# Structural Reinforcement Learning for Heterogeneous Agent Macroeconomics

Yucheng Yang*
Zurich

Chiyuan Wang*
Peking University

Andreas Schaab
Berkeley

Benjamin Moll
LSE

*equal contribution

NBER Summer Institute 2026

# Heterogeneous agent models with aggregate risk

- Huge literature since Krusell-Smith and Den Haan from late 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

# Heterogeneous agent models with aggregate risk

- Huge literature since Krusell-Smith and Den Haan from late 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).

# 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

# 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. portfolio choice à la KS1997 $\approx$ 1 min. Also: two prices clear two asset markets
3. HANK with forward-looking price/wage Phillips curve $\approx$ 4 min

# Literature and contribution

Global solution methods for HA models with aggregate risk

Fernández-Villaverde et al, Han-Yang-E, Maliar-Maliar-Winant, Azinovic-Gaegauf-Scheidegger, Schaab, Duarte-Silva...

- sidestep Master equation rather than "taming curse of dimensionality"
- Han-Yang-E DeepHAM = also RL-inspired

Global solution with bounded rationality Krusell-Smith, Den Haan, ...

- difference: no perceived law of motion

Self-confirming equilibrium & restricted perceptions equilibrium

- agents form price exp. from data generated by economy where they live
- but do so using restricted state space

Adaptive (least squares) learning Bray, Marcet-Sargent, Evans-Honkapohja, Jacobson, Giusto, ...

- similarity: stochastic approximation; differ: learning by economist, not agents

"Sequence space" Auclert-Bardóczy-Rognlie-Straub, AzinovicYang-Žemlicka

- global solution in sequence space (low-dim. prices) via Monte Carlo

# SRL Coding Tutorial https://github.com/StructuralRL/SRL

This repo presents a pedagogical JAX implementation of policy-gradient methods for heterogeneous-agent macro models in general equilibrium, including with aggregate risk.

SRL goes with the paper *Structural Reinforcement Learning for Heterogeneous Agent Macroeconomics* by Yucheng Yang, Chiyuan Wang, Andreas Schaab, and Benjamin Moll. We try to make the repo and our codes as accessible as possible. Our hope is that you will be able to clone the repo, read the notebooks top to bottom, and easily follow along with both the economics and the method. Note this is the *tutorial repo* for SRL, not the final replication codes.

## The tutorials

Start with the tutorials. They start with the simplest, canonical household problem and build up to our full implementation of a heterogeneous agent model with aggregate risk by adding one step at a time. Read them in order from notebook 0. TUTORIAL.md is the full syllabus.

| # | Notebook | Open in Colab | Colab G4 runtime | Free Colab T4 runtime |
|---|---|---|---|---|
| 0 | Household problem, in NumPy | Open in Colab | ~1 min | ~3 min |
| 1 | The same problem, in JAX | Open in Colab | ~6 s | ~15 s |
| 2 | The same problem, by policy gradient | Open in Colab | ~30 s | ~1 min |
| 3 | The household with moving prices | Open in Colab | ~45 s | ~1.5 min |
| 4 | Huggett without aggregate risk | Open in Colab | ~4.5 min | ~8 min |
| 5 | Huggett with aggregate risk | Open in Colab | ~35 s | ~1.5 min |

# Baseline Setup and Methodology

# Textbook HA model – Huggett (1993) with aggregate 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

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

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

- State of the economy: distribution $G_t(b, y)$ and agg. shock $z_t$
- Market clearing: bond price $q_t$ such that

$$\int b'_t(b, y) \, dG_t(b, y) = 0, \quad \text{all } t$$

Note: agent problem depends on $G_t$ only through low-dimensional price $q_t$

# Discretized representation with general notation

- Discretize individual state $s = (b, y) \in \{s_1, ..., s_J\}$ with $J = J_b \times J_y$
- Value function, distribution, etc are $J$-dimensional vectors

$$\mathbf{v}_t = \begin{bmatrix} v_t(s_1) \\ \vdots \\ v_t(s_J) \end{bmatrix}, \qquad \mathbf{g}_t = \begin{bmatrix} g_t(s_1) \\ \vdots \\ g_t(s_J) \end{bmatrix}$$

