An Experimental Study of Different Approaches to Solve the Market Equilibrium Problem

BRUNO CODENOTTI
IIT-CNR
and
BENTON MCCUNE, SRIRAM PEMMARAJU, RAJIV RAMAN,
and KASTURI VARADARAJAN
University of Iowa

3.3

Over the last few years, the problem of computing market equilibrium prices for exchange economies has received much attention in the theoretical computer science community. Such activity led to a flurry of polynomial time algorithms for various restricted, yet significant, settings. The most important restrictions arise either when the traders' utility functions satisfy a property known as *gross substitutability* or when the initial endowments are proportional (the Fisher model). In this paper, we experimentally compare the performance of some of these recent algorithms against that of the most used software packages. In particular, we evaluate the following approaches: (1) the solver PATH, available under GAMS/MPSGE, a popular tool for computing market equilibrium prices; (2) a discrete version of a simple iterative price update scheme called tâtonnement; (3) a discrete version of the welfare adjustment process; (4) convex feasibility programs that characterize the equilibrium in some special cases. We analyze the performance of these approaches on models of exchange economies where the consumers are equipped with utility functions, which are widely used in real world applications. The outcomes of our experiments consistently show that many market settings allow for an efficient computation of the equilibrium, well beyond the restrictions under which the theory provides polynomial time guarantees. For some of the approaches, we also identify models where they are are prone to failure.

Categories and Subject Descriptors: F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems; G.4 [**Mathematical Software**]; J.4 [**Social and Behavioral Sciences**]: *Economics*

A preliminary version of this paper appears in [Codenotti et al. 2005a]. This paper also includes additional results from [Codenotti et al. 2005c].
The work of B. McCune, R. Raman, and K. Varadarajan was supported by NSF CAREER award CCR-0237431.
Authors' addresses: Bruno Codenotti, IIT-CNR, Pisa, Italy; email: bruno.codenotti@iit.cnr.it. Benton McCune, Rajiv Raman, Sriram Pemmaraju, and Kasturi Varadarajan, Department of Computer Science, The University of Iowa, Iowa City IA 52242-1419; email: [bmccune, rraman, sriram, kvaradar]@cs.uiowa.edu.

General Terms: Algorithms, Economics, Experimentation

Additional Key Words and Phrases: Market equilibrium

ACM Reference Format:

Codenotti, B., McCune, B., Pemmaraju, S., Raman, R., and Varadarajan, K. 2008. An experimental study of different approaches to solve the market equilibrium problem. ACM J. Exp. Algor. 12, Article 3.3 (August 2008), 21 pages DOI = 10.1145/1227161.1402295 http://doi.acm.org/10.1145/1227161.1402295

1. INTRODUCTION

In its exchange version, the market equilibrium problem consists of finding prices and allocations of goods to traders such that each trader maximizes her utility function and the market clears (see below for precise definitions). A fundamental result in economic theory states that, under mild assumptions, market clearing prices exist [Arrow and Debreu 1954].

Soon after this existential result was shown, researchers started analyzing economic processes leading to the equilibrium. The most popular of these is *tâtonnement*, which, starting from an arbitrary price vector, updates it according to the market excess demand generated by such prices [Arrow et al. 1959; Arrow and Hurwicz 1960]. In its continuous version, the tâtonnement process is known to converge [Arrow et al. 1959] whenever the market satisfies *weak gross substitutability* (see below for the definition). However, it need not converge if the market does not satisfy this property (see Mas-Colell et al. [1995], Chapter 17).

The failure of tâtonnement to provide global convergence stimulated a substantial amount of work on the computation of the equilibrium. Scarf and some coauthors developed pivoting algorithms, which search for an equilibrium within the simplex of prices [Scarf 1967; Scarf 1982; Hansen and Scarf 1973; Eaves and Scarf 1976]. Unlike tâtonnement, these algorithms always reach a solution, but they lack a clear economic interpretation and require exponential time, even on certain simple instances.

Motivated by the lack of global convergence of tâtonnement and by the lack of a clear economic interpretation for Scarf's methods, Smale [1976] developed a global Newton's method for the computation of equilibrium prices. His approach provides a price-adjustment mechanism, which takes into account all the components of the Jacobian of the excess demand functions. However, Smale's technique does not come with polynomial time guarantees and its behavior, when the current price is far from equilibrium, seems complicated. For this reason, most solvers, based on Newton's method, including PATH, the solver used in Section 3 and which is available under the popular GAMS framework, do a line search within each Newton's iteration, in order to guarantee that some progress is made even far from equilibrium (see Ferris and Munson [2000a, 2000b]; Ferris et al. [2000]).

A different line of work has attempted to take advantage of the convexity of the set of equilibrium prices in certain exchange markets. For example, in Arrow et al. [1959] it is shown that when the market satisfies weak gross substitutability, a fundamental inequality holds, which defines an infinite collection of hyperplanes that separates equilibrium prices from the rest. A stream of work has extended this characterization to handle settings where the demand need not be a single-valued function of the prices. These settings include, in particular, the case of linear utility functions (see Newman and Primak [1992]; Primak [1984, 1993] and the references therein). Some of these papers build upon the characterization above to propose ellipsoid and cutting-plane algorithms to compute the equilibrium.

There has also been some work on writing the equilibria for certain special exchange economies as solutions to explicit convex programs—Nenakov and Primak [1983] for linear and Cobb–Douglas utilities, and Eaves [1985] for Cobb–Douglas utilities.

Another family of computational techniques follow from Negishi's characterization of the market equilibrium as the solution to a welfare maximization problem, where the *welfare function* that is maximized is a linear combination of individual utility functions obtained by using certain positive weights [Negishi 1960]. This characterization transforms the problem of computing equilibrium prices into the problem of computing the appropriate weights of the linear combination mentioned above. For this computation, there is a natural *welfare adjustment* process or *joint maximization* procedure that works in the space of the weights in a manner that is analogous to how the tâtonnement process works, in the space of prices. As a result, this process is convergent under conditions similar to those implying the convergence of the tâtonnement process [Mantel 1971].

Recently, the question of when the market equilibrium problem can be solved in polynomial time has received considerable attention, starting with the work of Deng et al. [2002]. The focus has been on isolating restrictive, yet important, families of markets for which the problem can be solved in polynomial time [Devanur et al. 2002; Jain 2004, Jain et al. 2003, Codenotti et al. 2005e, Codenotti and Varadarajan 2004, Codenotti et al. 2005d, Garg and Kapoor 2004, Garg et al. 2004, Ye 2005]. Several techniques have emerged in the process: primal–dual methods, auction–based algorithms, variants of welfare adjustment, tâtonnement, and convex programming formulations. (See Codenotti et al. [2004] for a review of this body of work.)

This paper aims to complement the flurry of recent theoretical advances in the design of polynomial time algorithms for the market equilibrium problem with an experimental investigation. The specific goal of this paper is to comparatively study four approaches to the problem.

1. The popular software tool based on the modeling language GAMS (short for "General Algebraic Modeling System") and specifically its subsystem MPSGE (short for "Mathematical Programming System for General Equilibrium Analysis"). GAMS/MPSGE is the most commonly used tool for practical applications involving the solution of market equilibrium problems. The solver we used for the market equilibrium problem within the GAMS/MPSGE framework is the Newton-based solver PATH [Ferris et al. 2000].

2. A version of the tâtonnement process. The continuous tâtonnement process converges for markets satisfying weak gross substitutability and is particularly attractive because of its simplicity. The main question is whether the theoretically well-understood continuous tâtonnement process can be turned into a simple discrete algorithm that has good convergence properties.
3. The sequential joint maximization algorithm of Rutherford [1999b]. Such an algorithm roughly corresponds to "Algorithm 2" in Jain et al. [2003] that computes an approximate equilibrium in an exchange market by iteratively solving a special case of exchange, which arises when the initial endowments are collinear (a.k.a. Fisher's model). Algorithm 2 in Jain et al. [2003] does not fit perfectly into the framework of sequential joint maximization, because it uses an extra fictitious trader.
4. Solving convex programming formulations for the market equilibrium problem for some speical cases [Eisenberg 1961; Codenotti et al. 2005b]. In the experiments reported in this paper, we use the "convex" option in the general purpose nonlinear solver LOQO, in combination with AMPL, its modeling language.

