Econ 202A Macroeconomics: Section 4

Kiyea Jin

November 12, 14, 2025

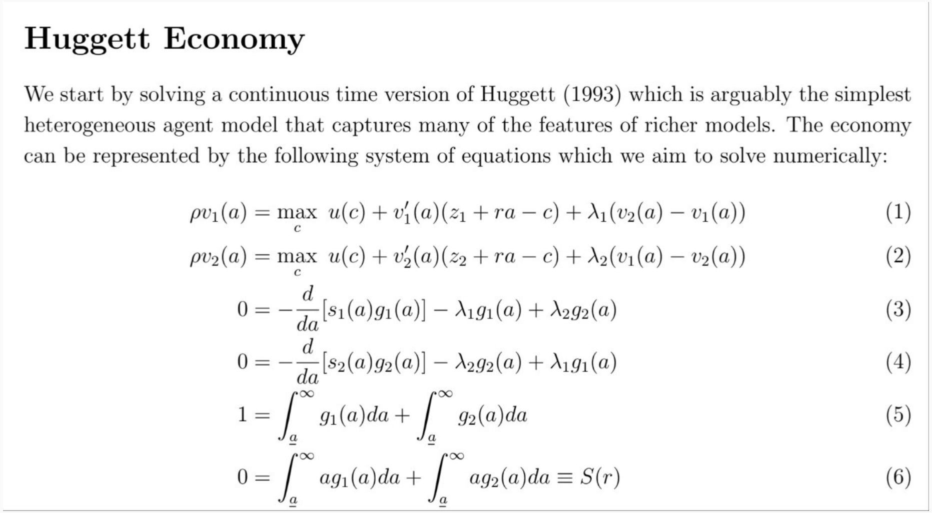
Section 4

Overview

1. Partial Equilibrium of the Huggett Model
   - HJB Equation
   - Kolmogorov Forward (Fokker-Planck) Equation
2. General Equilibrium of the Huggett Model
   - Aggregate Saving
   - Equilibrium Interest Rate
   - Bisection Method

Section 4-1: Partial Equilibrium of the Huggett Model

Huggett Economy: Stationary Equilibrium



Huggett Economy

We start by solving a continuous time version of Huggett (1993) which is arguably the simplest heterogeneous agent model that captures many of the features of richer models. The economy can be represented by the following system of equations which we aim to solve numerically:

$$\rho v_1(a) = \max_c \; u(c) + v_1'(a)(z_1 + ra - c) + \lambda_1(v_2(a) - v_1(a)) \tag{1}$$

$$\rho v_2(a) = \max_c \; u(c) + v_2'(a)(z_2 + ra - c) + \lambda_2(v_1(a) - v_2(a)) \tag{2}$$

$$0 = -\frac{d}{da}[s_1(a)g_1(a)] - \lambda_1 g_1(a) + \lambda_2 g_2(a) \tag{3}$$

$$0 = -\frac{d}{da}[s_2(a)g_2(a)] - \lambda_2 g_2(a) + \lambda_1 g_1(a) \tag{4}$$

$$1 = \int_{\underline{a}}^{\infty} g_1(a)da + \int_{\underline{a}}^{\infty} g_2(a)da \tag{5}$$

$$0 = \int_{\underline{a}}^{\infty} a g_1(a)da + \int_{\underline{a}}^{\infty} a g_2(a)da \equiv S(r) \tag{6}$$

Figure 1: Online Appendix for Achdou et al. (2022)

Implicit Method: Matrix Representation

1. Define I discrete grid points for a, denoted as a_i for $i = 1, \dots, I$, and form an $I \times 1$ vector $\mathbf{a} = [a_1, a_2, \dots, a_I]'$.

2. Let $V_{i,j} = V_j(a_i)$. For each a_i on the grid, make an initial guess for the value function as two $I \times 1$ vectors $\mathbf{V_e^0} = [V^0_{1,e}, V^0_{2,e}, \dots, V^0_{I,e}]'$ and $\mathbf{V_u^0} = [V^0_{1,u}, V^0_{2,u}, \dots, V^0_{I,u}]'$.

3. Construct the stacked $2I \times 1$ vector $\mathbf{V^0} = [\mathbf{V_e^0}, \mathbf{V_u^0}]'$.

4. Construct the $I \times I$ forward and backward difference matrix operators $\mathbf{D_F}$ and $\mathbf{D_B}$ such that $\mathbf{D_F V^n} \simeq (\mathbf{V^n})'_{\mathbf{F}}$ and $\mathbf{D_B V^n} \simeq (\mathbf{V^n})'_{\mathbf{B}}$.

Implicit Method: Matrix Representation

5. Construct a $2I \times 2I$ matrix as follows:

$$\mathbf{A} = \left(\begin{array}{cccc|cccc} -\lambda_e & 0 & \cdots & 0 & \lambda_e & 0 & \cdots & 0 \\ 0 & -\lambda_e & 0 & 0 & 0 & \lambda_e & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots & \ddots & \ddots \\ 0 & 0 & 0 & -\lambda_e & 0 & 0 & 0 & \lambda_e \\ \hline \lambda_u & 0 & \cdots & 0 & -\lambda_u & 0 & \cdots & 0 \\ 0 & \lambda_u & 0 & 0 & 0 & -\lambda_u & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots & \ddots & \ddots \\ 0 & 0 & 0 & \lambda_u & 0 & 0 & 0 & -\lambda_u \end{array}\right) \begin{array}{l} \left.\vphantom{\begin{array}{c}1\\1\\1\\1\end{array}}\right\} I \text{ elements} \\ \left.\vphantom{\begin{array}{c}1\\1\\1\\1\end{array}}\right\} I \text{ elements} \end{array}$$

Implicit Method: Matrix Representation

For iterations $n = 0, 1, 2, \dots$

6. Compute the derivative of the value function as an $I \times 1$ vector using both forward and backward difference matrix operators:

$$(\mathbf{V^n_{e,F}})' = \mathbf{D_F V^n_e} \qquad (\mathbf{V^n_{u,F}})' = \mathbf{D_F V^n_u}$$
$$(\mathbf{V^n_{e,B}})' = \mathbf{D_B V^n_e} \qquad (\mathbf{V^n_{u,B}})' = \mathbf{D_B V^n_u}$$

7. Set the first elements of $\mathbf{V^n_{j,B}}$ as:

$$(V^n_{1,e,B})' = U'(z_e + r\underline{a}) \qquad (V^n_{1,u,B})' = U'(z_u + r\underline{a})$$

and the last elements of $\mathbf{V^n_{j,F}}$ as:

$$(V^n_{I,e,F})' = U'(z_e + r\bar{a}) \qquad (V^n_{I,u,F})' = U'(z_u + r\bar{a})$$

Implicit Method: Matrix Representation

9. Compute the optimal consumption as an $I \times 1$ vector from:

$$\mathbf{c^n_{e,F}} = (U')^{-1}[(\mathbf{V^n_{e,F}})'] \qquad \mathbf{c^n_{u,F}} = (U')^{-1}[(\mathbf{V^n_{u,F}})']$$
$$\mathbf{c^n_{e,B}} = (U')^{-1}[(\mathbf{V^n_{e,B}})'] \qquad \mathbf{c^n_{u,B}} = (U')^{-1}[(\mathbf{V^n_{u,B}})']$$

