Computing Equilibria in Dynamic Models with Occasionally Binding Constraints

Johannes Brumm*[1] and Michael Grill†[2]

[1]University of Zurich
[2]Deutsche Bundesbank

First Version: July 2, 2010
This Version: September 22, 2013

Abstract

We propose a method to compute equilibria in dynamic models with several continuous state variables and occasionally binding constraints. These constraints induce non-differentiabilities in policy functions. We develop an interpolation technique that addresses this problem directly: It locates the non-differentiabilities and adds interpolation nodes there. To handle this flexible grid, it uses Delaunay interpolation, a simplicial interpolation technique. Hence, we call this method Adaptive Simplicial Interpolation (ASI). We embed ASI into a time iteration algorithm to compute recursive equilibria in an infinite horizon endowment economy where heterogeneous agents trade in a bond and a stock subject to various trading constraints. We show that this method computes equilibria accurately and outperforms other grid schemes by far.

Keywords: Adaptive Grid, Delaunay Interpolation, Non-Differentiabilities, Occasionally Binding Constraints, Simplicial Interpolation
JEL Classification: C63, C68, E21, G11

*Corresponding author: johannes.brumm@googlemail.com, Department of Banking and Finance, University of Zurich, Plattenstr. 32, CH-8032 Zurich, Switzerland, +41 44 6344107
†michael.grill@bundesbank.de, Financial Stability Department, Deutsche Bundesbank, Wilhelm-Epstein-Straße 14, DE-60431 Frankfurt am Main, Germany, +49 69 9566 2660

1 Introduction

In many applications of dynamic stochastic (general) equilibrium models, it is a natural modeling choice to include constraints that are occasionally binding. Examples are models with borrowing constraints, limited commitment, a zero bound on the nominal interest rate, or irreversible investments. These constraints induce non-differentiabilities in the policy functions, which make it challenging to compute equilibria. In particular, standard interpolation techniques using non-adaptive grids perform poorly both in terms of accuracy and shape of the computed policy function (see, e.g. Judd et al. (2003), pp.270-1). This paper proposes a method that overcomes these problems, even for models with several continuous state variables. We call this method Adaptive Simplicial Interpolation (ASI). Its working principle is to locate the non-differentiabilities that are induced by occasionally binding constraints, and to put additional interpolation nodes there.

We present our algorithm in the setting of a dynamic endowment economy where three or four (types of) agents face aggregate and idiosyncratic risk. To explain the main features of ASI we first compute equilibria in a simple two period version where agents trade in a bond subject to an ad hoc borrowing constraint. Second, we embed ASI into a time iteration algorithm to solve an infinite horizon version of the model. Finally, we add a Lucas tree-type stock, which is subject to a short sale constraint, and we replace the ad hoc borrowing constraint by a collateral constraint. Consequently, short positions in the bond need to be collateralized by stock holdings, while the stock may not be shorted.

Compared to earlier papers using a similar setup, such as Heaton and Lucas (1996), den Haan (2001) or Kubler and Schmedders (2003), the models we consider differ in two respects, which both make it harder to compute equilibria: First, we solve models with more agents, which results in a continuous state space of higher dimension. As the kinks[1] naturally form surfaces in the state space, they are of higher dimension as well. Second, in our extension, the trading constraints that agents face depend on tomorrow's equilibrium price of the

[1] In our terminology, a *kink* associated with a certain constraint is the set of points at which the policy function fails to be differentiable because the constraint is *just* binding, i.e. the constraint is binding *and* the associated multiplier is zero.

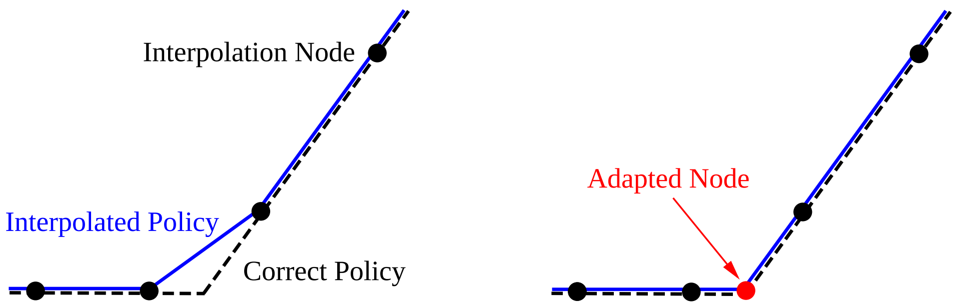

Figure 1: Non-Adaptive (lhs) and Adaptive (rhs) Linear Interpolation in 1D

stock, which is endogenously determined. Consequently, it is much harder to locate the kink and ad hoc methods fail.

Figure 1 illustrates the working principle of ASI. The dashed line displays a simple one-dimensional policy function with a kink. Suppose this function is approximated by linear interpolation between equidistant gridpoints. The resulting interpolated policy is displayed as a solid line in the left hand side of Figure 1. Clearly, the approximation error is comparatively large around the kink, and this is just because there is no interpolation node near the kink. If we knew the location of the kink and put a node there, then the approximation would be much better, as the right hand side of Figure 1 shows. This is the motivation for ASI, which directly addresses the problem of kinks in policy functions by placing additional gridpoints, called *adapted points*, at these non-differentiabilities. In higher dimensional state spaces and with complex constraints, this approach is not as simple as Figure 1 suggests. Hence, we need a flexible interpolation technique and a systematic adaptation procedure.
To be able to place gridpoints wherever needed, we use *Delaunay interpolation*, which consists of two steps. First, the convex hull of the set of gridpoints is covered with simplices, which results in a so-called *tessellation*. Then we linearly interpolate locally on each simplex.[2]
We adapt the grid as follows: First, we solve the system of equilibrium conditions on an initial grid. Second, we use these solutions to determine which edges of the tessellation cross kinks. Third, on each of these edges, we solve a modified

[2]Linear simplicial interpolation is only C^0 at the boundaries. For our purposes, this is desirable, because it provides a better fit at the kinks.

system of equilibrium conditions to determine the point of intersection with the kink. Finally, we place a new grid point there. Using this procedure with state spaces of more than one dimension, we get several adapted gridpoints for each kink. Delaunay tessellation connects these points by edges, such that the kinks are matched very accurately.
To solve the above described infinite horizon models, we embed adaptive simplicial interpolation in a standard time iteration algorithm (see, e.g. Judd (1998)). To assess the accuracy of the computed equilibria, we follow Judd (1992) in calculating relative errors in Euler equations, subsequently called *Euler errors*. Concerning the measured Euler errors, we find that our method accurately computes equilibria for the two economies considered, both for reasonable and extreme calibrations of our model. Furthermore, we assess the relative performance of the adaptive grid scheme by comparing it to a standard equidistant grid scheme using the same interpolation technique. We find that the adaptive grid scheme dominates by far: One needs to increase the number of equidistant gridpoints, and thereby computation time, by more than two orders of magnitude in order to reach the high accuracy of the adaptive grid scheme. Finally, we demonstrate that ad hoc update procedures that place additional points near the kinks are much less efficient than ASI.

In the literature, many algorithms have been applied to dynamic models with occasionally binding constraints. However, none of the existing algorithms addresses the problems of non-differentiabilities directly. Christiano and Fisher (2000) compare how several algorithms compute equilibria in a one sector growth model with irreversible investment, which has only one continuous state variable. None of the applied algorithms uses an adaptive grid scheme. A grid structure which is not adaptive, but endogenous, is proposed by Carrol (2006) and extended by Barillas and Fernández-Villaverde (2007), Hintermaier and Koeniger (2010), and Ludwig and Schön (2013). This so called endogenous grid method defines a grid on tomorrow's variables, resulting in an endogenous grid on today's variables. Its major advantage is that it avoids the root-finding step. Yet, as it exploits the specific mapping from next period's variables to today's variables, the applicability as well as the concrete implementation of this method depends on details of the model. Maybe most related to our paper, Gruene and Semmler

(2004) propose an adaptive grid scheme for solving dynamic programming problems. However, this method is designed for value function iteration, it interpolates on rectangular elements, and uses estimated local errors of the value function to update the grid. Along all these dimensions their method is orthogonal to our algorithm. The sparse grid Smolyak (1963) algorithm is a well-known approach to high-dimensional interpolation in economics. Krueger and Kubler (2004) use it to compute equilibria in OLG models with state spaces that have up to 30 dimensions. Certainly, this cannot be achieved in feasible time with our algorithm. However, the Smolyak algorithm requires policy functions to be smooth, which is not the case in models with occasionally binding constraints.

Section 2 explains ASI, which is based on Delaunay interpolation and an adaptive grid scheme. The example used to explain ASI is a two period exchange economy where several types of agents trade in a bond subject to ad hoc borrowing constraints. Section 3 shows how the infinite horizon version of this economy is solved by embedding ASI in a time iteration setup. In Section 4, ASI is applied to a model where trade in a bond and a stock is subject to collateral constraints and short-selling constraints. Sections 3 and 4 carefully examine how ASI performs in solving the respective models. Section 5 concludes.

2 Adaptive Simplicial Interpolation

The main innovation of this paper is ASI, which is tailor-made for interpolating policy functions in models with occasionally binding constraints. Section 2.1 gives a simple example of such a model: An exchange economy where heterogeneous agents trade in a one-period bond subject to ad hoc borrowing constraints. Section 2.2 provides a formal characterization of the problems we are considering. Section 2.3 outlines the adaptive simplicial interpolation algorithm we propose, while Sections 2.4 and 2.5 describe the two essential ingredients of the method: a simplicial interpolation technique based on Delaunay tessellation, and an adaptive grid scheme. Finally, Section 2.6 illustrates the workings of ASI with the help of the simple example from Section 2.1.

2.1 Simple Example: Borrowing Constraints

2.1.1 The Bond Economy

The economy is populated by H types of agents h $\in \mathbb{H} = \{1, \ldots, H\}$ living for T periods. Agents have identical preferences, but differ with respect to endowment realizations. They maximize expected time-separable lifetime utility

$$\mathbb{E}\left[\sum_{t=1}^{T} \beta^t \frac{c_t^{1-\gamma}}{1-\gamma}\right],$$

where c_t denotes consumption at t, β is the time discount factor, and γ is the coefficient of relative risk aversion.

Uncertainty is captured by a first-order Markov process with domain $X = \{1, ..., K\}$. Aggregate endowment of the single consumption good is given by a time invariant function $\bar{e} : X \to \mathbb{R}^{++}$. Similarly, agent h's individual endowment is given by $e^h : X \to \mathbb{R}^{++}$.

