

DeepHAM: A Global Solution Method for Heterogeneous Agent Models with Aggregate Shocks

Yucheng Yang

University of Zürich and SFI

Joint with Jiequn Han (Flatiron Institute) and Weinan E (Peking U)

BSE Summer Forum

Link to paper and code: <https://github.com/frankhan91/DeepHAM>

Major Development in Macroeconomics

*“Major focus of macro over the last 20 years has been the development of models that incorporate **rich specifications of heterogeneity and frictions** that can simultaneously can speak to **aggregate outcomes** while also addressing **a rich set of cross-sectional facts**.”*

- Richard Rogerson (2021).

Some Prominent Examples:

- Uneven dynamics across sectors and population after major economic and policy shocks.
- Aggregate impact of monetary/fiscal policy through heterogeneous households & firms.
- Distribution of investors matter for asset prices.

Paradigm shift: **Representative agent** models → **Heterogeneous agent** models.

New challenge: distribution dynamics and its feedback (to decision and aggregates).

Introduction

- **Heterogeneous agent (HA) models** with **aggregate shocks** are solved with global Krusell-Smith (KS) method or local perturbation method.

	KS method	Perturbation method
Multiple shocks	No	Yes
Multiple endogenous states	No	Yes
Estimation/Calibration	No	Yes
Large shocks	Yes	No
Risky steady state	Yes	No
Nonlinearity e.g. ZLB	Yes	No

Introduction

- **Heterogeneous agent (HA) models** with **aggregate shocks** are solved with global Krusell-Smith (KS) method or local perturbation method.

	KS method	Perturbation method
Multiple shocks	No	Yes
Multiple endogenous states	No	Yes
Estimation/Calibration	No	Yes
Large shocks	Yes	No
Risky steady state	Yes	No
Nonlinearity e.g. ZLB	Yes	No

This paper: a new efficient, reliable, and interpretable global solution method for high dimensional HA models with aggregate shocks using **deep learning**.

Deep Learning for High Dimensional Models

- Deep learning has been successful in high-dimensional scientific computing problems.
- We can use deep learning to solve high-dim value & policy functions in economics:

1. Use deep neural networks to approximate value function $V : \mathbb{R}^N \rightarrow \mathbb{R}$

$$V(\mathbf{x}) \approx \mathcal{L}^P \circ \dots \circ \mathcal{L}^p \circ \dots \circ \mathcal{L}^1(\mathbf{x}), \quad \mathbf{x}: \text{high-dim state vector},$$

$$\mathbf{h}_p = \mathcal{L}^p(\mathbf{h}_{p-1}) = \sigma(\mathbf{W}_p \mathbf{h}_{p-1} + \mathbf{b}_p), \quad \mathbf{h}_0 = \mathbf{x},$$

σ : element-wise nonlinear fn, e.g. $\text{Tanh}(\cdot)$. Want to solve unknown parameters $\Theta = \{\mathbf{W}_p, \mathbf{b}_p\}_p$.

2. Cast high-dim function into a loss function, e.g. Bellman equation residual.
 3. Optimize unknown parameters, Θ , to minimize average loss on a “global” state space, using a variant of stochastic gradient descent (SGD) method.
- Similar procedure to polynomial “projection”, but more efficient in practice.

This Paper: DeepHAM Method for HA Model

1. Use neural networks (NN) to represent value & policy functions.
2. Nest sub-NN of *generalized moments* to represent state distribution.
3. Iteratively update value & policy functions, and *generalized moments*.

This Paper: DeepHAM Method for HA Model

1. Use neural networks (NN) to represent value & policy functions.
2. Nest sub-NN of *generalized moments* to represent state distribution.
3. Iteratively update value & policy functions, and *generalized moments*.

Apply DeepHAM to three economies:

1. Krusell-Smith problem: competitive equilibrium.
2. Krusell-Smith problem with a financial sector (in the paper).
3. Constrained efficiency problem in HA models with aggregate shocks.

This Paper: DeepHAM Method for HA Model

1. Use neural networks (NN) to represent value & policy functions.
2. Nest sub-NN of *generalized moments* to represent state distribution.
3. Iteratively update value & policy functions, and *generalized moments*.