10. Calculate the optimal savings as an $I \times 1$ vector from:

$$\mathbf{s^n_{e,F}} = z_e + r\mathbf{a} - \mathbf{c^n_{e,F}} \qquad \mathbf{s^n_{u,F}} = z_u + r\mathbf{a} - \mathbf{c^n_{u,F}}$$
$$\mathbf{s^n_{e,B}} = z_e + r\mathbf{a} - \mathbf{c^n_{e,B}} \qquad \mathbf{s^n_{u,B}} = z_u + r\mathbf{a} - \mathbf{c^n_{u,B}}$$

11. Create indicator vectors:

$$\mathbf{I^n_{e,F}} = [I^n_{1,e,F}, I^n_{2,e,F}, \cdots, I^n_{I,e,F}]' \qquad \mathbf{I^n_{u,F}} = [I^n_{1,u,F}, I^n_{2,u,F}, \cdots, I^n_{I,u,F}]'$$
$$\mathbf{I^n_{e,B}} = [I^n_{1,e,B}, I^n_{2,e,B}, \cdots, I^n_{I,e,B}]' \qquad \mathbf{I^n_{u,B}} = [I^n_{1,u,B}, I^n_{2,u,B}, \cdots, I^n_{I,u,B}]'$$

where $I^n_{i,j,F} = 1$ if $s^n_{i,j,F} > 0$ and $I^n_{i,j,B} = 1$ if $s^n_{i,j,B} < 0$ for $i = 1, \cdots, I$ and $j = e, u$.

Implicit Method: Matrix Representation

12. Compute optimal consumption as follows:

$$\mathbf{c}_{\mathbf{e}}^{\mathbf{n}} = \mathbf{I}_{\mathbf{e},\mathbf{F}}^{\mathbf{n}} \cdot \mathbf{c}_{\mathbf{e},\mathbf{F}}^{\mathbf{n}} + \mathbf{I}_{\mathbf{e},\mathbf{B}}^{\mathbf{n}} \cdot \mathbf{c}_{\mathbf{e},\mathbf{B}}^{\mathbf{n}} + (1 - \mathbf{I}_{\mathbf{e},\mathbf{F}}^{\mathbf{n}} - \mathbf{I}_{\mathbf{e},\mathbf{B}}^{\mathbf{n}}) \cdot \mathbf{c}_{\mathbf{e},\mathbf{0}}^{\mathbf{n}}$$
$$\mathbf{c}_{\mathbf{u}}^{\mathbf{n}} = \mathbf{I}_{\mathbf{u},\mathbf{F}}^{\mathbf{n}} \cdot \mathbf{c}_{\mathbf{u},\mathbf{F}}^{\mathbf{n}} + \mathbf{I}_{\mathbf{u},\mathbf{B}}^{\mathbf{n}} \cdot \mathbf{c}_{\mathbf{u},\mathbf{B}}^{\mathbf{n}} + (1 - \mathbf{I}_{\mathbf{u},\mathbf{F}}^{\mathbf{n}} - \mathbf{I}_{\mathbf{u},\mathbf{B}}^{\mathbf{n}}) \cdot \mathbf{c}_{\mathbf{u},\mathbf{0}}^{\mathbf{n}}$$

13. Compute optimal savings as follows:

$$\mathbf{s}_{\mathbf{e}}^{\mathbf{n}} = \mathbf{I}_{\mathbf{e},\mathbf{F}}^{\mathbf{n}} \cdot \mathbf{s}_{\mathbf{e},\mathbf{F}}^{\mathbf{n}} + \mathbf{I}_{\mathbf{e},\mathbf{B}}^{\mathbf{n}} \cdot \mathbf{s}_{\mathbf{e},\mathbf{B}}^{\mathbf{n}}$$
$$\mathbf{s}_{\mathbf{u}}^{\mathbf{n}} = \mathbf{I}_{\mathbf{u},\mathbf{F}}^{\mathbf{n}} \cdot \mathbf{s}_{\mathbf{u},\mathbf{F}}^{\mathbf{n}} + \mathbf{I}_{\mathbf{u},\mathbf{B}}^{\mathbf{n}} \cdot \mathbf{s}_{\mathbf{u},\mathbf{B}}^{\mathbf{n}}$$

14. Construct two $I \times I$ diagonal matrices as follows:

$$\mathbf{S}_{\mathbf{e}}^{\mathbf{n}}\mathbf{D}_{\mathbf{e}}^{\mathbf{n}} = \mathrm{diag}(\mathbf{I}_{\mathbf{e},\mathbf{F}}^{\mathbf{n}} \cdot \mathbf{s}_{\mathbf{e},\mathbf{F}}^{\mathbf{n}})\,\mathbf{D}_{\mathbf{F}} + \mathrm{diag}(\mathbf{I}_{\mathbf{e},\mathbf{B}}^{\mathbf{n}} \cdot \mathbf{s}_{\mathbf{e},\mathbf{B}}^{\mathbf{n}})\,\mathbf{D}_{\mathbf{B}}$$
$$\mathbf{S}_{\mathbf{u}}^{\mathbf{n}}\mathbf{D}_{\mathbf{u}}^{\mathbf{n}} = \mathrm{diag}(\mathbf{I}_{\mathbf{u},\mathbf{F}}^{\mathbf{n}} \cdot \mathbf{s}_{\mathbf{u},\mathbf{F}}^{\mathbf{n}})\,\mathbf{D}_{\mathbf{F}} + \mathrm{diag}(\mathbf{I}_{\mathbf{u},\mathbf{B}}^{\mathbf{n}} \cdot \mathbf{s}_{\mathbf{u},\mathbf{B}}^{\mathbf{n}})\,\mathbf{D}_{\mathbf{B}}$$

Implicit Method: Matrix Representation

15. Construct a $2I \times 2I$ matrix $\mathbf{S^nD^n}$ as follows:

$$\mathbf{S^nD^n} = \begin{pmatrix} \mathbf{S^n_eD^n_e} & 0 \\ 0 & \mathbf{S^n_uD^n_u} \end{pmatrix} \tag{1}$$

16. Construct the matrix $\mathbf{P^n}$ as $\mathbf{P^n} = \mathbf{S^nD^n} + \mathbf{A}$

17. Find $\mathbf{V^{n+1}}$ from:

$$\frac{1}{\Delta}(\mathbf{V^{n+1}} - \mathbf{V^n}) + \rho\mathbf{V^{n+1}} = U(\mathbf{c^n}) + \mathbf{P^nV^{n+1}} \tag{2}$$

where $\mathbf{c^n} = [\mathbf{c^n_e}, \mathbf{c^n_u}]'$ is the stacked $2I \times 1$ vector.

