Operator Learning in Macroeconomics

Yaolang Zhong

University of Tokyo

The CREPE Conference Day

December 25, 2024

Background

Since around 2020, the Computational Economics Literature...

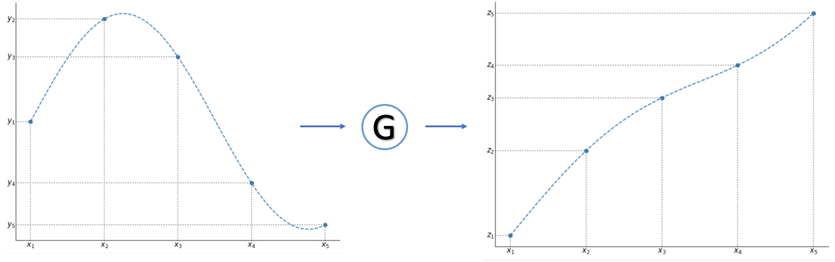
- Economics + Computer Science + Engineering (programming)
- Thanks to the power of neural network in overcoming the curse of dimensionality:
  - Feature Representation and Hierarchical Learning
  - Non-Linear Approximation
  - Shared Parameters and Regularization
  - Efficient Optimization

Overview

- This paper proposed a new numerical framework to solve a prevalent class of structural models: the heterogeneous agent (HA) models with aggregate shocks
  - In this context, the cross-sectional distribution of all individual states, which is an infinite-dimensional object, becomes part of the agents' state variable
- My approach demonstrated computational efficiency in experiments on a Bewley-Huggett-Aiyagari type model (Den Haan, Judd and Juillard, 2008), compared to alternatives in the current literature (Maliar, Maliar and Winant, 2021) ▸ Figure
- Three parts:
  - Reformulation of the problem of solving the model into learning an operator
  - Parameterization of the operator by the neural operator (Li et al., 2020)
  - Implementation by a specific training scheme

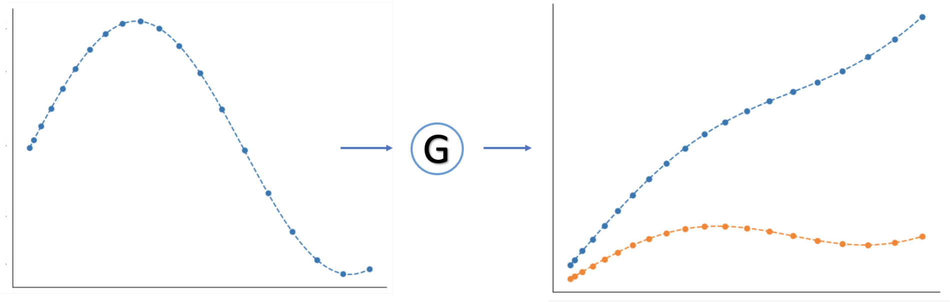
Introduction: Operator

- An operator $\mathbf{G}$ is a mapping between function spaces
- Examples: $\mathbf{h}(x) = \mathbf{G}(\mathbf{f})(x) = \frac{\mathrm{d}\mathbf{f}}{\mathrm{d}x}(x)$ and $\mathbf{h}(x) = \mathbf{G}(\mathbf{f})(x) = \int \mathbf{f}(x)\mathrm{d}x$
- $\mathbf{f} : (\{x_1, ..., x_J\}, \{y_1, ..., y_J\})$ and $\mathbf{h} : (\{x_1, ..., x_J\}, \{z_1, ..., z_J\})$
- We call $\{x_1, ..., x_J\}$ the "sensors"



Introduction: Operator (cont.)

- Increasing J-grid gives higher approximation accuracy
- The J-grid is not necessarily uniform
- We can have $\mathbf{G}(\mathbf{f}) = (\mathbf{h_1}, \mathbf{h_2})$



The Bewley-Huggett-Aiyagari Model

- lowercase letters for individual variables, UPPERCASE letters for aggregate variables and **bold** letters for functions and operators
- A continuum of infinitely lived and ex-ante identical agents, each period:
  - the time endowment $\bar{l}$
  - earn the after-tax wage $(1-\tau_t)\bar{l}W_t$ if employed $(\epsilon = 1)$
  - earn the unemployment benefit μW_t if unemployed $(\epsilon = 0)$
  - W_t is the per unit of time wage rate, τ_t is the tax rate, and μ is a model parameter denoting the fraction of wage
- Market is incomplete: non-zero capital holding $k_t \geqslant 0$
- The net rate of return for capital: $R_t - \delta$, R_t is market-determining interest rate and δ is the fixed depreciation rate

