# An experimental study of different approaches to solve the market equilibrium problem*

Bruno Codenotti Benton McCune Sriram Pemmaraju Rajiv Raman
Kasturi Varadarajan

**Abstract**

The market equilibrium problem has a long and distinguished history. Its computational version has recently received significant attention in the theoretical computer science community resulting in a flurry of polynomial time algorithms for computing market equilibria in various restricted but relevant settings. The most important special cases arise either when the traders have utility functions that satisfy a property known as *gross substitutability* or when the aggregate demand satisfies a property known as the *weak axiom of revealed preferences*.

In this paper we experimentally compare the performance of some of these recent algorithms against that of the algorithms traditionally used for market equilibrium problems. In particular, we evaluate the following approaches: (i) using GAMS/MPSGE, a popular commercial tool for computing market equilibrium, (ii) solving convex feasibility programs arising from some recently developed formulations of the market equilibrium problem, (iii) applying an iterative scheme, known as *sequential joint maximization*, (iv) implementing several discrete versions of a simple iterative price update scheme called *tâtonnement*. Our primary goals were to investigate the scalability of the last three approaches and to compare their performance with the first approach. Our main observations are:

- The convex programming approach compares favorably against PATH (the solver used within GAMS) and seems to be competitive in terms of scalability.
- The sequential joint maximization algorithm converges rapidly, even for certain instances for which its convergence is yet to be theoretically proved.
- The discrete versions of tâtonnement scale well compared to PATH.

*The first author is at the Toyota Technological Institute at Chicago, Chicago IL 60637. On leave from IIT-CNR, Pisa, Italy. E-mail: bcodenotti@tti-c.org. The rest of the authors are at the Department of Computer Science, The University of Iowa, Iowa City IA 52242-1419. E-mails: [bmccune, sriram, rraman, kvaradar]@cs.uiowa.edu. Research by the second, fourth, and fifth authors was partially supported by NSF CAREER award CCR-0237431.

## 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 [1]. This existential result has been the starting point of the development of effective procedures for the computation of the equilibrium.

Roughly speaking, we can identify three families of methods for the solution of *general* market equilibrium problems: (i) simplicial methods, (ii) path following methods or homotopy methods, and (iii) algorithms based on Newton's method.

Simplicial methods are techniques to approximate fixed points of continuous mappings of a simplex into itself, which have been pioneered by Scarf [38, 39, 14, 22, 26]. These algorithms do not achieve polynomial running times, and their performance on large size applications has been superseded by Newton's methods and homotopy methods. The global Newton's method fully uses the information embedded in the Jacobian of the excess demand function to guarantee convergence under very mild assumptions [23, 40, 41]. Homotopy methods have naturally evolved from simplicial methods and these follow a *path*, as a function of a parameter $t$, from an easy to solve problem ($t = 0$) to the original problem ($t = 1$) [13, 30, 42]. Both Newton's methods and homotopy methods enjoy global convergence, and, in spite of not being guaranteed to run in polynomial time, seem to be amenable for tackling real world applications (see [18], p. 670, and references therein). For instance, Newton's method is the main building block in the PATH solver [17], which is embedded in GAMS/MPSGE, the modeling language of choice for experimental work on market equilibrium applications

[6].

For markets with known additional structure, several alternative computational approaches have been considered. The simplest approach is the *tâtonnement* process which, starting from an arbitrary price vector, updates it according to the market excess demand generated by such prices [2, 3]. In its continuous version, the tâtonnement process is known to converge [2] whenever the market satisfies *weak gross substitutability* (see next section for the definition). However, it need not converge if the market does not satisfy this property (see [29], Chapter 17).

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 [2] it is shown that when the market satisfies weak gross substitutability, a fundamental inequality holds which defines a 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 [33, 34, 35] and the references therein). Some of these papers build upon the characterization above to propose Ellipsoid and cutting-plane algorithms to compute market equilibrium. However, as far as we can tell, none of these papers succeed in proving polynomial running time for these methods. For a discussion of this issue, especially pertaining to [33], see [9]. In [32], it is shown that for arbitrary endowments and linear utilities, the equilibrium is given as the solution to a finite convex program; this approach works for certain non-linear utilities as well.

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 [31]. This characterization transforms the problem of computing equilibrium prices into the problem of computing the 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 [28].

Over the last three years, the problem of computing equilibrium prices has been analyzed with the focus on its polynomial time solvability. In the process, some old results from mathematical economics literature have been rediscovered by computer scientists. For linear utilities, polynomial time algorithms have been developed both for the case of collinear endowments [12] and arbitrary endowments [24, 25, 19]; the approach of [24] which introduces the same convex programming formulation as [32], works for some non-linear utilities as well. For homothetic utilities and collinear endowments, polynomial time algorithms based on convex programming are given in [10]. Convex programs for some CES utilities and arbitrary endowments are presented in [11]. Polynomial time algorithms that exploit the characterization of [2] for markets satisfying weak gross substitutability are presented in [9].

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 [17].

2. Solving the recent convex programming formulations for market equilibrium problems, specifically those derived in [10, 11]. In the experiments reported in this paper, we use the "convex" option in the general purpose non-linear solver LOQO, in combination with AMPL, its modeling language.

3. The sequential joint maximization algorithm of [36]. Note that such an algorithm roughly corresponds to "Algorithm 2" in [25] 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 [25] does not fit perfectly into the framework of sequential joint maximization because it uses an extra fictitious trader. [25] proves that this algorithm converges in polynomial time for linear utility functions and it is not hard to extend this proof to whenever the exchange market enjoys weak gross substitutability and an efficient solver for Fisher's instances is available. However, [25] leaves open the

issue of convergence of the simpler scheme without the extra trader.