18. If $\mathbf{V^{n+1}}$ is close enough to $\mathbf{V^n}$: stop. Otherwise, go to step 6.

Implicit Method: Matrix Representation

Alternatively, solve the linear system:

$$\mathbf{V^{n+1}} = \left((\rho + \frac{1}{\Delta})\mathbf{I} - \mathbf{P^n}\right)^{-1} \left[U(\mathbf{c^n}) + \frac{1}{\Delta}\mathbf{V^n}\right] \tag{3}$$

Optimal Consumption

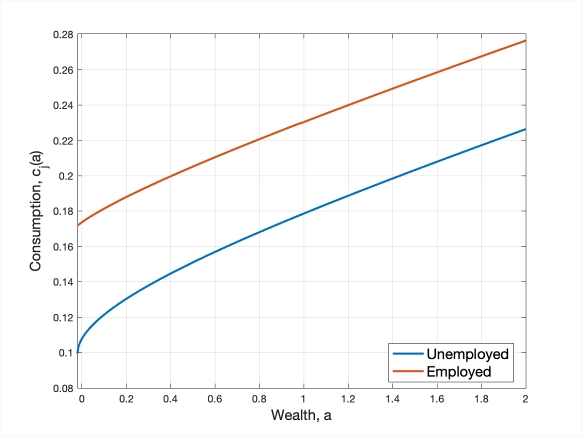


Figure 2: Optimal Consumption Policy for Employed and Unemployed States

Optimal Savings

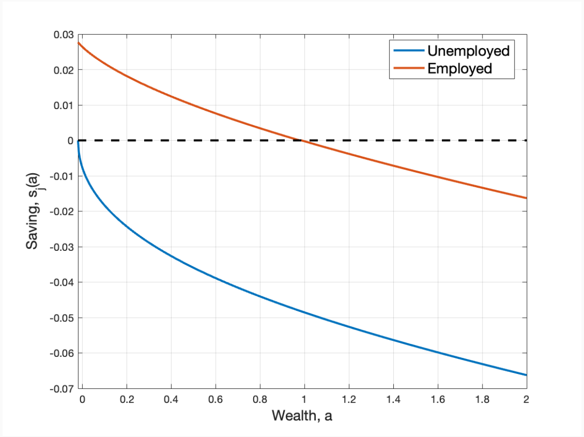
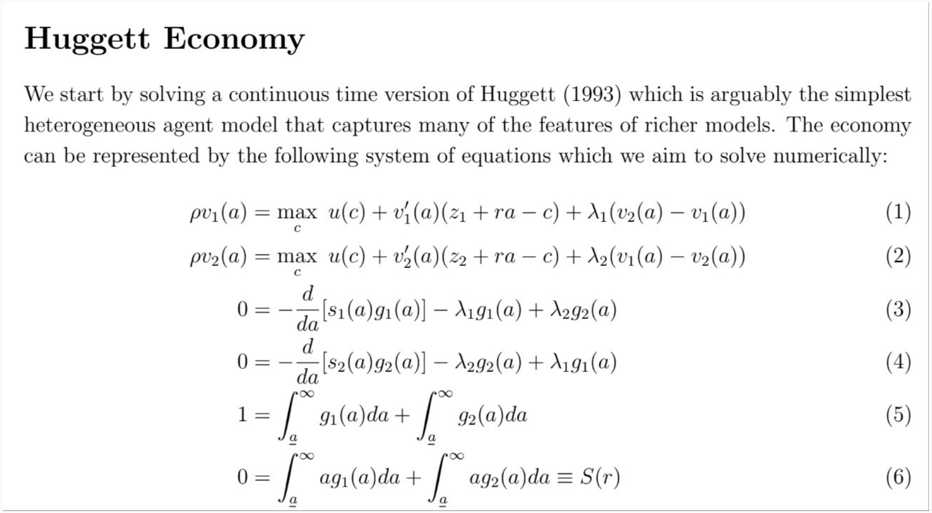


Figure 3: Optimal Savings Policy for Employed and Unemployed States

Huggett Economy: Stationary Equilibrium



Huggett Economy

We start by solving a continuous time version of Huggett (1993) which is arguably the simplest heterogeneous agent model that captures many of the features of richer models. The economy can be represented by the following system of equations which we aim to solve numerically:

$$\rho v_1(a) = \max_c \ u(c) + v_1'(a)(z_1 + ra - c) + \lambda_1(v_2(a) - v_1(a)) \quad (1)$$

$$\rho v_2(a) = \max_c \ u(c) + v_2'(a)(z_2 + ra - c) + \lambda_2(v_1(a) - v_2(a)) \quad (2)$$

$$0 = -\frac{d}{da}[s_1(a)g_1(a)] - \lambda_1 g_1(a) + \lambda_2 g_2(a) \quad (3)$$

$$0 = -\frac{d}{da}[s_2(a)g_2(a)] - \lambda_2 g_2(a) + \lambda_1 g_1(a) \quad (4)$$

$$1 = \int_{\underline{a}}^{\infty} g_1(a)da + \int_{\underline{a}}^{\infty} g_2(a)da \quad (5)$$

$$0 = \int_{\underline{a}}^{\infty} a g_1(a)da + \int_{\underline{a}}^{\infty} a g_2(a)da \equiv S(r) \quad (6)$$

Figure 4: Online Appendix for Achdou et al. (2022)

Consumption and Savings Policy Function

The Hamilton-Jacobi-Bellman (HJB) equation determines **optimal individual choices.**

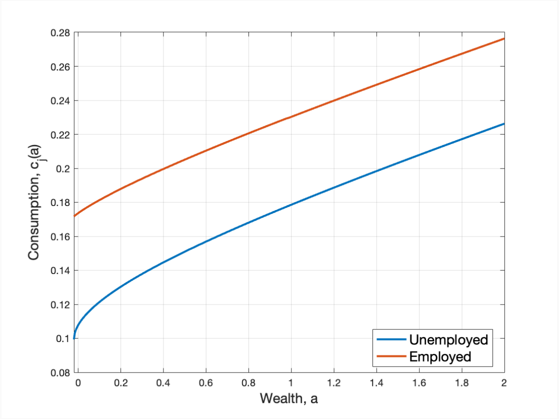


Figure 5: Consumption Policy Function

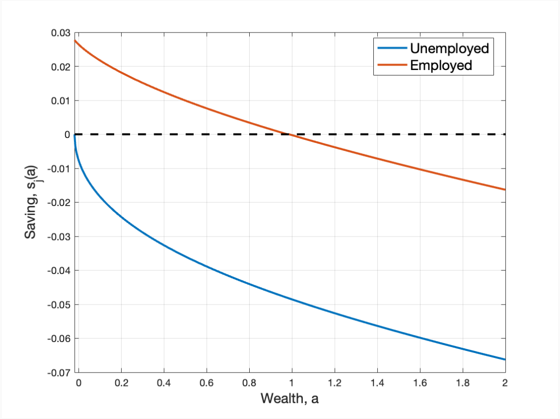


Figure 6: Saving Policy Function

Stationary Distribution for Wealth

The next question is: how does the endogenous stationary **distribution of wealth** evolve? In this context, what fraction of agents holds specific levels of assets and income once all transitions between states have stabilized?

