

Pricing Assets in Heterogeneous Agents Macro Economies

Goutham Gopalakrishna (Rotman-Toronto)
Zhouzhou Gu (Princeton University)
Jonathan Payne (Princeton University)

ML+Econ Reading Group

Princeton University

December, 2023

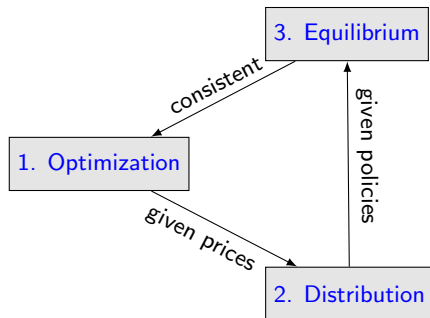
This Project

- ▶ We focus on **solving** macroeconomic models **not estimating** statistical models.
- ▶ We study continuous time DSGE models with the following features:
 1. A short-term risk free asset and a set of long-term risky assets,
 2. A large collection of price-taking agents who face idiosyncratic shocks/constraints
- ▶ We use deep learning to obtain a global solution.
- ▶ We are unaware of other global solution techniques for these models:
 1. *Traditional techniques*: are intractable unless the agent distribution can be approximated by low dimensional moments.
(e.g. Krusell and Smith, 1998, Fernández-Villaverde, Hurtado, and Nuno, 2023, Kubler and Scheidegger, 2019)
 2. *Other deep learning techniques*: have struggled to solve models with wealth distribution, long-lived assets and complicated portfolio choice.
(e.g. Duarte, 2018, Gopalakrishna, 2021, Sauzet, 2021, Gu et al., 2023, Fernández-Villaverde, Hurtado, and Nuno, 2023, Huang, 2023, Azinovic and Žemlička, 2023)

Three blocks

General equilibrium for this economy can be characterized by three blocks

1. **Optimization block:** High but finite dimensional PDE capturing agents optimization over consumption, portfolio holdings taking prices as given
2. **Distribution block:** High but finite dimensional law of motion of agent's wealth
3. **Equilibrium block:** Ensures that price processes are consistent with equilibrium



This is the first paper that can solve models with all three non-trivial blocks.

Roadmap

Illustrative model

Comparison to other models

Solution algorithm

Solutions to example models

Under the hood

General Equilibrium Endowment Asset Pricing Model

- ▶ Time is continuous with infinite horizon.
- ▶ Perishable consumption good created by a Lucas tree that produces output:

$$dy_t = \mu_y(y_t)dt + \sigma_y(y_t)dW_t$$

where W_t is a standard Brownian motion.

- ▶ Populated by price-taking households with flow utility $u(c_{i,t})$.
- ▶ Competitive markets for goods, risk-free bonds in net zero supply (with price r_t), and equity shares in Lucas tree (with price q_t). The price process follows

$$dq_t = \mu_t^q q_t dt + \sigma_t^q q_t dW_t$$

- ▶ **Financial frictions** restrict the agents' choice by $\Psi(a_{i,t}, b_{i,t}, \kappa_i) \geq 0$
 - ▶ $a_{i,t}$ is agent wealth, $b_{i,t}$ is agent bond holdings.
 - ▶ E.g. borrowing constraint is $a_{i,t} - \kappa_i \geq 0$.
 - ▶ E.g. equity non-participation constraint is $a_{i,t} - b_{i,t} = 0$.
 - ▶ Typically easier to model as penalty $\psi(a_{i,t}, b_{i,t}, \kappa_i) = |\Psi(a_{i,t}, b_{i,t}, \kappa_i)|^2$.

Optimization and Equilibrium

Given the belief about price processes \hat{r}, \hat{q} , agent i solves

$$\max_{c,b} \left\{ \mathbb{E}_0 \left[\int_0^\infty e^{-\rho t} (u(c_{i,t}) + \psi(a_{i,t}, b_{i,t}, \kappa_i)) dt \right] \right\} \quad (1)$$
$$s.t. \quad da_{i,t} = b_{i,t} \hat{r}_t - c_{i,t} + \left(\mu_t^q + \frac{y_t}{\hat{q}_t} \right) (a_{i,t} - b_{i,t}) + (a_{i,t} - b_{i,t}) \sigma_t^q dW_t$$

Optimization and Equilibrium

Given the belief about price processes \hat{r}, \hat{q} , agent i solves

$$\max_{c,b} \left\{ \mathbb{E}_0 \left[\int_0^{\infty} e^{-\rho t} (u(c_{i,t}) + \psi(a_{i,t}, b_{i,t}, \kappa_i)) dt \right] \right\} \quad (1)$$
$$s.t. \quad da_{i,t} = b_{i,t} \hat{r}_t - c_{i,t} + \left(\mu_t^q + \frac{y_t}{\hat{q}_t} \right) (a_{i,t} - b_{i,t}) + (a_{i,t} - b_{i,t}) \sigma_t^q dW_t$$

Equilibrium:

1. Individual i 's consumption decision c_t^i and bond holding b_t^i solves problem (1), given their believed price process (\hat{r}, \hat{q}) ;
2. Equilibrium prices (q_t, r_t) solves market clearing conditions: (i) goods market $\sum_i c_{i,t} = y$, (ii) stock market $\sum_i (a_{i,t} - b_{i,t}) = q_t$ and (iii) bond market $\sum_i b_{i,t} = 0$.
3. Agent beliefs about the price process are consistent with the optimal behaviour of other agents in the sense that $(\hat{r}, \hat{q}) = (r, q)$.