The Bewley-Huggett-Aiyagari Model

- Firms: Cobb-Douglas production function $Y_t = Z_t K_t^{\alpha} \left(\bar{l} L_t\right)^{1-\alpha}$
- K_t is the per capita capital, L_t is the employment rate, and $\alpha \in [0, 1]$ is the capital sharing. Z_t is a binary aggregate productivity shock: $Z_t \in \{Z_b, Z_g\}$
- Government: keep budget balanced by redistributing all taxation
- Firms' first-order optimality + Government's budget constraint:

$$R_t = \alpha Z_t \left(\frac{K_t}{\bar{l} L_t}\right)^{\alpha-1}, \quad W_t = (1-\alpha) Z_t \left(\frac{K_t}{\bar{l} L_t}\right)^{\alpha}, \quad \tau_t = \frac{\mu(1-L_t)}{\bar{l} L_t} \tag{1}$$

- Shocks: Z_t is first-order Markovian, ϵ_t is first-order Markovian conditional on the transition of Z_t, and confront to the law of the large number
- $(\epsilon_t, Z_t) \sim \mathbf{\Pi}$: the element $\pi_{\epsilon\epsilon' Z Z'}$ denotes $\mathrm{P}((\epsilon, Z) \to (\epsilon', Z'))$

The Bewley-Huggett-Aiyagari Model

- Denote $\mathbf{\Gamma}$ the distribution of agents over capital holdings
- Denote the law of motion of $\mathbf{\Gamma}$ by $\mathbf{H} : \mathbf{\Gamma}' = \mathbf{H}(\mathbf{\Gamma}, Z, Z')$
- The agents' problem can be therefore express recursively as

$$\mathbf{V}(k, \epsilon; Z, \mathbf{\Gamma}) = \max_{c,k'} \{\mathbf{U}(c) + \beta \mathbb{E}[\mathbf{V}(k', \epsilon'; Z, \mathbf{\Gamma}') \mid \epsilon, Z]\} \tag{2}$$

subject to

$$\begin{aligned} c + k' &= Rk + [(1-\tau)\bar{l}\epsilon + \mu(1-\epsilon)]W + (1-\delta)k, & (3) \\ \epsilon', Z' &\sim \mathbf{\Pi}(\epsilon, Z), & (4) \\ \mathbf{\Gamma}' &= \mathbf{H}(\mathbf{\Gamma}, Z, Z'), & (5) \\ k' &\geqslant 0 & (6) \end{aligned}$$

- Denote the solution to (2) subject to (3), (4), (5), (6) $\mathbf{V}^*(\cdot)$ and corresponding policy function $\mathbf{g}^*(\cdot)$

Technical Remarks

- In principle working on either $\mathbf{V}^*(\cdot)$ or $\mathbf{g}^*(\cdot)$ is fine:

$$\mathbf{g}^*(s) = \arg\max_{a \in \mathcal{A}} \left[R(s,a) + \beta \mathbb{E}_{s'} \mathbf{V}^*(s') \right],$$

$$\mathbf{V}^*(s) = \sum_{t=0}^{\infty} \beta^t R(s_t, \mathbf{g}(s_t))$$

- Solving for $\mathbf{V}^*$: value function iteration (VFI)

$$V(s) = \max_{a \in \mathcal{A}(s)} \left\{ u(s,a) + \beta \mathbb{E}\left[V(s') \mid s, a \right] \right\}$$

- Solving for $\mathbf{g}^*$: time iteration

$$u'(c_t) = \beta \mathbb{E}\left[u'(c_{t+1}) f'(k_{t+1}) \right]$$

where c is implicitly represented by $\mathbf{g}$

Technical Remarks

In practice, prefer $\mathbf{g}^*$ over $\mathbf{V}^*$

- Ultimately we are interested in $\mathbf{g}^*$
- There is a maximization operation in Bellman equation that is hard to deal with
- Approximation error is more informative in the case of $\mathbf{g}$
- ......