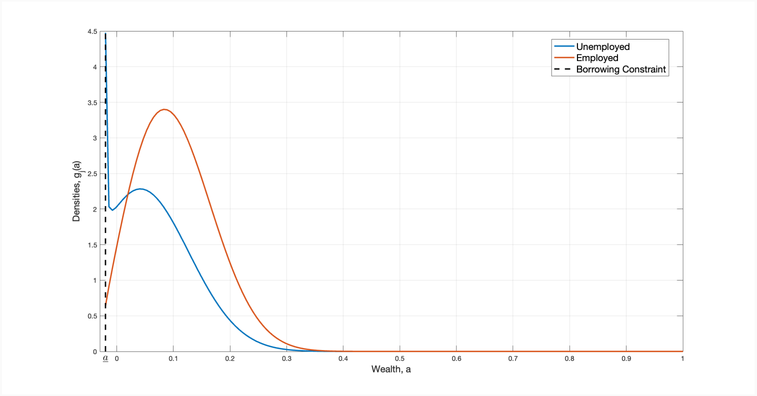


Figure 7: Implied Wealth Distribution

Kolmogorov Forward (Fokker-Planck) Equation

- The Kolmogorov Forward (KF) equation, also known as the Fokker-Planck Equation, describes how the endogenous distribution of wealth evolves over time.

Kolmogorov Forward (Fokker-Planck) Equation

- The Kolmogorov Forward (KF) equation, also known as the Fokker-Planck Equation, describes how the endogenous distribution of wealth evolves over time.
- The KF Equation is a partial differential equation (PDE) that describes the **time evolution of the probability density function (PDF) of the state** of a stochastic process.

Kolmogorov Forward (Fokker-Planck) Equation

- The Kolmogorov Forward (KF) equation, also known as the Fokker-Planck Equation, describes how the endogenous distribution of wealth evolves over time.
- The KF Equation is a partial differential equation (PDE) that describes the **time evolution of the probability density function (PDF) of the state** of a stochastic process.
- There is a tight link between solving the HJB and KF equations.

Derivation of Kolmogorov Forward Equation

Let $G_e(a,t)$ and $G_u(a,t)$ denote the cumulative distribution functions (CDFs) of employed and unemployed workers, respectively, representing the fraction of employed or unemployed agents with $a_t \leq a$ at time t. This must satisfy:

$$\begin{aligned} G_e(a,t+1) &= (1-\lambda_e)G_e(a^e_{-1},t) + \lambda_u G_u(a^u_{-1},t) \\ G_u(a,t+1) &= (1-\lambda_u)G_u(a^u_{-1},t) + \lambda_e G_e(a^e_{-1},t) \end{aligned} \tag{4}$$

where a^j_{-1} represents the level of assets in the previous period that optimally sets $a_{t+1} = a$ in employment status $j \in \{e,u\}$ in the previous period.

Discrete to Continuous-Time Transformation

Over a time interval of Δ units, the model can be expressed as:

$$\begin{aligned} G_j(a, t+\Delta) &= (1-\Delta\lambda_j)G_j({a^j}_{-\Delta}, t) + \Delta\lambda_{-j}G_{-j}({a^u}_{-\Delta}, t) \\ &= (1-\Delta\lambda_j)G_j(a-\Delta s_j, t) + \Delta\lambda_{-j}G_{-j}(a-\Delta s_{-j}, t) \end{aligned}$$

where $j \in \{e, u\}$ represents employment states.

Discrete to Continuous-Time Transformation

Subtracting $G_j(a,t)$ from both sides and dividing both sides by Δ gives:

$$\begin{aligned}\frac{G_e(a,t+\Delta)-G_j(a,t)}{\Delta} =& \frac{G_j(a-\Delta s_j,t)-G_j(a,t)}{\Delta} \\ & -\lambda_j G_j(a-\Delta s_j,t)+\lambda_{-j}G_{-j}(a-\Delta s_{-j},t)\end{aligned}$$

Taking the limit as $\Delta \to 0$, we obtain:

$$\begin{aligned}\dot{G}_j(a,t)=\frac{\partial G_j(a,t)}{\partial t} &= -\frac{\partial G_j(a,t)}{\partial a}s_j-\lambda_j G_j(a,t)+\lambda_{-j}G_{-j}(a,t) \\ &= -g_j(a,t)s_j-\lambda_j G_j(a,t)+\lambda_{-j}G_{-j}(a,t)\end{aligned}$$

Law of Motion for the Endogenous Distribution

To derive the law of motion for the probability density function (PDF), we differentiate with respect to a, yielding:

$$\dot{g}_j(a,t) = -\frac{\partial}{\partial a}\left[g_j(a,t)s_j(a,t)\right] - \lambda_j g_j(a,t) + \lambda_{-j} g_{-j}(a,t)$$

where $g_j(a,t) = \partial G_j(a,t)/\partial a$ denotes the probability density function.

Kolmogorov Forward Equation

These equations (5) are known as the Kolmogorov Forward equations:

$$\begin{aligned} \dot{g}_e(a,t) &= -\frac{\partial}{\partial a}\left[g_e(a,t)s_e(a,t)\right] - \lambda_e g_e(a,t) + \lambda_u g_u(a,t) \\ \dot{g}_u(a,t) &= -\frac{\partial}{\partial a}\left[g_u(a,t)s_u(a,t)\right] - \lambda_u g_u(a,t) + \lambda_e g_e(a,t) \end{aligned} \tag{5}$$

At each t, the densities integrate to one across asset holdings:

$$1 = \int_{\underline{a}}^{\infty} g_e(a,t)\,da + \int_{\underline{a}}^{\infty} g_u(a,t)\,da. \tag{6}$$

Kolmogorov Forward Equation

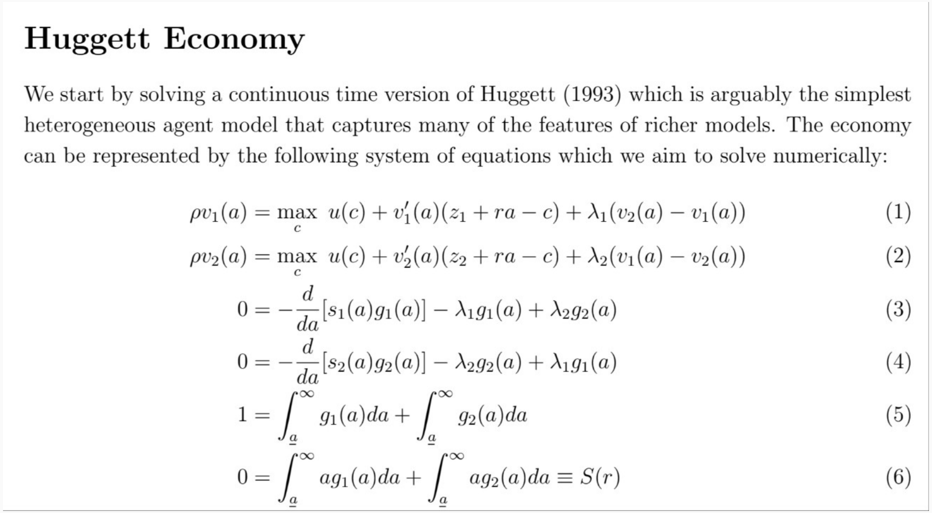
In a stationary equilibrium, aggregate objects are time-invariant, so t is no longer a state variable, and the time derivative is zero. Therefore, the stationary distribution must satisfy the following equations:

$$\begin{aligned} 0 = \dot{g}_e(a) &= -\frac{d}{da}\left[g_e(a)s_e(a)\right] - \lambda_e g_e(a) + \lambda_u g_u(a) \\ 0 = \dot{g}_u(a) &= -\frac{d}{da}\left[g_u(a)s_u(a)\right] - \lambda_u g_u(a) + \lambda_e g_e(a) \end{aligned} \tag{7}$$

