Online Appendix: Krusell and Smith (1998) Model

Zhouzhou Gu[*], Mathieu Laurière[†], Sebastian Merkel[‡], Jonathan Payne[§¶]

August 15, 2023

1 Introduction

The main paper Gu et al. (2023) describes how to use deep learning to solve a generic class of continuous time, heterogeneous agent models. This supplementary appendix focuses on the Krusell and Smith (1998) model, which is a special case that is particularly important to the macroeconomics literature. We start by setting up "master equations" in detail for the model. We then represent the numerical results from the paper.

2 Model

2.1 Environment

Setting: The model is in continuous time with infinite horizon. There is a perishable consumption good and a durable capital stock. The economy contains a unit continuum of households and a representative firm.

Production: The representative firm controls the production technology, which produces consumption goods according to the production function:

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

where K_t is the capital hired at time t, L_t is the labour hired at time t, and z_t is aggregate

[*]Princeton, Department of Economics. Email: zg3990@princeton.edu
[†]NYU Shanghai, NYU-ECNU Institute of Mathematical Sciences. Email: mathieu.lauriere@nyu.edu
[‡]University of Exeter, Department of Economics. Email: s.merkel@exeter.ac.uk
[§]Princeton, Department of Economics. Email: jepayne@princeton.edu
[¶]We would like to thank Adam Rebei for outstanding research assistance throughout this project. We are also very grateful to the comments and discussion from Goutham Gopalakrishna and Yucheng Yang.

productivity, which evolves according to:

$$dz_t = \eta(\bar{z} - z_t)dt + \sigma dB_t^0, \tag{2.1}$$

with lower and upper reflecting boundaries at $\{z_{min}, z_{max}\}$ and where B_t^0 denotes an aggregate Brownian motion process. We let $\mathcal{F}_t^0$ denote the filtration generated by B_t^0.

Households: Each household $i \in [0,1]$ has discount rate ρ and gets flow utility $u(c_t^i) = (c_t^i)^{1-\gamma}/(1-\gamma)$ from consuming c_t^i consumption goods at time t. Households have an idiosyncratic labor endowment $n_t^i \in \{n_1, n_2\}$, where $n_1 < n_2$ so n_1 is interpreted as unemployment and n_2 is interpreted as employment. Labor endowments switch idiosyncratically between n_1 and n_2 at Poisson rate $\lambda(n_t^i)$.

Assets, markets, and financial frictions: Each period, there are competitive markets for goods, capital rental, and labor. We use goods as the numeraire. We let r_t denote the rental rate on capital, w_t denote the wage rate on labor, and $q_t = [r_t, w_t]$ denote the price vector. Asset markets are incomplete so households cannot insure their idiosyncratic labor shocks. Instead, households can trade claims to the aggregate capital stock in a competitive asset market.

The original Krusell and Smith (1998) model imposes the "borrowing constraint" that each agent's net asset position, a_t^i, must satisfy $a_t^i \geq \underline{a}$, where $\underline{a}$ is an exogenous "borrowing limit". This generates an inequality boundary constraint and mass point, as discussed in Achdou et al. (2022). However, this causes difficulties for the neural network. So, to make the problem more tractable, we instead follow Brzoza-Brzezina et al. (2015) and introduce a penalty function ψ at the left boundary, replacing the agent flow utility by:

$$U(a_t, c_t) = u(c_t) + \mathbf{1}_{a_t \leq \underline{a}} \psi(a_t)$$

The penalty function we use here is the quadratic: $\psi(a) = -\frac{1}{2}\kappa(a - \underline{a})^2$ where κ is a positive constant.

2.2 Equilibrium

Household problem: Each household has two idiosyncratic states: their net-worth a_t^i and their labor endowment n_t^i. The evolution of $x_t^i = [a_t^i, n_t^i]$, follows:

$$dx_t^i = d\begin{bmatrix} a_t^i \\ n_t^i \end{bmatrix} = \begin{bmatrix} s(c_t^i, a_t^i, n_t^i, r_t, w_t) \\ 0 \end{bmatrix} dt + \begin{bmatrix} 0 \\ \check{n}_t^i - n_t^i \end{bmatrix} dJ_t^i \tag{2.2}$$

where $\check{n}_t^i$ is the complement of n_t^i, J_t^i is a Poisson process with arrival rate $\lambda(n_t^i)$, and the agent's saving function is given by:

$$s(c, a, n, r, w) = wn + ra - c.$$

Where convenient (with abuse of notation) we write the saving function using the condensed notation $s(c, x, q)$.