Apply DeepHAM to three economies:

1. Krusell-Smith problem: competitive equilibrium.
2. Krusell-Smith problem with a financial sector (in the paper).
3. Constrained efficiency problem in HA models with aggregate shocks.

Main features:

1. High accuracy compared to other global solution methods.
2. Efficient computational speed (no curse of dimensionality).
3. Interpretability of distribution representation and function mappings.

Methodology

Illustration of HA models: Krusell and Smith (1998)

- Production economy with a continuum of households: each household i solves

$$\max_{c_{i,t} \geq 0, a_{i,t+1} \geq \underline{a}} \mathbb{E}_0 \sum_{t=0}^{\infty} \beta^t u(c_{i,t})$$

subject to budget constraint

$$a_{i,t+1} = w_t \bar{\ell} y_{i,t} + R_t a_{i,t} - c_{i,t}$$

- **Idiosyncratic shocks** on employment status $y_{i,t}$.
- Representative firm produces $Y_t = Z_t F(K_t, \bar{L})$.
- **Aggregate shock** $Z_t \sim$ two-state Markov, entering household's problem via factor prices:

$$R_t = Z_t \partial_K F(K_t, \bar{L}) - \delta, \quad w_t = Z_t \partial_L F(K_t, \bar{L})$$

- Factor prices are uncertain because of aggregate shock, but households must know their stochastic process to choose optimally.

Computational Setup: Krusell-Smith Method

- **Curse of dimensionality** shows up in recursive form of household i 's problem:

$$V(a_i, y_i, Z, \mathbf{\Gamma}) = \max_{c_i, a'_i} \{u(c_i) + \beta \mathbb{E} V(a'_i, y'_i, Z', \mathbf{\Gamma}' | y_i, Z)\}$$

subject to budget & borrowing constraints. $\mathbf{\Gamma}$: distribution over (a, y) of all households.

- Household must know $\mathbf{\Gamma}$ to predict factor prices \Rightarrow infinite dimension $\mathbf{\Gamma}$ is state variable.

Computational Setup: Krusell-Smith Method

- **Curse of dimensionality** shows up in recursive form of household i 's problem:

$$V(a_i, y_i, Z, \Gamma) = \max_{c_i, a'_i} \{u(c_i) + \beta \mathbb{E} V(a'_i, y'_i, Z', \Gamma' | y_i, Z)\}$$

subject to budget & borrowing constraints. Γ : distribution over (a, y) of all households.

- Household must know Γ to predict factor prices \Rightarrow infinite dimension Γ is state variable.
- **Krusell-Smith method** (KS, 1998; Maliar et al., 2010):
 1. Approximate state: $\hat{s}_i = (a_i, y_i, Z, m_1)$. m_1 : first moment of individual asset distribution.
 2. Log linear law of motion for m_1 :

$$\log(m_{1,t+1}) = A(Z) + B(Z) \log(m_{1t}).$$

- Very costly in complex HA models with multiple assets or multiple shocks.
- **New approach**: “learn” high dimensional value & policy functions with **deep learning**.

DeepHAM: Represent Distribution with Neural Networks (NN)

- Consider N -agent Krusell-Smith problem (N finite but large). General form of value & policy functions are like (ignore y):

$$V(a_i; a_1, \dots, a_{i-1}, a_{i+1}, \dots, a_N; Z), \quad c(a_i; a_1, \dots, a_{i-1}, a_{i+1}, \dots, a_N; Z)$$

DeepHAM: Represent Distribution with Neural Networks (NN)

- Consider N -agent Krusell-Smith problem (N finite but large). General form of value & policy functions are like (ignore y):

$$V(a_i; a_1, \dots, a_{i-1}, a_{i+1}, \dots, a_N; Z), \quad c(a_i; a_1, \dots, a_{i-1}, a_{i+1}, \dots, a_N; Z)$$