4. A version of the tâtonnement process. The continuous tâtonnement process converges for markets satisfying weak gross substitutability, and is particularly attractive due to 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.

Our primary goal was to investigate the scalability of the last three approaches and to compare their performance with the first approach.

Prior to this paper, there has been some work analyzing the practical performance of different algorithms for the market equilibrium problem. In [7] the performance of a distributed implementation of tâtonnement is discussed; in [5] several complementarity solvers are implemented and their relative merits analyzed, while in [23] the efficiency of Newton's method is investigated. In [4] an approach based on global minimization is illustrated, and the outcomes some numerical experiments are reported. More recently, in [16] 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 fist attempt at an experimental evaluation of different algorithms for large-scale problems.

Our main observations are:

- The convex programming approach compares favorably against PATH (the solver used within GAMS) and seems to be competitive in terms of scalability.
- The sequential joint maximization algorithm converges rapidly, even for certain instances for which its convergence is yet to be theoretically proved.
- With an appropriately chosen price update rule, the discrete version of tâtonnement performed remarkably well and in particular scaled well compared to PATH. In fact, the excellent performance of tâtonnement has motivated a more careful theoretical investigation into its running time, resulting in a proof of polynomial time convergence, reported in [8].

To ensure reproducibility, we have made available at `http://www.cs.uiowa.edu/~rraman/eq/mkts.html`, many more details of our experiments, including raw data and source code.

## 2 Definitions

Let us consider $m$ economic agents which represent traders of $n$ goods. Let $\mathbf{R}^n_+$ denote the subset of $\mathbf{R}^n$ with all nonnegative coordinates. The $j$-th 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] $\pi \cdot x \leq \pi \cdot w_i$. 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: (i) For each good $j$, $\sum_i \bar{x}_{ij} \leq W_j$ and (ii) For each trader $i$, the vector $\bar{x}_i$ maximizes $u_i(x)$ subject to the constraints $\pi^T x \leq \pi^T w_i$ and $x \in \mathbf{R}^n_+$.

The celebrated result of Arrow and Debreu [1] states that, under quite mild assumptions, such an equilibrium exists. A special case occurs when the initial endowments are *collinear*, i.e., when $w_i = \delta_i w$, $\delta_i > 0$ and $w \in \mathbf{R}^n_+$, so that the relative incomes of the traders are independent of the prices. This special case 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 $\pi \cdot x \leq e_i$ 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 $\pi$, the vector $x_i(\pi)$ that maximizes $u_i(x)$ subject to the constraints $\pi^T x \leq \pi^T w_i$ and $x \in \mathbf{R}^n_+$ is called the *demand* [2] of trader $i$ at prices $\pi$. The *excess demand* of trader $i$ is $z_i(\pi) = x_i(\pi) - w_i$.

[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

Then $X_k(\pi) = \sum_i x_{ik}(\pi)$ denotes the *market demand* of good $k$ at prices $\pi$, and $Z_k(\pi) = X_k(\pi) - W_k = \sum_i z_{ik}(\pi)$ the *market excess demand* of good $k$ at prices $\pi$. 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.

When the market is defined 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 $j$.

Two properties that play a significant role in the theory of equilibrium and in related computational results are *gross substitutability* (GS) and the *weak axiom of revealed preferences* (WARP). A market is said to satisfy GS (resp. weak GS) if for any two sets of prices $\pi$ and $\pi'$ 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. The market excess demand is said to satisfy WARP if for any two sets of prices $\pi$ and $\pi'$ such that $Z(\pi) \neq Z(\pi')$ either $\pi^T Z(\pi') > 0$ or $(\pi')^T Z(\pi) > 0$. For the connection between WARP and market equilibrium see [29].

## 3 An Overview of our Experiments

**3.1 Generating Markets** An important aspect of our experiments is the generation of markets with enough variety so as to represent a wide range of phenomena. We start by assuming that every agent's preferences are represented by the *constant elasticity of substitution* (CES) functional form. A CES function is a concave function defined as

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

where $\alpha_j > 0$ for each $j$ and $\sigma > 0, \sigma \neq 1$. The $\alpha_j$'s and $\sigma$ are parameters that can be assigned values to obtain different utility functions. The parameter $\sigma$ 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 $\sigma$ models the elasticity of substitution, the $\alpha_j$'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.

Assuming that CES functional forms represent agent's preferences, generating a market corresponds to generating $\alpha_j$'s, $\sigma$, 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 $\alpha_{ij}$ denote the coefficient $\alpha_j^{\frac{1}{\sigma}}$ of the term $x_j^{\frac{\sigma}{\sigma-1}}$ in agent $i$'s utility function. For notational convenience let $A$ denote the $m \times n$ matrix of the $\alpha_{ij}$'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 $\alpha_{ij}$'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 $\sigma$ 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 value $\sigma$. We generate matrices $A$ and $W$ independently, using several generators we have implemented.

**Generators for the desirability matrices.** We have implemented three generators for the matrix $A$. To guarantee the existence of an equilibrium, we fix an $\epsilon > 0$, and make sure every agent desires every good to an extent of at least $\epsilon$.

**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 and uniformly at random from the space of vectors in $[0, 1]^n$ that sum up to 1 and each of whose entries is at least $\epsilon$.

**Concentrated Generator.** This constructs matrices in which for $1 \leq i \leq m$, agent $i$ desires a fraction .8 of good $i$. That is, $\alpha_{ii} = .8$. Two goods $j_1$ and $j_2$ are chosen at random from among the other

appropriately generalized to handle the case when the demand is a multi-valued function, which happens for instance when the utility functions are linear.

goods. Agent $i$'s desire for each good outside the set $\{i, j_1, j_2\}$, is set to $\epsilon$. The rest of the mass, $0.2 - \epsilon(n-3)$ is equally distributed between $j_1$ and $j_2$. 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 $0.2 - \epsilon(n-3)$. 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 $\epsilon$ and the rest of the mass, $1 - \epsilon|J_i|$ is distributed uniformly among the goods in $J_i$. Rows in the matrix are generated independently of each other.