Each period, agents trade in a one-period bond, which is in zero net supply. Hence, agents face the following budget constraints:

$$c_t^h + b_t^h p_t \leq e_t^h + b_{t-1}^h \;\; \forall t = 1, \ldots, T \;\; \forall h \in \mathbb{H},$$

where b_t^h denotes the bond holding that agent h acquires at time t, and p_t denotes the respective price. Moreover, agents face an ad hoc borrowing constraint:

$$b_t \geq \underline{b} \;\; \forall t = 1, \ldots, T, \;\text{where}\; \underline{b} \in \mathbb{R}^{-}.$$

2.1.2 State Space

The state of the economy at the beginning of a period is characterized by the exogenous shock and the asset distribution among agents. Because of bond market clearing, we may use the bond holdings of $H-1$ agents as the endogenous state variable:

$$y_t = \left(b_{t-1}^1, \ldots, b_{t-1}^{H-1}\right).$$

Assuming that last period's constraints of all agents were satisfied, agent h enters period t with bond holding restricted by

$$b_{t-1}^h \in [\underline{b}, -(H-1)\underline{b}].$$

Hence, we take the endogenous state space to be

$$Y \equiv \left\{ y \in [\underline{b}, -(H-1)\underline{b}]^{H-1} \middle| \sum_{i=1}^{H-1} y_i \in [\underline{b}, -(H-1)\underline{b}] \right\}.$$

The whole state space S is then given by the product of the exogenous part and the endogenous part, i.e. $S = X \times Y$.

2.1.3 Equilibrium Conditions

The endogenous choices and prices in period t are:

$$z_t \equiv \left(\left\{ c_t^h, b_t^h \right\}_{h \in \mathbb{H}}, p_t \right).$$

We call the collection of these endogenous variables *policies*, and denote the space of policies by Z.

The definition of competitive equilibrium is standard and given in Appendix A, where we also derive the first order necessary conditions for equilibrium. Here, we just state these conditions. Along an equilibrium path, policies satisfy market clearing in the bond market, budget constraints, Euler equations, borrowing constraints and complementary slackness conditions:

$$\begin{aligned}
&\sum_{h \in \mathbb{H}} b_t^h = 0, \\
&c_t^h + b_t^h p_t - e_t^h - b_{t-1}^h = 0 \ \forall h \in \mathbb{H}, \\
&-u'(c_t^h)p + \mu_t^h + \mathbb{E}\left[\beta u'(c_{t+1}^h)\right] = 0 \ \forall h \in \mathbb{H}, \\
&0 \leq b_t^h - \underline{b} \perp \mu_t^h \geq 0 \ \forall h \in \mathbb{H},
\end{aligned}$$

where μ_t^h denotes the Kuhn-Tucker multiplier on the borrowing constraint of agent h at time t and the sign $\perp$ denotes orthogonality.

2.1.4 Two Period Version

Now consider the simplest dynamic setting: $T = 2$. In this case there is no trade in the second period and agents simply consume all their funds:

$$c_2^h = e_2^h + b_1^h.$$

Consequently, in period one, equilibrium conditions for given initial bond holdings $\{b_0^h\}_{h \in \mathbb{H}}$ simplify to:

$$\begin{aligned}
&\sum_{h \in \mathbb{H}} b_1^h = 0, \\
&c_1^h + b_1^h p_1 - e_1^h - b_0^h = 0 \;\; \forall h \in \mathbb{H}, \\
&-u'(c_t^h) p_1 + \mu_1^h + \mathbb{E}\left[\beta u'(e_2^h + b_1^h)\right] = 0 \;\; \forall h \in \mathbb{H}, \\
&0 \leq b_1^h - \underline{b} \;\perp\; \mu_1^h \geq 0 \;\; \forall h \in \mathbb{H}.
\end{aligned}$$

In Section 2.6, we will use this two period model as a simple example to illustrate how ASI works. Before that, we describe the problem and its solution by ASI in its general form.

2.2 The General Problem

The above problem of finding an equilibrium policy for the two period bond economy with given initial bond holdings has the following structure:

Equilibrium Problem:
Given a state $s \in S$, and functions
$\phi : S \times \mathbb{R}^{m+n} \to \mathbb{R}^m, \;\; \psi : S \times \mathbb{R}^m \to \mathbb{R}^n,$
find policies and multipliers $(z, \mu) \in \mathbb{R}^m \times \mathbb{R}^n$,
s.t. $\phi(s, z, \mu) = 0, \quad 0 \leq \psi(s, z) \;\perp\; \mu \geq 0.$

In the case of our example, the equations $\phi = 0$ contain market clearing, budget constraints, and Euler equations. The inequalities $0 \leq \psi$ contain the borrowing

constrains, and μ contains the respective Kuhn-Tucker multipliers. To solve such a problem for a given state s, there are many well established procedures (see Section 3.3.1).
However, it is often not enough to solve the above problem for a given state. If one wants to solve and simulate dynamic models, then one typically needs the mapping from the state of the economy, s, into choices and prices, $f(s)$. In other words, one faces a parametric problem, with the state of the economy, s, being the parameter.

Parametric Equilibrium Problem:
Given $\phi : S \times \mathbb{R}^{m+n} \to \mathbb{R}^m,\ \ \psi : S \times \mathbb{R}^m \to \mathbb{R}^n$,
find $f : S \to \mathbb{R}^m,\ \ \mu : S \to \mathbb{R}^n$,
s.t. $\forall s \in S$: $\phi(s, f(s), \mu(s)) = 0, \quad 0 \leq \psi(s, f(s)) \ \perp\ \mu(s) \geq 0.$

One way to compute functions (f, μ) that approximately satisfy these conditions is collocation (see, e.g. Judd (1998)): First, choose a finite grid $G \subset S$.[3] Second, require that the above conditions have to be satisfied precisely on this grid, i.e.

$$\forall g \in G: \quad \phi(g, f(g), \mu(g)) = 0, \quad 0 \leq \psi(g, f(g)) \ \perp\ \mu(g) \geq 0.$$

For each point on the grid, $g \in G$, the solution $f(g)$ is determined by solving this complementarity problem. Aside from the grid G, collocation determines f by interpolating the solutions $\{f(g)\}_{g \in G}$ found on the grid. Clearly, this does not result in a perfect fit, and more importantly, the quality of the fit depends crucially on the location of the gridpoints $g \in G$. In particular, if there are kinks in the function f, it is desirable to put gridpoints there, as any method that interpolates over the kink provides only a poor approximation near the kink. In general, f is non-differentiable at the points k where for some j both $\psi_j(k, f(k))$ and $\mu_j(k)$ are equal to zero. The reason is as follows: $\psi_j(k, f(k)) = 0$ means that this constraint is binding, and $\mu_j(k) = 0$ means that the associated multiplier is zero though. Loosely speaking, the constraint is binding at one side and non-binding at the other side of the point. In general, this implies that the optimal solution is determined by different sets of equations on the two sides of

[3]If the space of exogenous shocks X is discrete, then the grid G on $S = X \times Y$ consists of several grids on Y, one for each $x \in X$.

the point, resulting in different slopes of the policy function.
All in all, the above reasoning suggests that we should put interpolation nodes at points where constraints are just binding. We achieve this using the algorithm presented in Sections 2.3 to 2.5.

2.3 The Algorithm

To solve the parametric equilibrium problem presented above, we propose Adaptive Simplicial Interpolation. An overview of this procedure is given below. Steps two and three are black boxes for now. Sections 2.4 and 2.5 explain these steps in detail. We will explain Delaunay interpolation first, as it includes the concept of tessellation, which we use in the grid adaptation procedure.

Adaptive Simplicial Interpolation:

1. *Initialization:*
   Start with an initial grid G_{init} and solve for the solutions $\{f(g)\}_{g \in G_{init}}$ using standard numerical procedures.

2. *Grid Adaptation:*
   Use the solutions $\{f(g)\}_{g \in G_{init}}$, as explained in *Section 2.5*, to solve jointly for adapted gridpoints G_{adapt} that lie directly on the kinks and for the solutions $\{f(k)\}_{k \in G_{adapt}}$ at these points.

3. *Simplicial Interpolation:*
   Interpolate f on $G = G_{init} \cup G_{adapt}$. To interpolate on a grid with such an irregular shape, use simplicial interpolation, namely Delaunay interpolation, which is explained in *Section 2.4*.

2.4 Delaunay Interpolation

To get as much flexibility as possible in adapting the collocation grid, we need to have a method that is able to interpolate between points from any arbitrary set of scattered points. In addition, we require the method to work in multiple dimensions. Delaunay interpolation fulfills both criteria. This interpolation technique consists of two main steps: First, the state space is divided into simplices, which is done by Delaunay tessellation. Second, simplicial interpolation interpolates locally on these simplices.

2.4.1 Delaunay Tessellation

Delaunay tessellation is a well established method to cover the convex hull of an arbitrary set of points with simplices. It was introduced by Delaunay (1934) and is widely used in engineering. However, as far as we know, we are the first to use this method in economics. For the sake of simplicity, we explain Delaunay tessellation for the two dimensional case. In this case, the simplices are just triangles and the method is called triangulation. In Figure 2, the picture on the left shows a set of scattered gridpoints. The picture on the right shows the Delaunay triangulation of this set of gridpoints. Delaunay triangulation is just one possible way to triangulate a set of gridpoints. However, it imposes discipline on the triangulation by satisfying the following *Delaunay property*: Inside the circumcircle of any triangle there is no point from the set of points. To make sense of this requirement, note that the vertices of a triangle lie on its circumcircle (by definition), and in a Delaunay triangulation other points might as well lie on this circumcircle but not inside. Simpson (1978) shows that this procedure maximizes the minimum angle among all angles within the triangulation. Hence, it avoids pointed triangles. From a numerical perspective, this is a convenient property, since it implies that the information used to interpolate at a particular point stems from points that are relatively close.

While the definition of Delaunay triangulation is straightforward, the efficient and robust computation of such a triangulation for an arbitrary set of points is the focus of a large literature in computational geometry. The most widely used approach is the so called *incremental algorithm*. This algorithm is initialized by adding three points (which are eventually removed) that form a triangle that contains all gridpoints. Starting with this triangle, the algorithm adds one grid point at a time to the triangulation. This works as follows: First, the triangle containing the new point has to be found. For this step, a point location procedure as described in Section 2.4.2 is used. In a second step, the identified triangle is split up into new triangles using the new point as a vertex. This is done in

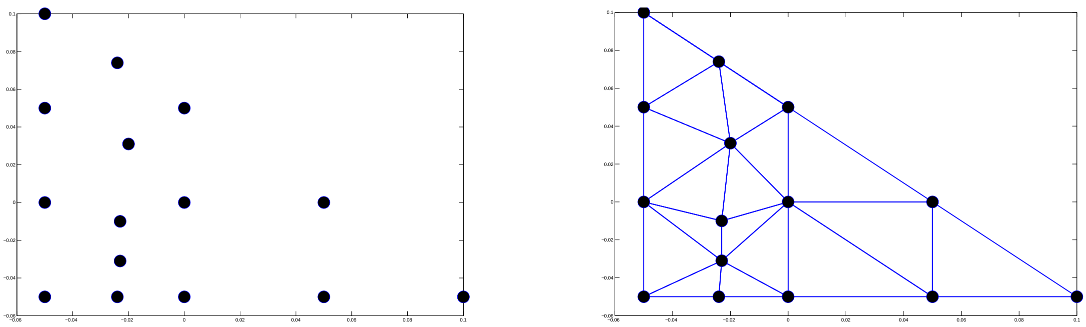

Figure 2: Set of Grid Points (lhs) and its Delaunay Triangulation (rhs)

a way that insures that all of the new triangles satisfy the Delaunay property described above. This procedure continues until all points are added to the triangulation. An extensive description of Delaunay tessellation is provided by de Berg et al. (2008),[4] while Liu and Snoeyink (2005) compare the performance of various algorithms for computing Delaunay tessellation.

