Deep Learning for Macroeconomics and Finance
(for solving models with complicated agent distributions)

Jonathan Payne

Princeton

based on work with Goutham Gopalakrishna, Zhouzhou Gu, Mathieu Laurière, Sebastian Merkel, Adam Rebei, and Yucheng Yang

June 5, 2024

PASC 2024: MS6F - Advances of Deep Learning in Economics

Many Economic Questions Involve Distributions Across Agents

- How do tax policies impact wealth inequality?
  (distribution of agent asset holdings)

- How do recessions impact wage and employment dispersion?
  (distribution of labor income across skills, education, and industries)

- How do innovation and trade shocks spread across firms (or countries)?
  (distribution of production technologies across firms)

- How do crisis shocks propagate through financial networks?
  (distribution of risk exposure across banks)

- What determines the distribution of city sizes?
  (distribution of population and firms across cities)

- ... These and many, many other questions require "**heterogeneous agent models**"

Technical Challenges and Solutions

- *Problem:* Hard to solve macro models with heterogeneous agents + aggregate shocks.
  - Infinite dimensional distribution becomes a state variable.
  - Traditional techniques: (linear/quadratic) perturbation, and approximate laws of motion.
- *Our goal:* develop a global solution technique for continuous macro time models:
  - Step 1: derive finite dimensional approximation to the distribution (finite agents, discrete state space, projections onto basis)
  - Step 2: train neural networks to solve the resulting high dimensional PDEs.
- *This talk:* discuss practical lessons from three of my papers:
  - Gu-Lauriere-Merkel-Payne (2024): solves business-cycle style macro models.
  - Payne-Rebei-Yang (2024): solves searching and maching models.
  - Gopalakrishna-Gu-Payne (2024): solves macro-finance models with implicit prices.

Deep learning offers the toolkit for heterogeneous agent macroeconomics

Table of Contents

Environment (Continuous Time Krusell-Smith '98)

- Continuous time, infinite horizon economy.
- Populated by $I = [0,1]$ households who consume goods, supply labor, and save wealth.
- Representative firm rents capital and labor to produce goods by $Y_t = e^{z_t} K_t^{\alpha} L_t^{1-\alpha}$:
  - K_t is capital hired, L_t is labor hired,
  - z_t is productivity (exogenous aggregate state variable): follows $dz_t = \eta(\bar{z} - z_t)dt + \sigma dB_t^0$
  - B_t^0 is a common Brownian motion process; it generates filtration $\mathcal{F}_t^0$.
- Competitive markets for goods (numeraire), capital (rental rate r_t), labor (wage w_t).

Household Problem

- Household i has idiosyncratic state $x_t^i = (a_t^i, n_t^i)$, where a_t^i is wealth, n_t^i is labor.
- Given belief about price processes, $(\tilde{r}, \tilde{w})$, household chooses consumption $c = \{c_t^i\}_{t\geq 0}$:

$$
\begin{aligned}
&\max_{\{c_t^i\}_{t\geq 0}} \mathbb{E}_0 \left[\int_0^\infty e^{-\rho t} u(c_t^i) dt \right] \\
&s.t. \quad da_t^i = (\tilde{w}_t n_t^i + \tilde{r}_t a_t^i - c_t^i) dt =: \mu_t^a dt, \quad a_t^i \geq \underline{a} \\
&\qquad\quad n_t^i \in \{n_1, n_2\}, \text{ switches at idiosyncratic Poisson rate } \lambda(n_t^i)
\end{aligned}
\tag{1}
$$

  - $u(c) = c^{1-\gamma}/(1-\gamma)$: utility function, ρ: discount rate, $\underline{a}$: borrowing limit.
  - $(\tilde{r}, \tilde{w}) = \{\tilde{r}_t, \tilde{w}_t\}_{t\geq 0}$ are agent beliefs about prices processes.
- Let $G_t = \mathcal{L}(a_t^i | \mathcal{F}_t^0)$ and g_t be population distribution and density of a_t^i, for history $\mathcal{F}_t^0$
  - Non degenerate since households get uninsurable idiosyncratic labor endowment shocks.