In our experiments, the utility functions of the traders are derived from the family of CES and nested CES functions. (See below for the definitions.) Our main motivation for studying these functions is that they are widely used to model production and consumption [Kehoe et al. 2005; Shoven and Whalley 1992]. Our main observations are:

1. The PATH solver (used within GAMS) typically converges when applied to CES functions. For certain choices of market types with nested CES functions, however, the PATH solver exhibits a large variance in performance and often fails to converge when applied to exchange economies with nested CES functions.
2. The convex programming approach, which is applicable to a subclass of CES functions [Codenotti et al. 2005b], compares favorably against PATH and seems to be competitive in terms of scalability.
3. With an appropriately chosen price update rule, the discrete version of tâtonnement performed remarkably well on CES functions and, in particular, scaled well compared to PATH. On nested CES functions, however, the tâtonnement algorithm also often fails to converge for certain market types, although it generally converges for most market types. The market types on which tâtonnement performs well complement those on which PATH performs well.
4. The welfare adjustment process is almost always convergent on CES exchange economies, where the initial endowment vectors are almost proportional. Even when the initial endowments are farthest from being proportional, the process converges, although with a significantly larger number of iterations, for all but a few market types.

Theoretical studies of the market equilibrium problem (for the exchange model) have revealed that polynomial time solvability depends very much on the nature of the utility functions and the nature of the initial endowments. In the case of CES functions, for example, gross substitutability implies polynomial time solvability whenever the *elasticity of substitution* (see below for the definition) is greater than or equal to one; the existence of convex feasibility formulations implies polynomial time solvability when the elasticity is greater than or equal to $\frac{1}{2}$ [Codenotti et al. 2005b]; when the elasticity is smaller than $\frac{1}{2}$ we do not know much about polynomial time solvability, but we do know that convex feasibility formulations are ruled out, because multiple disconnected equilibria can occur. For the extreme case of Leontief utility functions (zero elasticity), we know that polynomial time solvability is unlikely because the problem turns out to be *PPAD*-complete (see Papadimitriou [1994] for the definition of the complexity class *PPAD*).

In the special case where the initial endowments of the traders happen to be proportional, we have polynomial time solvability for all CES functions because of Eisenberg's [1961] convex program.

This background led us to the formulation of different input market types, and motivated us to study how the experimental performance of each algorithm varies with the market type.

Prior to this paper, there has been some work analyzing the practical performance of different algorithms for the market equilibrium problem. In Cheng and Wellman [1998] the performance of a distributed implementation of tâtonnement is discussed; in Billups et al. [1997] several complementarity solvers are implemented and their relative merits analyzed, while in Harker and Xiao [1990] the efficiency of Newton's method is investigated. In Bagirov and Rubinov [2001] an approach based on global minimization is illustrated and the outcomes of some numerical experiments are reported. More recently, in Esteban-Bravo [2004] the performance of interior point methods has been analyzed and computational data have been obtained for some small-scale benchmarks. A common feature of the experiments reported in these works is that the sizes of the problems considered were quite small. To the best of our knowledge, this paper provides the first attempt at an experimental evaluation of different algorithms for large-scale problems.

The rest of this paper is organized as follows. In Section 2, we provide the basic definitions, introduce the market models, describe the different types of input data, and the computational settings used for the experiments.

In Section 3, we present the results of the experimental work performed using the PATH solver, available under the GAMS/MPSGE system. We identify settings where PATH consistently performs well, and, by contrast, scenarios where the running-time of PATH is less predictable.

In Section 4, we analyze some simple price update schemes, which are discrete versions of the tâtonnement process, and show that, for certain market types, they rapidly converge well beyond what the theory predicts.

In Section 5, we describe the outcomes of the experiments done using the *sequential joint maximization* algorithm, which is based on Negishi's [1960] approach for establishing the existence of the equilibrium. This algorithm seems to converge in a few iterations, even for settings where the theory does not guarantee convergence. However the experiments have been limited to medium size instances, because of the fact that each iteration requires solving a convex program with a large number of variables.

Finally, in Section 6, we report on an experimental study of some of the convex-programming-based approaches for computing equilibria in various special cases. Here, too, the experiments on large-scale problems have not been possible, for the same reason as above.

2. DEFINITIONS AND MARKET MODELS

Let us consider an exchange economy with m economic agents which represent traders of n goods. Let $\mathbf{R}^n_+$ denote the subset of $\mathbf{R}^n$ consisting of the points with nonnegative coordinates. The jth coordinate in $\mathbf{R}^n$ will stand for good j. Each trader i has a concave utility function $u_i : \mathbf{R}^n_+ \to \mathbf{R}_+$, which represents her preferences for the different bundles of goods, and an initial endowment of goods $w_i = (w_{i1}, \ldots, w_{in}) \in \mathbf{R}^n_+$. At given prices $\pi \in \mathbf{R}^n_+$, trader i will sell her endowment and get the bundle of goods $x_i = (x_{i1}, \ldots, x_{in}) \in \mathbf{R}^n_+$, which maximizes $u_i(x)$ subject to the budget constraint[1] $\pi \cdot x \le \pi \cdot w_i$. Let $W_j = \sum_i w_{ij}$ denote the total amount of good j in the market.

An *equilibrium* is a vector of prices $\pi = (\pi_1, \ldots, \pi_n) \in \mathbf{R}^n_+$ at which there is a bundle $\bar{x}_i = (\bar{x}_{i1}, \ldots, \bar{x}_{in}) \in \mathbf{R}^n_+$ of goods for each trader i such that the following two conditions hold: (1) For each good j, $\sum_i \bar{x}_{ij} \le W_j$ and (2) for each trader i, the vector $\bar{x}_i$ maximizes $u_i(x)$ subject to the constraints $\pi^T x \le \pi^T w_i$ and $x \in \mathbf{R}^n_+$.

The celebrated result of Arrow and Debreu [1954] states that, under quite mild assumptions, such an equilibrium exists. A special case occurs when the initial endowments are *collinear*, i.e., when $w_i = \delta_i w$, $\delta_i > 0$, so that the relative incomes of the traders are independent of the prices. This special case is called the *Fisher setting*, because it is equivalent to the Fisher model, which is a market of n goods desired by m utility maximizing buyers with fixed incomes.

In the standard account of the Fisher model, each buyer has a concave utility function $u_i : \mathbf{R}^n_+ \to \mathbf{R}_+$ and an endowment $e_i > 0$ of *money*. There is a seller with an amount $q_j > 0$ of good j. An equilibrium in the Fisher setting is a nonnegative vector of prices $\pi = (\pi_1, \ldots, \pi_n) \in \mathbf{R}^n_+$ at which there is a bundle $\bar{x}_i = (\bar{x}_{i1}, \ldots, \bar{x}_{in}) \in \mathbf{R}^n_+$ of goods for each buyer i such that the following two conditions hold:

1. The vector $\bar{x}_i$ maximizes $u_i(x)$ subject to the constraints $\pi \cdot x \le e_i$ and $x \in \mathbf{R}^n_+$.
2. For each good j, $\sum_i \bar{x}_{ij} = q_j$.

Unless otherwise stated, any mention of the Fisher model refers to this standard account.

For any price vector π, the vector $x_i(\pi)$ that maximizes $u_i(x)$ subject to the constraints $\pi^T x \le \pi^T w_i$ and $x \in \mathbf{R}^n_+$ is called the *demand* [2] of trader i at prices

[1]Given two vectors x and y, we use $x \cdot y$ or $x^T y$ to denote their inner product.
[2]In the definitions, we assume that the demand is a single-valued function of the prices, which is the case with most of the commonly used utility functions. The definitions can be appropriately generalized to handle the case when the demand is a multivalued function, which happens, for instance, when the utility functions are linear.

π. The *excess demand* of trader i is $z_i(\pi) = x_i(\pi) - w_i$. Then $X_k(\pi) = \sum_i x_{ik}(\pi)$ denotes the *market demand* of good k at prices π, and $Z_k(\pi) = X_k(\pi) - W_k = \sum_i z_{ik}(\pi)$ the *market excess demand* of good k at prices π. The vectors $X(\pi) = (X_1(\pi), \ldots, X_n(\pi))$ and $Z(\pi) = (Z_1(\pi), \ldots, Z_n(\pi))$ are called *market demand* (or aggregate demand) and *market excess demand*, respectively.