Each agent, i, has a belief about the stochastic price process $\tilde{q} = \{\tilde{q}_t : t \geq 0\}$ adapted to $\mathcal{F}_t^0$. Given their belief, agent i chooses their consumption process, $c^i = \{c_t^i : t \geq 0\} \in \mathcal{C}(x, \tilde{q})$, to solve:

$$\begin{aligned} V(x_0^i, z_0) = \max_{c^i} \mathbb{E}_0 \left[\int_0^\infty e^{-\rho t} \left(u(c_t^i) + \mathbf{1}_{a_t \leq \underline{a}} \psi(a_t) \right) dt \right] \\ s.t. \quad (2.1),\ (2.2), \end{aligned} \tag{2.3}$$

where $\mathcal{C}(x, \tilde{q})$ is the set of admissible controls.

Firm problem: Firm optimization implies the following first order conditions for firm demand for renting capital and labor:

$$r_t = \partial_K F(K_t, L) - \delta, \qquad\qquad w_t = \partial_L F(K_t, L), \tag{2.4}$$

Distributions: The incomplete markets mean that idiosyncratic shocks potentially generate a non-degenerate cross sectional distribution of agent states. We let $G_t = \mathcal{L}(x_t^i | \mathcal{F}_t^0)$ and g_t denote the population distribution and density across x_t^i at time t, for a given history $\mathcal{F}_t^0$.

Equilibrium: Given an initial density g_0, an equilibrium for this economy consists of a collection of $\mathcal{F}_t^0$-adapted stochastic processes, $\{c_t^i, n_t^i, g_t, q_t, z_t, K_t : t \geq 0, i \in I\}$, that satisfy the following conditions: (i) each household's control process c_t^i solves problem (2.3) given their belief that the price process is $\tilde{q}$, (ii) firm demand for capital and labor satisfy the first order conditions (2.4), (iii) markets clear:

$$K_t = \sum_{j \in \{1,2\}} \int_{\underline{a}}^\infty a g_t(a, y_j) da, \qquad\qquad L = \sum_{j \in \{1,2\}} \int_{\underline{a}}^\infty n_j g_t(a, y_j) da,$$

and (iv) agent beliefs about the price process are consistent with the optimal behaviour of other agents in the sense that $\tilde{q} = q$.

2.3 Recursive Characterization of Equilibrium

States: We assume that there exists an equilibrium that is recursive in the aggregate state variables: $\{z, g\}$. Observe that we can express the price vector q explicitly in terms of $\{z, g\}$

by combining the firm optimization conditions and the market clearing conditions:

$$q = \begin{bmatrix} r_t \\ w_t \end{bmatrix} = \begin{bmatrix} e^{z_t}\partial_K F\left(\sum_{j\in\{1,2\}}\int_{\underline{a}}^{\infty} a g_t(a,n_j)da, L\right) - \delta \\ e^{z_t}\partial_L F\left(\sum_{j\in\{1,2\}}\int_{\underline{a}}^{\infty} a g_t(a,n_j)da, L\right) \end{bmatrix} =: Q(z,g) \tag{2.5}$$

A belief about the evolution of the distribution, $dg_t(x) = \tilde{\mu}^g(z_t, g_t)dt$ implies a belief about the evolution of prices through $q = Q(z_t, g_t)$ so beliefs about the price process can be characterized by beliefs about the evolution of the distribution.

Hamilton Jacobi Bellman Equation (HJBE): Given their beliefs, for each $x = [a, n]$, each household chooses c to solve the HJBE:

$$\begin{aligned} 0 = \max_{c\in\mathcal{C}(x,z,g)} \Big\{ & -\rho V(x,z,g) + u(c) + \mathbf{1}_{a\le\underline{a}}\psi(a) \\ & + \partial_a V(x,z,g)s(c,x,Q(z,g)) + \lambda(n)(V(\tilde{x},z,g) - V(x,z,g)) \\ & + \partial_z V(x,z,g)\eta(\bar{z} - z_t) + \frac{1}{2}\sigma^2\partial_{zz}V(x,z,g) \\ & + \int_{\mathcal{X}} \tilde{\mu}^g(z_t,g_t)\frac{\partial V}{\partial g}(x,z,g)(y)dy \Big\} \end{aligned}$$

where $V(x,z,g)$ is the value function of the household, $\tilde{x} = [a,\tilde{n}]$ is household state after the change from n to $\tilde{n}$, and $\partial V/\partial g$ is the Frechet derivative of V with respect to the distribution.[1] From the HJBE, the optimal consumption c^* can be computed for every (x,z,g), which allows a representative player to react optimally to any population distribution. The optimal control, c^*, is characterised by the first order condition:

$$\partial_a V(x,z,g) = u'(c^*(x,z,g)).$$

Kolmogorov Forward Equation (KFE): Denote the recursive equilibrium optimal control of the individual households by $c^*(x_t, z_t, g_t; \tilde{\mu}_g)$ for belief $\tilde{\mu}_g$. Then, for a given z path, the evolution of the distribution under the optimal control c^* can be characterized by the Kolmogorov Forward Equation (KFE):[2]

$$\begin{aligned} dg_t(x) &= \mu^g(c^*(x_t,z_t,g_t;\tilde{\mu}^g), x_t, z_t, g_t)dt, \quad \text{where} \\ \mu^g(c_t^*, x_t, z_t, g_t) &:= -\partial_a[s(c_t^*, x_t, Q(z_t,g_t))g_t(x)] - \lambda(n)g(x) + \lambda_{\tilde{n}}g(\tilde{x}) \end{aligned}$$

Under this recursive characterization, the belief consistency condition becomes that $\mu^g = \tilde{\mu}^g$.

[1] There are technical difficulties with defining the "distributional" derivatives for mean field games, as discussed in Cardaliaguet et al. (2015). However, we do not engage with these difficulties because in all numerical applications we are going to discretize the population density.

[2] Observe that there is no noise in the KFE because dB_t^0 does not directly impact the evolution of idiosyncratic states.

Master Equation: We follow the approach of Lions (2011) and characterize the equilibrium in one PDE, which is often referred to as the "master equation" of the "mean-field-game". Conceptually, the master equation is derived by imposing belief consistency and substituting the equilibrium KFE into the HJBE. In equilibrium, the $V(x, z, g)$ solves the following PDE:

$$0 = (\mathcal{L}V)(x, z, g) := (\mathcal{L}^h V)(x, z, g) + (\mathcal{L}^g V)(x, z, g) \tag{2.6}$$

where the operators $\mathcal{L}^h$ and $\mathcal{L}^g$ are defined by:

$$\begin{aligned}
(\mathcal{L}^h V)(x, z, g) :=& -\rho V(x, z, g) + u(c^*(x, z, g)) + \mathbf{1}_{a \le \underline{a}} \psi(a) \\
&+ \partial_a V(x, z, g) s(c^*(x, z, g), x, Q(z, g)) + \lambda(n)(V(\tilde{x}, z, g) - V(x, z, g)) \\
&+ \partial_Z V(x, z, g) \eta(\overline{z} - z) + \frac{1}{2}\sigma^2 \partial_{ZZ} V(x, z, g) \\
(\mathcal{L}^g V)(x, Z, g) :=& \int_{\overline{a}}^{\infty} \partial_b \frac{\partial V}{\partial g}(x, z, g)(b) \times s\left(x, c^*(x, Z, g), Q(z, g)\right) g(b, n) db \\
&+ \int_{\overline{a}}^{\infty} \frac{\partial V}{\partial g}(x, z, g)(b) \times \left(\lambda(\tilde{n}) g(b, \tilde{n}) - \lambda(n) g(b, n)\right) db
\end{aligned}$$

where c^* satisfies (2.3) and $Q(z, g)$ satisfies (3.1). In this notation, $\mathcal{L}^h$ reflects the optimization problem of the household and $\mathcal{L}^g$ reflects how the evolution of the distribution affects the household value.

Intuitively, $V(x, z, g)$ can be interpreted as the optimal value of a representative player who starts at state x, with aggregate shock equal to z, and who faces a population that starts at the distribution g and then plays according to the Nash equilibrium control c. We refer to Cardaliaguet et al. (2015); Bensoussan et al. (2015) for more details. The goal of this paper is use deep learning techniques to find numerical solutions to equation (2.6). The challenge is that the master equation contains an infinite dimensional derivative with respect to the distribution g. We need to work with numerical approximations of the distribution and solution methods that can handle high dimensions.

3 Solution Approach

In this section, we outline how to apply the "Deep Galerkin" approach to solving the master equation (2.6). The first part of this approach is to find a finite dimensional approximation to the distribution so we can develop a finite, but high, dimensional approximation to the master equation. The second part is to approximate the solution to the finite dimensional master equation using a neural network. Finally, we use "deep-learning" to solve the approximate master equation.