Equilibrium

Definition: Given an initial density g_0, an equilibrium for this economy consists of a collection of $\mathcal{F}_t^0$-adapted stochastic process, $\{c_t^i, g_t, z_t, q_t := [r_t, w_t] : t \geq 0, i \in I\}$, s.t.:

1. Given price process belief $\tilde{q}$, household consumption process, c_t^i, solves problem (1),
2. Given price process belief $\tilde{q}$, firm choose capital and labor optimally:
$$r_t = e^{z_t} \partial_K F(K_t, L) - \delta, \qquad w_t = e^{z_t} \partial_L F(K_t, L)$$
3. The price vector $q_t = [r_t, w_t]$ satisfies market clearing conditions:
$$K_t = \sum_{j \in \{1,2\}} \int a g_t(a, n_j) da, \qquad L = \sum_{j \in \{1,2\}} \int n_j g_t(a, n_j) da$$
4. Agent beliefs about the price process are consistent: $\tilde{q} = q$

Equilibrium (Combining Equations For Prices)

Definition: Given an initial density g_0, an equilibrium for this economy consists of a collection of $\mathcal{F}_t^0$-adapted stochastic process, $\{c_t^i, g_t, q_t := [r_t, w_t], z_t : t \geq 0, i \in I\}$, s.t.:

1. Given price process belief $\tilde{q}$, household consumption process, c_t^i, solves problem (1),
2. The price vector $q_t = [r_t, w_t]$ satisfies:

$$q_t = \begin{bmatrix} r_t \\ w_t \end{bmatrix} = \begin{bmatrix} e^{z_t}\partial_K F(K_t, L) - \delta \\ e^{z_t}\partial_L F(K_t, L) \end{bmatrix} =: Q(z_t, g_t), \text{ where } K_t = \sum_{j \in \{1,2\}} \int a g_t(a, n_j) da,$$

3. Agent beliefs about the price process are consistent: $\tilde{q} = q$.

Having closed form expressions for prices in terms of (z_t, g_t) makes problem very tractable

Recursive Representation of Equilibrium

- Aggregate states: (z, g), individual states: $x = (a, n)$, household value fn: $V(a, n, z, g)$.
- Given a belief $dg_t(x) = \tilde{\mu}_g(z_t, g_t)dt$, household at $x = (a, n)$ choose c to solve HJBE:

$$
\begin{aligned}
0 = \max_c \Big\{ & -\rho V(a,n,z,g) + u(c) + \partial_a V(a,n,z,g)(w(z,g)n + r(z,g)a - c) \\
& + \lambda(n)\left(V(a,\check{n},z,g) - V(a,n,z,g)\right) + \partial_z V(a,n,z,g)\mu^z(z) + 0.5\left(\sigma^z\right)^2 \partial_{zz} V(a,n,z,g) \\
& + \int_{\mathcal{X}} \partial_g V(y,z,g)\tilde{\mu}^g(y,z,g)dy \Big\}, \quad s.t. \quad \text{BC: } \partial_a V|_{a=\underline{a}} \geq u'(wn + r\underline{a})
\end{aligned}
$$

where $\check{n}$ is complement of n.

- For optimal policy rule $c^*(a, n, z, g; \tilde{\mu}^g)$ and z_t, population density, g, evolves by KFE:

$$
dg_t(a,n) = \underbrace{\left[-\partial_a[(w(z,g)n + r(z,g)a - c^*)g_t(a,n)] - \lambda(n)g_t(a,n) + \lambda(\check{n})g_t(a,\check{n})\right]}_{=:\mu^g(a_t,n_t,z_t,g_t;\tilde{\mu}^g)} dt
$$

- In equilibrium $\tilde{\mu}^g = \mu^g$.