Recursive Characterization of Equilibrium

Individual state = a_i , Aggregate states = $(y, \{a_j\}_{j \neq i}) = (\cdot)$.

Given belief about evolution of other agents, $(\hat{\mu}_{a_j}(\cdot), \hat{\sigma}_{a_j}(\cdot))$, agent i solves:

$$\begin{aligned} \rho V_i(a_i, \cdot) = & \max_{c_i, b_i} \left\{ u(c_i) + \psi(a_i, s_i, \kappa_i) + \frac{\partial V_i}{\partial a_i} \mu_{a_i}(c_i, b_i, \cdot) + \frac{\partial V_i}{\partial y} \mu_y \right. \\ & + \frac{1}{2} \frac{\partial^2 V_i}{\partial a_i^2} \sigma_{a_i}^2(b_i, \cdot) + \frac{1}{2} \frac{\partial^2 V_i}{\partial y^2} \sigma_y^2 + \frac{\partial^2 V_i}{\partial a_i \partial y} \sigma_{a_i}(b_i, \cdot) \sigma_y + \sum_{j \neq i} \frac{\partial V_i}{\partial a_j} \hat{\mu}_{a_j}(\cdot) \\ & \left. + \frac{1}{2} \sum_{j \neq i, j' \neq i} \frac{\partial^2 V_i}{\partial a_j \partial a_{j'}} \hat{\sigma}_{a_j}(\cdot) \hat{\sigma}_{a_{j'}}(\cdot) + \sum_{j \neq i} \frac{\partial^2 V_i}{\partial a_j \partial y} \hat{\sigma}_{a_j}(\cdot) \sigma_y + \sum_{j \neq i} \frac{\partial^2 V_i}{\partial a_i \partial a_j} \sigma_{a_i}(b_i, \cdot) \hat{\sigma}_{a_j}(\cdot) \right\} \end{aligned}$$

Recursive Characterization of Equilibrium

Individual state = a_i , Aggregate states = $(y, \{a_j\}_{j \neq i}) = (\cdot)$.

Given belief about evolution of other agents, $(\hat{\mu}_{a_j}(\cdot), \hat{\sigma}_{a_j}(\cdot))$, agent i solves:

$$\begin{aligned} \rho V_i(a_i, \cdot) = & \max_{c_i, b_i} \left\{ u(c_i) + \psi(a_i, s_i, \kappa_i) + \frac{\partial V_i}{\partial a_i} \mu_{a_i}(c_i, b_i, \cdot) + \frac{\partial V_i}{\partial y} \mu_y \right. \\ & + \frac{1}{2} \frac{\partial^2 V_i}{\partial a_i^2} \sigma_{a_i}^2(b_i, \cdot) + \frac{1}{2} \frac{\partial^2 V_i}{\partial y^2} \sigma_y^2 + \frac{\partial^2 V_i}{\partial a_i \partial y} \sigma_{a_i}(b_i, \cdot) \sigma_y + \sum_{j \neq i} \frac{\partial V_i}{\partial a_j} \hat{\mu}_{a_j}(\cdot) \\ & \left. + \frac{1}{2} \sum_{j \neq i, j' \neq i} \frac{\partial^2 V_i}{\partial a_j \partial a_{j'}} \hat{\sigma}_{a_j}(\cdot) \hat{\sigma}_{a_{j'}}(\cdot) + \sum_{j \neq i} \frac{\partial^2 V_i}{\partial a_j \partial y} \hat{\sigma}_{a_j}(\cdot) \sigma_y + \sum_{j \neq i} \frac{\partial^2 V_i}{\partial a_i \partial a_j} \sigma_{a_i}(b_i, \cdot) \hat{\sigma}_{a_j}(\cdot) \right\} \end{aligned}$$

FOCs after defining $\xi_i := \frac{\partial V_i}{\partial a_i}$, and applying Ito's lemma to ξ_i are

$$\xi_i = u'(c_i) \quad \xi_i(r - r^q) = \sigma_{\xi_i} \sigma^q + (\partial \psi_i / \partial b_i)$$

In equilibrium, beliefs are consistent: $(\hat{\mu}_{a_j}(\cdot), \hat{\sigma}_{a_j}(\cdot)) = (\mu_{a_j}(\cdot), \sigma_{a_j}(\cdot))$

Imposing Equilibrium

We use aggregate state $(y, \{\eta_i\}_{1 \leq i \leq I})$, where $\eta_i := a_i/A$ is agent i 's wealth share.

Once equilibrium is imposed, we know ξ_i is a function of $(y, \{\eta_i\}_{1 \leq i \leq I})$

... so we can write μ_{ξ_i} and σ_{ξ_i} in terms of derivatives of $(y, \{\eta_i\}_{1 \leq i \leq I})$.

We group the resulting equilibrium equations into three blocks.

