Mathematical Programming 14 (1978) 98–107.
North-Holland Publishing Company

A QUADRATIC NETWORK OPTIMIZATION MODEL FOR EQUILIBRIUM SINGLE COMMODITY TRADE FLOWS

C. Roger GLASSEY

University of California, Berkeley, CA, U.S.A.

Received 29 December 1976
Revised manuscript received 23 August 1977

When supply and demand curves for a single commodity are approximately linear in each of N regions and interregional transportation costs are linear, then equilibrium trade flows can be computed by solving a quadratic program of special structure. An equilibrium trade flow exists in which the routes carrying positive flow form a forest, and this solution can be efficiently computed by a tree growing algorithm.

Key words: Network Flows, Equilibrium Trade, Quadratic Programming.

1. Introduction

The optimization problem considered in this paper arises from a simple equilibrium model of international or interregional trade in a single commodity. We assume that in each region there is a market characterized by classical supply and demand curves. The equilibrium price and quantity produced and consumed will be determined, in the absence of imports or exports, by the intersection of these curves. If imports are introduced into this local market, consumption will exceed production but at a lower equilibrium price.

This type of equilibrium model has been extensively discussed in the literature of economics. Samuelson in [2] pointed out that an equilibrium solution is also the maximizer of a function which he called net social payoff. Lemmas 1 and 2 below are implicit in his discussion, but not stated and proved formally. More recently, Takayama and Judge [3] examine a number of models in which supply and demand curves are assumed to be linear, as in Fig. 1, and formulate a quadratic program which, when solved, yields equilibrium supplies, demands, trade flows and prices. The purpose of this paper is to show how their solution procedure can be greatly simplified by taking advantage of the very special structure of the problem.

2. Equilibrium conditions

The first simplification is to observe that, from the global view, the internal supply and demand of each region are irrelevant; it is only the net import quantity and the local market equilibrium price that matter. If local supply and

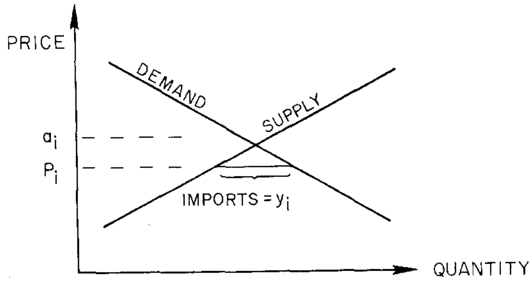


Fig. 1.

demand functions are linear, then there is a linear relation between price and net imports of the form

$$p_i = a_i - b_i y_i \tag{1}$$

where

- p_i is the equilibrium price in the ith region,
- y_i is the net import of the ith region,
- a_i is the equilibrium price in the absence of imports (and exports) and is positive,
- b_i is related to elasticity of supply and demand, and is also positive.

Clearly, if p_i exceeds a_i, then supply locally exceeds demand, the difference being available for export. Thus, y_i is not restricted to be nonnegative in (1); negative values of y_i are simply interpreted as exports.

Once the price p_i has been determined from global equilibrium considerations, the local supply and demand quantities are uniquely determined.

Now introduce nonnegative flow variables x_{ij} = exports from region i to region j, and transportation costs c_{ij} = cost per unit shipped from i to j.

We assume all shipments take place over the least cost route, therefore the c_{ij} will obey the triangle inequality:

$$c_{ij} \leq c_{ik} + c_{kj}.$$

The additional interregional trade equilibrium conditions are:

$$p_i + c_{ij} \geq p_j \quad \text{for all } i, j \tag{2}$$

and

$$(p_i + c_{ij} - p_j)x_{ij} = 0 \quad \text{for all } i, j. \tag{3}$$

The rationale behind these conditions is that if inequality (2) fails to hold, exporters will buy in market i at price p_i, transport to market j at unit cost c_{ij} and sell at price p_j thus making a profit. Exports from i to j will increase until the elasticity effects in markets i and j raise (and lower) these prices so that additional profit to exporters is no longer possible. Thus, if $x_{ij} > 0$, (2) must be satisfied as an equality, and we have the complementary slackness condition (3).