In terms of the excess demand function, the equilibrium is defined as a vector of prices $\pi = (\pi_1, \ldots, \pi_n) \in \mathbf{R}^n_+$, such that $Z_j(\pi) \leq 0$, for each j.

The property of *gross substitutability* (GS) plays a significant role in the theory of equilibrium and in related computational results. A market is said to satisfy GS (resp. weak GS) if for any two sets of prices π and π', such that $0 < \pi_j \leq \pi'_j$, for each j, and $\pi_j < \pi'_j$ for some j, we have that $\pi_k = \pi'_k$ for any good k implies $Z_k(\pi) < Z_k(\pi')$ (resp. $Z_k(\pi) \leq Z_k(\pi')$). That is, increasing the prices for some of the goods while keeping some others fixed can only cause an increase (resp. cannot cause a decrease) in demand for the goods whose price is fixed.

In this paper, we address the computational problem of finding the equilibrium price vector for an exchange economy given the initial endowment and an appropriate representation of the utility function for each trader.

2.1 Utility Functions

An important aspect of our experiments is the generation of markets with enough variety so as to represent a wide range of phenomenon. The utility function of every agent is a generalization of the *constant elasticity of substitution* (CES) functional form. A CES function is a concave function defined as

$$u(x_1, x_2, \ldots, x_n) = \left(\sum_{j=1}^{n} \alpha_j^{\frac{1}{\sigma}} x_j^{\frac{\sigma-1}{\sigma}} \right)^{\frac{\sigma}{\sigma-1}}$$

where $\alpha_j \geq 0$ for each j and $\sigma > 0, \sigma \neq 1$. The α_j's and σ are parameters that can be assigned values to obtain different utility functions. The parameter σ represents the *elasticity of substitution*, a natural measure of the curvature of the indifference curves of the utility function. We call an *elastic market* a market where consumers are highly sensitive to price changes. In the case of CES functions, this happens when all the consumers have a utility function with elasticity of substitution at least one. The CES functions range from *linear utility functions* (when $\sigma \to \infty$) that are fully elastic to *Leontief functions* (when $\sigma \to 0$) that are completely inelastic. When the utility function is linear, goods are perfect substitutes and when the utility function is Leontief, goods are perfect complements. In between, when $\sigma \to 1$, CES functions become the *Cobb–Douglas functions* that express a balance between substitution and complementarity effects. While σ models the elasticity of substitution, the α_j's capture how much an agent desires good j. CES functions are ubiquitous in economics literature, because of their power to express a wide variety of substitution and complementarity effects as well as their mathematical tractability, which allows for explicit computation of the associated demand function.

In our experiments, the traders' utility functions $u(x)$ are chosen from a simple family of nested CES functions that generalize CES functions. Let $u^1(x)$

be a CES function with elasticity σ_b that depends only on the quantity of the first $\lfloor n/3 \rfloor$ goods. That is,

$$u^1(x_1, x_2, \ldots, x_n) = \left(\sum_{j=1}^{\lfloor n/3 \rfloor} \alpha_j^{\frac{1}{\sigma_b}} x_j^{\frac{\sigma_b - 1}{\sigma_b}} \right)^{\frac{\sigma_b}{\sigma_b - 1}}$$

where each $\alpha_j \geq 0$. Similarly, let

$$u^2(x_1, x_2, \ldots, x_n) = \left(\sum_{j=\lfloor n/3 \rfloor + 1}^{\lfloor 2n/3 \rfloor} \alpha_j^{\frac{1}{\sigma_b}} x_j^{\frac{\sigma_b - 1}{\sigma_b}} \right)^{\frac{\sigma_b}{\sigma_b - 1}}$$

and

$$u^3(x_1, x_2, \ldots, x_n) = \left(\sum_{j=\lfloor 2n/3 \rfloor + 1}^{n} \alpha_j^{\frac{1}{\sigma_b}} x_j^{\frac{\sigma_b - 1}{\sigma_b}} \right)^{\frac{\sigma_b}{\sigma_b - 1}}$$

Then

$$u(x) = \left(u^1(x)^{\frac{\sigma_t - 1}{\sigma_t}} + u^2(x)^{\frac{\sigma_t - 1}{\sigma_t}} + u^3(x)^{\frac{\sigma_t - 1}{\sigma_t}} \right)^{\frac{\sigma_t}{\sigma_t - 1}}$$

for some elasticity σ_t. We refer to $u()$ as a two-level nested CES function with three nests, the *bottom elasticities* being all equal to σ_b, and the *top elasticity* being σ_t. Note that if the bottom and top elasticities are equal, then u becomes a CES function with elasticity $\sigma_b = \sigma_t$.

Nested CES functions are used extensively to model both production and consumption in applied general equilibrium: We refer the reader to the book by Shoven and Whalley [1992] for a sense of their pervasiveness. The popular modeling language MPSGE [Rutherford 1999a] uses nested CES functions to model production and consumption.

2.2 Input Generators

Assuming that we use two-level nested CES functional forms, as described above, to represent agent's preferences, generating a market corresponds to generating α_j's, σ_t, σ_b, and the endowments for each agent. Let m be the number of agents and n the number of goods. For $1 \leq i \leq m$, $1 \leq j \leq n$, let α_{ij} denote the coefficient $\alpha_j^{\frac{1}{\sigma_b}}$ of the term $x_j^{\frac{\sigma_b}{\sigma_b - 1}}$ in agent i's utility function. For notational convenience, let A denote the $m \times n$ matrix of the α_{ij}'s. We will call this the *desirability matrix*, since it represents the distribution of agents extent of desire for different goods. Without loss of generality, we can assume that the α_{ij}'s are normalized so that the entries in each row in A sum to 1. Let W denote the $m \times n$ matrix of endowments. Without loss of generality, we assume that endowments are normalized so that entries in W are are in the range $[0, 1]$ and all column sums are 1. This implies that the total quantity of each good is one. While the σ_t (and σ_b) values for different agents can be different, in general, for our experiments, we typically assume that these are all identical. Thus generating a market corresponds to generating $m \times n$ matrices A and W and the two values σ_t and σ_b. We generate matrices A and W independently, using several generators we have implemented.

Generators for the Desirability Matrices. We have implemented several generators for the desirability matrix A.

- *Uniform Generator*. This constructs matrices A such that each agent's desire is uniformly distributed among the n goods. Specifically, each row in A is chosen independently by first picking uniformly a random vector from $[0, 1]^n$ and then normalizing so that the sum of the numbers in the vector is 1.
- *Concentrated Generator*. This constructs matrices in which for $1 \leq i \leq m$, agent i desires a fraction 0.8 of good $n - i$. That is, $\alpha_{i,n-i} = 0.8$. Two goods j_1 and j_2 are chosen at random from among the other goods, and their desires are set to 0.1 each. The goods j_1 and j_2 are chosen at random with replacement and so they may be identical in which case the agent's desire for this good is just 0.2.[3] This generator assumes that[4] $m \leq n$.
- *Sharply Concentrated Generator*. The desire of agent i is 1 for good i and zero for the remaining goods. This generator assumes that $m \leq n$.
- *Subset Generator*. For each agent i, a random subset J_i of the goods with expected size $n/4$ is chosen. Agent i's desire for each good outside J_i is set to 0 and the rest of the mass is distributed uniformly among the goods in J_i. Rows in the matrix are generated independently of each other.
- *Introducing Correlation in Desires*. In the description of the uniform generator and the subset generator above, we mentioned that rows of the desirability matrix are independently generated. We get matrices with richer structure and asymmetry between goods by introducing some dependence. We explored the following type of dependence.
- *Replicated Desires*. In this case, after the first row of A is generated, the remaining rows are generated by simply copying the first row.

By combining desirability matrices of two types, we can get a new type of desirability matrix. For example, we can generate a desirability matrix A_1 using the sharply concentrated generator, a desirability matrix A_2 using the subset generator, pick a parameter β in the range $[0, 1]$, and output $\beta A_1 + (1 - \beta)A_2$. We do, in fact, use such combinations in some our experiments.