- Encode economics in NN architecture: approximate distn with **permutation invariant generalized moments** $\frac{1}{N} \sum_i Q(a_i)$, basis function $Q(\cdot)$ parameterized by (sub) NN:

$$\tilde{V}(a_i; \frac{1}{N} \sum_i Q_1(a_i), \dots, \frac{1}{N} \sum_i Q_J(a_i); Z)$$

$$\tilde{c}(a_i; \frac{1}{N} \sum_i \tilde{Q}_1(a_i), \dots, \frac{1}{N} \sum_i \tilde{Q}_J(a_i); Z)$$

DeepHAM: Represent Distribution with Neural Networks (NN)

- Consider N -agent Krusell-Smith problem (N finite but large). General form of value & policy functions are like (ignore y):

$$V(a_i; a_1, \dots, a_{i-1}, a_{i+1}, \dots, a_N; Z), \quad c(a_i; a_1, \dots, a_{i-1}, a_{i+1}, \dots, a_N; Z)$$

- Encode economics in NN architecture: approximate distn with **permutation invariant generalized moments** $\frac{1}{N} \sum_i Q(a_i)$, basis function $Q(\cdot)$ parameterized by (sub) NN:

$$\tilde{V}(a_i; \frac{1}{N} \sum_i Q_1(a_i), \dots, \frac{1}{N} \sum_i Q_J(a_i); Z)$$

$$\tilde{c}(a_i; \frac{1}{N} \sum_i \tilde{Q}_1(a_i), \dots, \frac{1}{N} \sum_i \tilde{Q}_J(a_i); Z)$$

- Special case: $Q(a) = a$ yields the first moment.

DeepHAM: Represent Distribution with Neural Networks (NN)

- Consider N -agent Krusell-Smith problem (N finite but large). General form of value & policy functions are like (ignore y):

$$V(a_i; a_1, \dots, a_{i-1}, a_{i+1}, \dots, a_N; Z), \quad c(a_i; a_1, \dots, a_{i-1}, a_{i+1}, \dots, a_N; Z)$$

- Encode economics in NN architecture: approximate distn with **permutation invariant generalized moments** $\frac{1}{N} \sum_i Q(a_i)$, basis function $Q(\cdot)$ parameterized by (sub) NN:

$$\tilde{V}(a_i; \frac{1}{N} \sum_i Q_1(a_i), \dots, \frac{1}{N} \sum_i Q_J(a_i); Z)$$

$$\tilde{c}(a_i; \frac{1}{N} \sum_i \tilde{Q}_1(a_i), \dots, \frac{1}{N} \sum_i \tilde{Q}_J(a_i); Z)$$

- Special case: $Q(a) = a$ yields the first moment.
- Algorithm solves **generalized moments** (GMs) that matter most for policy and value functions. (“numerically determined sufficient statistics”)

DeepHAM: Represent Distribution with Neural Networks (NN)

- Consider N -agent Krusell-Smith problem (N finite but large). General form of value & policy functions are like (ignore y):

$$V(a_i; a_1, \dots, a_{i-1}, a_{i+1}, \dots, a_N; Z), \quad c(a_i; a_1, \dots, a_{i-1}, a_{i+1}, \dots, a_N; Z)$$

- Encode economics in NN architecture: approximate distn with **permutation invariant generalized moments** $\frac{1}{N} \sum_i Q(a_i)$, basis function $Q(\cdot)$ parameterized by (sub) NN:

$$\tilde{V}(a_i; \frac{1}{N} \sum_i Q_1(a_i), \dots, \frac{1}{N} \sum_i Q_J(a_i); Z)$$

$$\tilde{c}(a_i; \frac{1}{N} \sum_i \tilde{Q}_1(a_i), \dots, \frac{1}{N} \sum_i \tilde{Q}_J(a_i); Z)$$

- Special case: $Q(a) = a$ yields the first moment.
- Algorithm solves **generalized moments** (GMs) that matter most for policy and value functions. (“numerically determined sufficient statistics”)
- GMs provide **interpretability** on how heterogeneity matters.

DeepHAM Algorithm: General Procedure

- Formulate discrete time N -agent HA models, solve value and policy functions **parameterized by neural nets** $V(s), c(s)$. $s = (a_i, y_i, Z, \Gamma)$.
- Parameterize two parts of mapping:
 1. Distribution $\Gamma \mapsto J$ generalized moments $\frac{1}{N} \sum_i Q_j(a_i)$.
 2. $(a_i, y_i, Z, \{\frac{1}{N} \sum_i Q_j(a_i)\}) \mapsto c, V$.