The model is completed by the flow conservation equations (a definition of net imports):

$$y_i - \sum_{j=1}^{N} x_{ji} + \sum_{j=1}^{N} x_{ij} = 0, \quad i = 1, 2, \dots, N. \tag{4}$$

We seek a procedure for calculating the equilibrium prices and flows from the data $(a_i, b_i, c_{ij}; i = 1, N; j = 1, N)$. One such procedure begins by formulating an optimization problem which has the interesting property that the optimality conditions of the problem (Kuhn–Tucker conditions) are the equilibrium conditions (1)–(4). The quadratic program that has this property is:

$$\text{(P):} \quad \begin{aligned} &\text{maximize} \quad z = \sum_i \Big(a_i y_i - \tfrac{1}{2} b_i y_i^2 - \sum_j c_{ij} x_{ij}\Big), \\ &\text{subject to} \quad (4) \quad \text{and} \quad x_{ij} \geq 0, \quad \text{for all } i, j. \end{aligned} \tag{5}$$

This objective can be shown to be equivalent to the net social pay-off function of [2]. The quadratic program has the special feature that the nonlinear part of the objective is separable in unrestricted variables. Since all b_i are positive, the objective function is negative definite, hence a unique global maximum exists.

It is easy to show that a dual of (P) is (see [1]):

$$\text{(D):} \quad \text{minimize} \quad \sum_i \frac{(a_i - \lambda_i)^2}{2b_i}, \tag{6}$$

$$\text{subject to} \quad \lambda_j - \lambda_i \leq c_{ij} \quad \text{for all } i, j. \tag{7}$$

The equilibrium solution can be obtained by solving (P) or (D) by any of a number of algorithms for the quadratic program.

3. A tree growing algorithm

We discuss here an algorithm that takes advantage of the special network structure of an equilibrium solution.

Lemma 1. *An equilibrium solution exists in which the trade routes of positive flow form a forest (a collection of trees).*

Proof. An equilibrium solution exists because (P) and (D) have optimal solutions. Recall that a tree is a connected graph of k nodes and $k-1$ arcs and contains no loops. Now suppose in an equilibrium solution, we find a subset of k regions connected by k or more trade routes (arcs) over which positive flow occurs. Then there must be a subset of these arcs that form a loop. Now we superimpose a flow around this loop in a direction opposite to at least one pre-existing arc flow, and increase the value of this loop flow until it exactly cancels one of the pre-existing arc flows. Observe that net imports remain unchanged and hence so do prices. Thus conditions (2) and (3) remain satisfied. Hence, any loops in the structure of an equilibrium trade flow pattern can be eliminated, leaving a collection of trees.

Thus we see that an equilibrium solution exists in which the set of N regions is partitioned into trading coalitions. The members of each coalition trade only with each other, and the set of active trade routes within each coalition forms a spanning tree for the coalition.

Stated formally, a coalition C is a set of k nodes and $k-1$ active arcs having the following properties:

(P1) *Internal equilibrium*: conditions (1), (2), (3) and (4) are satisfied for all nodes i, j in C.

(P2) *Tree structure*: the set of active arcs (ij), such that $x_{ij} > 0$, is a spanning tree for C. Hence, there are no loops of active arcs.

In addition, without loss of generality (see Lemma 2), we require

(P3) *Alternating arc orientation*: each node in a coalition (which contains more than one) is either an exporter or an importer, i.e., no transshipment occurs.

Thus, the unique path connecting a pair of nodes in C consists of arcs of alternating orientation. Starting from an importing node, for example, all adjacent nodes are exporters and the movement along this unique path is against the direction of goods flow. All nodes adjacent to exporters are importers, so movement is with the orientation of the arc on which goods flow.

Lemma 2. *Any coalition having properties* (P1) *and* (P2) *has a tree of active routes such that property* (P3) *obtains.*

Proof. Suppose in a coalition, $x_{ij} > 0$ and $x_{jk} > 0$ so node j is both an importer and an exporter. We call j a transshipment node, and the chain $i \to j \to k$ a transshipment route. The equilibrium conditions (2) and (3) then imply that $c_{ik} \geq c_{ij} + c_{jk}$. This inequality is the reverse of the triangle inequality which holds because of the least cost shipment route assumption, so we conclude that

$$c_{ik} = c_{ij} + c_{jk}.$$

Thus a flow around the loop $i \to k \to j \to i$ can be increased, with no change in objective function value, till either x_{ij} or x_{jk} becomes zero. Then arc (ik) is included in the tree of active arcs and at least one of (ij), (jk) is no longer active and is deleted from the tree. As in the proof of Lemma 1, the equilibrium conditions (1)–(4) still hold.

Now observe that if i (and k) was not a transshipment node before, it is not after this step is completed. Furthermore, the number of transshipment routes passing through node j has been reduced by one. A finite number of repetitions of this step will eliminate all such routes through j so a finite number of repetitions will produce a tree having property (P3).