- Consumption policy $c = \pi(s, \cdot) \Rightarrow J \times J$ transition matrix for $s$

$$\mathbf{A}_{\pi, z_t, p_t} \quad \text{with entries} \quad \Pr(s_{j'}|s_j) = \mathcal{T}_s(s_{j'}|s_j, \pi(s_j, \cdot), z_t, p_t)$$

- Law of motion for distribution $\mathbf{g}_t$ (Chapman-Kolmogorov equation):

$$\mathbf{g}_{t+1} = \mathbf{A}'_{\pi, z_t, p_t} \mathbf{g}_t, \quad z_{t+1} \sim \mathcal{T}_z(\cdot|z_t)$$

- High-dimensional state $(\mathbf{g}_t, z_t)$ is Markov

# Key difficulty: equilibrium prices are not Markov

- Low-dim equilibrium prices satisfy $p_t = P^*(\mathbf{g}_t, z_t)$: not Markov
  In Huggett, $p_t = q_t$, and $P^*(\mathbf{g}_t, z_t)$ is market clearing condition.
- ... only extremely high-dimensional $(\mathbf{g}_t, z_t)$ is
- 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!
- Our solution: use RL to approximate value/policy functions with low-dim non-Markov state variables

# What is reinforcement learning? A simple toy example

RL = learning value & policy functions from Monte Carlo ▸ RL Primer

Example: compute value function (eliminate actions & individual states for now)

$$v(p) = \mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t u(p_t) \middle| p_0 = p\right], \quad p_t = \text{exogenous stochastic process}$$

Two approaches:

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

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

# What is reinforcement learning? A simple toy example

RL = learning value & policy functions from Monte Carlo ▸ RL Primer

Example: compute value function (eliminate actions & individual states for now)

$$v(p) = \mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t u(p_t) \middle| p_0 = p\right], \quad p_t = \text{exogenous stochastic process}$$

Two approaches:

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

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

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

$$v(p) \approx \widehat{v}(p) = \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T} \beta^t u(p_t^i)$$

# What is reinforcement learning? A simple toy example

RL = learning value & policy functions from Monte Carlo ▸ RL Primer

Example: compute value function (eliminate actions & individual states for now)

$$v(p) = \mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t u(p_t) \middle| p_0 = p\right], \quad p_t = \text{exogenous stochastic process}$$

Two approaches:

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

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

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

$$v(p) \approx \widehat{v}(p) = \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T} \beta^t u(p_t^i)$$

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

# Sidestepping the Master Equation via Structural RL

Recall: states $s = (b, y)$, price $p = q$, agents choose $c_{i,t}$ to maximize

$$v_{i,0} = \max_{\{c_{i,t}\}} \mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t u(c_{i,t})\right] \text{ s.t. } s_{i,t+1} \sim \mathcal{T}_s(\cdot|s_{i,t}, c_{i,t}, z_t, p_t), \quad p_t = P^*(G_t, z_t) \tag{1}$$

# Sidestepping the Master Equation via Structural RL

Recall: states $s = (b, y)$, price $p = q$, agents choose $c_{i,t}$ to maximize

$$v_{i,0} = \max_{\{c_{i,t}\}} \mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t u(c_{i,t})\right] \text{ s.t. } s_{i,t+1} \sim \mathcal{T}_s(\cdot|s_{i,t}, c_{i,t}, z_t, p_t), \quad p_t = P^*(G_t, z_t) \tag{1}$$

Assumption 1: agents observe prices $p_t$ but not distribution $G_t(s)$ ▸ Wold repr.

- $p_t$ can be moments of $G_t$, e.g. GDP, as long as low-dimensional

# Sidestepping the Master Equation via Structural RL

Recall: states $s = (b, y)$, price $p = q$, agents choose $c_{i,t}$ to maximize

$$v_{i,0} = \max_{\{c_{i,t}\}} \mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t u(c_{i,t})\right] \text{ s.t. } s_{i,t+1} \sim \mathcal{T}_s(\cdot|s_{i,t}, c_{i,t}, z_t, p_t), \quad p_t = P^*(G_t, z_t) \tag{1}$$

Assumption 1: agents observe prices $p_t$ but not distribution $G_t(s)$ ▸ Wold repr.