Block-1

1. Optimization

- ▶ Given prices $(r, r_q, q, \mu_q, \sigma_q)$, agent optimization implies:

Consumption: $\xi_i = u'(c_i)$

Euler equations:

$$\rho \xi_i = r \xi_i + \sum_j \frac{\partial \xi_i}{\partial \eta_j} \mu_{j,\eta} + \frac{\partial \xi_i}{\partial y} \mu_y + \frac{1}{2} \sum_{j,j'} \frac{\partial^2 \xi_i}{\partial \eta_j \partial \eta_{j'}} \sigma_{j,\eta} \sigma_{j',\eta} + \frac{1}{2} \frac{\partial^2 \xi_i}{\partial y^2} \sigma_y^2 + \sum_j \frac{\partial^2 \xi_i}{\partial \eta_j \partial y} \sigma_{j,\eta} \sigma_y + \frac{\partial \psi_i}{\partial \mathbf{a}_i}, \quad \text{for all } i$$

Asset pricing: $\xi_i(r - r_q) = \left(\sum_j \frac{\partial \xi_i}{\partial \eta_j} \sigma_{j,\eta} + \frac{\partial \xi_i}{\partial y} \sigma_y \right) \sigma_q + \frac{\partial \psi_i}{\partial \mathbf{b}_i}$

Block-2

2. Distribution

Given the prices $(r, r_q, q, \mu_q, \sigma_q)$ and (c, ξ, b) , the law of motion of wealth shares is given as

$$\begin{aligned}d\eta_{j,t} &= \mu_{j,\eta,t}dt + \sigma_{j,\eta,t}dW_t \\ \mu_{j,\eta,t} &= \frac{y_t}{q_t} + \frac{b_{j,t}}{a_{j,t}}(r_t - r_{q,t}) - \frac{(u')^{-1}(\xi_{j,t})}{\eta_{j,t}q_t} + \frac{b_{j,t}}{a_{j,t}}(\sigma_t^q)^2 \\ \sigma_{j,\eta,t} &= \left[\frac{1}{a_j} (a_{j,t} - b_{j,t}) \sigma_t^q - \sigma_t^q \right] = -\frac{b_{j,t}}{a_{j,t}}\sigma_t^q\end{aligned}$$

Block-3

3. Equilibrium

In equilibrium, the following market clearing conditions must hold:

$$\sum_i b_i = 0 \qquad \sum_i \eta_i = 1 \qquad \sum_i c_i = y$$

Clearing conditions pin down the prices but LOM of price, q , is implicit.

We impose consistency conditions on the LOM of the price q to close the model:

$$q\mu^q = \sum_j \frac{\partial q}{\partial \eta_j} \mu_{j,\eta} + \frac{\partial q}{\partial y} \mu_y + \frac{1}{2} \sum_{j,j'} \frac{\partial^2 q}{\partial \eta_j \partial \eta_{j'}} \sigma_{j,\eta} \sigma_{j',\eta} + \sum_j \frac{\partial^2 q}{\partial \eta_j \partial y} \sigma_{j,\eta} \sigma_y + \frac{1}{2} \frac{\partial^2 q}{\partial y^2} \sigma_y^2$$
$$q\sigma^q = \sum_j \frac{\partial q}{\partial \eta_j} \sigma_{j,\eta} + \frac{\partial q}{\partial y} \sigma_y$$

Roadmap

Illustrative model

Comparison to other models

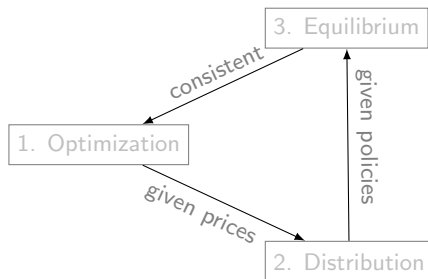
Solution algorithm

Solutions to example models

Under the hood

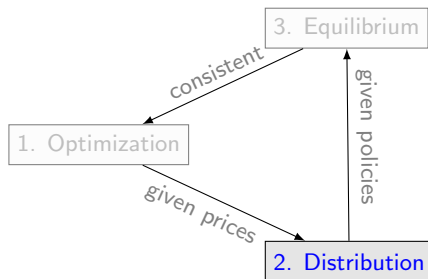
Comparison to other models

Models	Blocks			Method
	1	2	3	
Representative Agent (à la Lucas, 1978)	simple	NA	simple	Finite difference



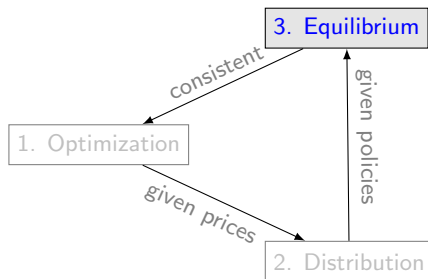
Comparison to other models

Models	Blocks			Method
	1	2	3	
Representative Agent (à la Lucas, 1978)	simple	NA	simple	Finite difference
Heterogeneous Agents (à la Krusell and Smith, 1998)	simple	✓	simple	Gu et al., 2023