**Generators for the endowment matrices.** We have implemented three generators for the endowment matrix that are very similar to the generators for the desirability matrix. Again, for some fixed $\epsilon > 0$, we make sure that for each good $j$ and each agent $i$, the initial endowment of good $j$ to agent $i$ is at least $\epsilon$. We have a uniform generator 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. The subset generator for the endowment matrix assumes that $m \geq n$ and for each good $j$, assigns an amount of .8 to agent $(m - j + 1)$. Thus the distribution of the goods is concentrated along the reverse diagonal of the endowment matrix. In all other ways, this generator is identical to the subset generator for desirability matrices.

**Introducing correlation in desires and in endowments** In the description of the uniform generator and the subset generator above, we mentioned that rows of the desirability matrix and columns of the endowment matrix are independently generated. We get matrices with richer structure and asymmetry between goods by introducing some dependency. We implement two kinds of dependence.

**Replicated desires or endowments.** In this case, after the first row of $A$ is generated the remaining rows are generated by simply copying the first row. The columns of the matrix $W$ can also be replicated similarly.

**Clustered desires or endowments.** In this case, the agents are randomly partitioned into some $k$ sets $A_1, A_2, \ldots, A_k$ with the stipulation that the rows for all agents in a set $A_i$ are identical. The columns of the matrix $W$ can also be clustered similarly.

By combining the generators for the desirability matrix with the generators for the endowment matrix, and by choosing a certain kind of dependence for the desirability matrix, and independently choosing a dependence for the endowment matrix, we get a large number of market generators that we use to investigate a variety of market equilibrium algorithms. We believe our market generators are able to provide to our algorithms input markets with a wide variety of properties.

**3.2 The Two Types of Experiments.** Most of our experiments fall in one of two categories. The first category of experiments involved estimating the running time of a variety of market equilibrium algorithms as a function of the number of agents and the number of goods. One aim is to get a sense of the rate of growth of the running times as a function of $m$ and $n$, while a second aim is to get a sense of the absolute running time of these algorithms on fairly large instances. Specifically, we are interested in knowing how well the convex programming approach, sequential joint maximization, and tâtonnement scale relative to PATH.

The second category of experiments involved determining how sensitive the running time of these algorithms is with respect to $\sigma$. The tractability of the problem of computing market equilibrium for exchange markets consisting of agents with CES utility functions depends significantly on the values of $\sigma$. For example, it is known that tâtonnement converges if $\sigma \geq 1$, but not necessarily for $\sigma < 1$ [2]. Also, Codenotti and Varadarajan [11] have shown that when the space of prices is appropriately transformed, the set of equilibrium price vectors is guaranteed to correspond to a convex set in the transformed space if $\sigma \geq 1/2$. This result complements the example in [21] that shows that for any $\sigma < 1/2$, it is possible to construct markets with two goods and two traders such that have multiple disconnected equilibria.

**3.3 Computational Environment.** We now describe the computational environment we used to run our experiments.

- GAMS is a modeling system for mathematical programming problems. It consists of a language compiler integrated with high-performance solvers, and it is tailored for complex, large scale modeling applications (see [6]). GAMS has been extended to GAMS/MPSGE by Rutherford [37] to efficiently 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. GAMS uses the PATH solver for the *mixed-complementarity* representation of

the market equilibrium problem. The GAMS experiments were performed on a machine with an AMD Athlon, 64 bit, 2202.865 Mhz processor, 2GB of RAM, and running Red Hat Linux Release 3, Kernel Version - 2.4.21-15.0.4

- The convex programming experiments used the general nonlinear solver LOQO [43], which is equipped with a *convex* option, which prevents the solver for calling on special code to handle non-convexities. For more details on LOQO see `http://www.orfe.princeton.edu/~loqo/`. The experiments that use convex-programming were performed on a machine with a Pentium III, 997.461 MHz processor, 512 MB of RAM, and running Red Hat Linux Release 3, Kernel Version - 2.4.21-15.0.4.

- The sequential joint maximization experiments also used the solver LOQO. These experiments were also performed on a machine with a Pentium III, 997.461 MHz processor, 512 MB of RAM, and running Red Hat Linux Release 3, Kernel Version - 2.4.21-15.0.4.

- The tâtonnement experiments were performed on a machine with an AMD Athlon, 64 bit, 2202.865 Mhz processor, 2GB of RAM, and running Red Hat Linux Release 3, Kernel Version - 2.4.21-15.0.4.

## 4 Experimenting with GAMS/MPSGE

In this section, we present the outcomes of the experiments we have carried out with the PATH solver available under GAMS/MPSGE[3], in order to give indications of the affordable problem sizes, as well as of the sensitivity of GAMS/MPSGE/PATH to specific properties of the market, e.g., gross-substitutability and WARP.

We have tested the performance of PATH against a variety of input data, and the results did consistently show a fairly rapid growth of the running time as a function of the size of the markets. An example is illustrated in Figure 1, where the input size ranges from 10 to 100. The runs on the other benchmarks lead to very similar figures. This holds true also for experiments on markets with collinear endowments, thus hinting at the fact that PATH is not particularly sensitive to the WARP property enjoyed by the aggregate demand in this case.

[3]The MPSGE subsystem is a preprocessor that provides easy input to market equilibrium problems in terms of the excess demand functions. The MPSGE system then converts this input into GAMS code; we then call PATH on this auto-generated code to solve the equilibrium problem.