- $p_t$ can be moments of $G_t$, e.g. GDP, as long as low-dimensional

Assumption 2: consumption policy $\pi$ does not condition on price histories

$$c_{i,t} = \pi(s_{i,t}, z_t, p_t)$$

- Companion paper: track price histories ($h$ lags or RNN) $\Rightarrow$ similar results

# Sidestepping the Master Equation via Structural RL

Recall: states $s = (b, y)$, price $p = q$, agents choose $c_{i,t}$ to maximize

$$v_{i,0} = \max_{\{c_{i,t}\}} \mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t u(c_{i,t})\right] \text{ s.t. } s_{i,t+1} \sim \mathcal{T}_s(\cdot|s_{i,t}, c_{i,t}, z_t, p_t), \quad p_t = P^*(G_t, z_t) \qquad (1)$$

Assumption 1: agents observe prices $p_t$ but not distribution $G_t(s)$ ▸ Wold repr.

- $p_t$ can be moments of $G_t$, e.g. GDP, as long as low-dimensional

Assumption 2: consumption policy $\pi$ does not condition on price histories

$$c_{i,t} = \pi(s_{i,t}, z_t, p_t)$$

- Companion paper: track price histories ($h$ lags or RNN) $\Rightarrow$ similar results

RL: optimize $\pi_\theta(s, z, p)$ to solve (1) on simulated paths via policy gradient.

Similarity to standard RL: don't know transition prob. of prices $p_t$

Difference to standard RL: know "local environment" $\mathcal{T}_s$ and $u$ (hybrid method)

# Restricted perceptions equilibrium

A pair of mappings $(\pi^*, P^*)$ 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 $\pi^*(s, z, p)$ to solve:

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

   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 $\pi^*$:

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

# Simulating the economy given policy $c = \pi_\theta(s, z, p)$

Given policy $\pi_\theta(s, z, p)$ and $(\mathbf{g}_0, z_0)$, want to simulate economy forward:

$$\{\mathbf{g}_t, p_t, z_t\}_{t \geq 0}$$

- important: very cheap computationally with JAX on GPUs

Recall: discrete $s \Rightarrow$ vectors $\boldsymbol{\pi}(z, p)$, $\mathbf{g}_t$, sparse transition matrix $\mathbf{A}_{\pi,z,p}$

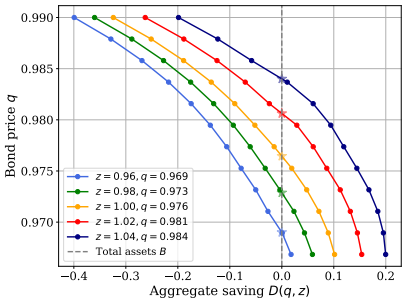
- Simulating $z_{t+1} \sim \mathcal{T}_z(\cdot|z_t)$: easy
- Simulating $\mathbf{g}_{t+1} = \mathbf{A}^\top_{\pi,z_t,p_t}\mathbf{g}_t$ given $p_t$: easy
- Simulating $p_t = P^*(\mathbf{g}_t, z_t)$: hard! Pinned down implicitly by market clearing each period

# Efficient handling of non-trivial market clearing

$D_t(z, p) = 0, \qquad D_t(z, p) := \int b'(s, z, p) dG_t(s) = \text{agg. demand for bonds}$

Key: policies $b'(s, z, p)$ double as individual demand functions

$$p_t \text{ solves } D_t(z_t, p_t) = \mathbf{b}'(z_t, p_t)^\top \mathbf{g}_t = 0$$




Market clearing is part of environment, not another loop!