2.4.2 Simplicial Interpolation

Having the triangulation of a set of points at hand, linear simplicial interpolation proceeds in two steps. First, the triangle containing the interpolation point is located. Second, the point's *barycentric coordinates* within this triangle are determined. A standard approach to find the triangle where the interpolation point is located is the *stochastic walk algorithm*. As described in Devillers et al. (2002), this algorithm 'walks' through the triangulation: Given a specific triangle, it randomly chooses one of its edges and checks whether the line supporting this edge separates the point from the triangle. If so, the point is obviously not contained in the triangle and the algorithms proceeds by considering the neighboring triangle that shares the chosen edge. If no edge separates the point from the triangle, the triangle containing the point is found. Devillers et al. (2002) prove that this stochastic walk terminates with probability one.

Having found the triangle that contains the point p, the interpolation value at p is given as a linear combination of the function values (v_1, v_2, v_3) at the corners (c_1, c_2, c_3) of the triangle. The weights of this linear combination are the

[4]Chapter 9 of de Berg et al. (2008) describes Delaunay triangulation. It can be found on www.cs.uu.nl/geobook/.

so called barycentric coordinates (b_1, b_2, b_3) of the point p with respect to the corners (c_1, c_2, c_3). The physical interpretation of these coordinates is as follows: If one puts weights (b_1, b_2, b_3) at the corners (c_1, c_2, c_3), then their center of mass lies at the point p. Mathematically, the barycentric coordinates (b_1, b_2, b_3) for a point p with Cartesian coordinates (p_1, p_2) are given by

$$\begin{aligned} b_1 &= \frac{(y_2 - y_3)(p_1 - x_3) + (x_3 - x_2)(p_2 - y_3)}{(y_2 - y_3)(x_1 - x_3) + (x_3 - x_2)(y_1 - y_3)} \\ b_2 &= \frac{(y_3 - y_1)(p_1 - x_3) + (x_1 - x_3)(p_2 - y_3)}{(y_2 - y_3)(x_1 - x_3) + (x_3 - x_2)(y_1 - y_3)} \\ b_3 &= 1 - b_1 - b_2, \end{aligned}$$

where $(x_1, y_1), (x_2, y_2), (x_3, y_3)$ are the Cartesian coordinates of the corners (c_1, c_2, c_3). The interpolation value v at point p is then given by

$$v = b_1 v_1 + b_2 v_2 + b_3 v_3.$$

2.5 An Adaptive Grid Scheme

Let us now turn to the process of adapting the grid. Our aim is to detect kinks and place points on these kinks in order to match them precisely. In terms of the notation of Section 2.2, we want to determine points that lie on

$$K = \{s|\ \exists j\ \ \psi_j(s, f(s)) = 0 \text{ and } \mu_j(s) = 0\}.$$

Hence, we are looking for points where a constraint holds with equality but the respective multiplier is zero, i.e. where the constraint is just binding. To determine such points we proceed as follows.

2.5.1 How to Determine Which Edges Cross Kinks

To determine the location of kinks, we use the solutions $\{f(g)\}$ computed on the initial grid G_{init}. Clearly, if $\psi_j(g, f(g)) = 0$, we know that this constraint, which we call constraint j, is binding at g. Otherwise it is not binding. Furthermore, we make use of the tessellation of the initial grid. We consider each edge of the tessellation and check whether constraint j is binding at one corner and non-binding at the other corner of this edge. If this is the case, we conclude that the associated kink, which we call kink j, crosses this edge. In this way, we find sets of edges $\{\mathcal{E}_j\}$ crossing the kinks $j = 1, \ldots, m$.

2.5.2 How to Put Points Exactly on the Kink

Given the sets of edges $\{\mathcal{E}_j\}$ crossing the kinks $j = 1, \ldots, m$, we need to determine where exactly to put points on these edges. For each individual edge $E \in \mathcal{E}_j$ this is done by solving a modified version of the equation system that characterizes equilibrium. The key conceptional difference is that we let the state variable vary on the edge and do not solve the equation system at a given point in the state space. To pin down the one point that lies on the kink, we force that both ψ_j and μ_j are equal to zero. Hence, we solve jointly for the equilibrium solution *and* for a point in the state space on which the equilibrium solution fulfills a certain requirement, namely that the considered constraint is just binding. More formally, we solve for the point k, policies z, and multipliers μ such that:

$$\begin{aligned}
&\phi(k, z, \mu) = 0, \quad 0 \leq \psi_{-j}(k, z) \perp \mu_{-j} \geq 0, \\
&\psi_j(k, z) = 0, \quad \mu_j = 0, \\
&k \in E.
\end{aligned}$$

By demanding $\psi_j(k, z) = 0$, $\mu_j = 0$ instead of $0 \leq \psi_j(k, z) \perp \mu_j \geq 0$, we reduce the degrees of freedom by one. But letting the state variable k vary on the one-dimensional object E, in contrast to fixing a point in the state space, increases the degrees of freedom by one. Hence, the modified equation system has a (locally unique) solution (k, z, μ), if (z, μ) is a (locally unique) solution to the original equation system at k. This solution does not only provide the point k that lies on the kink, but at the same time it provides the optimal policy at this point, namely $f(k) = z$.

In this way—for all edges E in all sets $\mathcal{E}_j$—we compute points k and policies $f(k)$. We call these points adapted, and denote the set containing them by G_{adapt}. Finally, we add them to the initial points to generate the adapted grid: $G = G_{init} \cup G_{adapt}$.[5]

[5] Instead of this fine tuned adaptation procedure, one could also use a rather mechanical update of the grid. Instead of locating the kink exactly, one could just add arbitrary points into the triangles of interest, e.g. the center point of the triangle or say 5 randomly distributed points. This is easier to program, but comes at the cost of a less accurate result, as we show in Section 3.4.3.

2.6 ASI at Work

Figure 3 visualizes the working principle of ASI. The left hand side displays an initial grid for a given exogenous state of the 2-period bond economy. On the x-axis we have wealth of agent 1, on the y-axis wealth of agent 2—remember that the wealth of agent 3 is given by market clearing. We place 15 equidistant gridpoints on this state space, and we solve the equilibrium problem on this initial grid. Knowing the optimal policies at these points, we now consider each constraint at a time. We start with the borrowing constraint of agent 1. In the left picture, black dots indicate that the constraint of agent 1 is binding, while white dots indicate that it is not binding. Hence, we know on which edges of the triangulation the constraint change from binding to non-binding. On these edges, we apply the second part of our adaptation scheme: we solve the modified equation system that allows us to find the particular point on the edge where the constraint is just binding (e.g. where the kink crosses the edge). Doing this for all relevant edges, we end up with 8 adapted points in this example, which are displayed in the right picture in Figure 3. Finally, a new triangulation is computed for the set of all gridpoints, initial and adapted. After this, we consider the next constraint. However, all other constraints are always non binding in this simple example. Hence, there are no further points to be added. Note that the new triangulation connects the adapted points by edges, thus kinks are matched very accurately. This can also be seen in Figure 4, where the left graph shows the equilibrium bond demand function of agent 1. The range where agent 1 is constrained by the borrowing limit is displayed by the dark shaded area. The kink induced by the inequality constraint is well approximated by the adapted points. The solid line in the right graph displays a slice of the bond demand function of agent 1. The dashed line represents the policy one gets if an equidistant grid is used. Clearly, this policy is quite inaccurate at the kink.

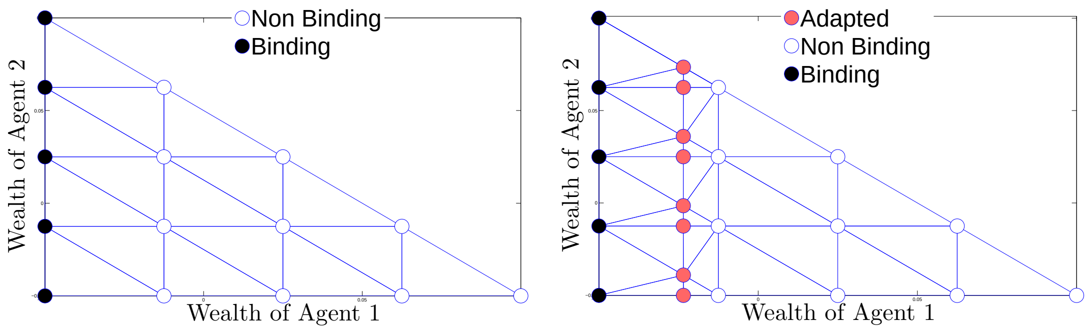

Figure 3: Initial Grid (lhs) and Adapted Grid (rhs) Using ASI in 2D

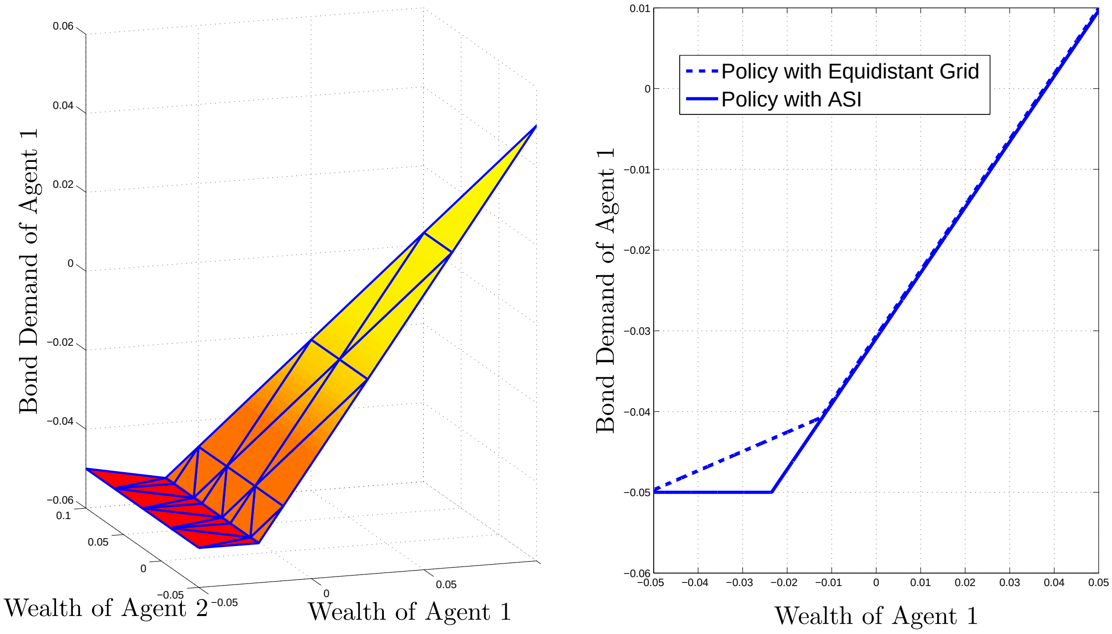

Figure 4: 2D Policy with ASI (lhs) and 1D Slice with and without ASI (rhs)

3 Time Iteration with ASI

We now consider the infinite horizon version of the bond economy from section 2.1. Section 3.1 characterizes recursive equilibrium policies for this model. Section 3.2 shows how such policies may be computed by embedding ASI into a standard time iteration setup. Details of how we implement this algorithm are given in Section 3.3. Finally, Section 3.4 analyzes the computational performance of time iteration with ASI.