The algorithm below finds an equilibrium solution by building up a set of coalitions which are, at termination, in equilibrium with each other as well as being in internal equilibrium.

Algorithm A1

Step 0. Initialization: set $p_i = a_i$, $y_i = 0$, $i = 1, \dots, N$; $x_{ij} = 0$, all arcs (ij). Each node is a coalition of one member.

Step 1. Find a pair of nodes (j, k) such that $p_j + c_{jk} < p_k$. If no such pair exists, the current solution is equilibrium.

Step 2. Let E and I be the coalitions containing j and k respectively. Let e in E and i in I be, respectively, exporting and importing nodes (i.e., $y_e \le 0$ and $y_i \ge 0$) such that

$$\bar{c}_{ei} = p_e + c_{ei} - p_i = \min_{j \in E, k \in I} \{p_j + c_{jk} - p_k\}.$$

(a) Increases x_{ei}, the flow from e to i, while maintaining equilibrium conditions (1), (3), and (4) within each coalition, until *either*:

(b) $p_e(x_{ei}) + c_{ei} = p_i(x_{ei})$ where $p_k(x_{ei})$ is the equilibrium price at k as a function of x_{ei}. This equilibrium value of x_{ei} is computed from (13) below. The new equilibrium prices in E and I are computed by adding δ_e and δ_i (see (12) below) to the old prices. Form a new coalition from the union of E and I plus the arc (ei). Then go to step 1. *Or*:

(c) $x_{jk} = 0$ for some arc (jk) in E or I. Then split the coalition containing (jk) at (jk) and call the part that contains e (or i) E (or I). Go to (2a).

Notes on computational details

Step 1. If the current solution is not equilibrium, additional profits are to be made by shipping from some exporting node to some importing node.

Step 2. The minimum computed in step 2 is attained at an arc (jk) where j is an exporting node and k is an importing node. This fact can be deduced as follows: suppose j is not an exporting node ($y_j > 0$). Then there is a node e in E such that $x_{ej} > 0$ and hence, by (3), $p_e + c_{ej} = p_j$. This equality, together with the triangle inequality $c_{ek} \le c_{ej} + c_{jk}$, implies that $\bar{c}_{ek} \le \bar{c}_{jk}$. The case $y_k < 0$ is analysed similarly.

This result implies that the search process of step 1 can be confined to arcs (jk) with j exporting and k importing.

Step 2 preserves property (P3) in the new coalition.

Step 2a. Let $y_j(x_{ei})$ be the equilibrium net import at node j as a function of x_{ei}. Conservation of flow implies that

$$-\sum_{j\in E}\Delta y_j = x_{ei} = \sum_{k\in I}\Delta y_k \tag{9}$$

where $\Delta y_j = y_j(x_{ei}) - y_j(0)$.

Let

$$\delta_e = p_e(x_{ei}) - p_e(0),$$
$$\delta_i = p_i(x_{ei}) - p_i(0).$$

Observe that the complementary slackness condition (3) implies all prices in each coalition E and I change by δ_e and δ_i respectively, since the nodes in E, and I, are connected by a tree of active arcs. Note that, while δ_e is positive, δ_i is negative. Hence, (1) and (9) combine to give:

$$\delta_e \sum_{j\in E}\beta_j = x_{ei} = -\delta_i \sum_{k\in I}\beta_k \tag{10}$$

where $\beta_j = 1/b_j$.

Step 2b. This condition is met when

$$\delta_i - \delta_e = p_e(0) + c_{ei} - p_i(0) = \bar{c}_{ei}. \tag{11}$$

(11) and (10) yield

$$\delta_e = -\bar{c}_{ei}\sum_{k\in I}\beta_k / \sum_{k\in E\cup I}\beta_k \tag{12a}$$

and

$$\delta_i = \bar{c}_{ei}\sum_{k\in E}\beta_k / \sum_{k\in E\cup I}\beta_k \tag{12b}$$

and

$$x_{ei} = \bar{c}_{ei}\sum_{k\in E}\beta_k \cdot \sum_{j\in I}\beta_j / \sum_{k\in E\cup I}\beta_k. \tag{13}$$

Step 2c. Now consider an arc (jk) in I which is oriented opposite to (ei), that is, the path from i to j in I traverses (jk) in the direction from k to j. Increasing x_{ei} will reduce x_{jk}. Flow conservation shows that

$$x_{jk} = -\sum_{n\in T_j} y_n \tag{14}$$

where T_j is the subtree of I rooted at j which is obtained when arc (jk) is deleted from I. Using (1), (10) and (14) we obtain

$$\Delta x_{jk} = -x_{ei}\cdot \sum_{n\in T_j}\beta_n / \sum_{n\in I}\beta_n. \tag{15}$$

This relationship is illustrated in Fig. 2.