- *Generators for the Endowment Matrices*. We have implemented several generators for the endowment matrix that are very similar to the generators for the desirability matrix. We have a uniform and the subset generator for endowment matrices that are identical to the corresponding generators for the desirability matrices, except that now columns are independently generated, instead of rows. (Recall that the entries in each column add up to 1.) The sharply concentrated generator for the endowment matrix assumes that $m \geq n$ and gives the entire 1 unit of the jth good to the jth trader. The concentrated generator for the endowment matrix also assumes that $m \geq n$. It takes the output of the concentrated generator and perturbs it so that the off-diagonal entries are small positive numbers.

[3]To ensure the existence of an equilibrium, we sometimes perturb the desirability matrix so that each entry is at least some very small positive number ε.
[4]In most of our experiments, we have $m = n$.

Some dependence can also be introduced among the columns of the endowment matrix with the uniform and subset generators. The main kind of dependence we use is that of replicated columns—having generated the first column, the remaining columns are simply copies of it. Note that this leads to the situation where the endowments of the agents are proportional—the Fisher model.

Finally we may combine the output of two generators, as indicated for the desirability matrix, to get a new type of generator.

2.3 Computational Environment

All of our experiments were performed on a machine with an AMD Athlon, 64-bit, 2202.865 MHz processor, 2 GB of RAM, and running Red Hat Linux Release 3, Kernel Version—2.4.21–15.0.4

—The experiments in Section 3 use the mixed-complementarity solver PATH that is available under GAMS, which is a modeling system for mathematical programming problems. GAMS consists of a language compiler integrated with high-performance solvers, and it is tailored for complex, large-scale modeling applications (see Brooke et al. [1988]). GAMS has been extended to GAMS/MPSGE by Rutherford [1995] to easily handle economic equilibrium problems. The web site `http://www.gams.com/solvers/mpsge/pubs.htm` lists a number of scientific papers, which have been using MPSGE.

—The experiments in Sections 5 and 6 involve solving convex programs. For this task, we used the general nonlinear solver LOQO [Vanderbei 2000], which is based on an infeasible primal-dual interior point method, and runs faster on convex programs (for which it can be run with the “convex option” on) than on nonconvex ones.

3. THE PERFORMANCE OF AN EFFICIENT GENERAL PURPOSE SOLVER

In this section, we present the outcomes of the experiments we have carried out with the PATH solver available under GAMS/MPSGE.

PATH is a sophisticated solver, based on Newton’s method, which is the most used technique to solve systems of nonlinear equations [Dirkse and Ferris 1996; Ferris and Munson 2000a]. Newton’s method constructs successive approximations to the solution and works very well in the proximity of the solution. However, there are no guarantees of progress far from the solution. For this reason, PATH combines Newton’s iterations with a line search, which makes sure that at each iteration the error bound decreases, by enforcing the decrease of a suitably chosen *merit function*. This line search corresponds to a linear complementarity problem, which PATH solves by using a pivotal method [Ferris and Munson 2000b].

Our experiments with GAMS/PATH are of two kinds. The first kind aims at understanding the sensitivity of PATH to specific market types and parameter ranges of interest. The second kind studies how the running time scales with input size.

	0.1	0.3	0.5	0.9	1.3	1.7
0.1	1.96	1.11	1.11	1.00	1.00	1.03
0.3	52.31	1.67	1.68	1.64	1.33	1.37
0.5	82.08	86.06	1.85	1.75	1.76	1.68
0.9	40.87	99.08	2.15	1.72	1.76	1.76
1.3	85.29	122.69	1.95	1.44	1.40	1.50
1.7	88.73	58.34	61.11	1.36	1.35	1.39

(a)

	0.1	0.3	0.5	0.9, 1.3, 1.7
0.1	0	0	0	0
0.3	5	0	0	0
0.5	3	2	0	0
0.9	5	5	0	0
1.3	5	3	0	0
1.7	4	1	1	0

(b)

Fig. 1. PATH on markets with 50 traders and goods. The desirability matrix is obtained by adding β times the output of a sharply concentrated generator and $(1-\beta)$ times the output of a subset generator, with $\beta = 0.95$. The endowment matrix is from the sharply concentrated generator. Rows of the table correspond to top elasticity σ_t, and columns to bottom elasticity σ_b. Six values—0.1, 0.3, 0.5, 0.9, 1.3, and 1.7—were chosen for these elasticities. (a) Each entry of this table corresponds to a choice of σ_t and σ_b; the number shown is the average running time in seconds over five inputs. (b) Each entry shows the number of failures out of the five runs.

	0. 1	0.3	0.5	0.9	1.3	1.7
0.1	1.77	1.23	1.23	1.19	1.15	1.07
0.3	110.01	1.84	1.88	1.97	1.90	1.86
0.5	64.38	20.00	2.18	2.01	1.97	2.06
0.9	76.65	43.25	2.29	2.04	1.98	1.99
1.3	2.42	1.82	1.68	1.61	1.68	1.72
1.7	1.62	1.58	1.67	1.64	1.56	1.54

(a)

	0. 1	0.3	0.5, 0.9, 1.3, 1.7
0.1	0	0	0
0.3	5	0	0
0.5	3	1	0
0.9	3	0	0
1.3	0	0	0
1.7	0	0	0

(b)

Fig. 2. PATH on markets with 50 traders and goods. The desirability and endowment matrices are generated using the concentrated generators.

	0.1	0.3	0.5	0.9	1.3	1.7
0.1	1.25	1.04	1.03	1.04	1.04	1.04
0.3	2.60	1.37	1.24	1.25	1.16	1.08
0.5	2.04	1.58	1.45	1.24	1.25	1.25
0.9	2.85	1.74	1.66	1.58	1.41	1.33
1.3	3.13	1.77	1.91	1.78	1.54	1.49
1.7	3.12	1.84	1.90	1.98	1.66	1.70

(a)

	0.1, 0.3, 0.5, 0.9, 1.3, 1.7
0.1	0
0.3	0
0.5	0
0.9	0
1.3	0
1.7	0

(b)

Fig. 3. PATH on markets with 50 traders and goods. The desirability and endowment matrices are generated using the uniform generators.

- *Sensitivity to Market Type*. Figures 1, 2, and 3 summarize the performance of PATH on economies with 50 traders and goods for three different choices of generators for the desirability and endowment matrices. One phenomenon that clearly stands out is that the performance is quite good when the top elasticity σ_t is less than or equal to the bottom elasticity σ_b. This range includes the special case of CES functions. In contrast, the performance can be significantly worse for some instances where $\sigma_t > \sigma_b$. This phenomenon can be seen both in the running time and in the number of runs in which PATH declares a failure in computing the equilibrium.

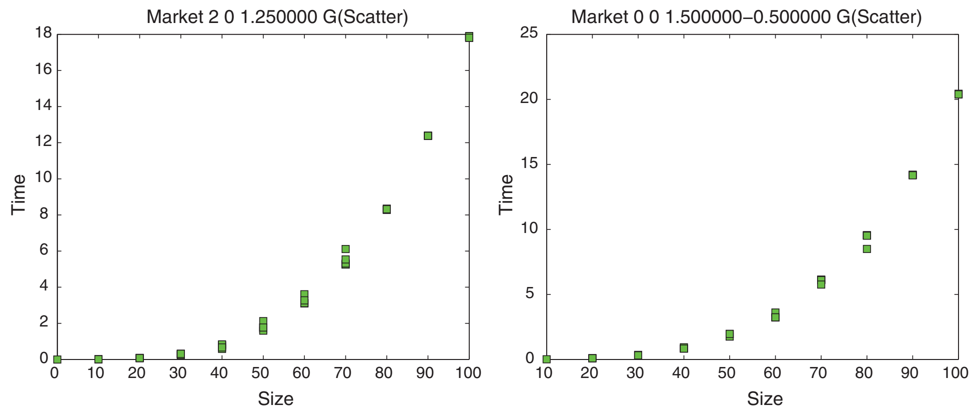


Fig. 4. The running time of PATH, in seconds, as a function of the input size ($m = n$). (a) The concentrated generator is used for the desirability matrix and the uniform generator for the endowment matrix; $\sigma_t = \sigma_b = 1.25$. (b) The uniform generator is used for both the desirability and endowment matrix; $\sigma_t = 1.5$ and $\sigma_b = 0.5$. There were five runs for each input size.