3.1 Finite Dimensional Master Equation

In the main paper, we discuss different ways of approximating the distribution. Here, we only consider approximating the economy by an environment with a large, finite number of agents $I < \infty$. In this case, the density, g_t, is replaced by the individual states of the I agents, which we denote by $\hat{g}_t$:

$$\hat{g}_t := \{x_t^i : i \le I\}.$$

The market clearing conditions now become (with some abuse of notation) $q_t = \hat{Q}(z_t, \hat{g}_t)$ where:

$$q = \begin{bmatrix} r_t \\ w_t \end{bmatrix} = \begin{bmatrix} e^{z_t} \partial_K F\left(\sum_i a_i, L\right) - \delta \\ e^{z_t} \partial_L F\left(\sum_i a_i, L\right) \end{bmatrix} =: \hat{Q}(z, g) \tag{3.1}$$

However, to maintain the price taking assumption in the finite agent model, we impose that agent i behaves as if their individual actions do not influence prices. Formally, this means that agent i perceives the pricing function to be:

$$q_t = \hat{Q}(z_t, \hat{g}_t^{-i})$$

where $\hat{g}_t^{-i} = \{x_t^j \in I^{-i}\}$ is the position of the other agents $I^{-i} := \{j \le I : j \neq i\}$. Ultimately, this will ensure that the neural network trains the policies rules as if the agents believe that their assets do not influence the market prices. Aside from this change to the belief process, the optimization problem for household remains the same.

Let $c^*(x^i, z, \hat{g})$ denote the equilibrium optimal control . Let $V(x^i, z, \hat{g})$ denote the value function for the master equation in the economy with I price taking agents. Then $V(x^i, z, \hat{g})$ solves $(\hat{\mathcal{L}}V)(x^i, z, \hat{g}) = 0$ subject to the boundary conditions, where the master equation operator is:

$$\begin{aligned} (\hat{\mathcal{L}}V) &= (\hat{\mathcal{L}}^h V) + (\hat{\mathcal{L}}^g V), \quad \text{where} \\ (\hat{\mathcal{L}}^h V)(x^i, z, \hat{g}) &:= (\mathcal{L}^h V)(x^i, z, \hat{g}) \\ (\hat{\mathcal{L}}^g V)(x^i, z, \hat{g}) &= \sum_{j \neq i} \frac{\partial V}{\partial x^j}(x^i, z, \hat{g}) s(c^*(x^j, z, \hat{g}), x^j, z, \hat{Q}(z, \hat{g}^{-j})) \\ &\quad + \sum_{j \neq i} \lambda(x^j) \left(V(x^i, z, (\tilde{x}^j, \hat{g}^{-ij})\}) - V(x^i, z, \hat{g}^{-i}) \right), \end{aligned}$$

The operator for the household optimization problem, $\hat{\mathcal{L}}^h$, is the same as in the general problem but with the distribution replaced by the finite collection of agents for the calculating the market clearing conditions (and some abuse of notation). The operator for the impact of distributional changes on the household, $\hat{\mathcal{L}}^g$, become finite dimensional because the economy only needs to track the evolution of a finite number of agents.

3.2 Neural Network Approximations

Section 3.1 derived finite approximations to the density, $\hat{g}$, and the master equation operator $\hat{\mathcal{L}}$. However, the resulting master equations are high dimensional and so cannot be solved by traditional techniques. Instead, we approximate the solution to the master equation using a neural network and deploy tools from the "deep learning" literature to "train" the neural network to solve the approximate master equation.

A neural network is a type of parametric functional approximation that is built by composing affine and non-linear functions in a chain or "network" structure (see Goodfellow et al. (2016) for a detailed discussion). We let $\hat{X} := \{x, z, \hat{g}\}$ denote the collection of inputs into the approximate value function. We denote the neural network approximation to the value function by $V(\hat{X}) \approx \hat{V}(\hat{X}; \theta)$, where θ are the parameters in the neural network approximation that depend upon the form of the approximation. There are many types of neural network approximations. The simplest form is a "feedforward" or "deep feedforward" neural network which is defined by:

$$
\begin{aligned}
h^{(1)} &= \phi^{(1)}(W^{(1)}\hat{X} + b^{(1)}) && \ldots \text{Hidden layer 1} \\
h^{(2)} &= \phi^{(2)}(W^{(2)}h^{(1)} + b^{(2)}) && \ldots \text{Hidden layer 2} \\
&\vdots \\
h^{(H)} &= \phi^{(H)}(W^{(H)}h^{(H-1)} + b^{(H)}) && \ldots \text{Hidden layer H} \\
o &= W^{(H+1)}h^{(H)} + b^{(H+1)} && \ldots \text{Output layer} \\
\hat{V} &= \phi^{H+1}(o) && \ldots \text{Output}
\end{aligned}
\tag{3.2}
$$

where the $\{h^{(i)}\}_{i \le H}$ are vectors referred to as "hidden layers" in the neural network, $\{W^{(i)}\}_{i \le (H+1)}$ are matrices referred to as the "weights" in each layer, $\{b^{(i)}\}_{i \le (H+1)}$ are vectors referred to as the "biases" in each layer, $\{\phi^{(i)}\}_{i \le (H+1)}$ are non-linear functions applied element-wise to each affine transformation and referred to as "activation functions" for each layer. The length of hidden layer, $h^{(i)}$, is referred to as the number of *neurons* in hidden layer i. The total collection of parameters is denoted by $\theta = \{W^{(i)}, b^{(i)}\}_{i \le (H+1)}$. The goal of deep learning is to train the parameters, θ, to make $\hat{V}(\hat{X}; \theta)$ a close approximation to $V(\hat{X})$.

