Structural Reinforcement Learning for Heterogeneous Agent Macroeconomics

Yucheng Yang*
Zurich

Chiyuan Wang*
Peking University

Andreas Schaab
Berkeley

Benjamin Moll†
LSE

*equal contribution †corresponding author

ICMS Workshop on AI and Economics

Heterogeneous agent models with aggregate risk

- One of the most important developments in macroeconomics since 90s
- Key challenge: rational expectations + general equilibrium
  $\Rightarrow$ distribution = state variable in Bellman equation ("Master equation")
  - true even though households/firms only care about prices
  - intuition: equilibrium prices are not Markov, only the distribution is
    $\Rightarrow$ forecast distributions to forecast prices
- Despite recent impressive advances to solve it directly, still lack of efficient global solution methods for advanced HA models with aggregate risk
- This paper: sidestep master eqn with structural reinforcement learning

Sidestep Master eqn using structural reinforcement learning

RL: learn value & policy functions in Markov decision process with Monte Carlo.

Unlike dynamic programming, RL can handle non-Markov states (e.g. prices).

This paper:

- Replace dist'n with low-dim. prices in state space
- RL about equilibrium prices but not individual states $\Rightarrow$ "Structural RL"
- Efficient asset market clearing using policy functions (= demand curves)
- Structural RL for both households and firms $\Rightarrow$ solving HANK

Outcome: efficient & flexible global solution method for HA models w agg risk, solves problems traditional methods struggle with:

1. non-trivial market clearing (Huggett w agg. risk) $\approx$ 1 min on Google Colab
2. HANK with forward-looking Phillips curve $\approx$ 3 min

Illustrative Model Setup and Methodology

Textbook HA model – Huggett (1993) with agg. risk

- Continuum of agents i, heterog. in $(b_{i,t}, y_{i,t})$, $y_{i,t}$ = id. risk, agg. shock z_t
- Households choose consumption $c_{i,t}$ to maximize

$$v_{i,0} = \max_{\{c_{i,t}\}} \mathbb{E}_0 \sum_{t=0}^{\infty} \beta^t u(c_{i,t}) \quad \text{subject to}$$

$$c_{i,t} + b_{i,t+1} = R_t b_{i,t} + y_{i,t} z_t, \quad y_{i,t+1} \sim \mathcal{T}_y(\cdot|y_{i,t}), \quad b_{i,t+1} \geq \underline{b}$$

- State of economy: $z_t \sim \mathcal{T}_z(\cdot|z_{t-1})$ and distribution $G_t(b, y)$. Prices: R_t.
- Asset market clearing: interest rates R_t such that

$$\int b_t'(b, y)\, dG_t(b, y) = \bar{B}, \quad \text{all } t$$

Note: agent problem depends on G_t only via low-dim. prices $R_t = P^*(G_t, z_t)$

Key difficulty: equilibrium prices are not Markov

- Households care about price R_t, not directly about distribution G_t
- But low-dim equilibrium prices $R_t = P^*(G_t, z_t)$ is not first-order Markov
- ... only extremely high-dimensional (G_t, z_t) is
- Dynamic programming can only handle Markov states $\Rightarrow$ Master equation

$$V(b, y, G, z) = \max_{c, b'} \left\{ u(c) + \beta \mathbb{E}\left[V(b', y', G', z') \mid y, z \right] \right\}$$

subject to $c + b' = R(G, z)b + yz$, $b' \geq \underline{b}$.