The degradation in performance is more striking for some configurations of generators than others. For instance, PATH performs much better on the configuration corresponding to Figure 3, than on the configuration corresponding to Figure 1.

- *Running Time as a Function of Input Size*. To get an estimate of how the running time of PATH varies with input size, we experimented with some generator configurations and choices of σ_t and σ_b where PATH does not report failure. Figure 4 illustrates how the running time grows fairly rapidly with size for two such parameter choices. The runs on other benchmarks (in PATH's good range) yield very similar figures.

4. AN ALGORITHM DERIVED FROM THE TÂTONNEMENT PROCESS

In 1874 Léon Walras proposed that an equilibrium price vector could be reached via a discrete price-adjustment process that he called *tâtonnement*. In Samuelson's [1947] now standard version of tâtonnement, competitive agents receive a price signal, and report their excess demands at these prices to the central auctioneer. The auctioneer then computes aggregate excess demands, adjusts the prices incrementally in proportion to the magnitude of excess demands, and announces the new incrementally adjusted price level. In each round, agents recalculate their excess demands upon receiving the newly adjusted price signal and report these to the auctioneer. The process continues until prices converge to an equilibrium.

In its continuous version, the tâtonnement process is governed by the differential equation system: $\frac{d\pi_i}{dt} = G_i(Z_i(\pi))$ for each $i = 1, 2, \ldots, n$, where $G_i()$ is some continuous and differentiable, sign-preserving function and the derivative of π_i is with respect to time. The continuous version of tâtonnement is more amenable to analysis of convergence and it is this process that is shown to be

	0. 1	0.3	0.5	0.9	1.3	1.7
0.1	0.21	100.00	100.00	0.07	3.96	5.66
0.3	0.14	0.55	75.48	0.06	6.79	2.04
0.5	0.35	0.85	0.45	0.04	5.78	2.68
0.9	0.83	0.12	0.52	0.22	4.10	4.19
1.3	0.59	0.27	1.24	0.54	0.03	0.18
1.7	0.10	0.78	0.67	1.67	0.02	0.01

(a)

	0.1	0.3	0.5	0.9, 1.3, 1.7
0.1	0	5	5	0
0.3	0	0	3	0
0.5	0	0	0	0
0.9	0	0	0	0
1.3	0	0	0	0
1.7	0	0	0	0

(b)

Fig. 5. Performance of tâtonnement on markets with 50 traders and goods. σ_t varies with the rows and σ_b with the columns. The desirability matrix is obtained by adding β times the output of a sharply concentrated generator, and $(1-\beta)$ times the output of a subset generator, with $\beta = 0.95$. The endowment matrix is from the sharply concentrated generator. (a) The number of iterations, in thousands, averaged over five runs. (b) The number of failures out of five runs.

convergent by Arrow, Block, and Hurwicz [1959] for markets satisfying weak GS.

In our implementation of tâtonnement, the starting price vector is $(1, 1, \ldots, 1)$. Let π^k be the price vector after k iterations (price updates). In iteration $(k+1)$, the algorithm computes the excess demand vector $Z(\pi^k)$ and then updates each price using the rule $\pi_i^{k+1} \leftarrow \pi_i^k + c_{i,k} \cdot Z_i(\pi^k)$. One specific choice of $c_{i,k}$ that we have used in many of our experiments is

$$c_{i,k} = \frac{\pi_i^k}{i \cdot \max_j |Z_j(\pi^k)|}$$

This choice of $c_{i,k}$ ensures that $|c_{i,k} \cdot Z_i(\pi)| \leq \pi_i^k$ and, therefore, π continues to remain nonnegative. Also noteworthy is the role of i that ensures that the "step size" diminishes as the process (hopefully) approaches the equilibrium.

Our experiments with tâtonnement were of three kinds. In the first, we attempt to understand the sensitivity of its performance, measured in terms of the number of iterations, to the market type. In the second, we study how the performance scales with size. For these two kinds of experiments, we terminated tâtonnement when $\max_i |Z_i(\pi^k)|$ fell below a threshold value of $\epsilon = 10^{-4}$ (success) or when the number of iterations exceeded 100, 000 (failure), whichever happened first. In the third kind of experiment, where we study how the performance depends on ε, we increased the limit on the number of iterations to 10 million.

- *Sensitivity to Market Type*. Figures 5 and 6 illustrate the performance of tâtonnement on economies with 50 traders and goods for two different choices of generators for the desirability and endowment matrices. As in the case of PATH, the choice of generators significantly impacts the performance of tâtonnement. In contrast with what we observed for PATH, tâtonnement performs better on markets with $\sigma_t \geq \sigma_b$ than on markets with $\sigma_t < \sigma_b$. This can be seen in Figure 5 in the number of failures as well as the number of iterations used. We observed this phenomenon consistently in generator configurations for which there was a degradation in the performance of tâtonnement.

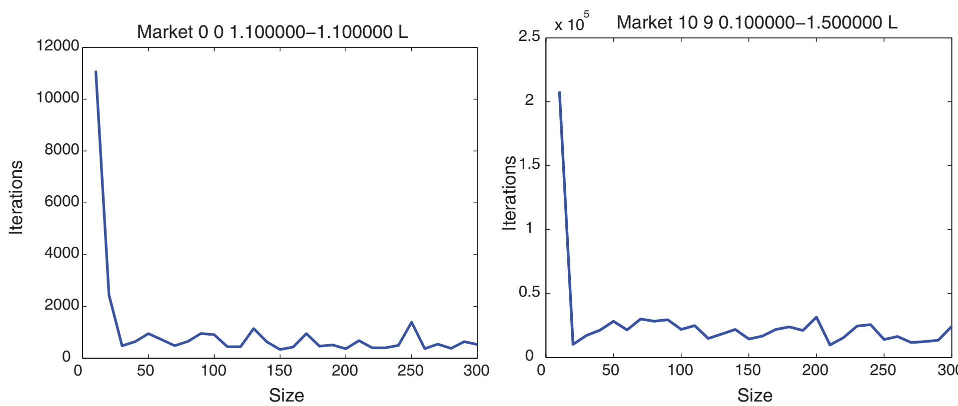
	0. 1	0.3	0.5	0.9	1.3	1.7
0.1	0.29	1.27	2.90	1.07	2.64	0.63
0.3	0.10	1.10	0.16	0.60	1.73	1.29
0.5	0.12	1.20	1.85	7.69	0.14	0.48
0.9	0.90	0.10	2.51	2.91	2.70	3.54
1.3	0.50	0.13	0.33	0.21	0.29	0.32
1.7	0.75	0.15	0.21	0.48	1.21	3.41

(a)

	0.1, 0.3, 0.5, 0.9, 1.3, 1.7
0.1	0
0.3	0
0.5	0
0.9	0
1.3	0
1.7	0

(b)

Fig. 6. Performance of tâtonnement on markets with 50 traders and goods. The desirability and endowment matrices are generated using the concentrated generators.

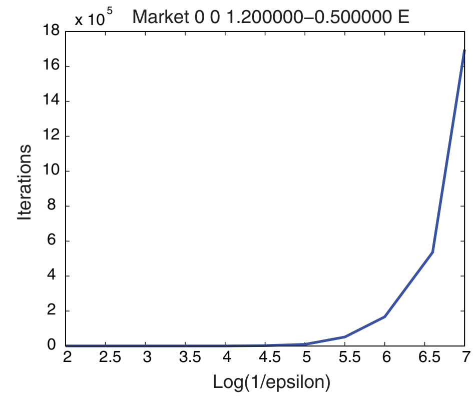


(a) (b)

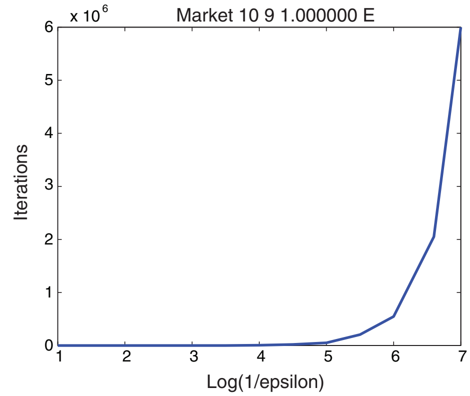
Fig. 7. The number of iterations of tâtonnement, as a function of the input size , with $m = n$. (a) The uniform generator is used for both the desirability matrix and the endowment matrix; $\sigma_t = \sigma_b = 1.1$. (b) The desirability matrix is obtained by adding β times the output of a sharply concentrated generator and $(1 - \beta)$ times the output of a subset generator, with $\beta = 0.95$. The endowment matrix is from the sharply concentrated generator; $\sigma_t = 0.1$ and $\sigma_b = 1.5$. The number of iterations for each input size is averaged over five runs.