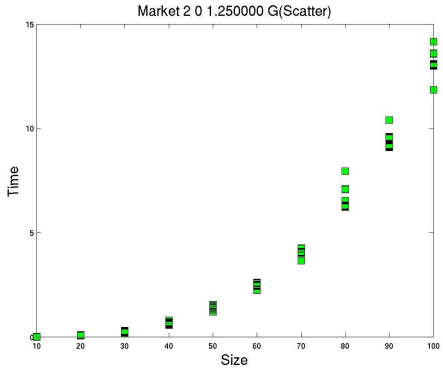



**Figure 1**: Running time as a function of problem size, for a market whose desirability matrix is generated by the concentrated generator and whose endowment matrix is generated by the uniform generator.

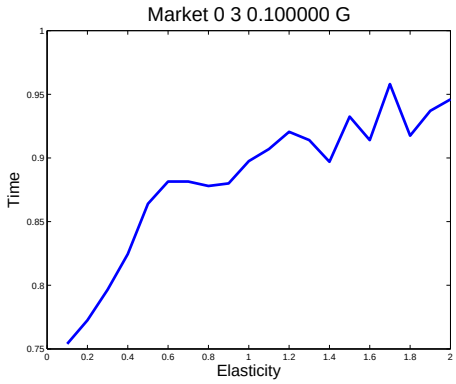
We have also tested PATH performance with respect to the elasticity of substitution parameter. There seems to be no dependence of the running time on the elasticity. Figure 2(a), (b) provide fairly typical indications of the pattern of indifference of the running time observed over a greater range of elasticities.

## 5 Algorithms using the Convex Programming approach

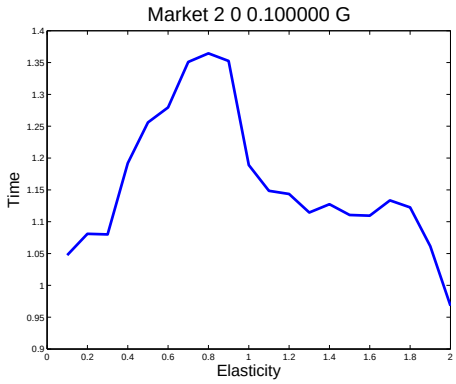
In this section, we report on an experimental study of some of the convex-programming based approaches for computing equilibria in various special cases.

**The Fisher Setting.** We implemented the convex program of Eisenberg [15] 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 3(a) 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 to complete. We suspect that this is because of the number of variables in the program being $n * m$, though this claim needs further investigation.

**Exchange with CES utilities.** We implemented the convex feasibility program of Codenotti and Varadarajan [11] for computing the equilibrium in an exchange market with traders who have CES utility functions with elasticity at least one. The program has $n + m$ variables for a market with $m$ traders and $n$ goods and

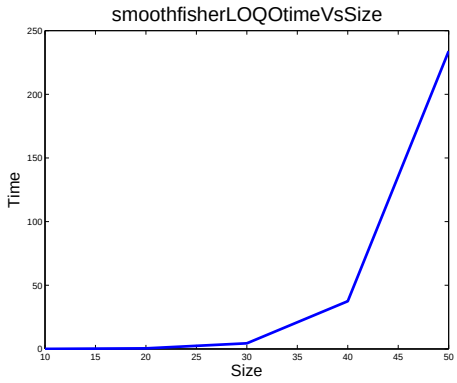



**(a)**

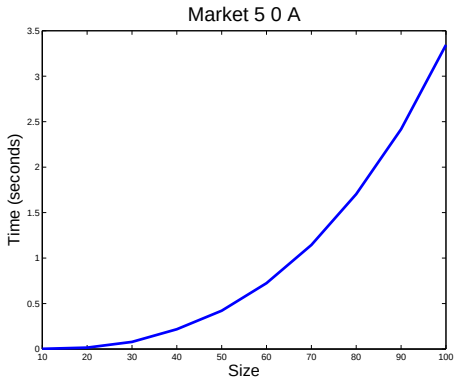



**(b)**

**Figure 2**: Running time of GAMS as a function of the elasticity of substitution, **(a)** for a market of type whose desires are produced by a uniform generator and endowments by a subset generator and **(b)** for a market where desires are produced by a concentrated generator and endowments by a uniform generator. The market size is 50.




**(a)**




**(b)**

**Figure 3**: 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 $\sigma = 1.25$.

$n + m$ 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 3(b), which is quite typical, depicts how the average running time varies with $n$ for $\sigma = 1.25$.

We also measured how the running time varies with elasticity. For a market with $n = 25$, we varied the elasticity 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.

## 6 Algorithms derived from tâtonnement

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 $\pi_i$ 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 convergent by Arrow, Block, and Hurwicz [2] for weak GS markets.

In our implementation of tâtonnement, the starting price vector is $(1, 1, \ldots, 1)$. Let $\pi^k$ 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 $\pi$ continues to remain non-negative. Also noteworthy is the role of $i$ that ensures that the "step size" diminishes as the process (hopefully) approach equilibrium.