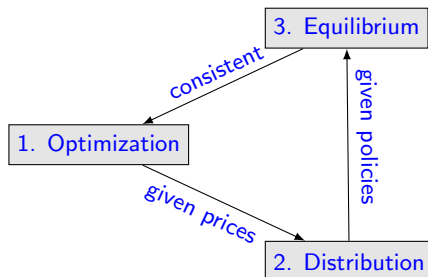
Comparison to other models

Models	Blocks			Method
	1	2	3	
Representative Agent (à la Lucas, 1978)	simple	NA	simple	Finite difference
Heterogeneous Agents (à la Krusell and Smith, 1998)	simple	✓	simple	Gu et al., 2023
Long-lived assets (à la Brunnermeier and Sannikov, 2014)	closed-form	low-dim	✓	Gopalakrishna, 2021



Comparison to other models

Models	Blocks			Method
	1	2	3	
Representative Agent (à la Lucas, 1978)	simple	NA	simple	Finite difference
Heterogeneous Agents (à la Krusell and Smith, 1998)	simple	✓	simple	Gu et al., 2023
Long-lived assets (à la Brunnermeier and Sannikov, 2014)	closed-form	low-dim	✓	Gopalakrishna, 2021
HA + Long-lived assets	✓	✓	✓	This paper



Roadmap

Illustrative model

Comparison to other models

Solution algorithm

Solutions to example models

Under the hood

Model summary

- ▶ Need to solve a system of PDEs + algebraic equations for (ξ_i, q) .

$$\begin{aligned} \text{Euler eqns.:} \quad 0 &= (r - \rho)\xi_i + \sum_j \frac{\partial \xi_i}{\partial \eta_j} \mu_{j,\eta} + \frac{\partial \xi_i}{\partial y} \mu_y + \frac{1}{2} \sum_{j,j'} \frac{\partial^2 \xi_i}{\partial \eta_j \partial \eta_{j'}} \sigma_{j,\eta} \sigma_{j',\eta} \\ &\quad + \frac{1}{2} \frac{\partial^2 \xi_i}{\partial y^2} \sigma_y^2 + \sum_j \frac{\partial^2 \xi_i}{\partial \eta_j \partial y} \sigma_{j,\eta} \sigma_y + \frac{\partial \psi_i}{\partial \mathbf{a}_i} \end{aligned}$$

$$\text{Price:} \quad q = \frac{y}{\sum_i \omega_i \eta_i}$$

$$\begin{aligned} \text{Consistency-1: } q\mu^q &= \sum_j \frac{\partial q}{\partial \eta_j} \mu_{j,\eta} + \frac{\partial q}{\partial y} \mu_y + \frac{1}{2} \sum_{j,j'} \frac{\partial^2 q}{\partial \eta_j \partial \eta_{j'}} \sigma_{j,\eta} \sigma_{j',\eta} + \\ &\quad + \frac{1}{2} \frac{\partial^2 q}{\partial y^2} \sigma_y^2 + \sum_j \frac{\partial^2 q}{\partial \eta_j \partial y} \sigma_{j,\eta} \sigma_y \end{aligned}$$

$$\text{Consistency-2: } q\sigma^q = \sum_j \frac{\partial q}{\partial \eta_j} \sigma_{j,\eta} + \frac{\partial q}{\partial y} \sigma_y$$

where $\omega_i := c_i/a_i$ is the consumption-wealth ratio of agent i .

Neural networks

We can parameterize functions (ξ_i, q) as neural network objects. Why?

- ▶ Automatic differentiation can be used for derivatives, and
- ▶ High-dimensional non-linear optimizers have been developed for neural networks

Let's start with the Euler equation of agent i and the unknown function ξ_i

$$0 = (r - \rho)\xi_i + \sum_j \frac{\partial \xi_i}{\partial \eta_j} \mu_{j,\eta} + \frac{\partial \xi_i}{\partial y} \mu_y + \frac{1}{2} \sum_{j,j'} \frac{\partial^2 \xi_i}{\partial \eta_j \partial \eta_{j'}} \sigma_{j,\eta} \sigma_{j',\eta} \\ + \frac{1}{2} \frac{\partial^2 \xi_i}{\partial y^2} \sigma_y^2 + \frac{1}{2} \sum_j \frac{\partial^2 \xi_i}{\partial \eta_j \partial y} \sigma_{j,\eta} \sigma_y + \frac{\partial \psi_i}{\partial \mathbf{a}_i}$$

- ▶ This is a high-dimensional PDE.
- ▶ In our model, the evolution of price function q also follows a PDE of the type above (the consistency conditions)

Neural networks

One way to solve this PDE is to parameterize the unknown function ξ_i as a neural network object $\hat{\xi}_i$ with parameters Θ_ξ .

The approximated function $\hat{\xi}_i$ takes as input a vector of training samples $\mathbf{X}^n = (y^n, \{\eta_i\}^n)_{n=1}^N$ drawn uniformly from the state space $\mathbf{X} := (y, \{\eta_i\}_{1 \leq i \leq l})$.

Neural networks

One way to solve this PDE is to parameterize the unknown function ξ_i as a neural network object $\hat{\xi}_i$ with parameters Θ_ξ .

The approximated function $\hat{\xi}_i$ takes as input a vector of training samples $\mathbf{X}^n = (y^n, \{\eta_i\}^n)_{n=1}^N$ drawn uniformly from the state space $\mathbf{X} := (y, \{\eta_i\}_{1 \leq i \leq I})$.