3.1 The Infinite Horizon Bond Economy

Consider the bond economy of Section 2.1 with $T = \infty$. We want to describe equilibrium in terms of policy functions that map the current state into current policies:

$$f_t : S \to Z, \ \ f_t : \left(x_t, \left(b^1_{t-1}, \ldots, b^{H-1}_{t-1}\right)\right) \mapsto \left(\left\{c^h_t, b^h_t\right\}_{h \in \mathbb{H}}, p_t\right).$$

For the components of the policy function, we use the same notation as for their values, hence

$$f_t = \left(\left\{c^h_t, b^h_t\right\}_{h \in \mathbb{H}}, p_t\right).$$

For all states, these functions $\{f_t\}$ have to satisfy the period-to-period first order equilibrium conditions (see Appendix A):

$$\begin{aligned}
\forall s : \qquad & \sum_{h \in \mathbb{H}} b^h(s) = 0, \\
& c^h_t(s) + b^h_t(s) p_t(s) - e^h_t(s) - b^h_{t-1}(s) = 0, \ \ \forall h \in \mathbb{H}, \\
& -u'(c^h_t(s)) p_t(s) + \mu^h_t(s) + \mathbb{E}\left[\beta u'\left(c^h_{t+1}\left(s_{t+1}\right)\right)\right] = 0, \ \ \forall h \in \mathbb{H}, \\
& 0 \leq b^h_t(s) - \underline{b} \ \perp \ \mu^h_t(s) \geq 0, \ \ \forall h \in \mathbb{H},
\end{aligned}$$

where $s_{t+1} = \left(x_{t+1}, \left(b^1_t, \ldots, b^{H-1}_t\right)\right)$.

A recursive equilibrium[6] policy function of this economy is a time invariant policy function f that satisfies these conditions, i.e. the sequence $\{f_t\}$ with $f_t = f \ \forall t$ satisfies the above conditions.

[6]As the focus of this paper is computational, we do not discuss existence of equilibria in detail. Note, however, that existence of recursive equilibrium in this setup and also in the extension presented in Section 4 is not guaranteed, while ϵ-recursive Markov equilibria do exist (see, e.g., Kubler and Schmedders (2003)).

3.2 The Algorithm

The above period-to-period equilibrium conditions have the following structure:

$$\forall s: \ \ \phi[f^{next}]\,(s, f(s), \mu(s)) = 0, \quad 0 \leq \psi\,(s, f(s)) \ \perp \ \mu(s) \geq 0,$$

where time t variables have no index, and the policy in $t+1$ is denoted by f^{next}. The equations $\phi[f^{next}] = 0$, which depend on f^{next}, contain market clearing, budget constraints and Euler equations. Only the latter depend on f^{next}—in this case on the consumption policies only. The inequalities $0 \leq \psi$ contain the borrowing constraints, and μ contains the respective Kuhn-Tucker multipliers. A recursive equilibrium policy function f satisfies:

$$\forall s: \ \ \phi[f]\,(s, f(s), \mu(s)) = 0, \quad 0 \leq \psi\,(s, f(s)) \ \perp \ \mu(s) \geq 0.$$

The problem of finding a policy function that (approximately) satisfies this condition is very hard to address directly. In a time iteration procedure, the recursive equilibrium policy function is approximated iteratively: in each step, a simpler problem is solved, where next period's policy, f^{next}, is taken as given. This brings us back to the period-to-period equilibrium conditions:

$$\forall s: \ \ \phi[f^{next}]\,(s, f(s), \mu(s)) = 0, \quad 0 \leq \psi\,(s, f(s)) \ \perp \ \mu(s) \geq 0.$$

This problem takes exactly the form of the parametric equilibrium problem discussed in Section 2.2. Hence, we may use adaptive simplicial interpolation for each step in the time iteration algorithm. The formal structure of the full algorithm is given below. We deviate from a standard time iteration procedure only with regard to the interpolation procedure, which is contained in the inner box.

Time Iteration with Adaptive Simplicial Interpolation:

1. Select a grid G_{init}, an initial policy function f^{init}, and an error tolerance ϵ. Set $f^{next} \equiv f^{init}$.

2. Make one time iteration step: For all $g \in G_{init}$, find $f(g)$ that solves

$$\phi[f^{next}]\,(s, f(s), \mu(s)) = 0, \quad 0 \leq \psi\,(s, f(s)) \;\perp\; \mu(s) \geq 0.$$

   Interpolate f by *adaptive simplicial interpolation*:
   First, use the solutions $\{f(g)\}_{g \in G_{init}}$ to solve jointly for adapted points G_{adapt} that lie directly on kinks and for the optimal policy $\{f(g)\}_{g \in G_{adapt}}$ at these points.
   Second, use solutions at all gridpoints $G = G_{init} \cup G_{adapt}$ to interpolate f by *simplicial interpolation*.

   If $\|f - f^{next}\|_\infty < \epsilon$, go to step 3.
   Else set $f^{next} \equiv f$ and repeat step 2.

3. Set the numerical solution to the infinite horizon optimization problem: $\tilde{f} = f$.

3.3 Implementation of the Algorithm

To demonstrate that our algorithm works well with standard equipment, we use Matlab on an Intel Core 2 Duo 2.40 GHz computer to implement our algorithm. The implementation of our algorithm in other programming languages is also possible. There exists open source software in C/C++ or R for the computation of Delaunay triangulations (see, e.g. CGAL (2013) or Barber et al. (1996)). Combined with standard root-finding algorithms one can then set-up a time iteration procedure with ASI as described above.

3.3.1 Solving the System of Equilibrium Conditions

To solve the complementarity problem at each grid point, one could use a solver that directly applies to complementarity problems. However, we prefer to transform the complementarity problem into a system of equations (see Appendix C) and then apply a standard non-linear equation solver, e.g. Matlab's fsolve or Ziena's Knitro.[7] We are able to solve our models with both solvers. However, we find that the more equations the equlibrium system involves the better is the performance of Knitro compared to fsolve.

3.3.2 Adaptive Simplicial Interpolation

Our method of choice for interpolation is Delaunay interpolation as described in Section 2.4. Delaunay interpolation is widely used in many areas, and hence code in several languages like Fortran, C/C++ or R is available on the web. In Matlab, routines for computing Delaunay tessellations and simplicial interpolation come with the standard version. In 2D and 3D, Matlab adopted the Delaunay interpolation routines from the Computational Geometry Algorithms Library (CGAL).[8] CGAL (2013) uses an incremental algorithm for computing Delaunay tessellation and a stochastic walk algorithm for simplicial interpolation. We describe these algorithms in Section 2.4.1 and 2.4.2, respectively.

3.3.3 Time Iteration

For the computation exercise presented below we set the error tolerance $\epsilon = 10^{-5}$. We set the initial policy function f^{init} such that agents consume all their wealth and the price of all assets is equal to zero. Hence, f^{init} corresponds to the policy function in the final period of a finite horizon economy. This is not an efficient starting guess, but it makes the computing times of our examples comparable. As a starting guess for solving the equilibrium problem at a given point, we use the solution from the previous iteration. In case the solver cannot find a root we use solutions from neighboring points as new starting guesses. In this way

[7] Matlab's fsolve is part of Matlab's optimization toolbox. For Knitro there are free trial licenses available for academic use.

[8] For higher dimensions, Matlab relies on routines from the Qhull project (see Barber et al. (1996)).

we always find solutions that satisfy the error tolerance.
To decrease CPU time, we start the time iteration procedure with a relatively coarse equidistant grid, and increase the density of the grid as the error in $\|f - f^{next}\|_\infty$ falls below $\epsilon \cdot 10$. We repeat this several times until we reach a grid of certain predefined size. In the comparison studies below, this refinement of the equidistant grid is done in exactly the same way for the adaptive grid method and the equidistant benchmark.
To further decrease CPU time, we do not use adaptive simplicial interpolation at each iteration step. The first step is not done until all refinements of the equidistant grid are carried out and the error in $\|f - f^{next}\|_\infty$ falls below $\epsilon \cdot 10$ again. Note that kinks in policy functions change their location along the time iteration procedure. Hence, it is important to use a sufficient number of adaptation steps. Furthermore, note that at each adaptation step, we compute new adapted nodes and do not use the adapted nodes from the last step any more.
Note that the above mentioned measures to reduce computation time, i.e. starting with a coarse grid and adapting the grid only later, are beneficial for infinite horizon economies only. When solving models with finite horizon, one is interested in the policy functions at each iteration step. Therefore, one would use adaptive simplicial interpolation right from the beginning, which makes its comparative advantage even stronger.

3.4 Computational Performance

To evaluate the computational performance of *time iteration with adaptive simplicial interpolation*, we first report the accuracy of the computed equilibria for various examples. Second, we compare time iteration with ASI to two other grid structures: an equidistant grid, and an ad hoc update scheme that places additional gridpoints randomly into simplices that are cut by kinks.

3.4.1 Measuring Accuracy

Following Judd (1992) we evaluate the accuracy of a computed equilibrium by calculating *relative errors in Euler equations* (EEs). An EE measures the error that an agent would make in terms of his period-to-period consumption decision, if he used the computed policy function. The unit of measure is the relative deviation of computed (i.e. interpolated) consumption, c_t^{int}, from the one that is optimal, c_t^{opt}, given next periods interpolated consumption, c_{t+1}^{int}. To derive c_t^{opt} from c_{t+1}^{int} one uses an Euler equation. For instance, in the Bond economy of Section 2.1 the Euler error $EE^h(\cdot)$ for agent h at a particular point s in the state space is given by

$$EE^h(s) = \left| \frac{c_t^{opt}}{c_t^{int}} - 1 \right| = \left| \frac{u'^{-1}\left(\beta \mathbb{E}_t \left[\frac{u'(c_{t+1}^{int})}{p_{int}} \right] \right)}{c_t^{int}(s)} - 1 \right|,$$

where p_{int} is the interpolated price of the bond today. However, it is possible to back out c_t^{opt} from c_{t+1}^{int} only if the Kuhn-Tucker multiplier entering the Euler equation is zero, i.e. if the respective constraint is non-binding. If it is binding, we set the Euler error equal to zero. Note however, that there is at least one unconstrained agent at each point in the state space. For this agent, the Euler error is non-trivial and enters our overall accuracy measures that we introduce next.

To evaluate the accuracy of computed equilibria, we calculate the Euler errors of all agents at many points in the state space. Concerning the choice of points, we make two alternative choices. First, we draw 10.000 random points from a uniform distribution over the whole state space (EE state space), and compute Euler errors for all agents at these points. Second, we take the points reached along the equilibrium path, when the economy is simulated for 5.000 periods (EE equilibrium path). In both cases, we report both the maximum over all agents and points (max EE) as well as the average across points of the maximum across agents (∅ EE). This results in four different statistics, which we all report in $\log_{10}$ scale.