The neural network defined in (3.2) is called a "feedforward" because hidden layer i cannot depend on hidden layers $j > i$. This is in contrast to a "recursive" neural networks where any hidden layer can be a function of any other hidden layer. It is called "fully connected" if all the entries in the weight matrices can be non-zero so each layer can use all the entries in the previous layer. In this paper, we will consider a fully connected "feedforward" network to be the default network. This is because these networks are the quickest to train and so we typically start by trying out this approach. However, there are applications where we find that more complicated neural network formulations are useful. In particular, we find that the type of recursive neural network suggested by the "Deep Galerkin" approach in Sirignano and Spiliopoulos (2018) is helpful for finite state space approximations.

3.3 Solution Algorithm

We train the neural network to learn parameters θ that minimize the error in the master equation and boundary conditions. We describe the key steps in in Algorithm 1. Essentially, the algorithm generates random points in the discretized states space $\{x, z, \hat{g}\}$, then calculates the error in the master equation on those points, and updates the parameters to decrease the error in the master equation. In the deep learning literature, this approach is sometimes referred to as "unsupervised" learning (e.g. Azinovic et al. (2022)) because we do not have direct observations of the value function, $V(x, z, \hat{g})$, and instead have to learn the value function indirectly via the master equation.

Algorithm 1: Solution Algorithm

1. Approximate the value function by a neural network: $V(x, z, \hat{g}) \approx \hat{V}(x, z, \hat{g}; \theta)$, where θ are the neural network parameters for the value function
2. Make initial parameter guess θ^0.
3. At iteration n with guess θ^n:
   (a) Generate M sample points, $S = \{(x_m, z_m, \hat{g}_m)\}_{m \le M}$ for evaluating the master equation error.
   (b) Calculate the average error in the master equation for the sample:
   $$\mathcal{E}(\theta^n, S) := \frac{1}{M} \sum_{m \le M} |\hat{\mathcal{L}}(x_m, z_m, \hat{g}_m)|^2$$
   where the derivatives in the differential operator are calculated using automatic differentiation.
   (c) Update the the parameters using "deep learning" toolkit. We typically use a "stochastic gradient descent" style method: at each point:
   $$\theta^{n+1} = \theta^n - \alpha_n D_\theta \mathcal{E}(\theta^n, S^n)$$
   where α_n is the "learning rate" and $D_\theta \mathcal{E}$ is the vector differential operator.
   (d) Repeat until $\mathcal{E}(\theta^n, S^n) \le \epsilon$ where ϵ is a precision threshold.

4 Implementation

We solve the model for the parameters outlined in table 2. The precise details of the algorithm, sampling, and neural network specification are outlined in the main paper Gu et al. (2023).

The error in the master equation is shown in Table 1 below. We don't have a clear benchmark for Krusell-Smith model because there is no existing technique that provides an accurate solution to the model with aggregate shocks. However, we can compare to widely used approximation techniques in the literature. In particuar, we compare the approach suggested by Fernández-Villaverde et al. (2018), which uses a neural network to approximate a statistical law of motion rather than developing the fully global solution. We compare to Fernández-Villaverde et al. (2018) by computing sample paths from both solution approaches. Essentially, we draw a series of productivity shocks from the Ornstein-Uhlenbeck process: $dz_t = \eta(\bar{z} - z)dt + \sigma dW_t$ and then evolve the population distribution.

	Master equation loss
Finite Agent NN	3.037×10^{-5}

Table 1: Neural Nets' results for solving Master Equations with aggregate shocks.

Figure 1 shows the comparison between our neural network solution and Fernández-Villaverde et al. (2018) for a particular path of productivity shocks. The upper-left panel shows the draw from the Ornstein-Uhlenbeck process: $dz_t = \eta(\bar{z} - z)dt + \sigma dB_t$. The upper left compares the evolution of capital stock. The middle plots compares the evolution of prices. The bottom plots compare the evolution of the population. As can be seen in figure 1, we get a similar path for aggregate capital stock, interest rates, wage rates, and the population distribution.