Using sixteen different market generators, we generated markets with $n$ agents and $n$ goods, for $n = 5, 6, \ldots, 14$ and $\sigma = 0.2, 0.4, 0.6 0.8, 1, 1.2$ for a total of 960 markets. We ran tâtonnement on these markets to understand how the rate of convergence of tâtonnement changes with elasticity. Similarly, using sixteen different market generators, we generated markets with $n$ agents and $n$ goods, for $n = 10, 20, \ldots, 100$ and $\sigma = 0.75, 1, 1.25 1.5, 1.75, 2.0$. For each market type and $(n, \sigma)$ pair we generated 20 market instances for a total of 19,200 markets. We ran tâtonnement on these markets to understand the how the rate of convergence of tâtonnement changes as a function of $n$. Finally, with $n$ fixed at 10, using each of 16 market generators, for $\sigma = 0.75, 1, 1.25 1.5, 1.75, 2.0$, and for each $\epsilon$ such that $\log_{10}(1/\epsilon) = 1, 1.5, 2, 2.5, \ldots, 5.5, 6$, we generated 20 instances and ran tâtonnement with the aim of investigating the convergence of tâtonnement as a function of $\epsilon$. These were our three main tâtonnement experiments. For the first two experiments, we terminated tâtonnement when $\max_i |Z_i(\pi^k)|$ fell below a threshold value of $\epsilon = 10^{-4}$ or when the number of iterations exceeded 100,000, whichever happened first. For the third experiment, we increased the limit on the number of iterations to 10 million.

Our main observations are below.

**Convergence as a function of $\sigma$.** For the "small" experiments (with $n = 5, 6, \ldots, 14$) as well as for the "large" experiments (with $n = 10, 20, \ldots, 100$), our implementation of tâtonnement converged to within the threshold $\epsilon = 10^{-4}$ in less than 100,000 iterations in *all* cases. There is a clear indication that the number of iterations to convergence increases slowly with respect to $\sigma$. This trend is summarized in the table in Figure 4 and further below, two graphs showing the trend for two distinct markets are given in Figure 5.

| $\sigma$ | mean | max |
|---|---|---|
| 0.2 | 368.3 | 2787 |
| 0.4 | 642.78125 | 4602 |
| 0.6 | 1216.14375 | 5508 |
| 0.8 | 1473.48125 | 8038 |
| 1.0 | 1923.45625 | 8587 |
| 1.2 | 1738.4125 | 7722 |

**Figure 4**: Summary of tâtonnement convergence data from experiments that investigate convergence as a function of elasticity. For each $\sigma$, we tested 160 markets. The column label "mean" is the mean number of iterations and "max" is the maximum number of iterations over 160 runs.

These observations may seem to contradict theoretical predictions. For example, for small $\sigma$ ($\sigma < 1$) the market excess demand does not satisfy weak GS and therefore, in general, the continuous version of tâtonnement is not guaranteed to converge. Examples of markets which force tâtonnement into a non-convergent behavior use fairly specific initial endowment distributions. In practice and in experiments such as ours where initial endowments are generated at random, discrete versions of tâtonnement seem to converge even in the absence of weak GS. Similar observations are reported in [7], though for far fewer instances, all consisting of markets of size smaller than ours. The "degree" of subsitutability among goods increases with $\sigma$. Therefore one could expect, in principle, that tâtonnement converges faster as $\sigma$ increases. We have instead observed the opposite phenomenon: the number of iterations slowly increases with $\sigma$, provided that $\sigma$ is not too close to zero. This fact can be explained in terms of the behavior of the excess demand function, which varies more rapidly (with respect to prices) as $\sigma$ increases, thus making it harder for discrete versions of tâtonnement to converge.

**Convergence as a function of $n$.** Here we report on experiments that investigate the dependence of the number of iterations of tâtonnement on the value of $n$. For these experiments trends were much harder to identify and the results were very sensitive to the type of market. For most of the 16 markets we used, the number of iterations does not increase with $n$ and actually seems to fall in some cases. See Figure 6(a) for an example of such a market phenomenon. We believe this is because the way we generate markets leads to "symmetry" among

goods, that is, each good is desired by roughly the same number of agents to roughly the same extent; also, each good is endowed to roughly the same number of agents to roughly the same extent. This leads to a situation where all components of an equilibrium price vector are roughly the same and therefore very close to our starting price vector $(1, 1, \ldots, 1)$. Given this, the main determinant of the number of iterations is the distance between $(1, 1, \ldots, 1)$ and an equilibrium price, rather than $n$. To get a more reliable estimate of the running time of tâtonnement as a function of $n$, we ran tâtonnement on markets in which there is clear asymmetry among goods. For such asymmetric markets, Figure 6(b) shows a typical plot of the number of iterations with respect to $n$.

**Convergence as a function of $\epsilon$.** The number of iterations of tâtonnement seems to grow rapidly with respect to $\log(\frac{1}{\epsilon})$, quite independently of the market type. Figures 7(a) and (b) show typical plots.

## 7 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 approach for establishing the existence of competitive equilibrium [31]. Let $\mathbf{R}^n_{++}$ denote the subset of $\mathbf{R}^n$ with all positive coordinates. Let $\alpha = (\alpha_1, \ldots, \alpha_m) \in \mathbf{R}^m_{++}$, be any vector. Consider the allocations that solve the following optimization problem over $x_i \in \mathbf{R}^n_+$:

$$\text{Maximize} \quad \sum_{i=1}^{m} \alpha_i u_i(x_i)$$

$$\text{Subject to} \quad \sum_i x_{ij} \leq \sum_i w_{ij} \text{ for each good } j.$$

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

Under some standard assumptions on the utility functions, the following properties hold for the map $f : \mathbf{R}^m_{++} \to \mathbf{R}^m$ (see Chapter 7 of the book by Ginsburgh and Waelbroeck [20] 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 $\alpha^* \in \mathbf{R}^m_{++}$ 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's theorem [31]. 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 [20, 28, 36]. In particular, Mantel shows [28] 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 analogue 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.