The stationary densities satisfy the same normalization:

$$1 = \int_{\underline{a}}^{\infty} g_e(a)\, da + \int_{\underline{a}}^{\infty} g_u(a)\, da \tag{8}$$

Huggett Economy: Stationary Equilibrium



Huggett Economy

We start by solving a continuous time version of Huggett (1993) which is arguably the simplest heterogeneous agent model that captures many of the features of richer models. The economy can be represented by the following system of equations which we aim to solve numerically:

$$\rho v_1(a) = \max_c \; u(c) + v_1'(a)(z_1 + ra - c) + \lambda_1(v_2(a) - v_1(a)) \tag{1}$$

$$\rho v_2(a) = \max_c \; u(c) + v_2'(a)(z_2 + ra - c) + \lambda_2(v_1(a) - v_2(a)) \tag{2}$$

$$0 = -\frac{d}{da}[s_1(a)g_1(a)] - \lambda_1 g_1(a) + \lambda_2 g_2(a) \tag{3}$$

$$0 = -\frac{d}{da}[s_2(a)g_2(a)] - \lambda_2 g_2(a) + \lambda_1 g_1(a) \tag{4}$$

$$1 = \int_{\underline{a}}^{\infty} g_1(a)da + \int_{\underline{a}}^{\infty} g_2(a)da \tag{5}$$

$$0 = \int_{\underline{a}}^{\infty} a g_1(a)da + \int_{\underline{a}}^{\infty} a g_2(a)da \equiv S(r) \tag{6}$$

Figure 8: Online Appendix for Achdou et al. (2022)

Finite Difference Approximations

The finite difference approximations to KF equation (7), associated with (8) is:

$$\begin{aligned} 0 &= -(g_{i,e}s_{i,e})' - \lambda_e g_{i,e} + \lambda_u g_{i,u} \\ 0 &= -(g_{i,u}s_{i,u})' - \lambda_u g_{i,u} + \lambda_e g_{i,e} \end{aligned} \tag{9}$$

$$1 = \Sigma_{i=1}^{I} g_{i,e}\Delta a + \Sigma_{i=1}^{I} g_{i,u}\Delta a \tag{10}$$

where $i = 1, \cdots, I$ and $\Delta a = a_{i+1} - a_i$ is the distance between equispaced grid points.

Finite Difference Approximations

The finite difference approximations to KF equation (7), associated with (8) is:

$$\begin{aligned} 0 &= -(g_{i,e}s_{i,e})' - \lambda_e g_{i,e} + \lambda_u g_{i,u} \\ 0 &= -(g_{i,u}s_{i,u})' - \lambda_u g_{i,u} + \lambda_e g_{i,e} \end{aligned} \tag{9}$$

$$1 = \Sigma_{i=1}^{I} g_{i,e}\Delta a + \Sigma_{i=1}^{I} g_{i,u}\Delta a \tag{10}$$

where $i = 1, \cdots, I$ and $\Delta a = a_{i+1} - a_i$ is the distance between equispaced grid points.

Because equations (7) and (8) are linear in g_e and g_u, so is their finite difference approximation. As a result, **no iterative procedure like the one used for solving the HJB equation is necessary**, and the equation can be solved in a single step.

Upwind Scheme

There is again a question when to use a forward and a backward approximation for the derivative $(g_{i,j}s_{i,j})'$.

Upwind Scheme

There is again a question when to use a forward and a backward approximation for the derivative $(g_{i,j}s_{i,j})'$.

It turns out that the most *convenient* and correct approximation is by upwind scheme:

$$-\frac{g_{i,j}(s^n_{i,j,F})^+ - g_{i-1,j}(s^n_{i-1,j,F})^+}{\Delta a} - \frac{g_{i+1,j}(s^n_{i+1,j,B})^- - g_{i,j}(s^n_{i,j,B})^-}{\Delta a} - g_{i,j}\lambda_j + g_{i,-j}\lambda_{-j} = 0$$

Upwind Scheme

There is again a question when to use a forward and a backward approximation for the derivative $(g_{i,j}s_{i,j})'$.

It turns out that the most *convenient* and correct approximation is by upwind scheme:

$$-\frac{g_{i,j}(s^n_{i,j,F})^+ - g_{i-1,j}(s^n_{i-1,j,F})^+}{\Delta a} - \frac{g_{i+1,j}(s^n_{i+1,j,B})^- - g_{i,j}(s^n_{i,j,B})^-}{\Delta a} - g_{i,j}\lambda_j + g_{i,-j}\lambda_{-j} = 0$$

$*$ Note that we define the "upwind scheme" differently here: we use a backward difference when the drift is positive and a forward difference when the drift is negative. This distinction arises because the HJB and KF equations evolve in opposite directions in time. The HJB equation is solved backward in time, whereas the Kolmogorov Forward equation is solved forward in time, describing how the probability density evolves from an initial distribution. Consequently, the numerical direction of information flow—and thus the appropriate finite-difference orientation—is reversed. Because $g_{0,j}$ and $g_{I+1,j}$ are outside the state space, the density at these points is zero and so $(s_{0,j,F})^+$ and $(s_{I+1,j,B})^-$ are never used.

Upwind Scheme

Collecting terms, we can write

$$g_{i-1,j} z_{i-1,j} + g_{i,j} y_{i,j} + g_{i+1,j} x_{i+1,j} + g_{i,-j} \lambda_{-j} = 0$$

where

$$x_{i+1,j} = -\frac{\left(s^n_{i+1,j,B}\right)^-}{\Delta a}$$
$$y_{i,j} = -\frac{\left(s^n_{i,j,F}\right)^+}{\Delta a} + \frac{\left(s^n_{i,j,B}\right)^-}{\Delta a} - \lambda_j$$
$$z_{i-1,j} = \frac{\left(s^n_{i-1,j,F}\right)^+}{\Delta a}$$