The examples that we consider have three or four agents and a borrowing limit of $\underline{b} = 0.1$ or 1.0, i.e. borrowing is restricted to 10% or 100% of average individual yearly income. Concerning all other parameters, we choose values that are considered standard in the literature, which we report in Appendix D. Tables 3.1 and 3.2 report the accuracy measures for the three and four agent examples respectively. Maximal Euler errors over the state space range from about -3 (for three agents and $\underline{b} = 0.1$) to -1.7 (for four agents and $\underline{b} = 1.0$). All errors are reasonably low, but could be improved much further by increasing the number of initial gridpoints, which would in turn also increase the number of adapted points. Generally speaking, a looser borrowing limit $\underline{b}$ and/or a greater number of agents—which both enlarge the state space—result in higher Euler errors. In the case of four agents, we are dealing with a three-dimensional state space, and kinks become two dimensional objects. This is illustrated in Figure 5, which displays a three dimensional grid that is adapted to a kink that lies approximately orthogonal to the horizontal axis.

Bond Economy with Three Agents

			EE state space		EE equilibrium path	
$\underline{b}$	points	time(min)	max EE	avg EE	max EE	avg EE
0.1	**40**(45)	**0.5**(0.4)	**-3.0**(−1.2)	**-3.8**(−2.1)	**-2.4**(−1.2)	**-4.4**(−2.1)
0.1	**113**(120)	**1.1**(1.0)	**-3.2**(−1.6)	**-4.2**(−2.8)	**-3.2**(−1.6)	**-4.8**(−3, 4)
1.0	**185**(190)	**6.5**(4.5)	**-2.1**(−1.1)	**-3.1**(−2.6)	**-2.2**(−1.1)	**-3.1**(−1.8)
1.0	**941**(946)	**13**(11)	**-3.2**(−1.2)	**-4.2**(−2.9)	**-3.2**(−1.6)	**-4.8**(−3.4)

Table 3.1: Accuracy of Adaptive Grid (Equidistant Grid in Brackets)

Bond Economy with Four Agents

			EE state space		EE equilibrium path	
$\underline{b}$	points	time(min)	max EE	avg EE	max EE	avg EE
0.1	**112**(120)	**4.5**(4)	**-2.7**(−1.3)	**-3.3**(−2.0)	**-2.7**(−1.3)	**-3.9**(−1.7)
1.0	**914**(969)	**60**(51)	**-1.7**(−1.1)	**-2.6**(−2.4)	**-1.8**(−1.1)	**-2.6**(−3.9)

Table 3.2: Accuracy of Adaptive Grid (Equidistant Grid in Brackets)

3.4.2 Comparison to Equidistant Grid

In order to assess the relative performance of ASI, we also compute equilibria on a standard equidistant grid, but still use Delaunay interpolation. To assess the gains from using an adaptive grid scheme, we ask the following questions: First, how do solutions on equidistant grids compare to solutions on adaptive grids, if (almost) the same number of gridpoints is used? Second, how many equidistant gridpoints are needed to match the accuracy of ASI?

When using the same number or slightly more points, the equidistant grid scheme is slightly faster. However, the difference is quite small, reinforcing our claim that adapting the grid takes very little time compared to overall computing time. More importantly, our algorithm outperforms the standard grid scheme by up

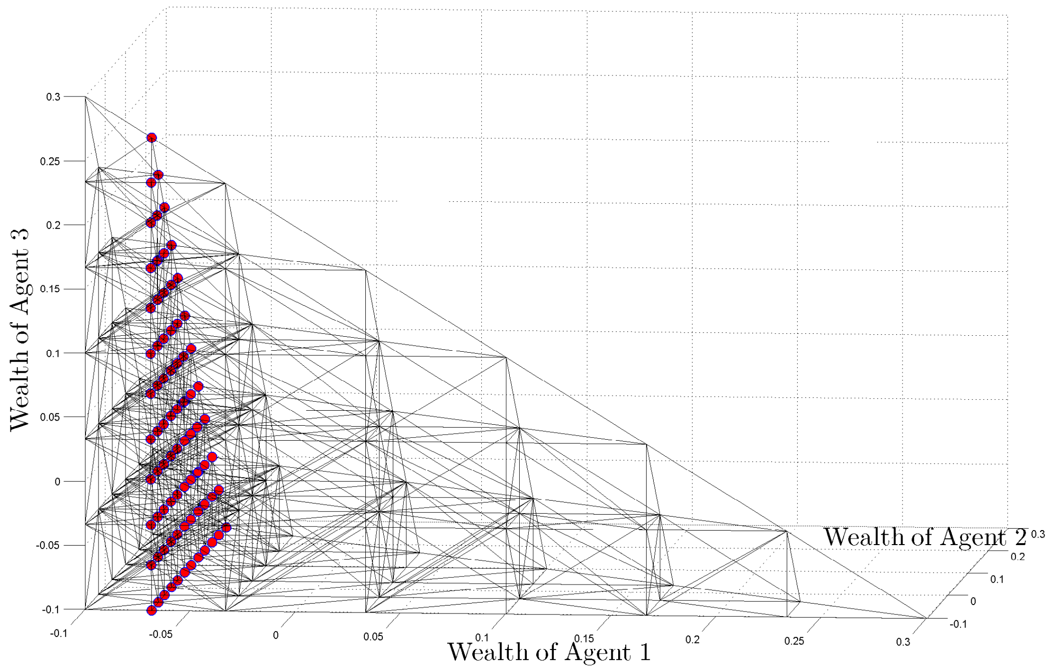

Figure 5: Adapted Grid with Three Continuous State Variables

to two orders of magnitude in terms of maximum Euler errors. This is true for Euler errors drawn over the whole state space and also along the equilibrium path. In the first example of Table 3.1, where we compare our results to an equidistant grid with about the same number of points, the adaptive grid yields maximum Euler errors that are about 70 times lower both on the state space and along the equilibrium path. Regarding the average Euler error, these factors are slightly lower but still substantial. We get these lower factors for average Euler errors, because the adaptive grid scheme rather targets the maximum Euler error by placing gridpoints on kinks, and not elsewhere in the state space. However, for two reasons the impact on average errors is also quite substantial. First, errors at the kinks are lowered dramatically, having a sizable effect on the average error. And second, even at a point located elsewhere, kinks may still play a role, because agents potentially end up near a kink tomorrow.

As a second exercise, we ask how many equidistant gridpoints are needed to get the same maximum Euler error as with a given adapted grid. Instead of targeting the number of gridpoints as above, we therefore target the maximum Euler

Bond Economy with Three Agents: Match Accuracy

			EE state space		EE equilibrium path	
$\underline{b}$	points	time(min)	max EE	avg EE	max EE	avg EE
0.1	**40**(20301)	**0.5**(79)	**-3.0**(-2.8)	**-4.1**(-5.3)	**-3.0**(-3.1)	**-4.4**(-6.2)
1.0	**185**(21945)	**6.5**(300)	**-2.1**(-1.9)	**-3.1**(-4.4)	**-2.2**(-1.9)	**-3.1**($-4,6$)

Table 3.3: Accuracy of Adaptive Grid (Equidistant Grid in Brackets)

Bond Economy with Four Agents: Match Accuracy

			EE state space		EE equilibrium path	
$\underline{b}$	points	time(min)	max EE	avg EE	max EE	avg EE
0.1	**112**(20825)	**4.5**(895)	**-2.7**(-2.0)	**-3.3**(-3.6)	**-2.7**(-2.1)	**-3.9**(-4.0)
1.0	**914**(20825)	**90**(3655)	**-1.7**(-1.1)	**-2.6**(-3.0)	**-1.8**(-1.1)	**-2.6**(-1.9)

Table 3.4: Accuracy of Adaptive Grid (Equidistant Grid in Brackets)

error over the state space. For the first example with $\underline{b} = 0.1$, we increase the grid size by a factor of 500. Interestingly, adaptive simplicial interpolation still outperforms the equidistant grid in terms of maximum Euler errors as reported in Table 3.3. Obviously, in terms of average Euler errors, taking 500 times more points makes a big difference, resulting in a lower error for the equidistant grid. For $\underline{b} = 1.0$, we cannot multiply the number of gridpoints by 500, as memory constraints restrict us to just above 20 thousend points. We therefore increase the grid size by a factor of 120, which yields maximum errors that are still higher than with adaptive simplicial interpolation.
When it comes to four agents, we also find that ASI outperforms equidistant gridpoints by far, as the results in Table 3.4 suggest. Trying to match the maximum Euler Error from the ASI example, we increase the amount of gridpoints by a factor of 200 for $\underline{b} = 0.1$ and 20 for $\underline{b} = 1.0$. For both cases we find that the maximum Euler Error on the equidistant grid is still far higher.

So far we have measured accuracy in terms of Euler errors. We now discuss whether the inaccuracies reflected in Euler errors really matter. For this purpose we consider the average interest rate over very long simulations of the bond economy with four agents and $\underline{b} = 0.1$. Figure 6 shows the deviation of the interest rate from its value in the precise solution (computed on a very fine grid) as a function of the number of gridpoints, both for adaptive and equidistant grids. One can see from the figure that the interest rate for adaptive grids is much more accurate than for equidistant grids of similar size, it converges much faster to the precise solution. Thus, the adaptive grid not only provides lower Euler errors but also more precise statistics from model simulations. However, Figure 6 also shows that equidistant grids of moderate size still generate solutions with acceptable accuracy for the model under consideration. One could thus argue, that locating the kink precisely is not worth the effort. For some applications this is clearly true. Yet in other applications, statistics of interest might be more sensitive than in this model, or a very precise solution might be needed. An example for the first case are models with heterogeneous preferences, where movements in the wealth distribution can have a strong impact on moments of asset prices. Therefore, capturing these movements inaccurately leads to imprecise measures of the moments. An example for the second case, that very precise solutions are needed, is welfare analysis: As welfare often changes only very little when policies or market arrangements are changed, it is important that welfare for the different regimes is measured very precisely, otherwise the computed welfare effect could be far off, and even its sign could be wrong. Finally, we would like to point out that one often cannot properly judge whether an imprecise solution is good enough for a given purpose if one does not have a more precise solution to compare with.

3.4.3 Comparison to ad hoc Update

Finally we compare the accuracy of equilibria computed with ASI to the accuracy of equilibria computed with an ad hoc update scheme. Using the solution from the initial grid this scheme detects which simplices are cut by a kink. Instead of adding points exactly on the kink as done by ASI, the ad hoc update randomly places additional gridpoints into these simplices. To compare this ad hoc update scheme with ASI we now compute equilibria for the examples considered above using the same initial grid as with ASI. As the results in Table 3.5 and 3.6 suggest ASI outperforms such an ad hoc update, even if we use up to 200 times more gridpoints.

