.05	4	5.18	131.00
10	100	99000	1000	26	0.16	325.38	0.03	8	6.87	349.40
10	120	142800	1200	24	0.15	449.66	0.04	4	4.51	470.90
10	140	194600	1400	26	0.26	1183.52	0.03	5	2.75	1217.00
20	10	1800	200	8	0.07	1.44	0.68	1	1.37	1.46
20	20	7600	400	13	0.13	14.92	0.17	3	13.66	17.28
20	30	17400	600	16	0.14	45.90	0.09	5	14.40	53.62
20	40	31200	800	20	0.19	157.87	0.05	6	13.74	183.00
20	50	49000	1000	21	0.20	277.37	0.03	7	12.72	317.80
20	60	70800	1200	24	0.19	476.12	0.01	8	12.78	545.90
20	70	96600	1400	21	0.15	492.74	0.02	3	7.38	532.00
20	80	126400	1600	25	0.13	750.42	0.02	6	8.83	823.10
30	10	2700	300	11	0.10	7.97	0.24	4	3.28	8.24
30	20	11400	600	14	0.11	41.71	0.06	5	12.28	47.55
30	30	26100	900	17	0.12	126.68	0.03	5	13.94	147.20
30	40	46800	1200	18	0.14	300.79	0.02	4	11.82	341.10
30	50	73500	1500	23	0.18	839.70	0.01	8	18.48	1030.00
30	60	106200	1800	22	0.11	756.40	0.02	5	14.16	881.20
40	10	3600	400	10	0.08	12.84	0.07	3	7.23	13.83
40	20	15200	800	12	0.07	49.75	0.12	2	8.98	54.66
40	30	34800	1200	19	0.12	318.09	0.02	7	17.23	384.30
40	40	62400	1600	21	0.12	655.50	0.02	7	19.41	813.40
40	50	98000	2000	24	0.15	1653.40	0.01	8	16.33	1976.00

5 Concluding remarks

In this paper we introduce a truncated Newton interior-point method for the solution of a monotone LCP arising from a multicommodity spatial equilibrium problem. An implementation of the algorithm which fully exploits the network structure of the model is described. The numerical results show that the approach is quite efficient. We believe that this type of methodology may also be employed in some other network models such as those discussed in [22].

Acknowledgment

The authors acknowlwdge several insightful discussions with Luís Nunes Vicente.

References

[1] R. Ahuja, T. Magnanti, and J. Orlin. *Network Flows*. Prentice Hall, Englewood Cliffs, New Jersey, USA, 1993.

[2] R. Asmuth, B. Eaves, and E. Peterson. Computing economic equilibria on affine networks with Lemke's algorithm. *Mathematics of Operations Research*, 4:209–214, 1979.

[3] O. Axelsson. *Iterative Solution Methods*. Cambridge University Press, 1994.

[4] R. Barret, M. Berry, T. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. Van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*. SIAM Publications, 1993.

[5] R. Dembo, S. Eisenstat, and T. Steihaug. Inexact newton methods. *SIAM Journal on Numerical Analysis*, 19:400–408, 1982.

[6] R. Dembo and T. Steihaug. Truncated-netwon algorithms for large scale unconstrained optimization. *Mathematical Programming*, 26:190–212, 1983.

[7] L. Fernandes, M. Coelho, J. Júdice, and J. Patricio. On the solution of a spatial equilibrium problem. *Investigação Operacional*, 13:119–134, 1993.

[8] L. Fernandes, J. Júdice, and J. Patricio. An investigation of interior-point and block pivoting algorithms for large-scale symmetric monotone linear complementarity problems. Technical report, Department of Mathematics, University of Coimbra, Coimbra, Portugal, 1994. To appear in *Computational Optimization and Applications*.

[9] A. Fiacco and G. McCormick. *Nonlinear programming: Sequential unconstrained minimization technique*. Wiley, New York, 1968.

[10] R. Freund and N. Nachtigal. An inplementation of the QMR method based on coupled two-term recurences. Technical Report 92.15, NASA Ames, Ames, CA, USA, 1992.

[11] A. George and M. Heath. Solution of sparse linear least squares problems using Givens rotations. *Linear Algebra and Its Applications*, 34:69–83, 1980.

[12] C. Glassey. A quadratic network optimization model for equilibrium single commodity trade flows. *Mathematical Programming*, 14:98–107, 1978.