Comparing the Computational Strategies

- The goal is to solve for the optimal policy function $\mathbf{g}(k, \epsilon; Z, \mathbf{\Gamma})$ with $\mathbf{\Gamma}' = \mathbf{H}(\mathbf{\Gamma}, Z, Z')$
- Krusell-Smith (KS) Framework: $\mathbf{g}(k, \epsilon; Z, m)$, m is a set of moments of $\mathbf{\Gamma}$
  - Lack: Full Information of Distribution - how to approximate $\mathbf{H}$?
- Neural Network (NN) Framework: $\mathbf{g}_{\text{NN}}(k, \epsilon; Z, k^N) \approx \mathbf{g}(k, \epsilon; Z, k^N)$
  - Lack: Discretization-Invariance - N determines the parameterization Figure
  - Lack: Permutation-Invariance - $\mathbf{g}_{\text{NN}}(k, \epsilon; Z, k^N) = \mathbf{g}_{\text{NN}}(k, \epsilon; Z, \hat{k}^N)$ (Han and Yang, 2021)
  - Lack: Sharing-Aggregation - $k_i' = \mathbf{g}_{\text{NN}}(k_i, \epsilon; Z, k^N)$ for $i = 1, ..., N$ Figure

Comparing the Computational Strategies

Table 1: Comparison of Three Numerical Frameworks for the Desirable Properties

Property	Framework		
	KS[1]	NN[2]	Operator[3]
Full Information of Distribution	×	✓	✓
Discretization-Invariance	✓	×	✓
Permutation-Invariance	✓	×	✓
Sharing-Aggregation	✓	×	✓

[1] Krusell-Smith

[2] Deep Learning with feed-forward neural network

[3] Deep Learning with neural operator (This Paper)

Reformulation into Operator

- Goal is $\mathbf{g}(k, \epsilon; Z, \mathbf{\Gamma})$
- Permutation-Invariance: $\mathbf{g}(k, \epsilon; Z, \tilde{\mathbf{\Gamma}})$ with k^N
  - use the empirical cumulative distribution function (ECDF) $\tilde{\mathbf{\Gamma}} \in \mathcal{T} : [k_{\text{min}}, k_{\text{max}}] \to [0, 1]$ to characterize $k^N = (k_1, ..., k_N)$
  - $\tilde{\mathbf{\Gamma}}$ is represented by the interpolation of the tuple $(\tilde{k}_1, ..., \tilde{k}_N), (\frac{1}{N}, ..., \frac{N}{N})\}$, where $(\tilde{k}_1, ..., \tilde{k}_N)$ is in ascending order
  - In practice, a large N but a small number of sensors J
- Sharing-Aggregation: $\mathbf{g}(k, \epsilon; Z, \tilde{\mathbf{\Gamma}}) = \mathbf{G}(\tilde{\mathbf{\Gamma}})(k, \epsilon, Z)$
  - $\mathbf{G} : \mathcal{T} \to \mathcal{H}$ such that $\mathbf{h}_{\tilde{\mathbf{\Gamma}}} = \mathbf{G}(\tilde{\mathbf{\Gamma}})$ is the "conditional policy function"
  - Process the high-dimensional part $\tilde{\mathbf{\Gamma}}$ once-for-all Figure

Parameterization of the Operator

- Parameterize the operator $\mathbf{G}$ by the neural operator θ
- In the case of $\mathbf{g}_{\mathsf{NN}}(k, \epsilon, Z, k^N)$, the neural network approximates the operation in vector space through a series of matrix multiplications
- In the case of $\mathbf{G}_\theta(\boldsymbol{\Gamma})$, the neural operator approximates the operation in function space through a series of convolutions (*Universal Approximation Theorem for Operator*)
- Fourier Neural Operator: transforms the input function into the Fourier domain, imposes a series of matrix multiplications, and then returns the output to the spatial domain (*The Convolution Theorem*) Figure
- Discretization-Invariance: Parameterization in the Fourier domain is independent of the discretization of the input and output function in spatial domain (choice of N)