DeepHAM Algorithm: General Procedure

- Formulate discrete time N -agent HA models, solve value and policy functions **parameterized by neural nets** $V(s), c(s)$. $s = (a_i, y_i, Z, \Gamma)$.
- Parameterize two parts of mapping:
 1. Distribution $\Gamma \mapsto J$ generalized moments $\frac{1}{N} \sum_i Q_j(a_i)$.
 2. $(a_i, y_i, Z, \{\frac{1}{N} \sum_i Q_j(a_i)\}) \mapsto c, V$.
- Iteratively update value and policy functions. In each iteration:
 1. Simulate stationary distribution with the latest policy.
 2. Given policy function, update value function. [details](#)
 3. Given value function, optimize policy function on **simulated paths**.

DeepHAM: Policy Function Optimization

In iteration k , given $V^{(k)}(s)$, optimize **policy** $\mathcal{C}^{(k)}(s)$ on **simulated paths**.

For N -agent competitive equilibrium, solve with **fictitious play**: separate it into N individual problems, when solving i 's problem, fix other agents' policy from last "play". Iterate:

1. At "play" $\ell + 1$, last play's policy $\mathcal{C}^{(k,\ell)}(s)$ is known.
2. For agent $i = 1$, update her optimal policy $\mathcal{C}^{(k,\ell+1)}(s)$ according to:

$$\max_{\mathcal{C}^{(k,\ell+1)}(s)} \mathbb{E}_{\mu(\mathcal{C}^{(k-1)}), \mathcal{E}} \left(\sum_{t=0}^T \beta^t u(c_{i,t}) + \beta^T V^{(k)}(s_{i,T}) \right)$$

subject to others all following $\mathcal{C}^{(k,\ell)}(s)$ in the first T periods.

3. All agents adopt the new policy $\mathcal{C}^{(k,\ell+1)}(s)$ in "play" $\ell + 1$.

DeepHAM: Policy Function Optimization

In iteration k , given $V^{(k)}(s)$, optimize **policy** $\mathcal{C}^{(k)}(s)$ on **simulated paths**.

For N -agent competitive equilibrium, solve with **fictitious play**: separate it into N individual problems, when solving i 's problem, fix other agents' policy from last "play". Iterate:

1. At "play" $\ell + 1$, last play's policy $\mathcal{C}^{(k,\ell)}(s)$ is known.
2. For agent $i = 1$, update her optimal policy $\mathcal{C}^{(k,\ell+1)}(s)$ according to:

$$\max_{\mathcal{C}^{(k,\ell+1)}(s)} \mathbb{E}_{\mu(\mathcal{C}^{(k-1)}), \mathcal{E}} \left(\sum_{t=0}^T \beta^t u(c_{i,t}) + \beta^T V^{(k)}(s_{i,T}) \right)$$

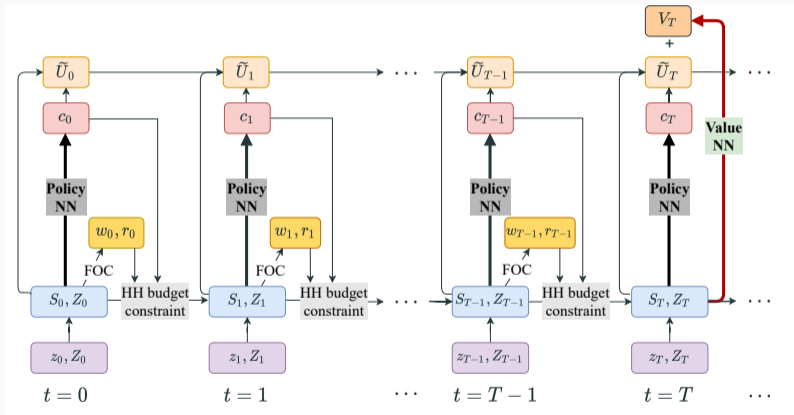
subject to others all following $\mathcal{C}^{(k,\ell)}(s)$ in the first T periods.