Recursive Representation of Equilibrium (Soft Borrowing Constraint)

- Aggregate states: (z, g), individual states: $x = (a, n)$, household value fn: $V(a, n, z, g)$.
- Given a belief $dg_t(x) = \tilde{\mu}_g(z_t, g_t)dt$, household at $x = (a, n)$ choose c to solve HJBE:

$$
\begin{aligned}
0 = \max_c \Big\{ & -\rho V(a,n,z,g) + u(c) - \mathbf{1}_{a_t \leq \underline{a}} \psi(a_t) + \partial_a V(a,n,z,g)(w(z,g)n + r(z,g)a - c) \\
& + \lambda(n)\left(V(a,\check{n},z,g) - V(a,n,z,g)\right) + \partial_z V(a,n,z,g)\mu^z(z) + 0.5\left(\sigma^z\right)^2 \partial_{zz} V(a,n,z,g) \\
& + \int_{\mathcal{X}} \partial_g V(y,z,g)\tilde{\mu}^g(y,z,g)dy \Big\}, \quad s.t. \quad \text{BC: } \cancel{\partial_a V|_{a=\underline{a}} \geq u'(\cdot)}, \ \psi(a) = -\frac{1}{2}\kappa(a-\underline{a})^2
\end{aligned}
$$

where $\check{n}$ is complement of n.

- For optimal policy rule $c^*(a, n, z, g; \tilde{\mu}^g)$ and z_t, population density, g, evolves by KFE:

$$
dg_t(a,n) = \underbrace{\left[-\partial_a[(w(z,g)n + r(z,g)a - c^*)g_t(a,n)] - \lambda(n)g_t(a,n) + \lambda(\check{n})g_t(a,\check{n})\right]}_{=:\mu^g(c_t^*, a_t, n_t, z_t, g_t; \tilde{\mu}^g)} dt
$$

- In equilibrium $\tilde{\mu}^g = \mu^g$.

"Master Equation" Representation of Equilibrium

- "Master equation" substitutes KFE, market clearing & belief consistency into HJBE.

- Equilibrium value function $V(a, n, z, g)$ characterized by one PDE (if it exists):

$$
\begin{aligned}
0 = \ & -\rho V(a,n,z,g) + u(c^*(a,n,z,g)) + \mathbf{1}_{a_t \leq \underline{a}} \psi(a_t) \\
& + \partial_a V(a,n,z,g)(w(z,g)n + r(z,g)a - c^*(a,n,z,g)) \\
& + \lambda(x)\left(V(a,\check{n},z,g) - V(a,n,z,g)\right) + \partial_z V(a,n,z,g)\mu^z(z) + 0.5\left(\sigma^z\right)^2 \partial_{zz} V(a,n,z,g) \\
& + \int_{\mathcal{X}} \partial_g V(y,z,g)\mu^g(c^*(y,z,g),y,z,g)dy =: \mathcal{L}V
\end{aligned}
$$

where the optimal control c^* is characterised by: $u'(c^*(a,n,z,g)) = \partial_a V(a,n,z,g)$.

- Theory: studies whether this master equation exists; e.g. [Cardaliaguet et al., 2015]

- Our paper: look for a "global" finite, but high, dimensional approximation to V

Table of Contents

Numerical Approximation Outline

- Goal: characterize approximate solution to Master equation numerically
- Problem: Master equation contains an infinite dimensional derivative.
- Solution: three main ingredients:
  1. High but finite dimensional approximation to distribution and Master equation:
     (i). Replace continuum of agents by a finite population of agents, or
     (ii). Discretize the wealth variable, or
     (ii). Project distribution onto a finite dimensional set of basis functions $b_i(x)$ (e.g. eigenfunctions, Chebyshev polynomials, neural network, ...).
  2. Parameterize V by neural network, and
  3. Train the parameters to minimize the (approximate) master equation residual.

Ingredient 1: Finite Dimensional "Distribution" Approximation

	Finite Population	Discrete State	Projection
Params $\hat{\varphi}$	Agent states $\hat{\varphi}_t = \{(a_t^i, n_t^i)\}_{i \leq N}$	Masses on grid $\hat{\varphi}_{i,t}, \ \forall (a^i, n^i)_{i \leq N}$	Basis coefficients $\hat{\varphi}_{i,t}, \ \forall b_i(a; n)\|_{i \leq N}$
Dist. approx.	$\frac{1}{N} \sum_{i=1}^N \delta_{(a_t^i, n_t^i)}$	$\sum_{i=1}^N \hat{\varphi}_{i,t} \delta_{(a^i, n^i)}$	$\sum_{i=0}^N \hat{\varphi}_{i,t} b_i(a; n)$
KFE approx. ($\mu^{\hat{\varphi}}$)	Evolution of other agents' states	Evolution of mass between grid points (e.g. finite diff.)	Evolution of projection coefficients (least squares)
Dimension (N)	≈ 50	≈ 200	≈ 5