Implementation

- A version of Time Iteration
- Semi-stochastic simulation: grids on k and simulation for the ergodic set $\mathbf{\Gamma}$ (Judd et al., 2011)
- Initialization:
  - Solve for $\mathbf{g}_{\text{static}}$ in the model without aggregated shock (Aiyagari, 1994)
  - Supervised Learning: $\mathbf{g}_\theta \approx \mathbf{g}_{\text{static}}$ (Transfer Learning)
  - Updating $\mathbf{g}_\theta$ using the ergodic set generated by $\mathbf{g}_{\text{static}}$ (Off-policy Learning)
- Fine-Tuning: not deliberately

Conclusion

- This paper introduces a novel numerical framework for solving the heterogeneous agent model
- The framework achieves computational efficiency by leveraging three key properties: Discretization-Invariance, Permutation-Invariance, and Sharing-Aggregation
- It offers a fresh perspective on handling the distribution function numerically

Figure

Return

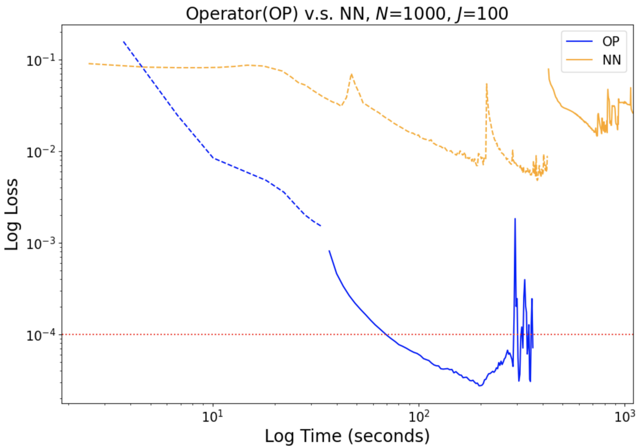


Figure 1: Training Loss vs. Time (seconds). My approach (blue) reached the 1% error (square root of loss) in around 5 mins and the alterative approach (yellow) took more than 20 mins