[13] G. Golub and C. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, USA, 1983.

[14] F. Güder, J. Morris, and S. Yoon. Parallel and serial sucessive overrelaxation for multicommodity spatial price equilibrium problems. *Transportation Science*, 26:48–58, 1992.

[15] M. Hestenes and E. Steifel. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards B*, 49:409–436, 1952.

[16] J. Júdice. Algorithms for linear complementarity problems. In E. Spedicato, editor, *Algorithms for Continuous Optimization. The State of the Art*, pages 435–474. Kluwer Academic Publishers, 1993.

[17] J. Kaliski and Y. Ye. A decomposition variant of the potential reduction algorithm for linear programming. *Management Science*, 39:757–776, 1993.

[18] M. Kojima, N. Megiddo, and S. Mizuno. A primal-dual infeasible-interior-point algorithm for linear programming. *Mathematical Programming*, 61:263–280, 1993.

[19] K. McShane, C. Monma, and D. Shanno. An implementation of a primal-dual interior point method for linear programming. *ORSA Journal on Computing*, 1:70–83, 1989.

[20] N. Megiddo. Pathways to the optimal set in linear programming. In N. Megiddo, editor, *Progress in Mathematical Programming, Interior Point and Related Methods*, pages 131–158. Springer, 1989.

[21] S. Mehrotra. On the implementation of a primal-dual interior point method. *SIAM Journal on Optimization*, 2:575–601, 1992.

[22] A. Nagurney. *Network Economics: A Variational Inequality Approach*. Kluwer Academic Publishers, London, England, 1993.

[23] J. Pang. A hybrid method for the solution of some multicommodity spatial equilibrium problems. *Management Science*, 27:1142–1157, 1981.

[24] J. Pang and P. Lee. A parametric linear complementarity technique for the computation of equilibrium prices in a single commodity spatial model. *Mathematical Programming*, 20:81–102, 1981.

[25] L. Portugal, F. Bastos, J. Júdice, J. Paixão, and T. Terlaky. An investigation of interior point algorithms for the linear transportation problem. Technical report, Department of Mathematics, University of Coimbra, Coimbra, Portugal, 1993. To appear in *SIAM Journal on Scientific Computing*.

[26] L. Portugal and J. Júdice. A hybrid algorithm for the solution of a single commodity spatial equilibrium model. Technical report, Department of Mathematics, University of Coimbra, Coimbra, Portugal, 1994. To appear in *Computers & Operations Research*.

[27] L. Portugal, M. Resende, G. Veiga, and J. Júdice. A truncated primal-infeasible dual-feasible network interior point method. Technical report, Department of Mathematics, University of Coimbra, Coimbra, Portugal, 1994.

[28] R. Prim. Shortest connection networks and some generalizations. *Bell System Technical Journal*, 36:1389–1401, 1957.

[29] M. Resende and P. Pardalos. Interior point algorithms for network flow problems. Technical report, AT&T Bell Laboratories, Murray Hill, New Jersey, USA, 1994. To appear in *Advances in Linear and Integer Programming*, J. Beasley, ed., Oxford University Press, 1995.

[30] M. Resende and G. Veiga. An efficient implementation of a network interior point method. In D. Johnson and C. McGeoch, editors, *Networks Flow and Matching: First DIMACS Implementation Challenge*, volume 12, pages 299–384. American Mathematical Society, 1993.

[31] M. Resende and G. Veiga. An implementation of the dual affine scaling algorithm for minimum cost flow on bipartite uncapacitated networks. *SIAM Journal on Optimization*, 3:516–537, 1993.

[32] E. Simantiraki and D. Shanno. An infeasible interior-point method for linear complementarity problems. RRR 7-95, RUTCOR, New Brunswick, New Jersey, USA, 1995.

[33] T. Takayama and G. Judge. *Spatial and temporal Price and Allocation Models*. North Holland, Amesterdam, 1993.

[34] Q. Yeh. *A reduced dual affine scaling algorithm for solving assignment and transportation problems*. PhD thesis, Columbia University, New York, USA, 1989.

[35] Y. Zhang. On the convergence of a class of infeasible interior-point methods for the horizontal linear complementarity problem. *SIAM Journal on Optimization*, 4:208–227, 1994.