3. All agents adopt the new policy $\mathcal{C}^{(k,\ell+1)}(s)$ in "play" $\ell + 1$.

Optimization solved on Monte Carlo simulation with N agents on a large number of sample paths in a **computational graph**.

Computational Graph for Policy Function Optimization

$$\max_{\Theta^C} \mathbb{E}_{\mu(\mathcal{C}^{(k-1)}), \mathcal{E}} \left(\tilde{U}_{i,T} + \beta^T V_{\text{NN}}(s_{i,T}; \Theta^V) \right)$$



Budget constraint $a_{i,t+1} = (r_t + 1 - \delta)a_{i,t} + w_t \bar{\ell} y_{i,t} - c_{i,t}$. $s_t = (a_{i,t}, y_{i,t}, Z_t, \Gamma_t)$. Cumulative utility $\tilde{U}_{i,t} = \sum_{\tau=0}^t \beta^\tau u(c_{i,\tau})$

Remarks on optimization over simulated paths

- Agents **formulate expectation** over future prices **through simulation**: no perceived law of motion needed.
 - Related to the idea of (model-based) reinforcement learning.
- Our formulation: easily extend to **constrained efficiency** problem.
 - Competitive equilibrium: fictitious play.
 - Constrained efficiency: optimize all agents' policy together.
- Finite agent approximation + fictitious play: could be used for solving **strategic equilibrium**.

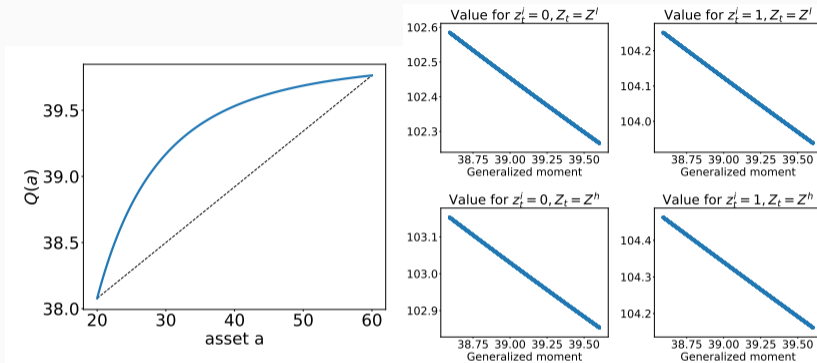
Accuracy Results for Krusell-Smith Problem

Method and Moment Choice	Bellman error	Std of error
KS Method (Maliar et al., 2010)	0.0253	0.0002
DeepHAM with 1st moment	0.0184	0.0023
DeepHAM with 1 generalized moments	0.0151	0.0015

Definition of Bellman Error

- Highly accurate compared to Krusell-Smith (KS) method. solution comparison
- Even only with first moment as model input, DeepHAM outperform KS method due to better capture of nonlinearity.
- Generalized moment yields more accurate solution than the first moment, as it extract more relevant information.

Interpretation of the Generalized Moment (GM)



Plot of $Q_1(a)$

Map $\frac{1}{N} \sum_i Q_1(a_i)$ to value function

- Basis function concave in asset, value function is linear wrt the GM.
- Heterogeneity matters! Unanticipated redistributive policy shock: asset from rich to poor HH \Rightarrow generalized moment $\uparrow \Rightarrow$ unshocked agent welfare \downarrow .
- KS method implies no effect, as first moment not change.

DeepHAM for Constrained Efficiency Problem

- **Constrained efficiency** problem: planner's allocation in incomplete market.
- Important “second best” allocation, but hard to solve in HA models.
- Literature only solves for HA models **without** aggregate shocks (Davila, Hong, Krusell, Rios-Rull, 2012; Nuno and Moll, 2018).