In figure 2, we generate multiple random paths for TFP, z_t, and show the evolution of our Neural Network solution and solution in Fernández-Villaverde et al. (2018) in a "fan chart" that displays percentiles for the evolution of the population. In particular, we generate 1,000 TFP paths starting from $z_0 = 0$ and calculate the corresponding aggregate capital evolution paths. We collect capital at different time t, sort to get the pth-quantile and plot the time series of the quantiles.

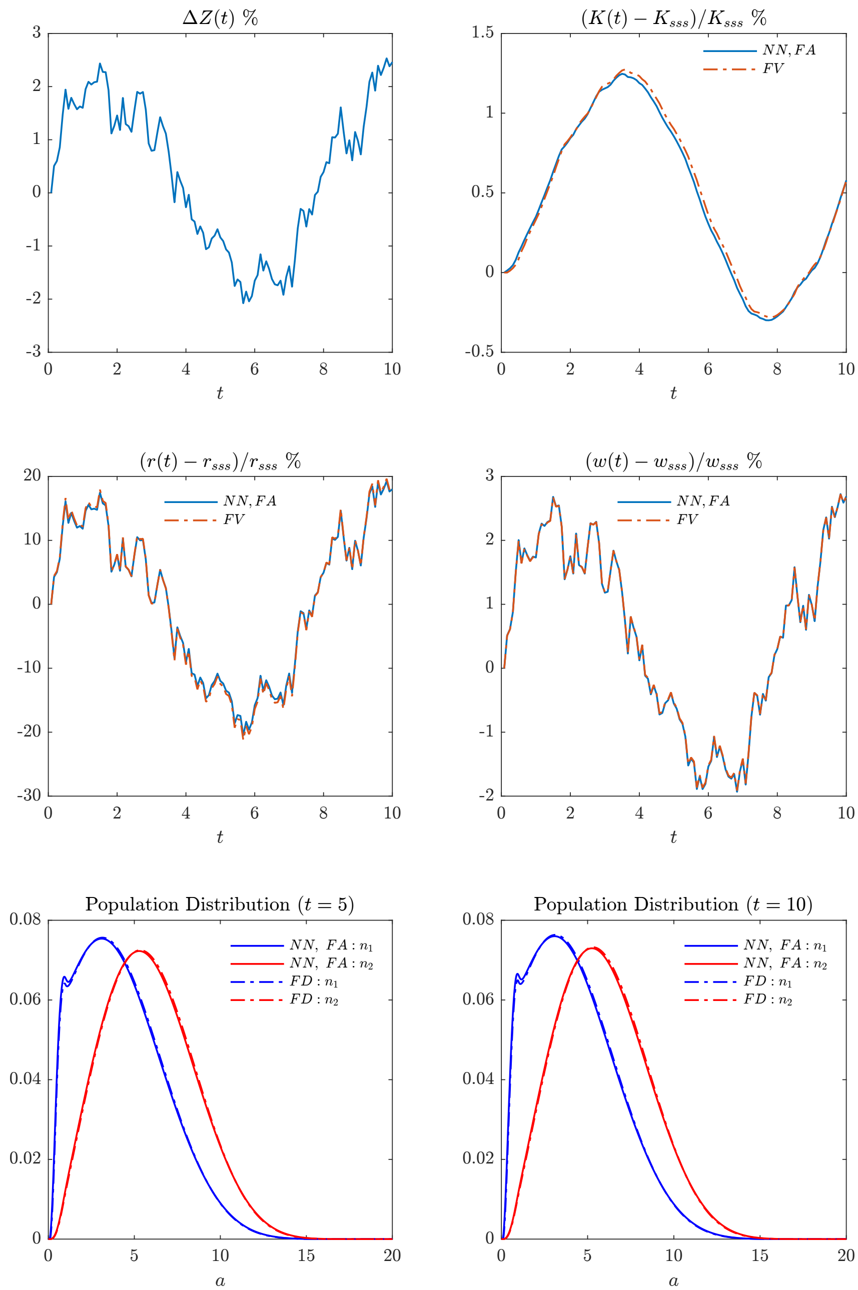


Figure 1: Impulse response functions for Krusell-Smith Model. The top left plot is the TFP shock path, the top right panel is the aggregate relative capital change, the middle left panel plots the relative average consumption change, and the middle right panel plots the relative capital return change. The bottom left is relative wage change, and the bottom right is the relative wealth change at different quantiles. *NN, FA* referes to the finite agent neural network and *FV* refers to the result generated from Fernández-Villaverde et al. (2018). Subscript *sss* refers to the stochastic steady state at $Z = 0$.