Then the Euler equation residual (or PDE residual) can be computed as

$$\begin{aligned} f_i := & (r - \rho)\hat{\xi}_i + \sum_j \frac{\partial \hat{\xi}_i}{\partial \eta_j} \mu_{j,\eta} + \frac{\partial \hat{\xi}_i}{\partial y} \mu_y + \frac{1}{2} \sum_{j,j'} \frac{\partial^2 \hat{\xi}_i}{\partial \eta_j \partial \eta_{j'}} \sigma_{j,\eta} \sigma_{j',\eta} \\ & + \frac{1}{2} \frac{\partial^2 \hat{\xi}_i}{\partial y^2} \sigma_y^2 + \frac{1}{2} \sum_j \frac{\partial^2 \hat{\xi}_i}{\partial \eta_j \partial y} \sigma_{j,\eta} \sigma_y + \frac{\partial \psi_i}{\partial \mathbf{a}_i} = 0 \end{aligned}$$

The Euler equation loss is given by $\frac{1}{N} \sum_{i=1}^N |f_i(\mathbf{X}^n)|$.

We can similarly compute the loss on consistency and goods market clearing conditions.

Parsimonious parameterization

Instead of parameterizing ξ_i and q , we parameterize the consumption-wealth ratio ω_i as $\hat{\omega}_i$ with Θ_ω and draw training sample \mathbf{X}^n

- ▶ Price q from the equilibrium condition $q = \frac{y}{\sum_i \hat{\omega}_i \eta_i}$
- ▶ Marginal value of wealth $\xi_i = u'(\hat{\omega}_i \eta_i q)$

We proceed to follow similar steps as before to compute Euler equation loss where the residual is computed as

$$f_i := (r - \rho)\xi_i(\hat{\omega}_i) + \sum_j \frac{\partial \xi_i(\hat{\omega}_i)}{\partial \eta_j} \mu_{j,\eta} + \frac{\partial \xi_i(\hat{\omega}_i)}{\partial y} \mu_y + \frac{1}{2} \sum_{j,j'} \frac{\partial^2 \xi_i(\hat{\omega}_i)}{\partial \eta_j \eta_{j'}} \sigma_{j,\eta} \sigma_{j',\eta} \quad (2)$$
$$+ \frac{1}{2} \frac{\partial^2 \xi_i(\hat{\omega}_i)}{\partial y^2} \sigma_y^2 + \sum_j \frac{\partial^2 \xi_i(\hat{\omega}_i)}{\partial \eta_j \partial y} \sigma_{j,\eta} \sigma_y + \frac{\partial \psi_i}{\partial \mathbf{a}_i} = 0$$

Parsimonious parameterization

Instead of parameterizing ξ_i and q , we parameterize the consumption-wealth ratio ω_i as $\hat{\omega}_i$ with Θ_ω and draw training sample \mathbf{X}^n

- ▶ Price q from the equilibrium condition $q = \frac{y}{\sum_i \hat{\omega}_i \eta_i}$
- ▶ Marginal value of wealth $\xi_i = u'(\hat{\omega}_i \eta_i q)$

We proceed to follow similar steps as before to compute Euler equation loss where the residual is computed as

$$f_i := (r - \rho)\xi_i(\hat{\omega}_i) + \sum_j \frac{\partial \xi_i(\hat{\omega}_i)}{\partial \eta_j} \mu_{j,\eta} + \frac{\partial \xi_i(\hat{\omega}_i)}{\partial y} \mu_y + \frac{1}{2} \sum_{j,j'} \frac{\partial^2 \xi_i(\hat{\omega}_i)}{\partial \eta_j \partial \eta_{j'}} \sigma_{j,\eta} \sigma_{j',\eta} \quad (2)$$
$$+ \frac{1}{2} \frac{\partial^2 \xi_i(\hat{\omega}_i)}{\partial y^2} \sigma_y^2 + \sum_j \frac{\partial^2 \xi_i(\hat{\omega}_i)}{\partial \eta_j \partial y} \sigma_{j,\eta} \sigma_y + \frac{\partial \psi_i}{\partial \mathbf{a}_i} = 0$$

- ▶ The derivatives can still be easily computed using automatic differentiation, which leads to the same Euler equation loss as before.
- ▶ With parsimonious parameterization, we don't require goods market clearing condition loss

Algorithm

Algorithm Pseudo Code

- 1: Initialize neural network objects $\{\hat{\omega}_i\}_{1 \leq i \leq I}$ with parameters $\{\Theta_{\omega_i}\}_{1 \leq i \leq I}$.
 - 2: Initialize optimizer.
 - 3: **while** Loss > tolerance, at iteration t **do**
 - 4: Sample training points: $\mathbf{X}^n = (y^n, \{\eta_i\}^n)_{n=1}^N$.
 - 5: Compute $q^n = y^n / (\sum_i \hat{\omega}_i(\mathbf{X}^n) \eta_i^n)$, $c_i^n = \hat{\omega}_i(\mathbf{X}^n) \eta_i^n q^n$, and $\xi_i^n = u'(c_i^n)$ for each i
 - 6: Determine σ_{η} , $r_q - r$ and then $\sigma^q, \mu^q, \mu_{\eta}$.
 - 7: Construct loss as $\sum_i \frac{1}{N} \sum_n |f_i(\mathbf{X}^n)|$, where f_i is defined in (2).
 - 8: Update $\{\Theta_{\omega_i}\}$ using ADAM optimizer.
 - 9: **end while**
-