We can define a function $g : \mathbf{R}^m_{++} \to \mathbf{R}^m$ in a manner similar to $f$ above, that is, $g_i(\alpha) = B_i(\alpha)/\alpha_i$, where $B_i(\alpha) = \pi \cdot w_i - \pi \cdot \bar{x}_i$, except that the price vector $\pi$ in the definition of $B_i(\alpha)$ is taken to be the equilibrium price for the Fisher instance that is generated by letting $\alpha_i$ be the income of the $i$-th trader, and $\sum_i w_i$ be the vector of goods available in the market. Under appropriate assumptions on the utility functions, properties (1)–(4) can be shown to hold for $g$. Furthermore, a generalization of an idea due to Jain, Mahdian, and Saberi [25] shows that, under a gross substitutability assumption on the $u_i$, we also have $\frac{\partial g_i(\alpha)}{\partial \alpha_i} < 0$ and $\frac{\partial g_j(\alpha)}{\partial \alpha_i} > 0$, for $j \neq i$. Note that the $u_i$'s are not required to be log-homogeneous for this consequence. For utility functions representing *homothetic preferences*, a result of Eisenberg [15] implies that $f$ and $g$ are identical.

Using the two characterizations above, we can conceive of polynomial-time algorithms working in the Negishi space that correspond to Ellipsoid and tâtonnement-based algorithms (see [9] and [8] for examples and pointers) in the price space. These would be polynomial-time analogs of computational methods (usually called *welfare adjustment*, or *joint maximization* methods) that have been well explored [36]. The second algorithm of Jain, Mahdian, and Saberi [25] may 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 algo-

rithm starts off from an arbitrary initial price $\pi^0$, and computes a sequence of prices as follows. Given $\pi^k$, the algorithm sets up a Fisher instance by setting the money of each trader to be $e_i^k = \pi^k \cdot w_i$, where $w_i$ is the $i$-th trader's initial endowment. (The goods in the Fisher instance are obtained by aggregating the initial endowment $w_i$ of each trader.) Let $\pi^{k+1}$ be the price vector that is the solution of the Fisher instance. If $\pi^{k+1}$ is within a specified tolerance of $\pi^k$, we stop and return $\pi^{k+1}$; one can show that $\pi^{k+1}$ must be an approximate equilibrium. Otherwise, we compute $\pi^{k+2}$ and proceed.

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

We tested our iterative algorithm for an exchange economy where traders have CES utility functions. We used the implementation described in Section 5 to solve the Fisher instance. Quite remarkably, the algorithm tends to converge fast. For instance, we ran the algorithm on instances with 10 goods and 10 traders, for elasticities $0.2, 0.4, \ldots, 1.2$. We generated 10 runs for each case, and the algorithm converged (more precisely, the Euclidean distance between successive prices became smaller than 0.001) in less than 10 iterations in each case. We also ran the algorithm on instances with 25 goods and traders, elasticities 0.25 and 1.25, and generated 5 runs of each. In all the instances, the Euclidean distance between successive prices was smaller than 0.02 after the 100-th iteration. In one case, the algorithm terminated earlier since the distance between successive prices became smaller than 0.001.

The explanation for these 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.

## 8 Analysis of the results

The experiments conducted so far suggest that tâtonnement (Section 6) and the convex programming approach to CES exchange markets (Section 5), where applicable, are competitive with PATH in terms of scalability. Experiments conducted for large values of $n$ strengthen this conclusion. The fact that the convex programming approach does comparatively well should be encouraging news to researchers working on polytime algorithms for equilibrium problems. The performance of tâtonnement has been, to some extent, anticipated by the observation in [9] that in some cases tâtonnement can be proved to run in pseudo-polynomial time. More recently, encouraged by the experimental performance of tâtonnement, we have been able to show polynomial time convergence of tâtonnement for exchange markets that have the weak GS property [8].

The running time of PATH is largely indifferent to $\sigma$. The performance of the CES convex program grows only mildly with $\sigma$ once $\sigma$ is beyond a threshold. The convergence of tâtonnement is sensitive to $\sigma$ and the number of iterations for convergence increases with increasing $\sigma$. This observation seems to contradict theoretical predictions and deserves further experimental study. As mentioned in the Section 6, for small $\sigma$ ($\sigma < 1$) the absence of weak GS seems to be overcome by properties of the market that are induced by initial endowments. Also, as $\sigma$ increases there is an increase in the rate of the change of excess demand relative to prices and this seems to lead to more iterations. Tâtonnement is also very sensitive to the type of the market, so much so that this effect sometimes overwhelms the effect that the size of market has, on its running time.

The fact that the simple algorithm based on iteratively solving Fisher (Section 7) tends to converge (even for markets where gross substitutability is not guaranteed to hold) is surprising and merits further study.

## References


[1] K.J. Arrow and G. Debreu, Existence of an Equilibrium for a Competitive Economy, Econometrica 22 (3), pp. 265–290 (1954).

[2] K. J. Arrow, H. D. Block, and L. Hurwicz. On the stability of the competitive equilibrium, II. Econometrica 27, 82–109 (1959).

[3] K. J. Arrow and L. Hurwicz. Competitive stability under weak gross substitutability: The "Euclidean distance" approach. International Economic Review 1, 1960, 38–49.

[4] A.M. Bagirov and A.M. Rubinov, Global Optimization of Marginal Functions with Applications to Economic Equilibrium, Journal of Global Optimization 20, pp. 215-237 (2001).

[5] S.C.Billups, S.P.Dirkse, and M.C. Ferris, A comparison of large scale mixed complementarity problem solvers, Comput. Optim. Appl. 7, pp. 3-25 (1997).

[6] A. Brooke, D. Kendrick, and A. Meeraus, GAMS: A user's guide, The Scientific Press, South San Francisco (1988).

[7] J. Q. Cheng, M. P. Wellman, The WALRAS Algorithm: A Convergent Distributed Implementation of General Equilibrium Outcomes, Computational Economics 12(1), pp. 1-24 (1998).