Structural RL to sidestep the master equation

- Our solution: use policy gradient to solve for low-dimensional policy

$$c_\theta(b, y, R, z)$$

to maximize expected lifetime utility on simulated equilibrium paths

$$v(\theta) = \mathbb{E}_{(b_0, y_0, G_0, z_0)}\left[\sum_{t=0}^{T} \beta^t u(c_\theta(b_t, y_t, R_t, z_t))\right]$$

subject to $c + b' = R(G, z)b + yz$, $b' \geq \underline{b}$. ▸ Restricted perceptions equilibrium

- Assumption 1: households observe prices R_t but not full distribution G_t
  1. Similarity to standard RL: don't know transition prob. of prices;
  2. Difference to standard RL: agents know individual state transitions & utility functions and exploit in computation $\Rightarrow$ Structural RL
- Assumption 2: policy depends only on current prices $\pi(b, y, R, z)$.
Companion paper: extend to price histories via Recurrent SRL

Simulating the economy given policy $c_\theta(b, y, R, z)$

Given (suboptimal) policy $c_\theta(b, y, R, z)$, want to simulate economy forward:

$$\{G_t, R_t, z_t\}_{t \geq 0}$$

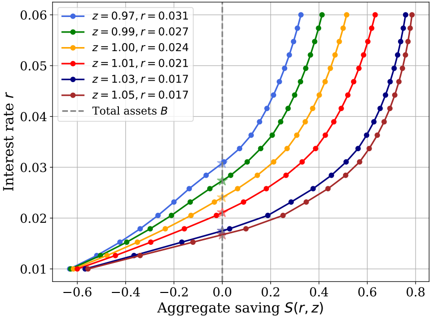
- z_{t+1} is easy: exogenous aggregate shock process
- G_{t+1} is easy when given R_{t+1}, z_t, G_t, and policy
- R_{t+1} is hard: pinned down implicitly by market clearing each period

Efficient handling of non-trivial market clearing

$$S_t(R,z) = \bar{B}, \qquad S_t(R,z) := \int b'(b,y,R,z)dG_t(b,y) = \text{agg. saving supply}$$

Key: integrate policy $b'(b,y,R,z) \Rightarrow$ aggregate saving on price grid $S(R,z_t)$

$$\Rightarrow R_t \text{ solves } S_t(R_t,z_t) = \mathbf{b}'(R_t,z_t)^\top \mathbf{g}_t = \bar{B}$$



Market clearing is part of environment, not another function loop!

Structural RL Summary

- Parameterize a low-dimensional household policy

$$c_\theta(b, y, R, z)$$

- Choose θ to maximize lifetime utility over simulated equilibrium rollouts:

$$v(\theta) \approx \widehat{v}(\theta) = \frac{1}{N}\sum_{n=1}^{N}\sum_{t=0}^{T}\beta^t u(c_\theta(b_t^n, y_t^n, R_t^n, z_t^n))$$

- No perceived law of motion; no inner loop / outer loop as in Krusell-Smith
- Optimize θ by policy gradient with structural knowledge on b, y dynamics and $u(\cdot)$
  - low-dimensional state $\Rightarrow$ grid-based π_θ works well in practice

Computational experiments

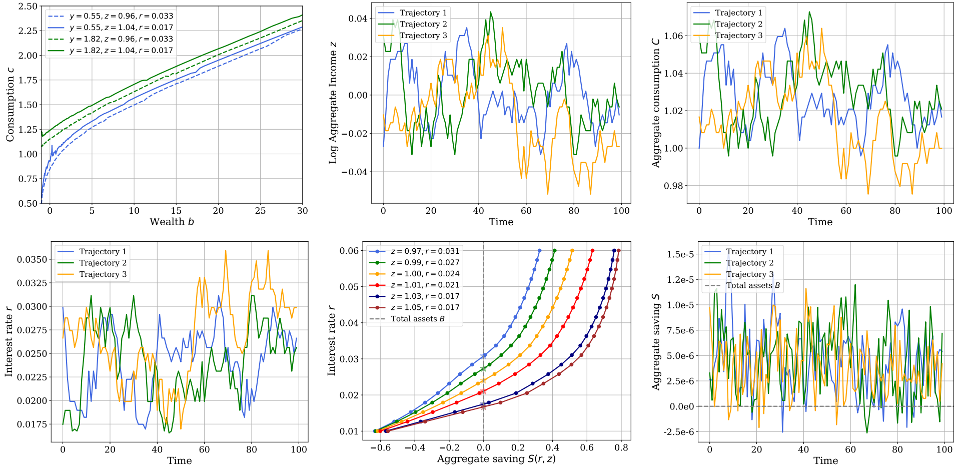
Runtimes

- Efficient implementation in JAX for GPUs, run on Google Colab
- Stochastic algorithm: present averages over multiple runs