Projections more easily capture shape but have complicated KFE approximation

Ingredient 2: Approximate V by Neural Network

- Let $\boldsymbol{\omega} = (a, n, z, \hat{\varphi})$. Approximate value function $V(\boldsymbol{\omega})$ by neural network with form:

$$\begin{aligned}
\boldsymbol{h}^{(1)} &= \phi^{(1)}(W^{(1)}\boldsymbol{\omega} + \boldsymbol{b}^{(1)}) && \ldots \text{Hidden layer 1} \\
\boldsymbol{h}^{(2)} &= \phi^{(2)}(W^{(2)}\boldsymbol{h}^{(1)} + \boldsymbol{b}^{(2)}) && \ldots \text{Hidden layer 2} \\
&\vdots \\
\boldsymbol{h}^{(H)} &= \phi^{(H)}(W^{(H)}\boldsymbol{h}^{(H-1)} + \boldsymbol{b}^{(H)}) && \ldots \text{Hidden layer H} \\
S &= \sigma(\boldsymbol{h}^{(H)}) && \ldots \text{Surplus}
\end{aligned}$$

- Terminology: a fully connected feed forward NN (finite agent params in blue):
  - H: is the number of *hidden layers*, $(H = 5)$
  - Length of vector $\boldsymbol{h}^{(i)}$: number of *neurons* in hidden layer i, $(Length = 64)$
  - $\phi^{(i)}$: is the *activation function* for hidden layer i, $(\phi^i = tanh)$
  - σ: is the *activation function* for the final layer, (soft-plus)
  - $\boldsymbol{\Theta} = (W^1, \ldots W^{(H)}, b^{(1)}, \ldots, b^{(H)})$ are the *parameters*,

- Discrete state, projection more complicated NN [Sirignano and Spiliopoulos, 2018].

Ingredient 3: Algorithm ("EMINN" or "Economic Deep Galerkin")

1: Initialize the neural networks approximation for $\hat{V}$ with parameters Θ.

2: **while** Loss > tolerance **do**

3: Sample M new training points: $\boldsymbol{S} = (\boldsymbol{S}^m = (a^m, n^m, z^m, (\hat{\varphi}_i^m)_{i \leq N}))_{m=1}^M$.

4: Calculate the weighted average error across sample points, given current Θ:

$$\mathcal{E}(\boldsymbol{S}; \Theta) = \kappa^e \frac{1}{M} \sum_{m \leq M} |\hat{\mathcal{L}}(a^m, n^m, z^m, (\hat{\varphi}_i^m)_{i \leq N}; \Theta)| + \kappa^s \mathcal{E}^s(\boldsymbol{S}; \Theta), \quad \text{where}$$

- $\hat{\mathcal{L}}(a^m, n^m, z^m, (\hat{\varphi}_i^m)_{i \leq N}; \Theta)$ is error in Master equation $\hat{\mathcal{L}}$ at training point $\boldsymbol{S}^m$ (where derivates in $\hat{\mathcal{L}}$ are calculated using automatic differentiation.)
- $\mathcal{E}^s(\boldsymbol{S}; \Theta)$ is penalty for "wrong" shape (e.g. penalty for non-concavity of V)

5: Update parameters Θ by stochastic gradient descent: $\Theta^{new} = \Theta^{old} - \alpha D_\Theta \mathcal{E}(\boldsymbol{S}; \Theta)$

6: **end while**

The Macroeconomic Deep Learning Conundrum

- Algorithm is straightforward to describe and code
- ...but hard to implement successfully!
- I will discuss some features that we have found helpful.

Sampling Approaches

- Sampling (a, n, z): draw from uniform distribution, then add draws where error high.
- Sampling the parameters in the distribution approximation $(\hat{\varphi}^i)_{i \leq N}$:
  - *Moment sampling:*
    1. Draw samples for selected moments of the distribution (that are important for $\hat{Q}(z, \hat{\varphi})$).
    2. Sample $\hat{\varphi}$ from a distribution that satisfies the moments drawn in the first step.
  - *Mixed steady state sampling:*
    1. Solve for the steady state for a collection of fixed aggregate states z.
    2. Draw random, perturbed mixtures of this collection of steady state distributions.
  - *Ergodic sampling:*
    1. Simulate economy using current value function approximation.
    2. Use simulated distributions as training points.