[8] B. Codenotti, B. McCune, K. Varadarajan, Market equilibrium via the excess demand function. Submitted. Manuscript available at `people.cs.uchicago.edu/~codenott`.

[9] B. Codenotti, S. Pemmaraju, and K. Varadarajan, On the Polynomial Time Computation of Equilibria for Certain Exchange Economies. SODA 2005, to appear.

[10] B. Codenotti and K. Varadarajan, Efficient Computa-

tion of Equilibrium Prices for Markets with Leontief Utilities. Proc. ICALP 2004.

[11] B. Codenotti and K. Varadarajan, Market Equilibrium in Exchange Economies with Some Families of Concave Utility Functions. Submitted, 2004.

[12] N. R. Devanur, C. H. Papadimitriou, A. Saberi, V. V. Vazirani, Market Equilibrium via a Primal-Dual-Type Algorithm. FOCS 2002, pp. 389-395. (Full version with revisions available on line, 2003.)

[13] B. C. Eaves, Homotopies for computation of fixed points, Mathematical Programming 3 (1), pp.1-22 (1972).

[14] B. C. Eaves and H. Scarf, The Solution of Systems of Piecewise Linear Equations, Mathematics of Operations Research, Vol. 1, No. 1, pp. 1-27 (1976).

[15] E. Eisenberg, Aggregation of Utility Functions. Management Sciences, Vol. 7 (4), 337–350 (1961).

[16] M. Esteban-Bravo, Computing Equilibria in General Equilibrium Models via Interior-point Methods, Computational Economics 23, pp. 147-171 (2004).

[17] M. C. Ferris, T. S. Munson, and D. Ralph. A homotopy method for mixed complementarity problems based on the PATH solver. In D. F. Griffiths and G. A. Watson, editors, Numerical Analysis 1999, Research Notes in Mathematics, pages 143-167, London, 2000. Chapman and Hall.

[18] M.C Ferris and J.S. Pang, Engineering and Economic Applications of Complementarity Problems, SIAM Review 39(4), pp.669-713 (1997).

[19] R. Garg and S. Kapoor, Auction Algorithms for Market Equilibrium. In *Proc. STOC*, 2004.

[20] V. A. Ginsburgh and J. L. Waelbroeck. Activity Analysis and General Equilibrium Modelling, North-Holland, 1981.

[21] S. Gjerstad, Multiple Equilibria in Exchange Economies with Homothetic, Nearly Identical Preference, University of Minnesota, Center for Economic Research, Discussion Paper 288, 1996.

[22] T. Hansen and H. Scarf, The Computation of Economic Equilibrium, Cowles Foundation Monograph No. 24, New Haven: Yale University Press (1973).

[23] P.T. Harker and B. Xiao, Newton's method for the nonlinear complementarity problem: a B-differential equation approach, Mathematical Programming 48, pp. 339-358 (1990).

[24] K. Jain, A polynomial time algorithm for computing the Arrow-Debreu market equilibrium for linear utilities, Discussion Paper, Microsoft Lab., Seattle, WA (2003). FOCS 2004.

[25] K. Jain, M. Mahdian, and A. Saberi, Approximating Market Equilibria, Proc. APPROX 2003.

[26] H.W. Kuhn, Simplicial Approximation of Fixed Points, Proc. National Academy of Sciences of the United States of America Vol. 61, n. 4, pp. 1238-1242 (1968).

[27] A. S. Manne, H. Chao and R. Wilson, Computation of Competitive Equilibria by a Sequence of Linear Programs, Econometrica, vol. 48, issue 7, pp. 1595-1615 (1980).

[28] R. R. Mantel, The welfare adjustment process: its stability properties. International Economic Review 12, 415-430 (1971).

[29] A. Mas-Colell, M. D. Whinston, J. R. Green, *Microeconomic Theory*, Oxford University Press, 1995.

[30] O.H. Merril. Applications and extension of an algorithm that computes fixed points of certain upper semicontinuous point to set mappings. Ph.D. thesis, Department of Industrial Engineering, University of Michigan (1972).

[31] T. Negishi, Welfare Economics and Existence of an Equilibrium for a Competitive Economy, Metroeconomica 12, 92-97 (1960).

[32] E. I. Nenakov and M. E. Primak. One algorithm for finding solutions of the Arrow-Debreu model, Kibernetica 3, 127–128 (1983).

[33] D. J. Newman and M. E. Primak. Complexity of Circumscribed and Inscribed Ellipsoid Methods for Solving Equilibrium Economical Models, Applied Mathematics and Computation 52: 223-231 (1992).

[34] M. E. Primak, An algorithm for finding a solution of the linear exchange model and the linear Arrow-Debreu model, Kibernetika 5, 76-81 (1984).

[35] M. E. Primak, A converging algorithm for a linear exchange model, Journal of Mathematical Economics 22, 181–187 (1993).

[36] T. Rutherford, Sequential joint maximization, in J. Weyant, ed., 'Energy and Environmental Policy Modeling', Kluwer, Dordrecht, the Netherlands. International series in operations research and management science, volume 18 (1998).

[37] T.F. Rutherford, Extensions of GAMS for complementarity problems arising in applied economic analysis. Journal of Economic Dynamics and Control 19 (1995) 1299–1324.

[38] H. Scarf, The Approximation of Fixed Points of a Continuous Mapping, SIAM J. Applied Math., 15, pp. 1328-1343 (1967).