- *Performance as a Function of Input Size*. We measured how the number of iterations taken by the tâtonnement algorithm changes with size for various configurations of the desirability and endowment generators, σ_t and σ_b. As with PATH, we focused on the parts of the configuration space where tâtonnement does not tend to fail. In these experiments, the input size is the number of traders that equals the number of goods. Quite remarkably, we find that the number of iterations does not grow significantly with input size, but stays nearly flat once the input size is beyond a certain threshold. This phenomenon happens consistently across the input configurations and is illustrated by the plots shown in Figures 7a and b. The running time, on the other hand, does grow with the input size.

At each iteration of the tâtonnement algorithm, the essential and most expensive task is the computation of the demand at the new price. The demand function for traders with CES utilities has an explicit formulation, which can



(a)



(b)

Fig. 8. The number of iterations of tâtonnement, as a function of $\log_{10}(1/\varepsilon)$, with $m = n = 50$. (a) The uniform generator is used for both the desirability matrix and the endowment matrix; $\sigma_t = 1.2$ and $\sigma_b = 0.5$. (b) The desirability matrix is obtained by adding β times the output of a sharply concentrated generator and $(1 - \beta)$ times the output of a subset generator, with $\beta = 0.95$. The endowment matrix is from the sharply concentrated generator; $\sigma_t = \sigma_b = 1.0$. The number of iterations for each input size is averaged over five runs.

be easily derived. If M_i denotes the income of the ith trader, then her demand for good j is

$$\left(\frac{\alpha_{ij}}{\pi_j}\right)^{\sigma} \frac{M_i}{\sum_{1 \le j \le n} \alpha_{ij}^{\sigma} \pi_j^{1-\sigma}}$$

Using this formula, the aggregate excess demand can be computed in $O(mn)$ time.

The same asymptotic bound applies to the computation of the demand for nested CES functions, when the number of nests is a constant, as was the case in the experiments we conducted.

- *Performance as a Function of* ε. The number of iterations of tâtonnement seems to grow rapidly with respect to $\log(\frac{1}{\epsilon})$, quite independently of the market type. Figures 8a and b show typical plots.

5. WELFARE ADJUSTMENT SCHEMES

In this section, we report on some experimental work for computing equilibria in the exchange model using the *sequential joint maximization* algorithm, which is based on Negishi's [1960] approach for establishing the existence of the equilibrium. Let R^n_{++} denote the subset of $\mathbf{R}^n$ with all positive coordinates. Let $\alpha = (\alpha_1, \ldots, \alpha_m) \in R^m_{++}$, be any vector. Consider the allocations that solve the following optimization problem over $x_i \in \mathbf{R}^n_+$:

$$\begin{aligned} \text{Maximize} \quad & \sum_{i=1}^{m} \alpha_i u_i(x_i) \\ \text{Subject to} \quad & \sum_i x_{ij} \le \sum_i w_{ij} \text{ for each good } j \end{aligned}$$

The optimal allocations $\bar{x_i}$ are called the *Negishi welfare optimum* at the *welfare weights* α_i. Let $\pi = (\pi_1, \dots, \pi_n) \in \mathbf{R}^n_+$, where the "dual price" π_j is the Lagrangian multiplier associated with the constraint in the program corresponding to the jth good. Define $B_i(\alpha) = \pi \cdot w_i - \pi \cdot \bar{x}_i$, the budget surplus of the ith trader at prices π and with allocation $\bar{x}_i$. Define $f_i(\alpha) = B_i(\alpha)/\alpha_i$, and $f(\alpha) = (f_1(\alpha), \dots, f_m(\alpha))$.

Under some standard assumptions on the utility functions, the following properties hold for the map $f : R^m_{++} \to \mathbf{R}^m$ (see Chapter 7 of the book by Ginsburgh and Waelbroeck [1981] for a systematic exposition.)

1. $f(\alpha)$ is single valued, continuous, and differentiable at each $\alpha \in \Re^m_{++}$.
2. $\sum_i \alpha_i f_i(\alpha) = 0$, which corresponds to Walras' Law.
3. For any real $\lambda > 0$, $f(\lambda\alpha) = f(\alpha)$, which is positive homogeneity.
4. There exists an $\alpha^* \in R^m_{++}$ such that $f(\alpha^*) = 0$. The corresponding dual prices constitute an equilibrium for the economy.

Properties 1–3 follow from definitions and basic optimization theory; property (4) follows from Negishi [1960] theorem. This characterization suggests an approach for finding an equilibrium by a search in the space of *Negishi weights*. This approach, which is complementary to the traditional price space search, is elaborated in Ginsburgh and Waelbroeck [1981], Mantel [1971] and Rutherford [1999b]. In particular, Mantel [1971] shows that if the utility functions are strictly concave and log-homogeneous, and generate an excess demand that satisfies gross substitutability, then we have $\frac{\partial f_i(\alpha)}{\partial \alpha_i} < 0$ and $\frac{\partial f_j(\alpha)}{\partial \alpha_i} > 0$ for $j \neq i$. This is the analog of gross substitutability in the "Negishi space." He also shows that a differential welfare-weight adjustment process, which is the equivalent of tâtonnement, converges to the equilibrium, in these situations. The related computational methods that work in the space of welfare weights to find an α^* such that $f(\alpha^*) = 0$ are usually called *welfare adjustment* or *joint maximization* methods.

If each u_i is the logarithm of a function that is homogeneous of degree one, which is the case in our experiments,[5] then a result of Eisenberg [1961] implies that the dual prices corresponding to welfare weights $\alpha = (\alpha_1, \dots, \alpha_m)$ are precisely the Fisher equilibrium prices for the model in which the traders have incomes $\alpha_1, \dots, \alpha_m$. The welfare adjustment methods can then be seen as attempting to compute an equilibrium for the exchange economy by iteratively solving Fisher instances. This idea has been explored by Ye [2005] for the case of linear utility functions. The second algorithm of Jain, Mahdian, and Saberi [2003] may also be seen in this light, although it uses an extra trader and thus does not fit directly into this framework.

We implemented an algorithm for computing the equilibrium for an exchange market that uses an algorithm for the Fisher setting as a black box. The algorithm starts off from an arbitrary initial price π^0, and computes a sequence of prices as follows. Given π^k, the algorithm sets up a Fisher instance by setting

[5]The nested CES functions we use are homogeneous of degree one; however, it is easy to verify that replacing each trader's utility function by its logarithm does not change the equilibria.

	0.0	0.2	0.4	0.6	0.8	1.0
0 .1	2	2	2	2	2	5.2
0.3	2	2	2	2	2	5
0.5	2	2	2	2	2	5
0.7	2	2	2	2	2	4.8
0.9	2	2	2	2	2	4.8
1.1	2	2	2	2	2	4.4
1.3	2	2	2	2	2	4

Fig. 9. Number of iterations of the iterative Fisher algorithm. The elasticity of the CES functions of the traders varies with the rows; β varies with the columns; each entry is the average number of iterations over five runs. We have $m = n = 25$ and the desirability matrix is computed using the uniform generator.

the money of each trader to be $e_i^k = \pi^k \cdot w_i$, where w_i is the ith trader's initial endowment. Let π^{k+1} be the price vector that is the solution of the Fisher instance with incomes $e_1^k, \ldots, e_m^k$. The goods in the Fisher instance are obtained by aggregating the initial endowment w_i of each trader. If π^{k+1} is within a specified tolerance of π^k, we stop and return π^{k+1}; one can show that π^{k+1} must be an approximate equilibrium. Otherwise, we compute π^{k+2} and proceed.