Need to choose economically relevant subspace on which to sample.

Implementation Details

	Finite Population	Discrete State	Projection
Neural Network			
(i) Structure	Feed-forward	Recurrent + embedding	Recurrent + embedding
(ii) Initialization	$W(a,\cdot) = e^{-a}$	random	random
Sampling			
(i) (a,l)	Active sampling $[\underline{a},\overline{a}] \times \{y_1, y_2\}$	Uniform sampling $[\underline{a},\overline{a}] \times \{y_1, y_2\}$	Uniform sampling $[\underline{a},\overline{a}] \times \{y_1, y_2\}$
(ii) $(\hat{\varphi}_i)_{i \leq N}$	Moment sampling: sample r, then random agents positions that give r	Mixed steady-state sampling then ergodic sampling	Sample K and then orthogonal coefficients from ergodic sampling
(iii) z	$U[z_{min}, z_{max}]$	$U[z_{min}, z_{max}]$	$U[z_{min}, z_{max}]$
Loss Function			
(i) Constraints	$\partial_{aa}V(a,\cdot), \partial_{za}V(a,\cdot) < 0$	$\partial_{aa}V(a,\cdot), \partial_{za}V(a,\cdot) < 0$	$\partial_{aa}V(a,\cdot), \partial_{za}V(a,\cdot) < 0$
(ii) Learning rate	Decay from 10^{-4} to 10^{-6}	Decay from 10^{-4} to 10^{-6}	Decay from 10^{-4} to 10^{-6}

Neural Network Q & A

- ***Q.*** *What are the main differences to discrete time?*
  - Need to calculate derivatives rather than expectations (we use automatic differentiation)
  - Need to choose where to sample (sample more where master equation error is large)
- ***Q.*** *Why do we need shape constraints?*
  - Neural network can find "bad" approximate solutions,
    (E.g. portfolio problem has approximate solution $V \approx 0$ for high γ.) More
  - Option: penalize shape that correspond to known "bad" solutions.
  - Option: train ϕ satisfying $V(a, n, z, g) = \phi(a, n, z, g; \theta)(a - \underline{a})^{1-\gamma}$ instead of training V

Neural Network Q & A

- ***Q.*** *What about slowing down the updating?*
  - For projection methods, we use "Howard improvement algorithm" to slow down the rate of updating (fix policy rule for some iterations and just update V).
  - [Duarte, 2018] and [Gopalakrishna, 2021] suggest introducing a "false" time step but so far we have not found this necessary (or found a way to implement at high scale).
  - We use shape constraints as a replacement.
- ***Q.*** *What about imposing symmetry and/or dimension reduction?*
  - [Han et al., 2021] and [Kahou et al., 2021] suggest feeding the distribution through a preliminary neural network that reduces the dimension and imposes symmetry.
  - We find we can solve the problem with and without this approach.

Table of Contents

Testing the Algorithm

- We test the model with fixed aggregate productivity (Aiyagari (1994)) Plots

	Master equation loss	MSE(NN, FD)
Finite Agent NN	3.135×10^{-5}	4.758×10^{-5}
Discrete State Space NN	9.303×10^{-6}	6.591×10^{-5}