Upwind Scheme

$$\mathbf{P}^{\mathsf{T}} = \left(\begin{array}{ccccc|ccccc}
y_{1,1} & x_{2,1} & 0 & \dots & 0 & \lambda_2 & 0 & 0 & \dots & 0 \\
z_{1,1} & y_{2,1} & x_{3,1} & \dots & 0 & 0 & \lambda_2 & 0 & \dots & 0 \\
0 & z_{2,1} & y_{3,1} & \dots & 0 & 0 & 0 & \lambda_2 & \dots & 0 \\
\vdots & \ddots & \ddots & \ddots & x_{I,1} & \vdots & \ddots & \ddots & \ddots & 0 \\
0 & \dots & 0 & z_{I-1,1} & y_{I,1} & 0 & \dots & 0 & 0 & \lambda_2 \\
\hline
\lambda_1 & 0 & 0 & \dots & 0 & y_{1,2} & x_{2,2} & 0 & \dots & 0 \\
0 & \lambda_1 & 0 & \dots & 0 & z_{1,2} & y_{2,2} & x_{3,2} & \dots & 0 \\
0 & 0 & \lambda_1 & \dots & 0 & 0 & z_{2,2} & y_{3,2} & \dots & 0 \\
\vdots & \ddots & \ddots & \ddots & 0 & \vdots & \ddots & \ddots & \ddots & x_{I,2} \\
0 & \dots & 0 & 0 & \lambda_1 & 0 & \dots & 0 & z_{I-1,2} & y_{I,2}
\end{array}\right)
\begin{array}{l}
\left.\vphantom{\begin{array}{c}1\\1\\1\\1\\1\end{array}}\right\} I \text{ elements} \\
\left.\vphantom{\begin{array}{c}1\\1\\1\\1\\1\end{array}}\right\} I \text{ elements}
\end{array}$$

Solving KF Equation

It turns out that equations (9) can be written in matrix form in a way closely related to the approximation used for solving the HJB equations.

$$0 = \dot{\mathbf{g}} = \mathbf{P^T g} = 0 \tag{11}$$

where $\mathbf{P^T}$ is the transpose of the intensity matrix $\mathbf{P}$ from the HJB equation ($\mathbf{P^n}$ from the final HJB iteration), and $\mathbf{g}$ is the $2I \times 1$ stacked vector $[\mathbf{g_e}, \mathbf{g_u}]'$.

$\mathbf{P}^\top$ is the discretized version of the *adjoint*—the infinite-dimensional analogue of a matrix transpose—of the HJB operator, the "Kolmogorov Forward operator."

Solving KF Equation

This approximation is *convenient* because, after constructing the matrix $\mathbf{P}$ for solving the HJB equation using an implicit method, almost no additional work is required.

Solving KF Equation

This approximation is *convenient* because, after constructing the matrix $\mathbf{P}$ for solving the HJB equation using an implicit method, almost no additional work is required.

The remaining task is to solve the *eigenvalue problem* $\mathbf{P^T g} = 0$, where $\mathbf{g}$ is an eigenvector of the matrix $\mathbf{P^T}$ corresponding to the eigenvalue 0, and is normalized to sum to one.

Solving KF Equation

This approximation is *convenient* because, after constructing the matrix $\mathbf{P}$ for solving the HJB equation using an implicit method, almost no additional work is required.

The remaining task is to solve the *eigenvalue problem* $\mathbf{P}^\mathbf{T}\mathbf{g} = 0$, where $\mathbf{g}$ is an eigenvector of the matrix $\mathbf{P}^\mathbf{T}$ corresponding to the eigenvalue 0, and is normalized to sum to one.

This makes sense because the matrix $\mathbf{P}$ captures the evolution of the stochastic process, and to find the stationary distribution, one solves the eigenvalue problem $\mathbf{P}^\mathbf{T}\mathbf{g} = 0$.

Stationary Distribution for Employment Status

In a somewhat similar but simpler setup, suppose we want to determine the long-run equilibrium employment rate when the dynamics of aggregate employment and unemployment in discrete time are given by:

$$\begin{aligned} e_{t+1} &= (1-\lambda_e)e_t + \lambda_u u_t \\ u_{t+1} &= \lambda_e e_t + (1-\lambda_u)u_t \end{aligned}$$

Stationary Distribution for Employment Status

In a somewhat similar but simpler setup, suppose we want to determine the long-run equilibrium employment rate when the dynamics of aggregate employment and unemployment in discrete time are given by:

$$e_{t+1} = (1 - \lambda_e)e_t + \lambda_u u_t$$
$$u_{t+1} = \lambda_e e_t + (1 - \lambda_u)u_t$$

In Δ units of time, we have:

$$e_{t+\Delta} = (1 - \Delta\lambda_e)e_t + \Delta\lambda_u u_t$$
$$u_{t+\Delta} = \Delta\lambda_e e_t + (1 - \Delta\lambda_u)u_t$$

Stationary Distribution for Employment Status

In a somewhat similar but simpler setup, suppose we want to determine the long-run equilibrium employment rate when the dynamics of aggregate employment and unemployment in discrete time are given by:

$$e_{t+1} = (1 - \lambda_e)e_t + \lambda_u u_t$$
$$u_{t+1} = \lambda_e e_t + (1 - \lambda_u)u_t$$

In Δ units of time, we have:

$$e_{t+\Delta} = (1 - \Delta\lambda_e)e_t + \Delta\lambda_u u_t$$
$$u_{t+\Delta} = \Delta\lambda_e e_t + (1 - \Delta\lambda_u)u_t$$

Rearranging and taking limits, we obtain the continuous-time dynamics:

$$\dot{e}_t = -\lambda_e e_t + \lambda_u u_t$$
$$\dot{u}_t = \lambda_e e_t - \lambda_u u_t$$

Stationary Distribution for Employment Status

The system can be expressed as:

$$\dot{\mathbf{s}}_{\mathbf{t}} = \mathbf{T}\mathbf{s}_{\mathbf{t}}$$

where

$$\mathbf{T} = \begin{pmatrix} -\lambda_e & \lambda_u \\ \lambda_e & -\lambda_u \end{pmatrix}$$

Stationary Distribution for Employment Status

The system can be expressed as:

$$\dot{\mathbf{s}}_\mathbf{t} = \mathbf{T}\mathbf{s}_\mathbf{t}$$

where

$$\mathbf{T} = \begin{pmatrix} -\lambda_e & \lambda_u \\ \lambda_e & -\lambda_u \end{pmatrix}$$

In the stationary equilibrium, the rates do not change over time, so we have:

$$0 = \dot{\mathbf{s}} = \mathbf{T}\mathbf{s}$$

Stationary Distribution for Employment Status

The system can be expressed as:

$$\dot{\mathbf{s}}_{\mathbf{t}} = \mathbf{T}\mathbf{s}_{\mathbf{t}}$$

where

$$\mathbf{T} = \begin{pmatrix} -\lambda_e & \lambda_u \\ \lambda_e & -\lambda_u \end{pmatrix}$$

In the stationary equilibrium, the rates do not change over time, so we have:

$$0 = \dot{\mathbf{s}} = \mathbf{T}\mathbf{s}$$

Therefore, **s** is an eigenvector associated with a zero eigenvalue, normalized to sum to one.

Stationary Distribution for Employment Status

The system can be expressed as:

$$\dot{\mathbf{s}}_{\mathbf{t}} = \mathbf{T}\mathbf{s}_{\mathbf{t}}$$

where

$$\mathbf{T} = \begin{pmatrix} -\lambda_e & \lambda_u \\ \lambda_e & -\lambda_u \end{pmatrix}$$

In the stationary equilibrium, the rates do not change over time, so we have:

$$0 = \dot{\mathbf{s}} = \mathbf{T}\mathbf{s}$$

Therefore, **s** is an eigenvector associated with a zero eigenvalue, normalized to sum to one.