Approach Q & A

- ▶ **Q.** *What about imposing dimension reduction?*
 - ▶ Han, Yang, and E, 2021 and Kahou et al., 2021 suggest feeding the distribution through a preliminary neural network that reduces the dimension.
 - ▶ We have solved the model **without this approach** but agree this could be useful.
- ▶ **Q.** *Neural networks are slow to train so how do we calibrate?*
 - ▶ Could add parameters to neural network so we only have to train the model once.
 - ▶ Let ζ denote economic parameters needing external calibration to match moments.
 - ▶ Add ζ as inputs into the neural network:

$$V(\hat{X}_t, \zeta) \approx \hat{V}(\hat{X}_t, \zeta; \theta), \quad \hat{X}_t = \{x_t, z_t, \hat{g}_t\}$$

- ▶ Train the neural network using sampling from \hat{X} and ζ .
- ▶ Use the equilibrium \hat{V} to calculate moments for different parameters.

Roadmap

Illustrative model

Comparison to other models

Solution algorithm

Solutions to example models

Under the hood

Representative Agent Model (Lucas (1978))

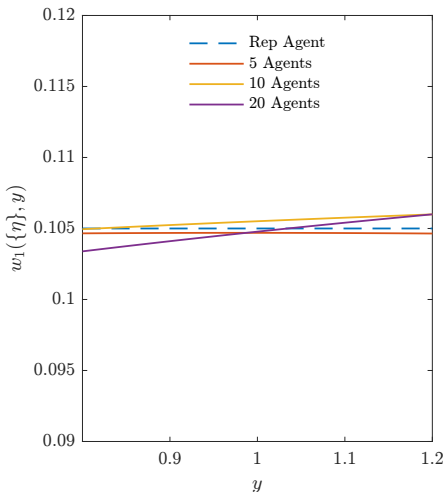
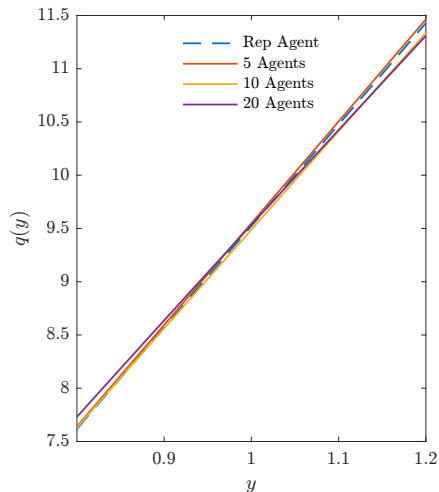
Suppose there are no financial frictions: $\psi(a_{i,t}, b_{i,t}, \kappa_i) = 0$.

In this case, households are identical so there is a “representative agent”.

The model has closed form solution:

$$q(y) = \frac{y}{\rho + (\gamma - 1)\mu - \frac{1}{2}\gamma(\gamma - 1)\sigma^2}$$
$$\omega(y) = \left[\rho + (\gamma - 1)\mu - \frac{1}{2}\gamma(\gamma - 1)\sigma^2 \right]$$

Model solution. MSE: $< 10^{-4}$



As-if Complete Market Model, $\gamma = 5$, $\mu = 2\%$, $\sigma = 5\%$, $\rho = 5\%$.

Limited Participation Model (Basak and Cuoco, 1998)

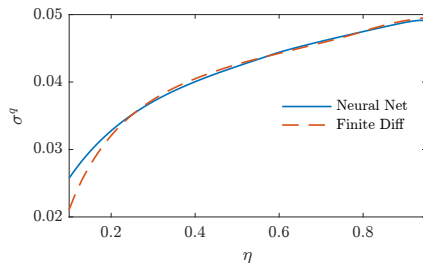
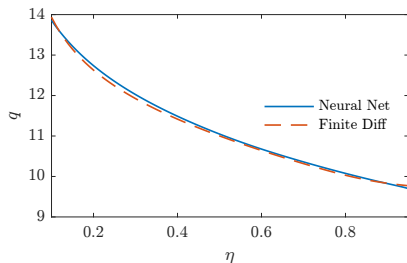
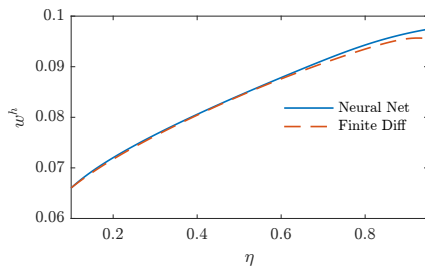
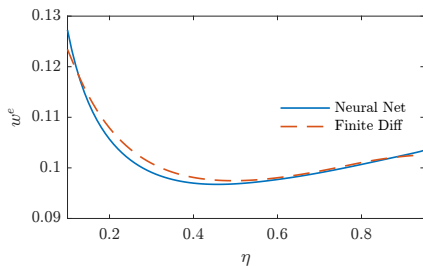
Two types of agents: experts (e) and households (h).

Expert sector can hold stocks and bonds.