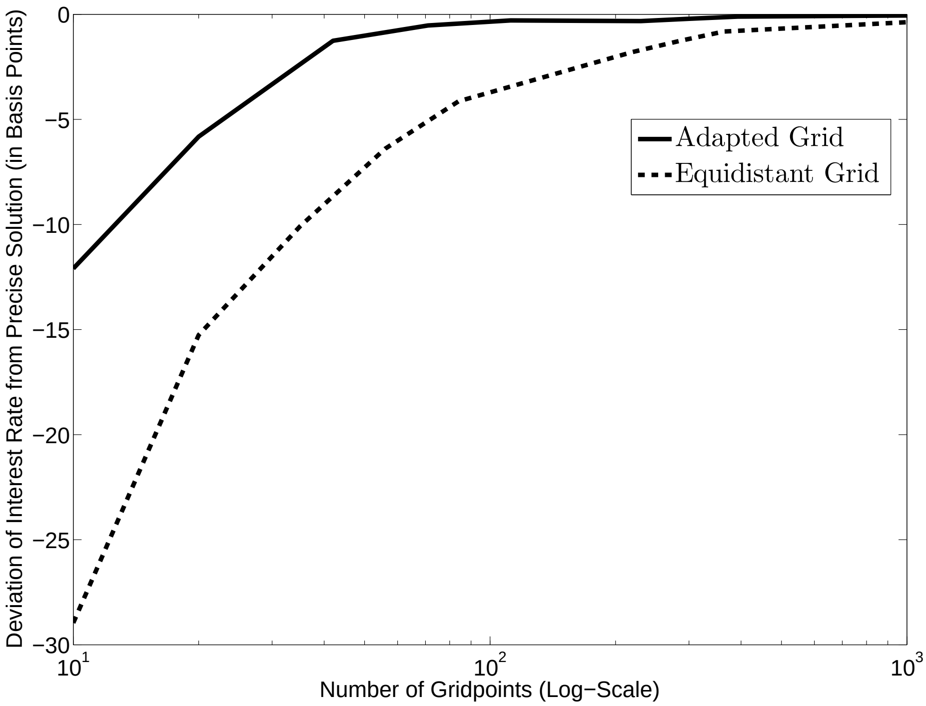

Figure 6: Accuracy of the Computed Interest Rate for Different Grids

Bond Economy with Three Agents: Comparison to ad hoc Update

			EE state space		EE equilibrium path	
$\underline{b}$	points	time(min)	max EE	avg EE	max EE	avg EE
0.1	**40**(8000)	**0.5**(40)	**-3.0**(-1.3)	**-4.1**(-2.7)	**-3.0**(-2.7)	**-4.4**(-4.5)
1.0	**185**(8000)	**6.5**(122)	**-2.1**(-2.0)	**-3.1**($-3,1$)	**-2.2**(-1.9)	**-3.1**($-4,6$)

Table 3.5: Accuracy of Adaptive grid (Grid with ad hoc Update in Brackets)

Bond Economy with Four Agents: Comparison to ad hoc Update

			EE state space		EE equilibrium path	
$\underline{b}$	points	time(min)	max EE	avg EE	max EE	avg EE
0.1	**112**(8000)	**4.5**(333)	**-2.7**(-2.3)	**-3.3**(-3.9)	**-2.7**(-2.4)	**-3.9**(-4.3)
1.0	**914**(20825)	**90**(3655)	**-1.7**(-1.3)	**-2.6**(-3.0)	**-1.8**(-1.1)	**-2.6**(-1.9)

Table 3.6: Accuracy of Adaptive Grid (Grid with ad hoc Update in Brackets)

4 Extension: Endogenous Collateral Constraints

The above setup with a bond as the only asset and simple ad hoc borrowing constraints was very convenient for explaining how ASI works. However, to show the potential of ASI, we now consider a richer setup, which includes a stock that can be used as collateral for borrowing. This collateral constraint makes the borrowing limit dependent on the current stock holding as well as on next period's price of the stock, which is endogenously determined. As a consequence, it is now much harder to locate the kink, and ad hoc methods would fail.

4.1 The Bond and Stock Economy

4.1.1 Setup

We extend the bond economy of Section 2.1 by introducing a Lucas tree-type stock which is in unit net supply. It pays out a fixed fraction δ of aggregate endowment each period, i.e. stock holders receive dividends $d(x) = \delta \cdot \bar{e}(x)$ per unit of the stock. Hence, aggregate endowment is given by the sum of individual endowments and dividends, i.e.

$$\bar{e}(x) = \sum_{h \in \mathbb{H}} e^h(x) + d(x) \;\; \forall x \in X.$$

The Lucas tree is traded each period after dividends are paid. Each agent h buys l^h shares of the stock at a price q. Hence, agents face the following budget constraints:

$$c_t^h + b_t^h p_t + l_t^h q_t \leq e_t^h + b_{t-1}^h + l_{t-1}^h (q_t + d_t) \;\; \forall t = 1, \ldots, T \;\; \forall h \in \mathbb{H}.$$

Moreover, trade in the bond and the stock is subject to constraints. First, we impose a *short-selling constraint* on the stock, i.e.

$$l_t^h \geq 0 \;\; \forall t = 1, \ldots, T \;\; \forall h \in \mathbb{H}.$$

In contrast to the stock, the bond may be shorted. However, only if the stock is used as collateral. More precisely, the short position in the bond may not exceed the minimal value—in terms of resale value plus dividends—that the stock has next period:

$$-b_t^h \leq \min_{x_{t+1} \in X} \left\{ l_t^h \left(q(s_{t+1}) + d(x_{t+1}) \right) \right\}, \forall t = 1, \ldots, T \ \forall h \in \mathbb{H},$$

where tomorrow's state is $s_{t+1} = (x_{t+1}, y_{t+1})$. The endogenous part of the state, y_{t+1}, will be specified below. This constraint is motivated by a bankruptcy law which makes it possible to seize an agents' stock holding, but not his income. To put it differently, all future income is exempted. As there is no further punishment for default, an agent will default on his asset position, if and only if his portfolio has a negative value. As this behavior is anticipated—and we assume that default premia may not be charged—no agent will be allowed to acquire such a portfolio, which imposes the above constraint.

4.1.2 State Space

With the above collateral constraint, financial wealth,

$$w_t^h \equiv l_{t-1}^h \left(q(s_t) + d(x_t) \right) + b_{t-1}^h,$$

cannot go below zero. Hence, the fraction of total financial wealth that an agent holds,

$$y^h = \frac{w^h}{\sum_{j \in \mathbb{H}} w^j},$$

is bounded between zero and one. By market clearing, we may use the fractions of financial wealth of the first $H - 1$ agents as the endogenous state space:

$$y = \left(y^1, \ldots, y^{H-1} \right) \in Y \equiv \left\{ y \in \mathbb{R}_+^{H-1} \middle| \sum_{i=1}^{H-1} y^i \leq 1 \right\} \subset \mathbb{R}_+^{H-1}.$$

Finally, we define the whole state space S as the product of the exogenous part and the endogenous part, i.e. $S = X \times Y$.
With this definition of the state space, reconsider the collateral constraint above, and note that: Todays choice of any agent, through its impact on tomorrows state, influences tomorrows price of the stock, and hence today's collateral constraint of agent h. In this sense, the collateral constraint is endogenous, which complicates the model considerably relative to the model with ad hoc borrowing constraints that we have considered above.

4.1.3 Equilibrium Conditions

The endogenous choices and prices in period t are

$$z_t \equiv \left(\left(c_t^h, b_t^h, l_t^h \right)_{h \in \mathbb{H}}, p_t, q_t \right).$$

In Appendix B we define competitive equilibrium and derive the first-order equilibrium conditions of this model. Along an equilibrium path, policies have to satisfy market clearing on both asset markets, budget constraints, Euler equations for both assets, and complementary slackness conditions for both kinds of multipliers:

$$\begin{aligned}
&\sum_{h \in \mathbb{H}} b_t^h = 0, \quad \sum_{h \in \mathbb{H}} l_t^h = 1, \text{and } \forall h \in \mathbb{H} \\
&c_t^h + b_t^h p_t + l_t^h q_t - e_t^h - b_{t-1}^h - l_{t-1}^h \left(q_t + d_t \right) = 0, \\
&-u'(c_t^h) p_t + \mu^h + \mathbb{E}\left[\beta u'(c_{t+1}^h) \right] = 0, \\
&-u'(c_t^h) q_t + \mu^h \min_{x_{t+1} \in X} \left\{ q(s_{t+1}) + d(x_{t+1}) \right\} + \nu_t^h + \mathbb{E}\left[\beta u'(c_{t+1}^h) \left(q_{t+1} + d_{t+1} \right) \right] = 0, \\
&0 \leq \min_{x_{t+1} \in X} \left\{ l_t^h \left(q(s_{t+1}) + d(x_{t+1}) \right) + b_t^h \right\} \perp \mu_t^h \geq 0, \\
&0 \leq l_t^h \perp \nu_t^h \geq 0,
\end{aligned}$$

where μ^h and ν^h denote the Kuhn-Tucker multipliers on the collateral and the short-selling constraint of agent h.

4.2 Computational Performance

Before we look at errors in Euler equations, we first discuss how the kinks induced by the short selling and collateral constraints are located within the state space. Figure 7 shows the adapted grid for an exogenous state where the first agent is hit by a bad idiosyncratic shock. To clearly visualize the kinks, we highlight the edges that connect adapted points. The short selling constraint of the first agent induces a kink which has two components, the one which lies almost on the y-axis and the curved one to the very right. Furthermore, each of the collateral constraints induces one kink, where the kink from the first agent's constraint runs approximately parallel to the y-axis at about 0.08 fraction of wealth of agent 1. In Figure 8 one can see how these kinks shape an equilibrium policy function. The left hand picture displays the stock demand over the full