In this context, the first element of **s** represents the stationary employment rate, and the second element represents the stationary unemployment rate.

Implementation in MATLAB

Since the eigenvector is only defined up to a scalar, we fix one element of the eigenvector to a specific value (e.g., 0.1). This step is essential to avoid having a singular matrix $\mathbf{T}$, which cannot be inverted otherwise.

Follow these steps:

1. Define the vector

$$\mathbf{b} = \begin{pmatrix} 0.1 \\ 0 \end{pmatrix}$$

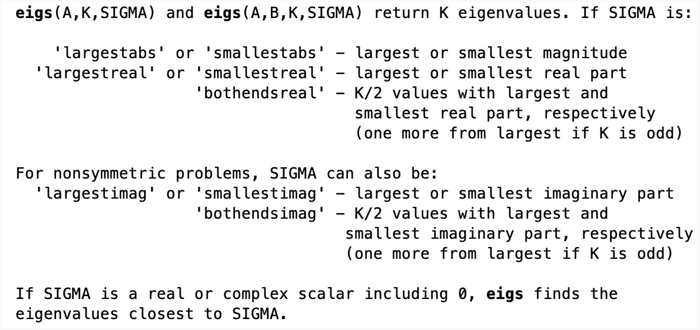
   and modify the matrix as

$$\hat{\mathbf{T}} = \begin{pmatrix} 1 & 0 \\ \lambda_e & -\lambda_u \end{pmatrix}.$$

2. Compute $\hat{\mathbf{s}}$ by solving $\hat{\mathbf{s}} = \hat{\mathbf{T}}^{-1}\mathbf{b}$.
3. Normalize $\hat{\mathbf{s}}$ so that the elements sum to one to find $\mathbf{s}$.

Implementation in MATLAB

Use MATLAB's `eigs` function to find the eigenvector associated with the eigenvalue closest to zero.



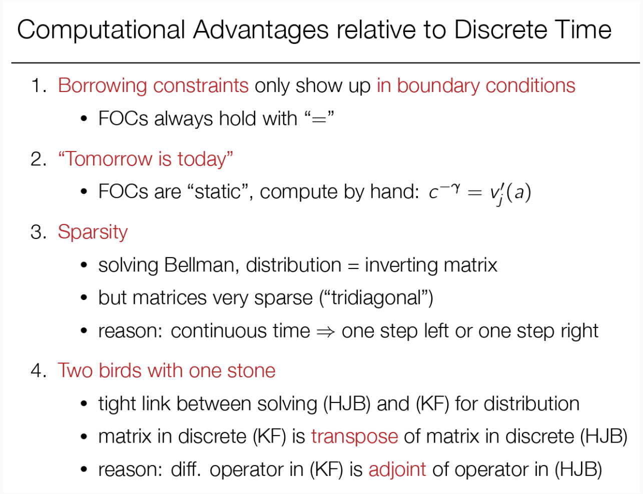
```
eigs(A,K,SIGMA) and eigs(A,B,K,SIGMA) return K eigenvalues. If SIGMA is:

    'largestabs' or 'smallestabs' - largest or smallest magnitude
  'largestreal' or 'smallestreal' - largest or smallest real part
                   'bothendsreal' - K/2 values with largest and
                                    smallest real part, respectively
                                    (one more from largest if K is odd)

For nonsymmetric problems, SIGMA can also be:
  'largestimag' or 'smallestimag' - largest or smallest imaginary part
                   'bothendsimag' - K/2 values with largest and
                                   smallest imaginary part, respectively
                                   (one more from largest if K is odd)

If SIGMA is a real or complex scalar including 0, eigs finds the
eigenvalues closest to SIGMA.
```

Final Comments



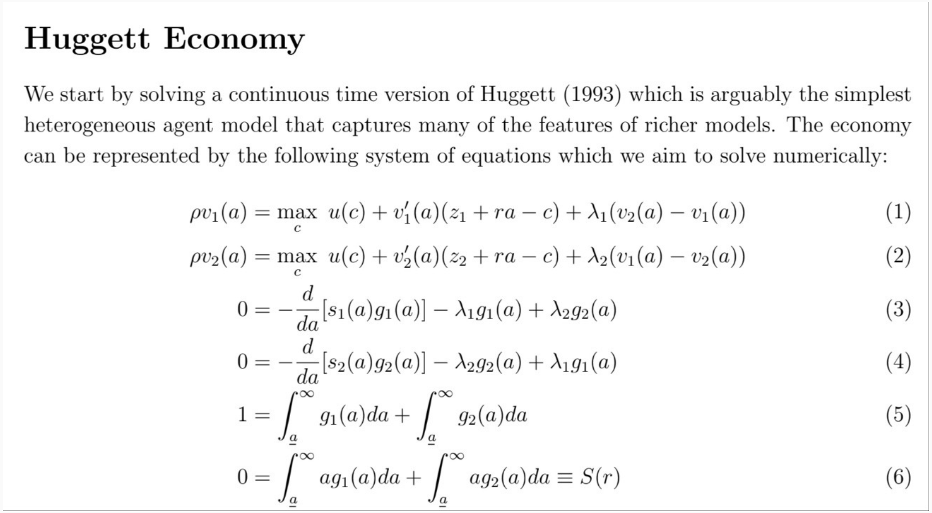
Computational Advantages relative to Discrete Time

1. Borrowing constraints only show up in boundary conditions
   - FOCs always hold with "="
2. "Tomorrow is today"
   - FOCs are "static", compute by hand: $c^{-\gamma} = v_j'(a)$
3. Sparsity
   - solving Bellman, distribution = inverting matrix
   - but matrices very sparse ("tridiagonal")
   - reason: continuous time $\Rightarrow$ one step left or one step right
4. Two birds with one stone
   - tight link between solving (HJB) and (KF) for distribution
   - matrix in discrete (KF) is transpose of matrix in discrete (HJB)
   - reason: diff. operator in (KF) is adjoint of operator in (HJB)

Figure 9: Computational Advantages Relative to Discrete Time (Achdou et al., 2022)

Section 4-2: General Equilibrium of the Huggett Model

Huggett Economy: Stationary Equilibrium



Huggett Economy

We start by solving a continuous time version of Huggett (1993) which is arguably the simplest heterogeneous agent model that captures many of the features of richer models. The economy can be represented by the following system of equations which we aim to solve numerically:

$$\rho v_1(a) = \max_c \ u(c) + v_1'(a)(z_1 + ra - c) + \lambda_1(v_2(a) - v_1(a)) \tag{1}$$

$$\rho v_2(a) = \max_c \ u(c) + v_2'(a)(z_2 + ra - c) + \lambda_2(v_1(a) - v_2(a)) \tag{2}$$

$$0 = -\frac{d}{da}[s_1(a)g_1(a)] - \lambda_1 g_1(a) + \lambda_2 g_2(a) \tag{3}$$

$$0 = -\frac{d}{da}[s_2(a)g_2(a)] - \lambda_2 g_2(a) + \lambda_1 g_1(a) \tag{4}$$

$$1 = \int_{\underline{a}}^{\infty} g_1(a)da + \int_{\underline{a}}^{\infty} g_2(a)da \tag{5}$$

$$0 = \int_{\underline{a}}^{\infty} a g_1(a)da + \int_{\underline{a}}^{\infty} a g_2(a)da \equiv S(r) \tag{6}$$