Figure

Figure

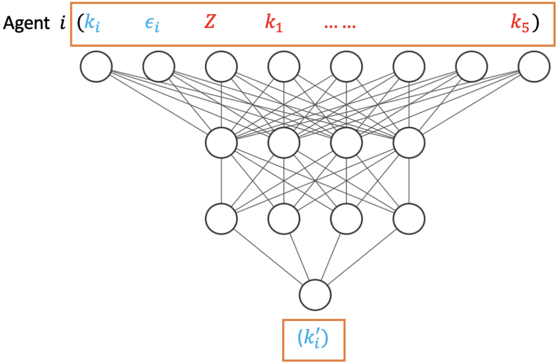


Figure 2: An example Neural Network in the case of $N = 5$ agents

Figure

Return

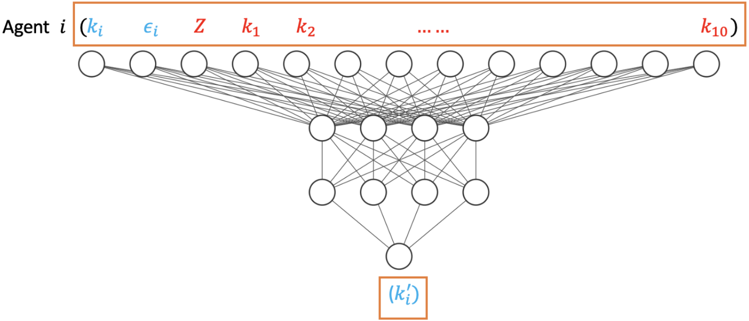


Figure 3: An example Neural Network in the case of $N = 10$ agents

Figure

Return

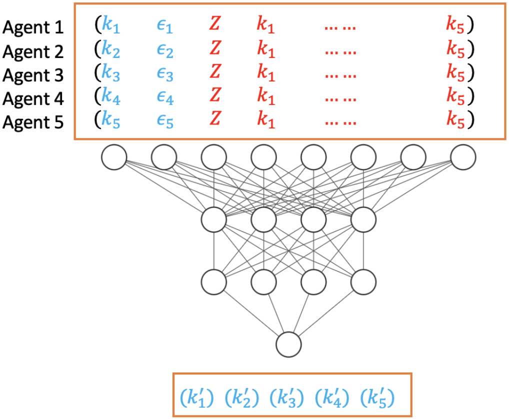


Figure 4: Processing the transition in the case of policy function

Figure

Return

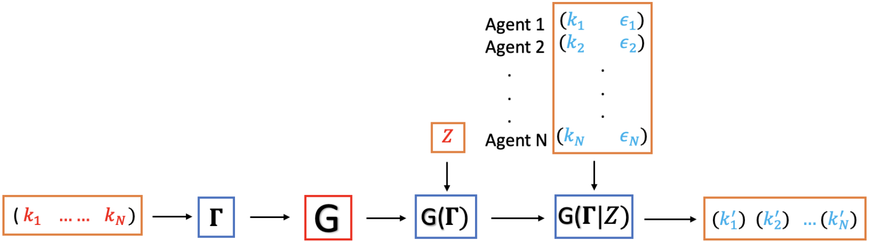


Figure 5: Processing the transition in the case of policy operator

Figure

▸ Layer

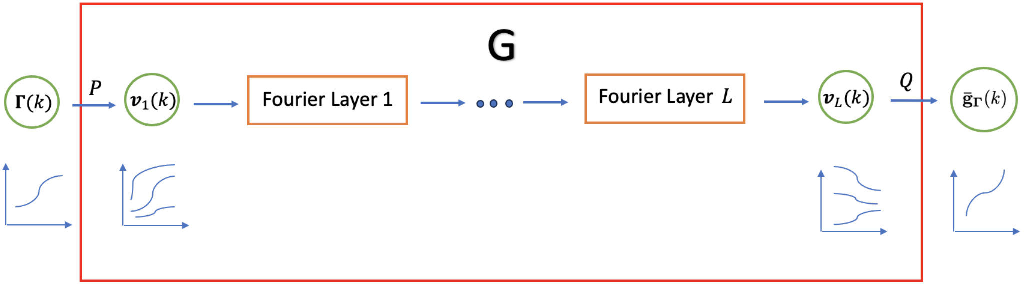


Figure 6: Fourier Neural Operator Architecture

Figure

▸ Return

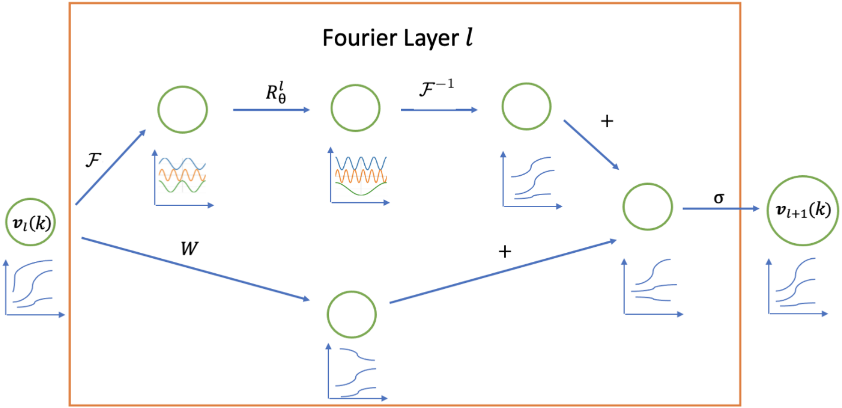


Figure 7: A Typical Fourier Layer l

Aiyagari, S Rao, "Uninsured idiosyncratic risk and aggregate saving," *The Quarterly Journal of Economics*, 1994, *109* (3), 659–684.

Haan, Wouter J Den, Kenneth L Judd, and Michel Juillard, "Computational suite of models with heterogeneous agents: model specifications," *Journal of Economic Dynamics and Control, this issue*, 2008.

Han, Jiequn and Yucheng Yang, "Deepham: A global solution method for heterogeneous agent models with aggregate shocks," *arXiv preprint arXiv:2112.14377*, 2021.

Judd, Kenneth L, Lilia Maliar, and Serguei Maliar, "Numerically stable and accurate stochastic simulation approaches for solving dynamic economic models," *Quantitative Economics*, 2011, *2* (2), 173–210.

Li, Zongyi, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar, "Fourier neural operator for parametric partial differential equations," *arXiv preprint arXiv:2010.08895*, 2020.

Maliar, Lilia, Serguei Maliar, and Pablo Winant, "Deep learning for solving dynamic economic models.," *Journal of Monetary Economics*, 2021, *122*, 76–101.