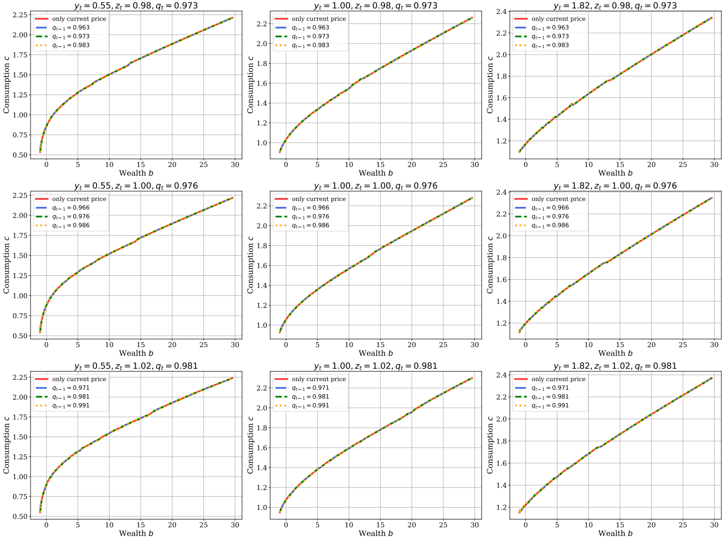
Model	Average converge epoch	# Runs	Average Runtime (sec)
Krusell-Smith	438.4	10	56.55
Huggett with agg. shocks	480.6	10	75.29
HANK with agg. shocks	496.5	10	199.53
Partial equilibrium (Huggett)	289.3	10	39.49

Note: all experiments were implemented on the A100 GPU on Google Colab

Huggett: simulated trajectories under the optimal policy



Huggett: adding one lagged price p_{t-1} into state space



HANK with forward-looking Phillips curve

Household block (similar to before): states $s = (b, y)$ and prices $p = (R, w)$

$$\text{policies} = \{c(s,p), n(s,p)\} \quad \text{that maximize PDV of utility}$$

Firm block: price setting $\Rightarrow$ forward-looking Phillips curve = added difficulty

$$\Pi_t = \frac{\varepsilon}{\theta}\left(\frac{w_t}{z_t} - m^*\right) + \mathbb{E}\left[R_{t+1}^{-1}\frac{Y_{t+1}}{Y_t}\Pi_{t+1}\middle| \mathcal{I}_t\right], \qquad m^* = \frac{\varepsilon - 1}{\varepsilon}$$

Conventional approach: parameterize $\mathbb{E}[\Pi_{t+1}|\mathcal{I}_t] \Rightarrow$ complicated fixed-point
(e.g. Kase-Melosi-Rottner, Fernández-Villaverde et al)

HANK with forward-looking Phillips curve

Household block (similar to before): states $s = (b, y)$ and prices $p = (R, w)$

$$\text{policies} = \{c(s,p), n(s,p)\} \quad \text{that maximize PDV of utility}$$

Firm block: price setting $\Rightarrow$ forward-looking Phillips curve = added difficulty

Our solution: solve firm price-setting problem using policy gradient method

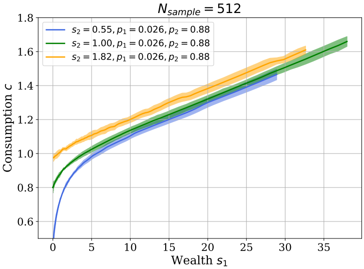
$$\text{policy} = \Pi(z, p) \quad \text{that maximizes}$$