Household sector can only hold bonds: $\Psi_h(a_{h,t}, b_{h,t}) = a_{h,t} - b_{h,t} = 0$.

State space is (y, η) , where η is expert's wealth share.

Model solution. L2 Loss: $< 10^{-5}$



2 Agents Limited Participation Model, $\gamma = 5$, $\rho_e = \rho_h = 5\%$, $\mu = 2\%$, $\sigma = 5\%$.

Production Economy With Idiosyncratic Risk

Production technology:

$$y_t = z_t K_t^\alpha L_t^{1-\alpha}$$

Capital accumulation (investment friction $\Phi(\iota) = \frac{1}{\phi} \log(\phi \iota + 1)$):

$$dK_t = (\Phi(\iota_t) - \delta) K_t dt$$

Dividend price ratio:

$$\frac{d_t}{q_t} = \frac{\alpha y_t - \iota_t K_t}{q_t K_t}$$

Labor endowment (Poisson switching):

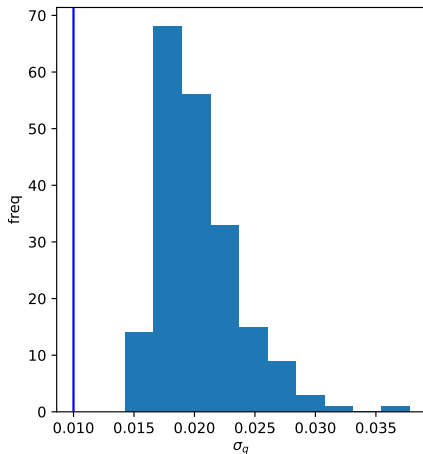
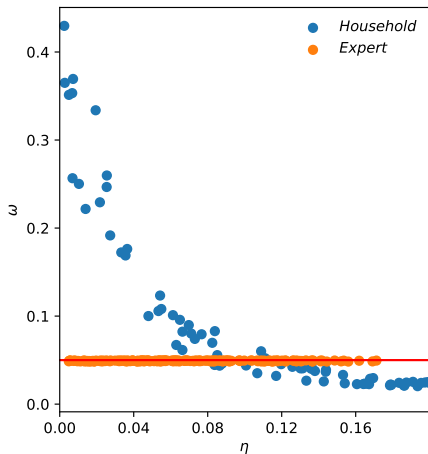
$$\ell_i \in \{\underline{\ell}, \bar{\ell}\}$$

Households wealth process:

$$da_{i,t} = [\hat{w}_t \ell_{i,t} + a_{i,t} \hat{r}_t - c_{i,t}] dt$$

Expert's wealth process: the same as before

Model solution. L2 loss: $< 10^{-4}$



11 Agents Limited Participation model with Production: $\gamma = 1, \sigma = 1\%$,
 $\alpha = 0.3, \phi = 10, \delta = 0.05, \lambda_1 = \lambda_2 = 0.4, \underline{\ell} = 0.3, \bar{\ell} = 1.7, \rho_e = \rho_h = 5\%$

Roadmap

Illustrative model

Comparison to other models

Solution algorithm

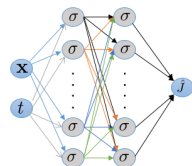
Solutions to example models

Under the hood

Automatic differentiation in practice

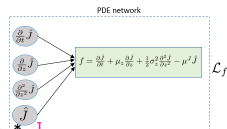
Approximating J using a neural network

```
def J(z,t):  
    J = neural_net(tf.concat([z,t],1),weights,biases)  
    return J
```



Constructing regularizer: 1D model

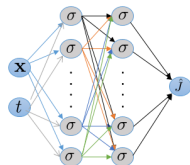
```
def f(z,t):  
    J = J(z,t)  
    J_t = tf.gradients(J,t)[0]  
    J_z = tf.gradients(J,z)[0]  
    J_zz = tf.gradients(J_z,z)[0]  
    f = J_t + advection * J_z + diffusion * J_zz - linearTerm * J  
    return f
```



Automatic differentiation in practice

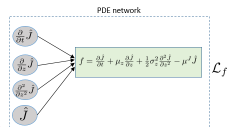
Approximating J using a neural network

```
def J(z,t):
    J = neural_net(tf.concat([z,t],1),weights,biases)
    return J
```

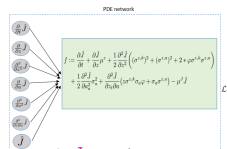


Constructing regularizer: 1D model

```
def f(z,t):
    J = J(z,t)
    J_t = tf.gradients(J,t)[0]
    J_z = tf.gradients(J,z)[0]
    J_zz = tf.gradients(J_z,z)[0]
    f = J_t + advection * J_z + diffusion * J_zz - linearTerm
    return f
```



```
def f(z,a,t):
    J = J(z,a,t)
    J_t = tf.gradients(J,t)[0]
    J_z = tf.gradients(J,z)[0]
    J_a = tf.gradients(J,a)[0]
    J_zz = tf.gradients(J_z,z)[0]
    J_aa = tf.gradients(J_a,a)[0]
    J_az = tf.gradients(J_a,z)[0]
    f = J_t + advection_z * J_z + advection_a * J_a + diffusion_z * J_zz +
        diffusion_a * J_aa + crossTerm * J_az - linearTerm * J
    return f
```



Conclusion

We have used deep learning to solve continuous time DSGE models with:





1. A short-term risk free asset and a set of long-term risky assets,
2. A large collection of price-taking agents who face idiosyncratic shocks/constraints

We believe this will allow the development of heterogeneous agent macro-finance.