DeepHAM for Constrained Efficiency Problem

- **Constrained efficiency** problem: planner's allocation in incomplete market.
- Important “second best” allocation, but hard to solve in HA models.
- Literature only solves for HA models **without** aggregate shocks (Davila, Hong, Krusell, Rios-Rull, 2012; Nuno and Moll, 2018).
- DeepHAM solves constrained efficiency problem as easily as solve competitive equilibrium, just to remove the fictitious play procedure.
- We solve constrained efficiency problem of Davila et al. (2012), and that **with** aggregate shocks and countercyclical unemployment risk.
- It takes DeepHAM **20 minutes** to solve Davila et al. (2012) on GPU, which takes conventional methods **> 10 hours** on CPU.

Constrained Efficiency for HA Models w or w/o Agg Shock

	No aggregate shock		Aggregate shock	
	Market	Constrained Opt.	Market	Constrained Opt.
Average assets	30.635	119.741	34.296	95.811
Wealth Gini	0.864	0.862	0.812	0.878
Consumption Gini	0.615	0.386	0.578	0.388

Findings:

1. Both models: K in constrained optimum \gg competitive equilibrium.
 - Why? Overcome pecuniary externality: $K \uparrow \Rightarrow w \uparrow, R \downarrow$, redistribute from rich to poor (high labor share).

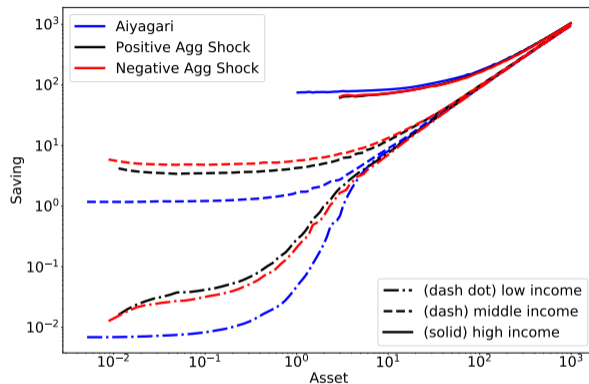
Constrained Efficiency for HA Models w or w/o Agg Shock

	No aggregate shock		Aggregate shock	
	Market	Constrained Opt.	Market	Constrained Opt.
Average assets	30.635	119.741	34.296	95.811
Wealth Gini	0.864	0.862	0.812	0.878
Consumption Gini	0.615	0.386	0.578	0.388

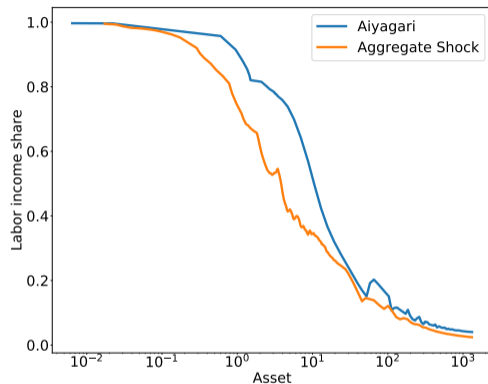
Findings:

- Both models: K in constrained optimum \gg competitive equilibrium.
 - Why? Overcome pecuniary externality: $K \uparrow \Rightarrow w \uparrow, R \downarrow$, redistribute from rich to poor (high labor share).
- Constrained optimal K in model w/ agg shock $<$ w/o agg shock.

Constrained Efficiency for HA Models w or w/o Agg Shock



Policy function



Labor share distribution

Agg shock \Rightarrow precautionary saving \uparrow by poor HHs \Rightarrow labor share lower than model w/o agg shock. So planner raises K less in constrained efficient equilibrium.

Conclusion

- We develop DeepHAM, an efficient, reliable, and interpretable deep learning based method to solve HA models with aggregate shocks globally.
- Deep learning based model reduction informs interpretable generalized moments.
- For the first time, we solve constrained efficiency in HA models with aggregate shock.
- “Neural network techniques will open up a new research avenue in macro-finance.” (RFS 2021). A few possible directions:
 1. Models with search and matching, or rich spatial structure (“DeepSAM”).
 2. HA(NK) models: asset pricing, welfare, and optimal policy (“DeepHANK”).
 3. Machine/Reinforcement learning for models of learning (“Machine Learning Squared”).
- As is happening in other disciplines (Nobel Prize Physics 2024, Chemistry 2024), deep learning may become (part of) the new generation of model solvers in economics.

Thank You!