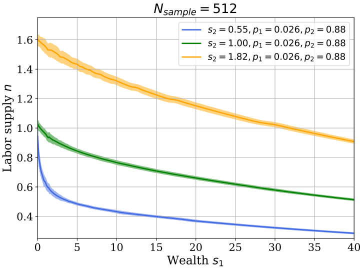
$$J_\Pi = \mathbb{E}_0 \left[\sum_{t=0}^{\infty} R_{0\to t}^{-1} \left\{ \text{Profits}\left(\frac{1+\Pi(z_t, R_t, w_t)}{1+\Pi_t}, \frac{w_t}{z_t}, Y_t \right) - \frac{\theta}{2} \left(\Pi(z_t, R_t, w_t)\right)^2 \right\} \right]$$

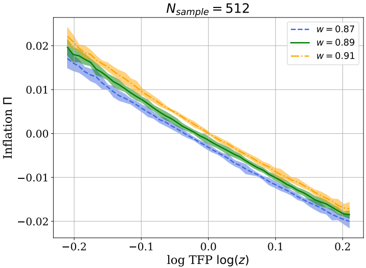
Symmetric treatment of firms and households, update policies simultaneously

In practice: good convergence properties

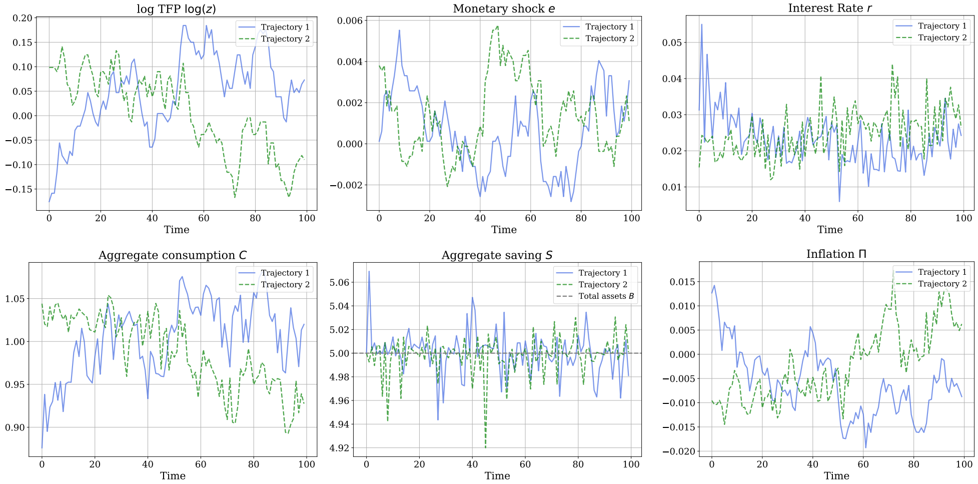
HANK: Household and firm policy functions







HANK simulations



Summary

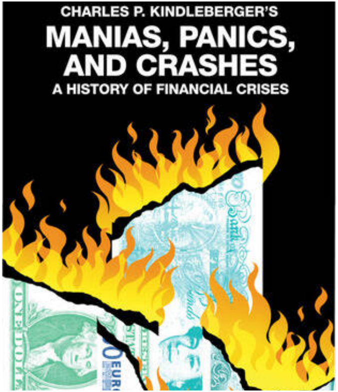
Efficient and flexible global solution method for non-stationary HA models

Reinforcement learning about equilibrium prices (but not individual states)

- sidestep infinite-dimensional Master equation
- solve much lower-dimensional problem

Solves problems traditional methods struggle with

- non-trivial market clearing conditions
- HANK with forward-looking Phillips curve
- next: models of large crises, booms/busts



Thanks!