This may be seen as a version of tâtonnement in the Negishi space. We are simply performing the step $e_i^{k+1} \leftarrow e_i^k + e_i^k f_i(e^k)$.

In our implementation of the iterative Fisher algorithm, we stop when the Euclidean distance between the successive prices falls below 0.001 (success) or when the number of iterations exceeds 100 (failure). In our experiments, we studied how the convergence of the iterative Fisher algorithm varied with elasticity of the CES utility functions (we set $\sigma_t = \sigma_b$) and the initial endowments. If the elasticity is greater than 1, then gross substitutability holds, and Mantel [1971] results show that a differential version of welfare adjustment converges to the equilibrium welfare weights. On the other hand, if the initial endowments of the traders are proportional, then Eisenberg [1961] result implies that our iterative Fisher algorithm should terminate in two iterations. Setting $m = n$, we generate the endowment matrix by taking β times the output of the sharply concentrated generator plus $(1-\beta)$ times the output of a uniform generator with replicated columns (recall that this yields proportional endowments). We varied the parameter β from 0 to 1. Note that when $\beta = 0$, the initial endowments are proportional, whereas when $\beta = 1$, the initial endowments are orthogonal. We varied the elasticity of the utility functions of the traders from 0.1 to 1.3.

Figure 9 shows the result of such an experiment when the uniform generator is used for the desirability matrix. The algorithm converges in a very small number of iterations for all elasticity and β values, although the number of iterations tends to be somewhat higher when β equals 1.

Figure 10a tabulates the number of iterations where we use the concentrated generator for the desirability matrix. Note that the number of iterations is quite small when $\beta \leq 0.8$, indicating that the iterative Fisher algorithm tends to converge quickly when the initial endowments are even reasonably close to being proportional. The number of iterations is significantly larger when

	0.0	0.2	0.4	0.6	0.8	1.0
0 .1	2	2	2.4	3	3	75.2
0.3	2	2	3	3	3	87
0.5	2	2	3	3	3	47.6
0.7	2	2	2.8	3	3	48.6
0.9	2	2	2.4	3	3	46.2
1.1	2	2	2.8	3	3	56.4
1.3	2	2	2.6	3	3	62.4

(a)

	0.0	0.2	0.4	0.6	0.8	1.0
0 .1	0	0	0	0	0	3
0.3	0	0	0	0	0	2
0.5	0	0	0	0	0	0
0.7	0	0	0	0	0	0
0.9	0	0	0	0	0	0
1.1	0	0	0	0	0	0
1.3	0	0	0	0	0	0

(b)

Fig. 10. The iterative Fisher algorithm when the concentrated generator is used for the desirability matrix; $m = n = 25$. (a) The average number of iterations over five runs. (b) The number of failures out of five runs.

the initial endowments are orthogonal ($\beta = 1$). Even in this case, failure to converge tends to happen only for small elasticity values, as can be seen from Figure 10b.

These results are quite representative of the behavior we observed over different choices of generators for the desirability matrix. The explanation for these good convergence results should be read in the light of the discussion above, which shows that we are actually performing a welfare adjustment process in the Negishi space.

Ye [2005] gives an example with two traders and two goods and linear utilities for which the simple iterative Fisher algorithm described above cycles between two prices (and therefore does not converge). As explained, the iterative update in the above algorithm corresponds to making the update $\alpha_i^{k+1} \leftarrow \alpha_i^k + \alpha_i^k f_i(\alpha^k)$ in the Negishi space. It is conceivable (and probably even provable using the technology in Arrow et al. [1959]) that a differential version of the above converges to an equilibrium when the utility functions satisfy GS; note that Mantel [1971] has shown that a differential version of $\alpha_i^{k+1} \leftarrow \alpha_i^k + f_i(\alpha^k)$ converges to an equilibrium in this situation. In our experiments, we worked with the simple iterative Fisher and not a differential version, motivated by the fact that this is done in practice [Rutherford 1999b].

6. EXPLICIT CONVEX PROGRAMS

In this section, we report on an experimental study of some of the convex-programming based approaches for computing equilibria in various special cases.

6.1 The Fisher Setting

We implemented the convex program of Eisenberg [1961] for computing the equilibrium in the Fisher setting when the traders have homogeneous utility functions. The program has $n * m$ variables for a market with m traders and n goods, has n linear constraints, and a concave objective function. Our interest is in measuring how well the running time scales with size. In our experiments, we set $n = m$. Figure 11a depicts how the average running time varies with n for a a typical run where traders having CES utilities with $\sigma = 0.25$. We did not run the experiments beyond $n = 50$, since the solver LOQO took too long

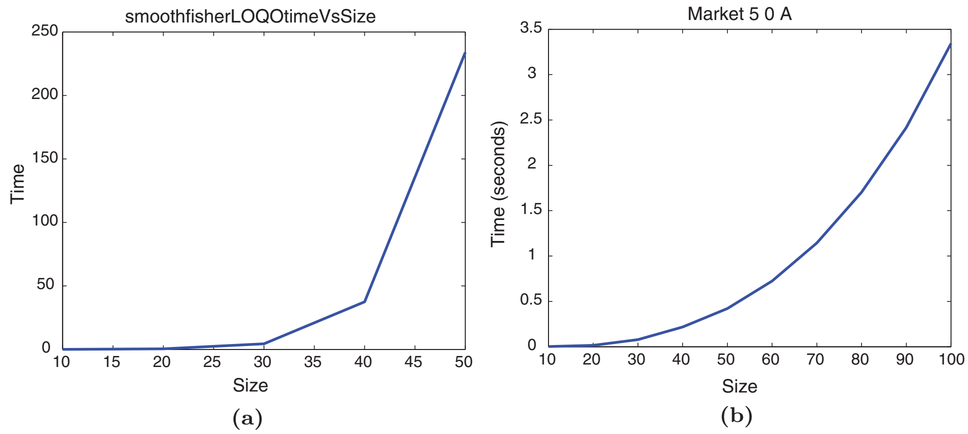


Fig. 11. Running time as a function of size for **(a)** the convex program for Fisher instances with $\sigma = 0.25$, and **(b)** for the convex program for exchange instances with $\sigma = 1.25$.

to complete. We suspect that this happens because the number of variables in the program is $n * m$.

6.2 Exchange Economies with CES Utilities

Codenotti et al. [2005b] present convex programs that characterize the equilibria in exchange economies where traders have CES functions with elasticity that lies in the range (a) $[1/2, 1)$, and (b) $(1, \infty)$. We implemented a version of their convex program when the elasticities lie in the latter range. This program has $n + m$ variables for a market with m traders and n goods and $n + m$ constraints. In our experiments, all the traders have the same elasticity and $n = m$. We measured how well the running time scales with problem size. Figure 11b, which is quite typical, depicts how the average running time varies with n for $\sigma = 1.25$. Note that the running time compares favorably with that of PATH for the same input configuration (see Figure 4a).

We also measured how the running time varies with elasticity. For a market with $n = 25$, we varied the elasticity of substitution from 1 to 20 and found that, beyond a certain point, the running time is stable and increases only very mildly with the elasticity. We experimented with different kinds of markets and found this behavior to be fairly typical.

ACKNOWLEDGMENTS

We wish to thank Thomas Rutherford for sharing his knowledge of GAMS/MPSGE with us.

REFERENCES

ARROW, K. AND DEBREU, G. 1954. Existence of an equilibrium for a competitive economy. *Econometrica 22*, 3, 265–290.

ARROW, K. J. AND HURWICZ, L. 1960. Competitive stability under weak gross substitutability: The “euclidean distance” approach. *International Economic Review 1*, 38–49.

Arrow, K. J., Block, H. D., and Hurwicz, L. 1959. On the stability of the competitive equilibrium, ii. *Econometrica 27*, 82–109.

Bagirov, A. and Rubinov, A. 2001. Global optimization of marginal functions with applications to economic equilibrium. *Journal of Global Optimization 20*, 215–237.