Thus the critical value of x_{ei} that causes x_{jk} to be reduced to zero is:

$$\theta_{jk} = x_{jk}\sum_{n\in I}\beta_n / \sum_{n\in T_j}\beta_n. \tag{16}$$

Similarly, when $(jk)\in E$ and is oriented opposite to (ei) we obtain

$$\theta_{jk} = x_{jk}\sum_{n\in E}\beta_n / \sum_{n\in T_k}\beta_n. \tag{17}$$

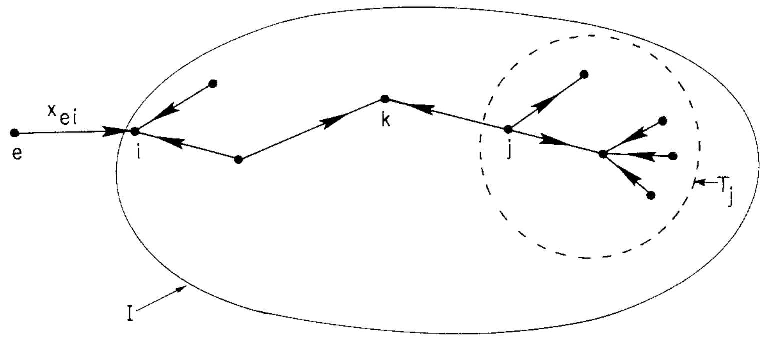


Fig. 2. An importing coalition.

Step 2c is accomplished by computing the minimum of θ_{jk} given by (16) and (17) for those arcs in I and E oriented opposite to (ei). Note that only the prices need be updated for each coalition, since y_j is computed from p_j in (1) and arc flows in each tree can be computed from (14).

Example (A Four Region System).

Data.

$a =$	9	13	16	30
$\beta =$	0.25	2.5	3.0	0.25
$c =$	0	2	1	1
	2	0	3	1
	1	3	0	2
	1	1	2	0

Cycle 1.

Step 1. min $\bar{c}_{ij} = \bar{c}_{14} = -20$.

Step 2. Form coalition $C_1 = (1\ 4)$.

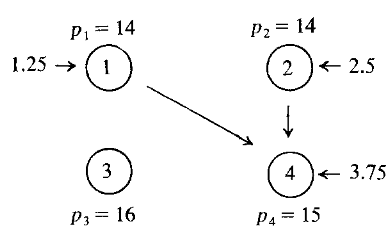
$$x_{14} = -y_1 = y_4 = 2.5$$

$$p = 19 \quad 13 \quad 16 \quad 20$$

Cycle 2.

Step 1. min $\bar{c}_{ij} = \bar{c}_{24} = -6$.

Step 2. Enlarge coalition by adding 2; $x_{24} = 2.5$, $c_1 = (1\ 2\ 4)$:



Cycle 3.

Step 1. $\min c_{ij} = \bar{c}_{13} = -1$.

Step 2b. $p_3 = 15.5$, $p_1 = 14.5$, $x_{31} = 1.5$.

Step 2c. (see eq. 17)

$$\theta_{13} = 1.25 \frac{3}{2.75} = 1.36.$$

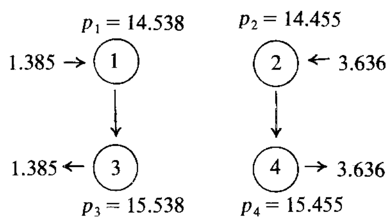
i.e., x_{14} is reduced to zero when $x_{31} = 1.36 < 1.5$. Hence E becomes the single node 1; the coalition (2, 4) is split from E when $x_{31} = 1.3636$ and

$$\delta_e = \frac{1.3636}{0.25 + 0.25 + 2.5} = 0.4545 \quad \text{(eq. 10)}.$$

Step 2b now gives

$$p_3 = 15.538, \quad p_1 = 14.538, \quad x_{31} = 1.385.$$

The equilibrium solution is:



Lemma 3. *The marginal profit,* $p_i - p_e - c_{ei}$, *is the partial derivative of the objective function* (5) *with respect to* x_{ei} *after* (4) *has been used to eliminate the* y_i *terms from* (5).