In stationary world, lagged prices are enough for RE ▸ back

Recall Assumption 1: agents observe prices p_t but not distribution $G_t(s)$

Important: Assumption 1 still consistent with rational expectations

Why? Wold representation theorem!

Step 1 (Wold): if p_t-process is stationary, it has Wold representation = VMA(∞)

$$p_t = \sum_{j=0}^{\infty} c_j \varepsilon_{t-j}, \qquad c_j = \text{some unknown coefficients}$$

Step 2: if VMA(∞) is invertible, it can be expressed as a VAR(∞) and hence

$$p_{t+1} \sim \mathcal{T}_p(\cdot | p_t, p_{t-1}, \ldots)$$

In practice, include finitely many lags

Assumption 2: extreme case with zero lags $p_{t+1} \sim \mathcal{T}_p(\cdot | p_t)$

Restricted perceptions equilibrium ▸ back

A pair of mappings (π^*, P^*) constitutes a restricted perceptions equilibrium if:

1. Optimality. For any price sequence $\{p_t\}$ generated by $p_t = P^*(\mathbf{g}_t, z_t)$ and exogenous sequence $\{z_t\}$, agents choose $\pi^*(s, p, z)$ to solve:

$$\max_{\pi} \mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t u(\pi(s_t, p_t, z_t))\right],$$

   subject to the individual budget constraint and state transition equations.

2. Market clearing. For every period t, all markets clear.

3. Consistency. The prices that agents use to form expectations coincide with the prices in the simulated economy when all agents follow π^*:

$$p_t = P^*(\mathbf{g}_t, z_t), \qquad \mathbf{g}_{t+1} = \mathbf{A}^{\top}_{\pi^*(p_t, z_t)} \mathbf{g}_t$$

Key difficulty: equilibrium prices are not Markov

- Equilibrium prices satisfy back

$$p_t = P^*(\mathbf{g}_t, z_t)$$
$$\mathbf{g}_{t+1} = \mathbf{A}^\top_{\pi_t(z_t)}\mathbf{g}_t$$
$$z_{t+1} \sim \mathcal{T}_z(\cdot|z_t)$$

- Difficulty: low-dimensional p_t does not have Markov property...
- ... only extremely high-dimensional $(\mathbf{g}_t, z_t)$ does
- Dynamic programming can only handle Markov states $\Rightarrow$ Master equation

$$V(s, \mathbf{g}, z) = \max_c u(c) + \beta\mathbb{E}\left[V(s', \mathbf{g}', z')|s, \mathbf{g}, z\right] \text{ s.t. } s' \sim \mathcal{T}_s(\cdot|s, c, P^*(\mathbf{g}, z))$$

- Without Markov transition prob's: cannot even write Bellman equation!
- But what if there was a way to approximate value and policy functions with p_t process for which there are no Markov transition probabilities?

Brief primer on reinforcement learning

RL: learning value & policy functions in incompletely-known Markov decision processes from experience (Monte Carlo sampling) a.k.a. "approximate DP"

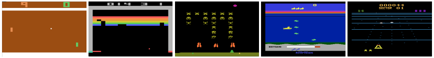
Playing Atari with Deep Reinforcement Learning

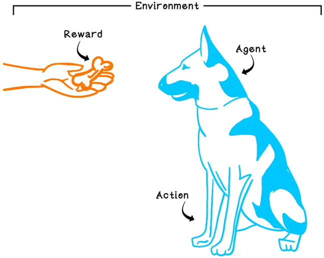
Volodymyr Mnih Koray Kavukcuoglu David Silver Alex Graves Ioannis Antonoglou

Daan Wierstra Martin Riedmiller

DeepMind Technologies

{vlad,koray,david,alex.graves,ioannis,daan,martin.riedmiller} @ deepmind.com





Computing an expected value

Random variable x

How compute expected value $\mathbb{E}[x]$? Two approaches:

1. Exact: know probability distribution $f(x) \Rightarrow$ calculate
$$\mathbb{E}[x] = \int x f(x) dx$$
2. Monte Carlo: don't know f but can sample $\{x_1, x_2, \ldots, x_N\}$
$$\mathbb{E}[x] \approx \bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i$$
Or update incrementally (stochastic approximation method):
$$\bar{x}_k = \frac{1}{k} \sum_{i=1}^{k} x_i \quad \text{satisfies} \quad \bar{x}_k = \bar{x}_{k-1} + \frac{1}{k} \left(x_k - \bar{x}_{k-1} \right), \quad \frac{1}{k} = \text{“learning rate”}$$

Computing a value function

For now: eliminate actions and individual states

$$v_0 = \mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t u(p_t)\right], \quad p_t = \text{exogenous stochastic process}$$

Two approaches:

1. Dynamic programming: p_t Markov and know $f(p'|p)$

$$v(p) = u(p) + \int v(p')f(p'|p)dp'$$

Computing a value function

For now: eliminate actions and individual states

$$v_0 = \mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t u(p_t)\right], \quad p_t = \text{exogenous stochastic process}$$

Two approaches:

1. Dynamic programming: p_t Markov and know $f(p'|p)$

$$v(p) = u(p) + \int v(p') f(p'|p) dp'$$

2. Monte Carlo: don't know f but sample N trajectories $\{p_t^i\}_{t=0}^{T}$

$$v_0 \approx \widehat{v}_0 = \frac{1}{N}\sum_{i=1}^{N}\sum_{t=0}^{T} \beta^t u(p_t^i) \quad \text{or} \quad \widehat{v}_0^k = \widehat{v}_0^{k-1} + \frac{1}{k}\left(\sum_{t=0}^{T} \beta^t u(p_t^k) - \widehat{v}_0^{k-1}\right)$$

Computing a value function

For now: eliminate actions and individual states

$$v_0 = \mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t u(p_t)\right], \qquad p_t = \text{exogenous stochastic process}$$

Two approaches:

1. Dynamic programming: p_t Markov and know $f(p'|p)$

$$v(p) = u(p) + \int v(p') f(p'|p) dp'$$

2. Monte Carlo: don't know f but sample N trajectories $\{p_t^i\}_{t=0}^T$

$$v_0 \approx \widehat{v}_0 = \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T} \beta^t u(p_t^i) \quad \text{or} \quad \widehat{v}_0^k = \widehat{v}_0^{k-1} + \frac{1}{k}\left(\sum_{t=0}^{T} \beta^t u(p_t^k) - \widehat{v}_0^{k-1}\right)$$

Can also be extended to compute optimal policy: policy gradient method

Computing a value function

For now: eliminate actions and individual states

$$v_0 = \mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t u(p_t)\right], \quad p_t = \text{exogenous stochastic process}$$

Two approaches:

1. Dynamic programming: p_t Markov and know $f(p'|p)$

$$v(p) = u(p) + \int v(p') f(p'|p) dp'$$

2. Temporal difference learning: N trajectories, update v incrementally

$$\widehat{v}^k(p_t^k) = \widehat{v}^{k-1}(p_t^k) + \frac{1}{k}\left(\left[\sum_{\tau=0}^{n-1} \beta^\tau u(p_{t+\tau}^k) + \beta^n \widehat{v}^{k-1}(p_{t+n}^k)\right] - \widehat{v}^{k-1}(p_t^k)\right)$$

Can also be extended to compute optimal policy: policy gradient method

Agent vs Environment, Rollout of a Policy

RL distinguishes between agent and environment = everything outside of agent

In HA macro:

- agents = households and their policies
- environment = everything else, including market clearing etc

Related: rollout = agent interacting with environment under given policy

- take any (generally suboptimal) policy, run it forward in time
- how optimize policy is completely separate question (policy improvement)

This viewpoint will be important, in particular for non-trivial market clearing

back