Comments and questions are welcome!

Email: `yucheng.yang@uzh.ch`

Appendix

- Solving HA models with aggregate shocks:
 1. Global KS method: Krusell and Smith (1998), Den Haan (2010) project, Fernandez-Villaverde et al. (2019), etc.
 2. Local perturbation method: Reiter (2009), Ahn et al. (2017), Winberry (2018), Bayer and Luetticke (2020); Boppart, Krusell and Mitman (2018), Auclert et al. (2021), etc.
- Deep learning for high dimensional problems:
 1. Stochastic control & PDE: Han and E (2016), Han, Jentzen and E (2018).
 2. Macroeconomics: Duarte (2018), Fernandez-Villaverde et al. (2020, 2021), Maliar et al. (2021), Azinovic et al. (2022), among many others.
- How heterogeneity matters in macro: Kaplan and Violante (2018), Kaplan et al. (2018), Auclert (2019), etc.
- Constrained efficiency problem in HA models: Davila et al. (2012), Nuno and Moll (2018), Bhandari et al. (2021), etc.

DeepHAM: Value Function Learning

Define cumulative utility for HH i up to t :

$$\tilde{U}_{i,t} = \sum_{\tau=0}^t \beta^{\tau} u(c_{i,\tau}).$$

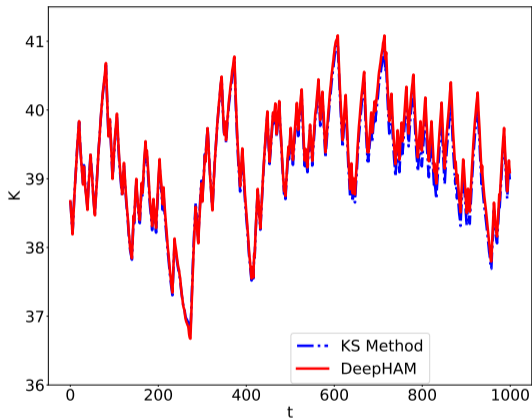
In iteration k , given policy function $\mathcal{C}^{(k-1)}(s)$:

1. Sample states s from the stationary distribution. Then the value of each state s can be approximately calculated as cumulative utility in the following T (T large enough) periods following policy $\mathcal{C}^{(k-1)}(s)$:

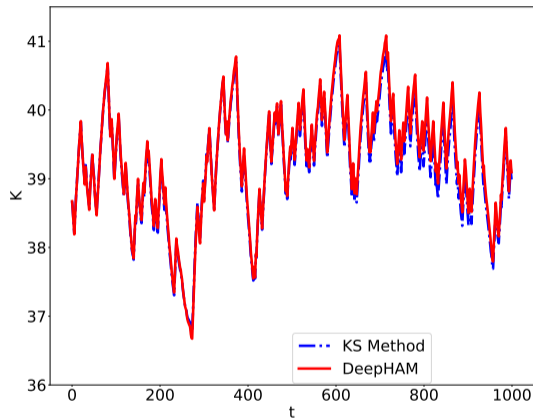
$$\tilde{V}^{(k)}(s) \approx \mathbb{E} \tilde{U}_T = \mathbb{E} \sum_{\tau=0}^T \beta^{\tau} u(c_{i,\tau})$$

2. Learn **value function** $V^{(k)}(s)$ parameterized by deep neural networks with **regression**. [back](#)

Solution Comparison



aggregate capital (K_t)



aggregate consumption (C_t)

Accuracy Measures: Bellman Equation Errors

For the KS problem, only using solved value function $V(\cdot)$, **Bellman equation error** is

$$\text{err}_B = V(a_i, y_i, Z, \mathbf{a}^{-i}, \mathbf{y}^{-i}) - \max_{c_i} \left\{ u(c_i) + \beta \sum_{y', Z', \mathbf{y}'^{-i}} V(a'_i, y'_i, Z', \widehat{\mathbf{a}}'^{-i}, \mathbf{y}'^{-i}) \right. \\ \left. \times \Pr \left(Z', y'^i, \mathbf{y}'^{-i} | Z, y^i, \mathbf{y}^{-i} \right) \right\}$$

back