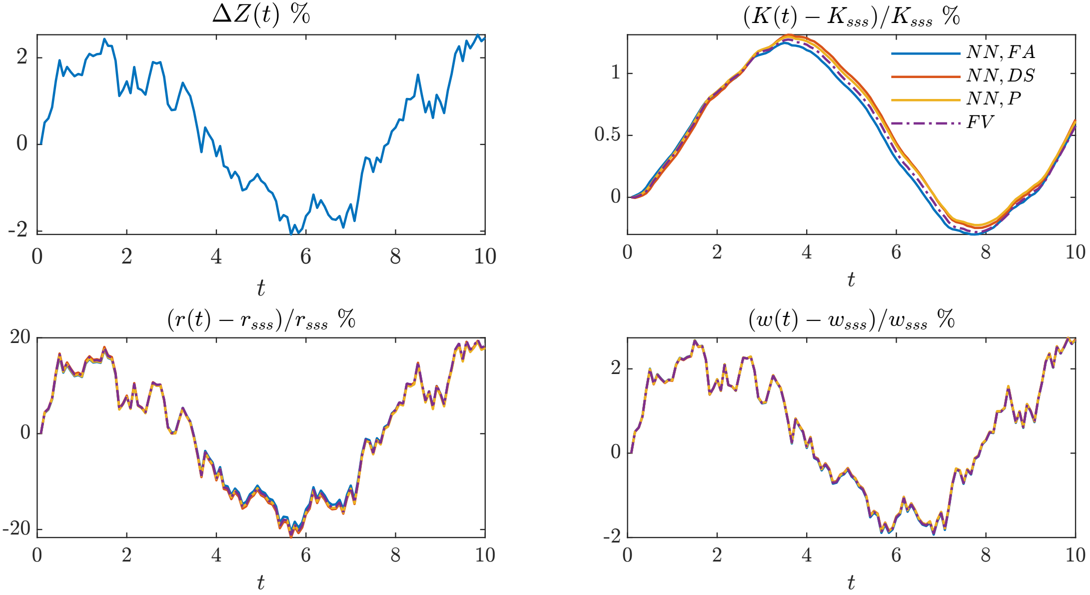
- Solve version with stochastic aggregate productivity (Krusell-Smith (1998)):

	Master equation training loss
Finite Agent NN	3.037×10^{-5}
Discrete State Space NN	9.639×10^{-5}
Projection NN	8.506×10^{-6}

  - Neural network solutions generate similar output to traditional methods.
  - Example plots: comparison to [Fernández-Villaverde et al., 2018]

Krusell-Smith: Sample Time Paths

More Plots



Krusell-Smith: Numerical Results

More Plots

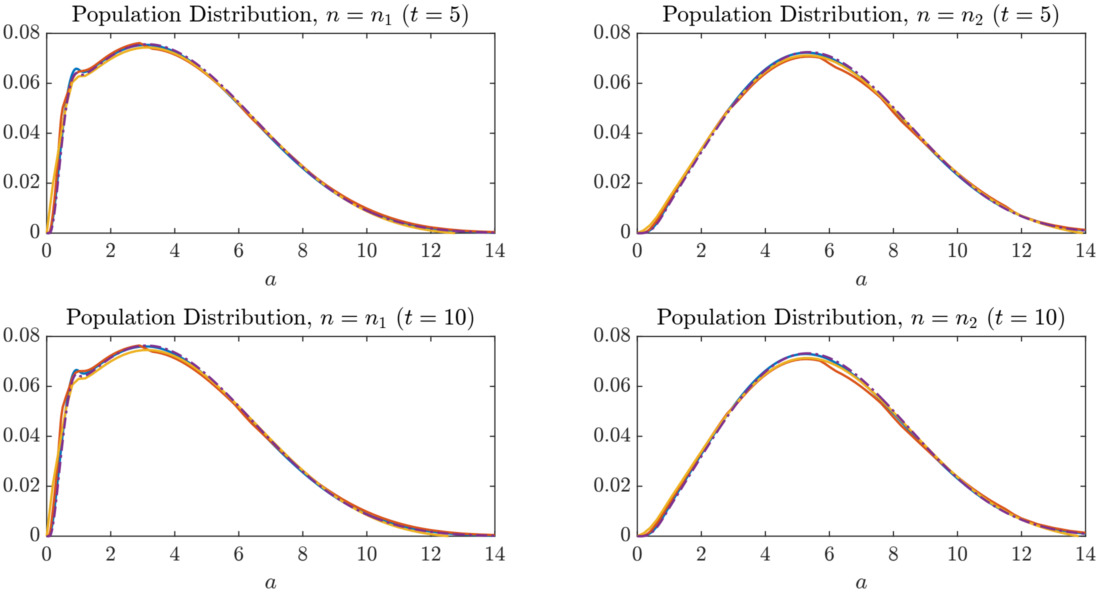


Table of Contents

Two Minor Extensions

1. Heterogeneous firm dynamics (in Gu-Lauriere-Merkel-Payne '24):
   - In KS-98, there is only one firm.
   - We extend our method to models with many firms that each have non-tradable capital and capital adjustment costs (a version of [Khan and Thomas, 2008])
   - Technical difference is that we need to price a firm equity.
2. Population movements (in Gu-Lauriere-Merkel-Payne '24):
   - In the KS-98 model, there was one wage in the entire economy
   - We extend our method to a "spatial" model where there are different wages in different locations and agents can move (a version of [Bilal, 2021]).
   - Technical difference is that the wage in location j only depends on the part of the population distribution at location j.