Figure 10: Online Appendix for Achdou et al. (2022)

Aggregate Saving

After having solved HJB equations and KF equations, the aggregate saving function $S(r)$ defined in the following equation can be easily computed. It can be approximated as:

$$S(r) \simeq \sum_{i=1}^{I} a_i g_{i,e} \Delta a + \sum_{i=1}^{I} a_i g_{i,u} \Delta a \tag{12}$$

Equilibrium in the Bond Market

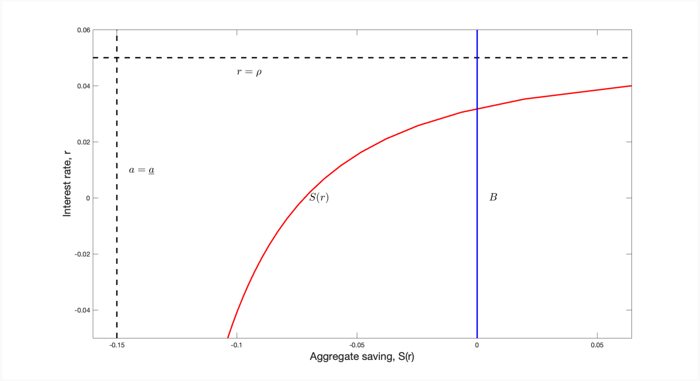
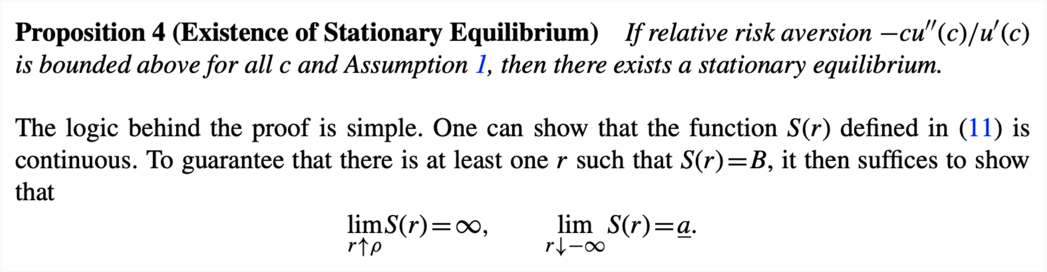


Figure 11: Equilibrium in the Bond Market

Existence of Stationary Equilibrium



Proposition 4 (Existence of Stationary Equilibrium) *If relative risk aversion $-cu''(c)/u'(c)$ is bounded above for all c and Assumption 1, then there exists a stationary equilibrium.*

The logic behind the proof is simple. One can show that the function $S(r)$ defined in (11) is continuous. To guarantee that there is at least one r such that $S(r) = B$, it then suffices to show that

$$\lim_{r \uparrow \rho} S(r) = \infty, \qquad \lim_{r \downarrow -\infty} S(r) = \underline{a}.$$

Figure 12: Existence of Stationary Equilibrium (Achdou et al., 2022)

Finding the Equilibrium Interest Rate

The equilibrium interest rate, where aggregate saving (asset demand) $S(r)$ equals zero, can be found using either a **bisection metho**d or Newton's method.

Bisection Method

- The bisection method is a numerical approach for finding roots, effective for functions that change sign over an interval.
- The obvious idea is to increase r whenever $S(r) < 0$ and decrease r whenever $S(r) > 0$.
- To find the interest rate r that sets $S(r) = 0$:
  1. Choose an interval $[r_{\text{low}}, r_{\text{high}}]$ where $S(r_{\text{low}}) < 0$ and $S(r_{\text{high}}) > 0$.
  2. Compute the midpoint $r_{\text{mid}} = \frac{r_{\text{low}}+r_{\text{high}}}{2}$
  3. Evaluate $S(r_{\text{mid}})$:
     - If $S(r_{\text{mid}}) = 0$, r_{mid} is the equilibrium rate
     - If $S(r_{\text{mid}}) < 0$, update $r_{\text{low}} = r_{\text{mid}}$
     - If $S(r_{\text{mid}}) > 0$, update $r_{\text{high}} = r_{\text{mid}}$
  4. Repeat until r_{mid} converges within a desired tolerance level.

Bisection Method

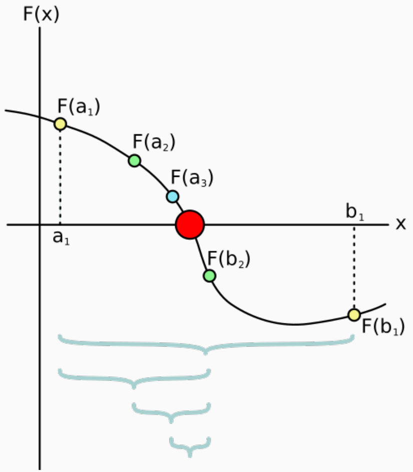


Figure 13: Graphical Representation of the Bisection Method for Finding *r*

General Equilibrium

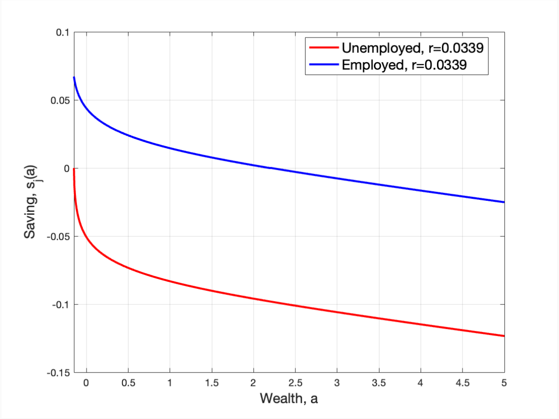


Figure 14: Savings Policy Function

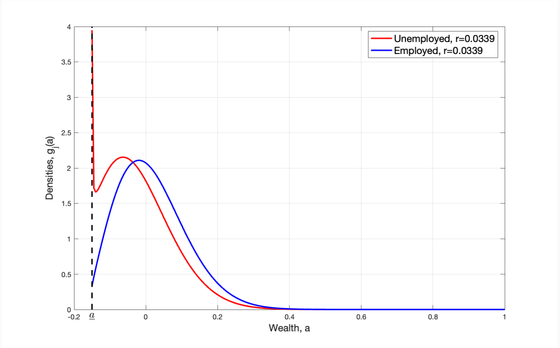


Figure 15: Implied Wealth Distribution

References

Achdou, Y., J. Han, J.-M. Lasry, P.-L. Lions, and B. Moll (2022). Income and wealth distribution in macroeconomics: A continuous-time approach. *The review of economic studies 89*(1), 45–86.