Billups, S., Dirkse, S., and Ferris, M. 1997. A comparison of large scale mixed complemetarity problem solvers. *Compututational Optimization and Applications 7*, 3–25.

Brooke, A., Kendrick, D., and Meeraus, A. 1988. *GAMS: A User's Guide*. The Scientific Press, San Francisco, CA.

Cheng, J. Q. and Wellman, M. P. 1998. The walras algorithm: A convergent distributed implementation of general equilibrium outcomes. *Computational Economics 12*, 1, 1–24.

Codenotti, B. and Varadarajan, K. R. 2004. Efficient computation of equilibrium prices for markets with leontief utilities. In *ICALP*. 371–382.

Codenotti, B., Pemmaraju, S., and Varadarajan, K. 2004. The computation of market equilibria. *ACM SIGACT News 35*, 4, 23–37.

Codenotti, B., McCune, B., Pemmaraju, S. V., Raman, R., and Varadarajan, K. 2005a. An experimental study of different approaches to solve the market equilibrium problem. In *ALENEX/ANALCO*. 167–179.

Codenotti, B., McCune, B., Penumatcha, S., and Varadarajan, K. 2005b. Market equilibrium for ces exchange economies: Existence, multiplicity, and computation. In *FSTTCS 2005: Foundations of Software Technology and Theoretical Computer Science*. Springer Verlag, New York. 505–516.

Codenotti, B., McCune, B., Raman, R., and Varadarajan, K. 2005c. Computing equilibrium prices: Does theory meet practice? In *Proceedings ESA 2005, 13th Annual European Symposium, Palma de Mallorca, Spain, October 3–6, 2005*. 83–94.

Codenotti, B., McCune, B., and Varadarajan, K. 2005d. Market equilibrium via the excess demand function. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*. ACM Press, New York. 74–83.

Codenotti, B., Pemmaraju, S., and Varadarajan, K. 2005e. On the polynomial time computation of equilibria for certain exchange economies. In *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics, Philadelphia, PA. 72–81.

Deng, X., Papadimitriou, C. H., and Safra, M. 2002. On the complexity of equilibria. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*. ACM Press, New York. 67–71.

Devanur, N. R., Papadimitriou, C. H., Saberi, A., and Vazirani, V. V. 2002. Market equilibrium via a primal-dual-type algorithm. In *43rd Symposium on Foundations of Computer Science*. IEEE Computer Science, Los Alamitos, CA. 389–395.

Dirkse, S. and Ferris, M. 1996. A pathsearch damped newton method for computing general equilibria. *Annals of Operations Research*, 211–232.

Eaves, B. C. 1985. Finite solution of pure trade markets with cobb-douglas utilities. *Mathematical Programming Study 23*, 226–239.

Eaves, B. C. and Scarf, H. 1976. The solution of systems of piecewise linear equations. *Mathematics of Operations Research 1*, 1, 1–27.

Eisenberg, E. 1961. Aggregation of utility functions. *Management Sciences 7*, 4, 337–350.

Esteban-Bravo, M. 2004. Computing equilibria in general equilibrium models via interior-point methods. *Computational Economic 23*, 147–171.

Ferris, M. C. and Munson, T. S. 2000a. Complementarity problems in gams and the path solver. *Journal of Economic Dynamics and Control 24*, 165–188.

Ferris, M. C. and Munson, T. S. 2000b. *Path 4.6*.

Ferris, M. C., Munson, T. S., and Ralph, D. 2000. *A Homotopy Method for Mixed Complementarity Problems Based on the PATH Solver*. Chapman and Hall, London. 143–167.

Garg, R. and Kapoor, S. 2004. Auction algorithms for market equilibrium. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing, Chicago, IL, June 13–16, 2004*. ACM, New York. 511–518.

Garg, R., Kapoor, S., and Vazirani, V. 2004. An auction-based market equilibrium algorithm for the separable gross substitutability case. In *Proceedings of the Approximation, Randomization, and Combinatorial Optimization, Algorithms and Techniques, 7th International Workshop on*

Approximation Algorithms for Combinatorial Optimization Problems, APPROX 2004, and 8th International Workshop on Randomization and Computation, RANDOM 2004, Cambridge, MA, August 22–24, 2004, Springer, New York. 128–138.

GINSBURGH, V. A. AND WAELBROECK, J. L. 1981. *Activity Analysis and General Equilibrium Modelling*. North Holland, Amsterdam.

HANSEN, T. AND SCARF, H. 1973. *The Computation of Economic Equilibrium, Cowles Foundation Monograph No. 24*. Yale University Press, New Haven.

HARKER, P. AND XIAO, B. 1990. Newton's method for the nonlinear complementarity problem: A b-differential equation approach. *Mathematical Programming 48*, 339–358.

JAIN, K. 2004. A polynomial time algorithm for computing the arrow-debreu market equilibrium for linear utilities. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS'04)*. IEEE Computer Society, Los Alamitos, CA. 286–294.

JAIN, K., MAHDIAN, M., AND SABERI, A. 2003. Approximating market equilibria. In *Proceedings of the Approximation, Randomization, and Combinatorial Optimization: Algorithms and Techniques, 6th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, APPROX 2003 and 7th International Workshop on Randomization and Approximation Techniques in Computer Science, RANDOM 2003, Princeton, NJ*, *August 24–26*, *2003*, Springer, New York. 98–108.

KEHOE, T., SRINIVASAN, T., AND WHALLEY, J. 2005. *Frontiers in Applied General Equilibrium Modelling*. Cambridge University Press, Cambridge.

MANTEL, R. R. 1971. The welfare adjustment process: its stability properties. *International Economic Review 12*, 415–430.

MAS-COLELL, A., WHINSTON, M. D., AND GREEN, J. R. 1995. *Microeconomic Theory*. Oxford University Press, Oxford.

NEGISHI, T. 1960. Welfare economics and existence of an equilibrium for a competitive economy. *Metroeconomica 12*, 92–97.

NENAKOV, E. I. AND PRIMAK, M. E. 1983. One algorithm for finding solutions of the Arrow-Debreu model. *Kibernetica 3*, 127–128.

NEWMAN, D. J. AND PRIMAK, M. E. 1992. Complexity of circumscribed and inscribed ellipsoid methods for solving equilibrium economical models. *Applied Mathematics and Computation 52*, 223–231.

PAPADIMITRIOU, C. 1994. On the complexity of the parity argument and other inefficient proofs of existence. *Journal of Computer and System Sciences 48*, 498–532.

PRIMAK, M. E. 1984. An algorithm for finding a solution of the linear exchange model and the linear Arrow-Debreu model. *Kibernetika 5*, 76–81.

PRIMAK, M. E. 1993. A converging algorithm for a linear exchange model. *Journal of Mathematical Economics 22*, 181–187.

RUTHERFORD, T. 1995. Extensions of gams for complementarity problems arising in applied economic analysis. *Journal of Economic Dynamics and Control 19*, 1299–1324.

RUTHERFORD, T. 1999a. Applied general equilibrium modeling with mpsge as a gams sybsystem: An overview of the modeling framework and syntax. *Computational Economics 14*, 1–46.

RUTHERFORD, T. 1999b. *Sequential Joint Maximization*. Vol. 18. Kluwer, Boston, MA.

Samuelson, P. 1947. Foundations of Economic Analysis. Cambridge, Mass.: Harvard University Press.

SCARF, H. 1967. The approximation of fixed points of a continuous mapping. *SIAM J. Applied Math 15*, 1328–1343.

SCARF, H. 1982. *The Computation of Equilibrium Prices: An Exposition*. Vol. 2. 1008–1061.

SHOVEN, J. AND WHALLEY, J. 1992. *Applying General Equilibrium*. Cambridge University Press, Cambridge.

SMALE, S. 1976. A convergent process of price adjustment and global newton methods. *Journal of Mathematical Economics 3*, 2, 107–120.

VANDERBEI, R. J. 2000. *LOQO Users' Manual: Version 4.05, Technical Report*.

YE, Y. 2008. A path to the arrow-debreu competitive market equilibrium. *Mathematical Programming*. *111*, 315–348.

Received November 2005; revised December 2006; accepted May 2007