Two Major Extensions

1. Search and matching (SAM) models (in Payne-Rebei-Yang '24):
   - In KS-98, the mean of distribution entered the master equation through the prices.
   - In search and matching models, the shape of the distribution is more important

	Distribution	Distribution impact on decisions
HAM	Asset wealth and income	Via aggregate prices
SAM	Type (productivity) of agents in two sides of matching	Via matching probability with other types

2. Macro finance models with complicated asset pricing (Gopalakrishna-Gu-Payne '24):
   - In the KS-98 model, we can express (r, w) as closed form functions of the state.
   - For pricing long-term assets, the price process is only an implicit function of the state.
   - We also have to handle portfolio choice, which deep learning has found hard.

Table of Contents

Conclusion

1. Deep learning in macroeconomics is a rapidly evolving field.
2. Much progress but still many challenges.
3. Some comments/questions for the computer science literature:
    - Neural networks often seem “too flexible” for macroeconomic models. Other options?
    - How should we rewrite economic models to make them more deep learning “friendly”?
    - How should we dynamically choose weights in the loss function?
    - How do we get the a better stability vs speed tradeoff?

Thank You!

References I

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

Azinovic, M. and Žemlička, J. (2023).
Economics-inspired neural networks with stabilizing homotopies.
arXiv preprint arXiv:2303.14802.

Basak, S. and Cuoco, D. (1998).
An equilibrium model with restricted stock market participation.
The Review of Financial Studies, 11(2):309–341.

Bilal, A. (2021).
Solving heterogeneous agent models with the master equation.
Technical report, University of Chicago.

Brunnermeier, M. K. and Sannikov, Y. (2014).
A Macroeconomic Model with a Financial Sector*.
The American Economic Review, 104(1042):379–421.

Cardaliaguet, P., Delarue, F., Lasry, J.-M., and Lions, P.-L. (2015).
The master equation and the convergence problem in mean field games.
arXiv.

References II

Carmona, R. (2020).
Applications of mean field games in financial engineering and economic theory.
To appear in: Machine Learning and Data Sciences for Financial Markets, Cambridge University Press.

Duarte, V. (2018).
Machine learning for continuous-time economics.
Available at SSRN 3012602.

Fernández-Villaverde, J., Hurtado, S., and Nuño, G. (2018).
Financial Frictions and the Wealth Distribution.
Working Paper, pages 1–51.

Gopalakrishna, G. (2021).
Aliens and continuous time economies.
Swiss Finance Institute Research Paper, (21-34).

Gu, Z., Laurière, M., Merkel, S., and Payne, J. (2023).
Deep learning solutions to master equations for continuous time heterogeneous agent macroeconomic models.

Han, J., Yang, Y., and E, W. (2021).
DeepHAM: A global solution method for heterogeneous agent models with aggregate shocks.
arXiv preprint arXiv:2112.14377.

References III

Kahou, M. E., Fernández-Villaverde, J., Perla, J., and Sood, A. (2021).
Exploiting symmetry in high-dimensional dynamic programming.
Technical report, National Bureau of Economic Research.

Khan, A. and Thomas, J. K. (2008).
Idiosyncratic shocks and the role of nonconvexities in plant and aggregate investment dynamics.
Econometrica, 76(2):395–436.

Krusell, P. and Smith, A. A. (1998).
Income and Wealth Heterogeneity in the Macroeconomy.
Journal of Political Economy, 106(5):867–896.

Lucas, R. E. (1978).
Asset prices in an exchange economy.
Econometrica, 46(6):1429–1445.

Sirignano, J. and Spiliopoulos, K. (2018).
DGM: A deep learning algorithm for solving partial differential equations.
Journal of computational physics, 375:1339–1364.