Proof. The equilibrium prices and flows within coalition I are an optimal solution to the subproblem derived by restricting (P) to the nodes of I and all the arcs directly connecting them. Consequently, the price p_i is the shadow price associated with eq. (4), and hence is the partial derivative of the optimal objective of this subproblem with respect to the right hand side of (4), which is x_{ei}. Similarly, p_e is the *negative* of the partial derivative of the corresponding subproblem defined on E (since positive x_{ei} *reduces* the available stock at node e). The change in total objective function of P with respect to x_{ei} is the sum of the changes of the two subproblem objective functions less the transportation cost $c_{ei}x_{ei}$.

We can now state a convergence result:

Theorem 1. *The Algorithm* A1 *reaches an equilibrium solution in a finite number of operations.*

Proof. Each step is finite. Violation of the condition (2) implies the partial derivative of the objective function with respect to x_{ij} is positive, so a positive objective increase will occur if x_{ij} is increased by a positive amount in step 2. Eq. (12) shows that δ_e and δ_i are positive and (13) shows the new equilibrium flow implied by step 2b is positive. The possible break up of the coalition in step 2c can only occur at a positive flow value x_{ei}, since internal flows within a coalition are always positive. Hence the objective is strictly increasing on each cycle of steps 1–2. The equilibrium prices and flows within each coalition are uniquely determined, consequently different values of the objective correspond to different coalition structures. Since only a finite number of possible coalitions exist, the proof is complete.

The reader will be struck by similarities with the primal simplex algorithm of linear programming, both in the motivation of the improvement step and the method of proof of finite convergence. Note, however, major differences. The set of arcs which can carry positive flow is, in general, not a basis for the primal problem (it is only a tree for the complete graph if all nodes are in the same coalition). Furthermore, increasing flow on a profitable arc may result in zero, one, or several arcs being reduced to zero flow.

4. An alternative algorithm

Some computational experience with Algorithm A1 suggests that during step 2 the increase in flow along arc (ei) is rarely limited by the feasibility test of step 2c. This observation suggests that some computations may be saved if the requirement that x_{ij} be nonnegative is relaxed at least temporarily and then enforced only when necessary, as in the following.

Algorithm A2.

Step 0. Initialization: set $p_i = a_i$, $y_i = 0$, $i = 1, 2, \dots, N$; $x_{ij} = 0$, all (ij); list L = empty set.

Step 1. Find a pair of nodes j, k such that $p_j + c_{jk} < p_j$. If no such pair exist, go to step 3.

Step 2. Same as step 2, Algorithm A1 *except* the feasibility test of step 2c is made only for arcs (jk) in list L. Update node prices.

Step 3a. For each coalition of more than 2 nodes, compute y_j from (1), x_{jk} from (14). If all arc flows are nonnegative, stop. The solution is equilibrium.

Step 3b. If negative flow exists in any coalition, split the coalition at the arc (jk) of most negative flow. Compute new node prices in each component as follows: let A be the subtree containing j, B the subtree containing k. Add δ_j and δ_k to all prices in A and B respectively, where

$$\delta_j = -x_{jk} / \sum_{i \in A} \beta_i$$

$$\delta_k = x_{jk} / \sum_{i \in B} \beta_i.$$

Add arc (jk) to list L. Compute net imports and flows in the newly formed coalitions, and repeat step 3b. After all negative arc flows have been eliminated by splitting coalitions, go to step 1.

Theorem 2. *The algorithm* A2 *obtains an equilibrium solution in a finite number operations.*

Proof. At worst, steps 1, 2 and 3 are repeated till all arcs appear on the list L, at which point Algorithm A2 reduces to A1. When the algorithm terminates, all equilibrium conditions are satisfied; step 3 guarantees nonnegativity of flows, step 1 tests for dual feasibility and all other equilibrium conditions are maintained at all steps.

The three growing algorithm A2 has been used to solve a few small test problems. For these problems, no use was made of the list L because an equilibrium solution was reached after two cycles of steps 1–3. The arcs deleted in step 3b of cycle 1 were not selected in step 1 of cycle 2.

While direct comparison of computing times of algorithm A2 with general purpose quadratic programming algorithms have not been made, it seems at least plausible that taking advantage of the forest structure of an optimal solution and using network concepts will result in a more efficient solution procedure.

References

[1] C.R. Glassey, "Explicit duality for convex homogeneous programs", *Mathematical Programming* 10 (1976) 176–191.

[2] P. Samuelson, "Spatial price equilibrium and linear programming", *American Economic Review* 42 (1952) 283–303.

[3] T. Takayama and G.G. Judge, *Spatial and temporal price and allocation models* (North-Holland, Amsterdam, 1971).