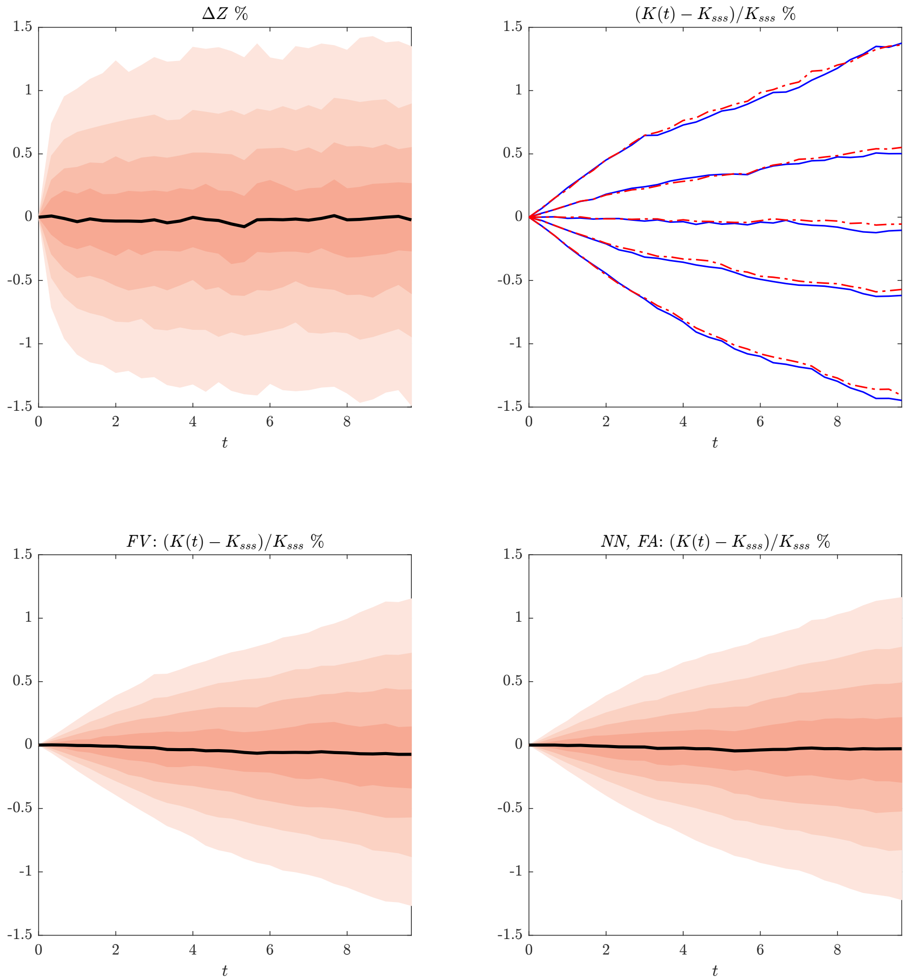


Figure 2: Forcasted aggregate capital dynamics starting from the stochastic steady state (sss) for the Krusell-Smith Model. The top left plot is the fan chart for the TFP shock path, generated from OU process with initial condition $Z_0 = 0$. The bottom left panel and right panel are fan charts (capital quantile) of corresponding responses. The top right panel is the time series plot for relative change in aggregate capital at quantile 10%, 30%, 50%, 70%, 90% (from the lowest to the highest), in which the blue solid lines are generated by neural network solution and the red dashed lines are generated by Fernández-Villaverde et al. (2018). *NN, FA* referes to the finite agent technique, and *FV* refers to Fernández-Villaverde et al. (2018)'s technique.

References

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., Gaegauf, L., and Scheidegger, S. (2022). Deep equilibrium nets. *International Economic Review*, 63(4):1471–1525.

Bensoussan, A., Frehse, J., and Yam, S. C. P. (2015). The master equation in mean field theory. *Journal de Mathématiques Pures et Appliquées*, 103(6):1441–1474.

Brzoza-Brzezina, M., Kolasa, M., and Makarski, K. (2015). A penalty function approach to occasionally binding credit constraints. *Economic Modelling*, 51:315–327.

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

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

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. MIT press.

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

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

Lions, P.-L. (2007-2011). Lectures at College de France.