Our current project uses this to solve a model of institutional asset pricing.

Thank You!

References I

-  Azinovic, Marlon and Jan Žemlička (2023). “Economics-Inspired Neural Networks with Stabilizing Homotopies”. In: *arXiv preprint arXiv:2303.14802*.
-  Basak, Suleyman and Domenico Cuoco (1998). “An equilibrium model with restricted stock market participation”. In: *The Review of Financial Studies* 11.2, pp. 309–341.
-  Brunnermeier, Markus K and Yuliy Sannikov (2014). “A macroeconomic model with a financial sector”. In: *American Economic Review* 104.2, pp. 379–421.
-  Duarte, Victor (2018). “Machine learning for continuous-time economics”. In: *Available at SSRN 3012602*.
-  Fernández-Villaverde, Jesús, Samuel Hurtado, and Galo Nuno (2023). “Financial frictions and the wealth distribution”. In: *Econometrica* 91.3, pp. 869–901.
-  Gopalakrishna, Goutham (2021). “Aliens and continuous time economies”. In: *Swiss Finance Institute Research Paper* 21-34.
-  Gu, Zhouzhou et al. (2023). “Deep Learning Solutions to Master Equations for Continuous Time Heterogeneous Agent Macroeconomic Models”. In.
-  Han, Jiequn, Yucheng Yang, and Weinan E (2021). “DeepHAM: A Global Solution Method for Heterogeneous Agent Models with Aggregate Shocks”. In: *arXiv preprint arXiv:2112.14377*.

References II

-  Huang, Ji (2023). “Breaking the Curse of Dimensionality in Heterogeneous-Agent Models: A Deep Learning-Based Probabilistic Approach”. In: *SSRN Working Paper*.
-  Kahou, Mahdi Ebrahimi et al. (2021). *Exploiting symmetry in high-dimensional dynamic programming*. Tech. rep. National Bureau of Economic Research.
-  Krusell, Per and Anthony A Smith (1998). “Income and Wealth Heterogeneity in the Macroeconomy”. In: *Journal of Political Economy* 106.5, pp. 867–896.
-  Kubler, Felix and Simon Scheidegger (2019). “Self-justified equilibria: Existence and computation”. In: *Available at SSRN 3494876*.
-  Lucas, Robert E. (1978). “Asset Prices in an Exchange Economy”. In: *Econometrica* 46.6, pp. 1429–1445.
-  Sauzet, Maxime (2021). “Projection methods via neural networks for continuous-time models”. In: *Available at SSRN 3981838*.

Appendix: FOCs

FOCs are

$$\begin{aligned} [c_i] : \quad 0 &= u'(c_i) - \partial_a V_i(a_i) \\ [b_i] : \quad 0 &= -\frac{\partial V_i}{\partial a_i} (r(\cdot) - r_q(\cdot)) + \frac{\partial^2 V_i}{\partial a_i^2} (a_i - b_i) (\sigma^q)^2(\cdot) \\ &\quad + \frac{\partial^2 V_i}{\partial a_i \partial y} \sigma^q(\cdot) \sigma_y(\cdot) + \sum_{j \neq i} \frac{\partial^2 V_i}{\partial a_i \partial a_j} \sigma^q(\cdot) \hat{\sigma}_{a_j}(\cdot) + \frac{\partial \psi}{\partial b_i} \end{aligned} \tag{3}$$

▶ Back

Appendix: FOCs

$$[c_i]: \quad 0 = u'(c_i) - \xi_i$$

$$[b_i]: \quad 0 = -\xi_i (r(\cdot) - r_q(\cdot)) + \frac{\partial \xi_i}{\partial a_i} (a_i - b_i) (\sigma^q)^2(\cdot) \\ + \frac{\partial^2 \xi_i}{\partial y^2} \sigma_y(\cdot)^2 + \frac{\partial \xi_i}{\partial y} \sigma^q(\cdot) \sigma_y(\cdot) + \sum_{j \neq i} \frac{\partial \xi_i}{\partial a_j} \sigma^q(\cdot) \hat{\sigma}_{a_j}(\cdot) + \frac{\partial \psi}{\partial b_i} \quad (4)$$

Rearranging the asset pricing equation and using

$$\sigma_{\xi_i} = \frac{\partial \xi_i}{\partial a_i} \sigma_{a_i} + \frac{\partial \xi_i}{\partial y} \sigma_y + \sum_{j \neq i} \frac{\partial \xi_i}{\partial a_j} \sigma_{a_j}$$

we get

$$\xi_i (r - r_q) = \sigma_{\xi_i} \sigma^q + \frac{\partial \psi_i}{\partial b_i}$$

▶ Back

Appendix: Grid sampling vs Simulation method

- ▶ Our method draws training points from throughout the state space
- ▶ Sampling procedure is complementary to simulation based methods ([Azinovic et al \(2018\)](#), [Villaverde et al \(2020\)](#)), but also works for models with **rare events** and financial constraints that bind far away from the steady state