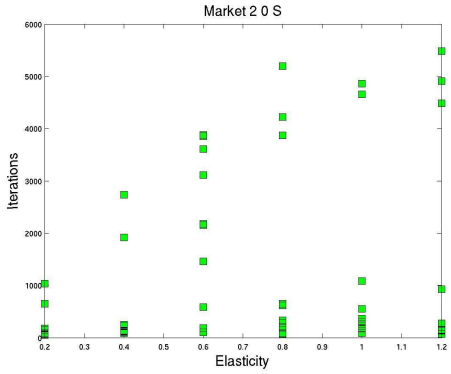
[39] H. Scarf, The Computation of Equilibrium Prices: An Exposition, in Arrow and Intriligator, editors, Handbook of Mathematical Economics, Volume II, pp. 1008-1061 (1982).

[40] S. Smale, A convergent process of price adjustment, Journal of Mathematical Economics, 3 (1976), pp. 107–120.

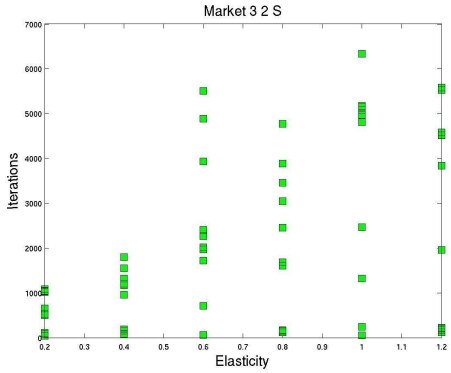
[41] S. Smale, Exchange processes with price adjustment, Journal of Mathematical Economics, 3 (1976), pp. 211–226.

[42] G. Van der Laan, and A.J.J. Talman, A restart algorithm for computing fixed points without an extra dimension. Mathematical Programming 17, 74-84 (1979).

[43] R. J. Vanderbei. LOQO users' manual: version 4.05, Technical Report, Operations Research and Financial Engineering, Princeton University, 2000.

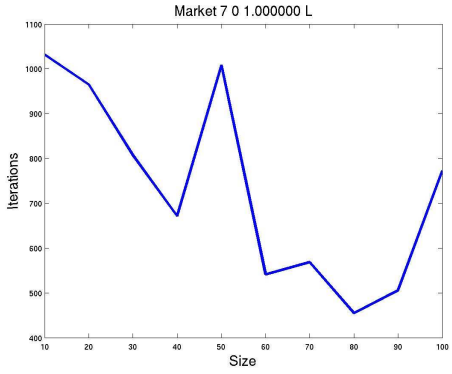



**(a)**

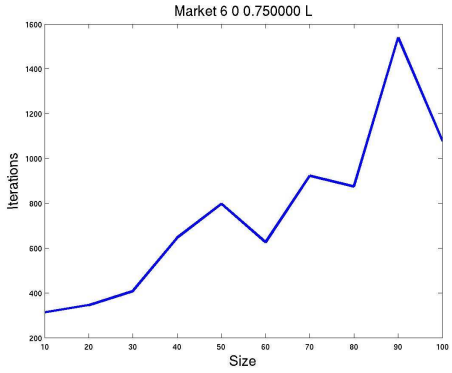



**(b)**

**Figure 5**: (a) This graph pertains to the market whose desirability matrix is generated by the concentrated generator and whose endowment matrix is generated by the uniform generator. The rows of the desirability matrix and the columns of the endowment matrix are generated independently. For each value of $\sigma$, the graph shows the running times of 10 runs, one for each value of $n = 5, 6, \ldots, 14$. (b) This graph is similar to the graph in (a), except that it pertains to a market whose desirability matrix is generated by the subset generator and whose endowment matrix is generated by the concentrated generator. Again, the rows of the desirability matrix and the columns of the endowment matrix are generated independently.

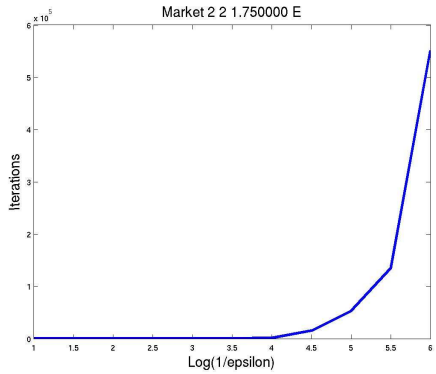



**(a)**

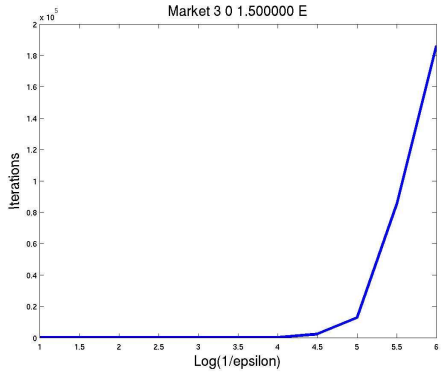



**(b)**

**Figure 6**: Number of iterations of tâtonnement as a function of $n$ for two distinct markets. (a) In this market, agents are clustered into 4 equal-sized groups and agents within each group have identical desires generated uniformly at random; endowments of goods is generated by the uniform generator. The value of $\sigma = 1.0$. For this market, the number of iterations seems to fall as $n$ increases. (b) In this market, all agents have identical desire and this desire is generated using the subset generator; the endowments are generated uniformly at random. This market is clearly asymmetric with about a quarter of the goods highly desired. The value of $\sigma = 0.75$. For this market, the number of iterations seems to increase with increasing $n$.




**(a)**




**(b)**

**Figure 7**: Number of iterations of tâtonnement as a function of $\log(1/\epsilon)$. (a) For this market, both agent desires and good endowments are generated using the concentrated generators. The value of $\sigma = 1.75$. (b) For this market, agent desires were generated with the subset generator and the endowments of goods were generated using the uniform generator. The value of $\sigma = 1.0$.