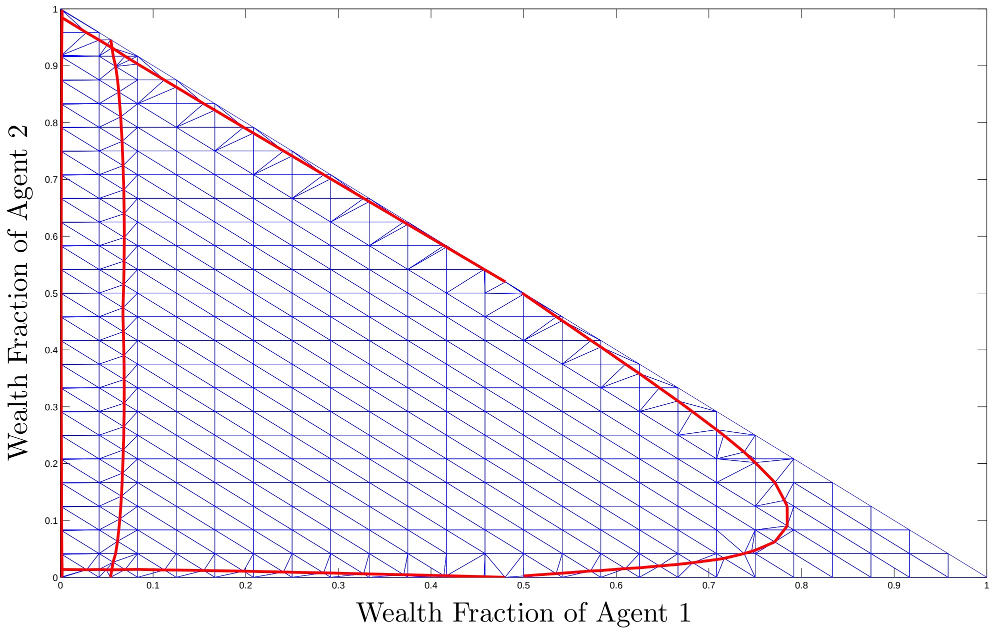

Figure 7: Bond and Stock Economy: Adapted Grid with Several Identified Kinks

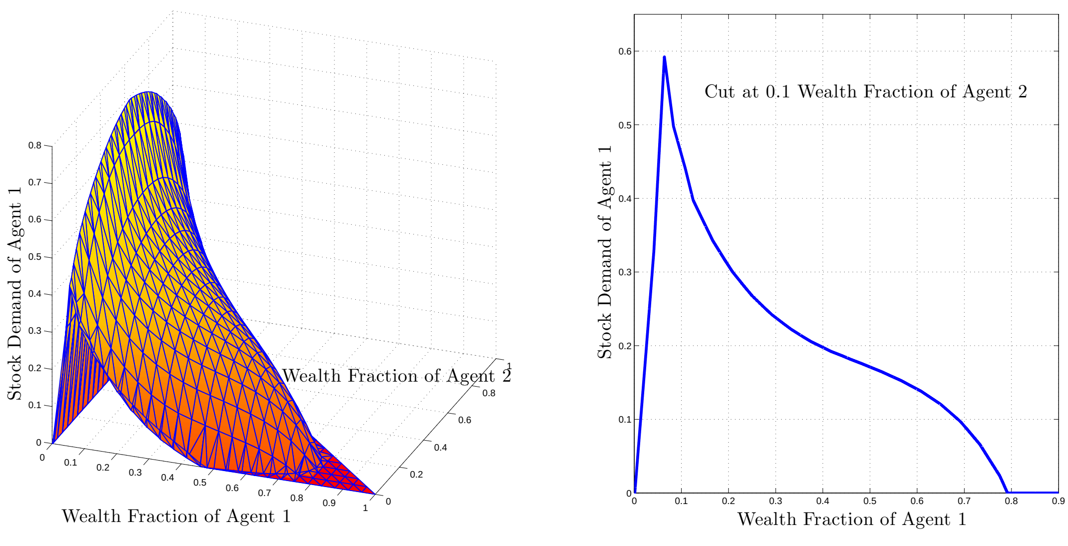

Figure 8: Bond and Stock Economy: 2D Stock Demand (lhs) and 1D Slice (rhs)

state space, whereas the picture on the right hand side displays a slice at 0.1 wealth fraction of agent 2. The distinct peak at 0.08 wealth fraction of agent 1 corresponds to the kink induced by his collateral constraint. To the left, the collateral constraint is binding as agent 1 is so poor that she borrows up to maximum amount possible. When she becomes wealthier she increases her stock holding until the collateral constraint is not binding anymore. At higher levels of wealth her demand for the stock goes down as she is no longer willing to pay the collateral premium for holding the tree. When she has a wealth share of around 80% the short selling constraint becomes binding. Here, the other two agents desire to hold the entire tree as to satisfy their collateral constraints.
As in Section 3.4, we evaluate the performance of our algorithm by computing relative errors in Euler equations. In Table 4.1, we show results for equilbria computed with ASI using two different values for the dividend parameter δ. For all other parameters, we use the same calibration as for the Bond economy (see Appendix D). Obviously, as the figures above suggest, more points are needed than in the bond model to bring Euler errors down to reasonable values. Comparing the results from ASI with results on equidistant grids, we find that for the same number of gridpoints, ASI outperforms equidistant grids by approximately one order of magnitude in terms of maximum Euler Error. Again, we ask how many points are needed to match the accuracy of ASI. Increasing the number of points up to a factor of 20 yields almost the same maximum Euler Error, as the results in Table 4.2 show. This factor is still substantial, however, not as high as for the Bond model. The reason are non-linearities away from the kink, as can be seen in Figure 8. We have developed an adaptation scheme that adapts the grid to non-linearities, which further improves the relative performance of our algorithm. However, as this is not the focus of this paper, we do not elaborate more on this.

5 Conclusion

This paper presents an algorithm that is tailor-made for computing equilibria in dynamic models with occasionally binding constraints. To directly address the problem of kinks in such models, we develop a new interpolation technique based on adaptive grids and simplicial interpolation. We show that Adaptive

Bond and Stock Economy

			EE state space		EE equilibrium path	
δ	points	time(min)	max EE	avg EE	max EE	avg EE
0.10	**1235**(1250)	**310**(260)	**-2.5**(-1.4)	**-3.8**(-3.2)	**-3.1**(-1.6)	**-4.1**(-3.5)
0.25	**1160**(1225)	**302**(251)	**-2.2**(-1.4)	**-3.3**(-2.9)	**-2.2**(-1.4)	**-3.4**(-2.7)

Table 4.1: Accuracy of Adaptive Grid (Equidistant Grid in Brackets)

Bond and Stock Economy: Match Accuracy

			EE state space		EE equilibrium path	
δ	points	time(min)	max EE	avg EE	max EE	avg EE
0.10	**1235**(25425)	**310**(4500)	**-2.5**(-2.4)	**-3.8**(-4.0)	**-3.1**(-2.6)	**-4.1**(-4.2)
0.25	**1160**(25425)	**302**(4812)	**-2.2**(-2.1)	**-3.3**(-4.2)	**-2.2**(-2.3)	**-3.4**(-4.4)

Table 4.2: Accuracy of Adaptive Grid (Equidistant Grid in Brackets)

Simplicial Interpolation accurately computes equilibria in dynamic models with several continuous state variables and various inequality constraints. Comparison studies show that our method outperforms other grid techniques by up to two orders of magnitude in terms of maximum errors in Euler equations.

We would like to point out that our paper makes two distinct contributions which could be useful on their own. First, we introduce the concept of Delaunay interpolation into economics which could prove to be a helpful tool for the interpolation of functions on irregular grids in various applications. Second, we propose a method to identify points that lie directly on the kinks induced by occasionally binding constraints. This method can also improve accuracy when combined with interpolation techniques other than Delaunay interpolation.

Potentially, ASI could be used to interpolate the value function in problems with discontinuous choices (e.g. in models with discrete default decisions). In such a setting, policy functions are discontinuous and thus induce kinks into value functions. The application of ASI to such problems is left for future research.

A Details Bond Economy

In this appendix, we define competitive equilibrium and derive first-order equilibrium conditions for the bond economy presented in Section 2.1. For this purpose, some additional notation is needed. We denote the shock at time t by x_t, but the history of shocks that occurred up to period t by x^t. The set of histories up to period t is denoted by X^t, and the set of all possible histories by $\mathbb{X} \equiv \bigcup_{t=1}^{T} X^t$. For x^{t+1} being a possible successor of x^t we write $x^{t+1} \geq x^t$. Finally, the probability of history x^t is denoted by $\pi(x^t)$ and the conditional transition probability by $\pi(x^{t+1}|x^t)$

A.1 Competitive Equilibrium

A competitive equilibrium for an economy with agents' initial bond holdings

$$\left(b_0^h\right)_{h\in\mathbb{H}}$$

is a collection

$$\{z(x^t)\}_{x^t\in\mathbb{X}} \equiv \left\{\left(c^h(x^t), b^h(x^t)\right)_{h\in\mathbb{H}}, p(x^t)\right\}_{x^t\in\mathbb{X}}$$

of consumption allocations, bond holdings, and bond prices that satisfy the following conditions:

1. Markets clear[9]:

$$\sum_{h\in\mathbb{H}} b^h(x^t) = 0 \quad \forall x^t \in \mathbb{X}.$$

2. Given prices $(p(x^t))_{x^t\in\mathbb{X}}$, each agent chooses

$$\left(c^h(x^t), b^h(x^t)\right)\right)_{x^t\in\mathbb{X}}$$

to maximize lifetime utility such that $\forall x^t \in \mathbb{X}$ the following constraints hold:

$$\begin{aligned} &\textit{budget constraint} && c^h(x^t) + b^h(x^t)p(x^t) \leq e^h(x^t) + b^h(x^{t-1}), \\ &\textit{borrowing constraint} && b^h(x^t) \geq \underline{b}. \end{aligned}$$

[9]By Walras' Law market clearing in the asset market(s) implies market clearing in the consumption goods market.

A.2 First-Order Equilibrium Conditions

Each individual agent faces the following optimization problem:

$$\begin{aligned} & \max_{(c(x^t),b(x^t))_{x^t\in\mathbb{X}}} && \mathbb{E}\left[\sum_{t=1}^{T}\beta^t u(c(x^t))\right] \\ & && \text{s.t. } \forall x^t \in \mathbb{X}: \\ & \textit{budget constraint} && c^h(x^t) + b^h(x^t)p(x^t) \leq e^h(x^t) + b^h(x^{t-1}), \\ & \textit{borrowing constraint} && b^h(x^t) \geq \underline{b}. \end{aligned}$$

Denote the multiplier associated with these constraints by $\lambda(x^t)$ and $\mu(x^t)$. Differentiating the Lagrangian with respect to the different choice variables gives

$$\begin{aligned} c(x^t): &\quad \pi(x^t)\beta^t u'(c(x^t)) - \lambda(x^t) = 0 \\ c(x^{t+1}): &\quad \pi(x^{t+1})\beta^{t+1} u'(c(x^{t+1})) - \lambda(x^{t+1}) = 0 \\ b(x^t): &\quad -\lambda(x^t)p(x^t) + \mu(x^t) + \sum_{x^{t+1}\geq x^t}\left(\lambda(x^{t+1})\right) = 0 \end{aligned}$$

Substituting the first two FOCs into the last one, we get the following Euler equation for the bond:

$$-u'(c(x^t))p(x^t) + \mu(x^t) + \sum_{x^{t+1}\geq x^t}\beta\pi(x^{t+1}|x^t)u'(c(x^{t+1})) = 0.$$

In addition, the Kuhn-Tucker FOCs include the following complementarity condition:

$$0 \leq b(x^t) - \underline{b} \;\perp\; \mu(x^t) \geq 0.$$

Combined with market clearing conditions and budget constraints, these are the equilibrium conditions stated in Section 2.1.

B Details Bond and Stock Economy

In this appendix, we define competitive equilibrium and derive first-order equilibrium conditions for the economy presented in Section 4. The notation is as introduced in the beginning of Appendix A.