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

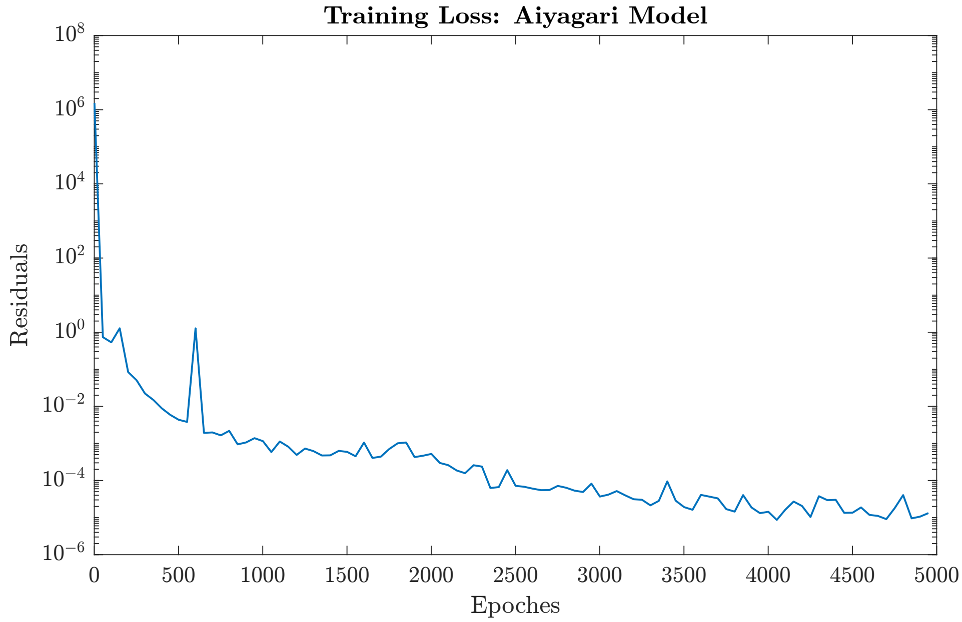
A Krusell-Smith Model

A.1 Parameters for Krusell-Smith Model

Parameter	Symbol	Value
Capital share	α	1/3
Depreciation	δ	0.1
Risk aversion	γ	2.1
Discount rate	ρ	0.05
Mean TFP	$\overline{Z}$	0.00
Reversion rate	η	0.50
Volatility of TFP	σ	0.01
Transition rate (1 to 2)	λ_1	0.4
Transition rate (2 to 1)	λ_2	0.4
Low labor productivity	n_1	0.3
High labor productivity	n_2	$1 + \lambda_2/\lambda_1(1 - n_1)$
Borrowing constraint	$\underline{a}$	10^{-6}
Maximum of asset	$\overline{a}$	20.0
Penalty Function	$\psi(a)$	$-\frac{1}{2}\kappa(a - a_{lb})^2$
Penalty parameters	a_{lb}	1.0
Penalty parameters	κ	3.0
Drift in O-U Process	η	0.5
Volatility in O-U Process	σ	0.01
Maximum TFP	Z_{max}	0.04
Minimum TFP	Z_{min}	-0.04

Table 2: Parameters.

B Additional Plots



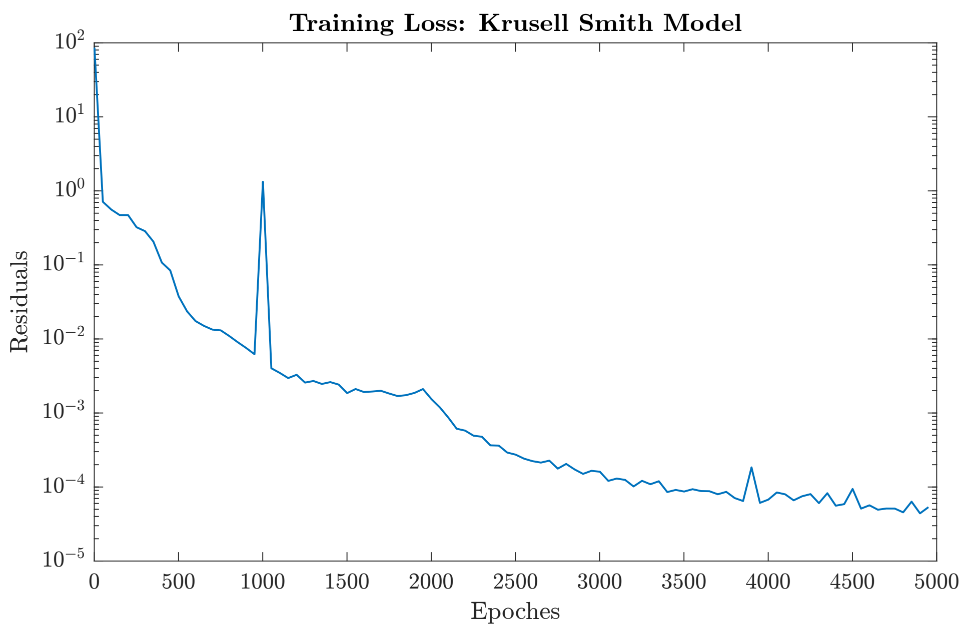


Figure 3: Training Loss vs Interation Plots (Finite Agent Method)