B.1 Competitive Equilibrium

A competitive equilibrium for an economy with agents' initial portfolios

$$\left(b_0^h, l_0^h\right)_{h\in\mathbb{H}}$$

is a collection

$$\{z(x^t)\}_{x^t\in\mathbb{X}} \equiv \left\{\left(c^h(x^t), b^h(x^t), l^h(x^t)\right)_{h\in\mathbb{H}}, p(x^t), q(x^t)\right\}_{x^t\in\mathbb{X}}$$

of consumption allocations, bond and stock holdings, and prices that satisfy the following conditions:

1. Markets clear:

$$\sum_{h\in\mathbb{H}} b^h(x^t) = 0, \quad \sum_{h\in\mathbb{H}} l^h(x^t) = 1 \quad \forall x^t \in \mathbb{X}.$$

2. Given prices $(p(x^t), q(x^t))_{x^t\in\mathbb{X}}$, each agent chooses

$$\left(c^h(x^t), b^h(x^t), l^h(x^t)\right)\big)_{x^t\in\mathbb{X}}$$

to maximize lifetime utility such that $\forall x^t \in \mathbb{X}$ the following constraints hold:

$$\begin{aligned}
&\textit{budget constraint} && c^h(x^t) + b^h(x^t)p(x^t) + l^h(x^t)q(x^t) \leq \\
& && e^h(x^t) + b^h(x^{t-1}) + l^h(x^{t-1})\left(q_t(x^t) + d_t(x^t)\right), \\
&\textit{short selling constraint} && l^h(x^t) \geq 0 \text{ and} \\
&\textit{collateral constraints} && \min_{x^{t+1}\geq x^t} \left\{l^h(x^t)\left(q(x^{t+1}) + d(x^{t+1})\right) + b^h(x^t)\right\} \geq 0.
\end{aligned}$$

B.2 First-Order Equilibrium Conditions

Each individual agent faces the following optimization problem:

$$\begin{aligned}
& && \max_{(c(x^t),b(x^t),l(x^t))_{x^t\in\mathbb{X}}} \quad \mathbb{E}\left[\sum_{t=1}^{T} \beta^t u(c(x^t))\right] \\
& && \text{s.t. } \forall x^t \in \mathbb{X}: \\
&\textit{budget constraint} && c^h(x^t) + b^h(x^t)p(x^t) + l^h(x^t)q(x^t) \leq \\
& && e^h(x^t) + b^h(x^{t-1}) + l^h(x^{t-1})\left(q_t(x^t) + d_t(x^t)\right), \\
&\textit{short selling constraint} && l^h(x^t) \geq 0 \text{ and} \\
&\textit{collateral constraints} && \min_{x^{t+1}\geq x^t} \left\{l^h(x^t)\left(q(x^{t+1}) + d(x^{t+1})\right) + b^h(x^t)\right\} \geq 0.
\end{aligned}$$

Denote the multipliers associated with these constraints by $\lambda(x^t)$, $\nu(x^t)$, and $\mu(x^t)$. Differentiating the Lagrangian gives

$$
\begin{aligned}
c(x^t): &\qquad \pi(x^t)\beta^t u'(c(x^t)) - \lambda(x^t) = 0 \\
c(x^{t+1}): &\qquad \pi(x^{t+1})\beta^{t+1} u'(c(x^{t+1})) - \lambda(x^{t+1}) = 0 \\
b(x^t): &\qquad -\lambda(x^t)p(x^t) + \mu(x^t) + \sum_{x^{t+1}\geq x^t} \left(\lambda(x^{t+1})\right) = 0 \\
l(x^t): &\qquad \nu(x^t) - \lambda(x^t)q(x^t) + \mu(x^t) \min_{x^{t+1}\geq x^t} \left\{q(x^{t+1}) + d(x^{t+1})\right\} \\
&\qquad + \sum_{x^{t+1}\geq x^t} \left(\lambda(x^{t+1})\right)\left(q(x^{t+1}) + d(x^{t+1})\right) = 0.
\end{aligned}
$$

Substituting the first two FOCs into the last two, we get the following Euler equations for the bond and the stock:

$$
\begin{aligned}
&-u'(c(x^t))p(x^t) + \mu(x^t) + \sum_{x^t+\geq x^t} \left(\beta\pi(x^{t+1}|x^t)u'(c(x^{t+1}))\right) = 0, \\
&\nu(x^t) - u'(c(x^t))q(x^t) + \mu(x^t) \min_{x^{t+1}\geq x^t} \left\{q(x^{t+1}) + d(x^{t+1})\right\} \\
&+ \sum_{x^{t+1}\geq x^t} \left(\beta\pi(x^{t+1}|x^t)u'(c(x^{t+1}))\right)\left(q(x^{t+1}) + d(x^{t+1})\right) = 0.
\end{aligned}
$$

In addition, the Kuhn-Tucker FOCs include the following complementarity conditions:

$$
\begin{aligned}
&0 \leq \min_{x^{t+1}\geq x^t} \left\{l^h(x^t)\left(q(x^{t+1}) + d(x^{t+1})\right) + b^h(x^t)\right\} \;\perp\; \mu(x^t) \geq 0 \\
&0 \leq l(x^t) \;\perp\; \nu(x^t) \geq 0.
\end{aligned}
$$

Combined with market clearing conditions and budget constraints, these are the equilibrium conditions stated in Section 4.

C Transforming Complementarities into Equations

At the initial gridpoints, ASI solves the follwing complementarity problem:

$$\begin{aligned}
&\text{Given a state } s \in S \text{ , and functions}\\
&\phi : S \times \mathbb{R}^{m+n} \to \mathbb{R}^m,\ \ \psi : S \times \mathbb{R}^m \to \mathbb{R}^n,\\
&\text{find policies and multipliers } (z, \mu) \in \mathbb{R}^m \times \mathbb{R}^n,\\
&\ \text{s.t. } \phi(s, z, \mu) = 0, \quad 0 \leq \psi(s, z) \ \perp\ \mu \geq 0.
\end{aligned}$$

Following Garcia and Zangwill (1981), we transform this complementarity problem into a system of equations, to be able to apply a standard non-linear equation solver. Key to the transformation are the following definitions:

$$\alpha \equiv \begin{cases} \mu & \text{for} \quad \mu \geq 0,\ \ \psi(s, z) = 0 \\ -\psi(s, z) & \text{for} \quad \mu = 0,\ \ \psi(s, z) > 0 \end{cases}$$

and

$$\begin{aligned}
\alpha^+ &= (\max(0, \alpha))^k \\
\alpha^- &= (\max(0, -\alpha))^k,
\end{aligned}$$

where $k \in \mathbb{N}^+$. Using these definitions, the problem reads:

$$\begin{aligned}
&\text{Given a state } s \in S \text{ , and functions}\\
&\phi : S \times \mathbb{R}^{m+n} \to \mathbb{R}^m,\ \ \psi : \mathbb{R}^m \to S \times \mathbb{R}^n,\\
&\text{find policies and alphas } (z, \alpha) \in \mathbb{R}^m \times \mathbb{R}^n,\\
&\ \text{s.t. } \phi(s, z, \alpha^+) = 0,\ \ \psi(s, z) - \alpha^- = 0.
\end{aligned}$$

D Parameterization

We set the discount factor $\beta = 0.95$ and the risk aversion parameter $\gamma = 1.5$ for all agents. Concerning the exogenous shock process, we make the following choices: We assume that agents may either receive a good or a bad idiosyncratic shock. One agent always gets the bad shock and all others get the good one. This results in three or four states per aggregate shock, depending on the number of agents. Allowing for two aggregate shocks the exogenous part of the state space comprises six or eight states respectively. We denote the ratios of good to bad idiosyncratic and aggregate shocks by ν_{idio} and ν_{agg}. We finally denote the persistence of idiosyncratic and aggregate shocks by ρ_{idio} and ρ_{agg}. We compute equilibria for two values of the borrowing limit $\underline{b}$, namely $\underline{b} = 0.1$ and 1, i.e. borrowing up to 10% or 100% of average individual yearly income. All parameter values can be found in Table D.1.

γ	ν_{idio}	ν_{agg}	ρ_{idio}	ρ_{agg}	β	$\underline{b}$
1.5	1.6	1.06	0.9	0.65	0.95	0.1/1.0

Table D.1: Parameter Values

Acknowledgements

We have benefited greatly from comments by Klaus Adam, Wouter den Haan, Georg Dürnecker, Philip Jung, Winfried Koeniger, Tom Krebs, Dirk Krueger, Felix Kubler, Michael Reiter, Karl Schmedders and participants to seminars at the Universities of Mannheim, Stockholm, and Zurich, SAET 2009 in Ischia, ICE 2009 in Chicago, SED 2010 in Montreal, CEF 2010 in London, SAET 2010 in Singapore, and ESWC 2010 in Shanghai. All remaining errors are ours. Johannes Brumm gratefully acknowledges financial support from the ERC. The views expressed in this paper are those of the authors and do not necessarily reflect the position of the Deutsche Bundesbank.

References

Barber, C. B., D. P. Dobkin, and H. Huhdanpaa (1996). The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software (TOMS) 22*(4), 469–483. http://www.qhull.org.

Barillas, F. and J. Fernández-Villaverde (2007). A generalization of the endogenous grid method. *Journal of Economic Dynamics and Control 31(8)*, 2698–2712.

Carrol, C. D. (2006). The method of endogenous gridpoints for solving dynamic stochastic optimization problems. *Economics Letters 91(3)*, 312–320.

CGAL (2013). CGAL, Computational Geometry Algorithms Library. http://www.cgal.org.

Christiano, L. J. and J. D. Fisher (2000). Algorithms for solving dynamic models with occasionally binding constraints. *Journal of Economic Dynamics and Control 24(8)*, 1179–1232.

de Berg, M., O. Cheong, M. van Kreveld, and M. Overmars (2008). *Computational Geometry: Algorithms and Applications.* Springer.

Delaunay, B. (1934). Sur la sphère vide. *Bulletion of Academy of Sciences of the USSR 7*, 793–800.

den Haan, W. J. (2001). The importance of the number of different agents in a heterogeneous asset-pricing model. *Journal of Economic Dynamics and Control 25(5)*, 721–746.

Devillers, O., S. Pion, and M. Teillaud (2002). Walking in a triangulation. *International Journal of Foundations of Computer Science 13*(02), 181–199.

Garcia, C. and W. Zangwill (1981). *Pathway to Solutions, Fixed Points and Equilibria.* Prentice Hall, Englewood Cliffs.

Gruene, L. and W. Semmler (2004). Using dynamic programming with adaptive grid scheme for optimal control problems in economics. *Journal of Economic Dynamics and Control 28*, 2427–2456.

Heaton, J. and D. Lucas (1996). Evaluating the effects of incomplete markets on risk sharing and asset pricing. *Journal of Political Economy 104*, 443–487.

Hintermaier, T. and W. Koeniger (2010). The method of endogenous gridpoints with occasionally binding constraints among endogenous variables. *Journal of Economic Dynamics and Control 34*(10), 2074–2088.

Judd, K. (1992). Projection methods for solving aggregate growth models. *Journal of Economic Theory 58*, 410–452.

Judd, K. (1998). *Numerical Methods in Economics*. MIT Press, Cambridge.

Judd, K., F. Kubler, and K. Schmedders (2003). Computational methods for dynamic equilibria with heterogeneous agents. In M. Dewatripont, L. P. Hansen, and S. Turnovsky (Eds.), *Advances in Economics and Econometrics*, pp. 243–290. Cambridge University Press.

Krueger, D. and F. Kubler (2004). Computing equilibrium in olg models with stochastic production. *Journal of Economic Dynamics and Control 28(7)*, 1411–1436.

Kubler, F. and K. Schmedders (2003). Stationary equilibria in asset pricing-models with incomplete markets and collateral. *Econometrica 71*, 1767–1793.

Liu, Y. and J. Snoeyink (2005). A comparison of five implementations of 3d delaunay tessellation. *Combinatorial and Computational Geometry 52*, 439–458.

Ludwig, A. and M. Schön (2013). Endogenous grids in higher dimensions: Hybrid methods and delaunay interpolaton. Mimeo, University of Cologne.

Simpson, R. (1978). Locally equiangular triangulations. *The Computer Journal 21*, 243–245.

Smolyak, S. (1963). Quadrature and interpolation formulas for tensor products of certain classes of functions. *Soviet Mathematics Doklady 4*, 240–243.