# Using structural knowledge of individual dynamics

Value function for given policy $\pi(s, z, p) = \{c(s, z, p), b'(s, z, p)\}$

$$v_\pi(s, z, p) = \mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t u(\pi(s_{i,t}, z_t, p_t)) \middle| s_{i,0} = s, z_0 = z, p_0 = p\right] \qquad (*)$$

Partition state space $(s, z, p)$ into known dynamics and unknown dynamics

# Using structural knowledge of individual dynamics

Value function for given policy $\pi(s, z, p) = \{c(s, z, p), b'(s, z, p)\}$

$$v_\pi(s, z, p) = \mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t u(\pi(s_{i,t}, z_t, p_t)) \middle| s_{i,0} = s, z_0 = z, p_0 = p\right] \qquad (*)$$

Partition state space $(s, z, p)$ into known dynamics and unknown dynamics

- use transition matrix $\mathbf{A}$ to keep track of all $s$-transitions
- expectation $\mathbb{E}$ only over price trajectories $\{p_t\}_{t=0}^{\infty}$ and $\{z_t\}_{t=0}^{\infty}$

Write $v_\pi(s, z, p)$ in $(*)$ as vector $\mathbf{v}_\pi(z, p)$:

$$\mathbf{v}_\pi(z, p) = \mathbb{E}\left[\mathbf{u}_0 + \beta \mathbf{A}_0 \mathbf{u}_1 + \beta^2 \mathbf{A}_0 \mathbf{A}_1 \mathbf{u}_2 + ... | z, p\right] = \mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t \mathbf{A}_{0\to t} \mathbf{u}_t \middle| z, p\right]$$

where $\mathbf{u}_t = u(\boldsymbol{\pi}(z_t, p_t))$ and $\mathbf{A}_t = \mathbf{A}_{\pi, z_t, p_t}$ and $\mathbf{A}_{0\to t} = \mathbf{A}_0 \cdots \mathbf{A}_{t-1}$

# Summary: problem to be solved

Find optimal policy $\pi(s, z, p)$ that maximizes

$$\mathbf{v}_\pi(z, p) = \mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t \mathbf{A}_{\pi,0\to t} u(\boldsymbol{\pi}(z_t, p_t)) \middle| z_0 = z, p_0 = p\right]$$

taking as given evolution of equilibrium prices $p_t$ (sg[·] = stop-gradient operator)

$$p_t = \text{sg}\left[P^*(\mathbf{g}_t, z_t)\right], \quad \mathbf{g}_{t+1} = \mathbf{A}^\top_{\pi,z_t,p_t}\mathbf{g}_t, \quad z_{t+1} \sim \mathcal{T}_z(\cdot|z_t), \quad t = 0, 1, \ldots$$

with $(\mathbf{g}_0, z_0)$ given and $\mathbf{A}_{\pi,0\to t} = \mathbf{A}_{\pi,z_0,p_0} \cdots \mathbf{A}_{\pi,z_{t-1},p_{t-1}}$

Key observation:

- State of economy = $(\mathbf{g}, z)$ = very high-dimensional
- But state in value/policy functions = $(s, z, p)$ = very low-dimensional!
- No perceived law of motion, No inner loop / outer loop ($\neq$ Krusell-Smith)
- GE problem only mildly more difficult than PE

# Structural policy gradient (SPG) method for maximizing $\mathbf{v}_\pi$

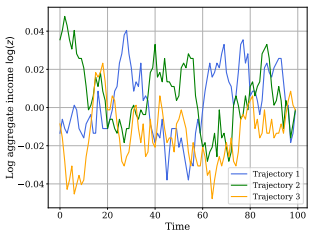
Find optimal policy $\boldsymbol{\pi}(z, p)$ that maximizes estimate of $\mathbb{E}_{z_0 \sim \psi_z, p_0 \sim \psi_p}[\mathbf{v}_\pi(z_0, p_0)]$:

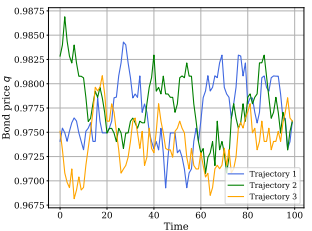
$$\widehat{\mathbf{v}}_\pi = \frac{1}{N} \sum_{n=1}^{N} \left[ \sum_{t=0}^{T} \beta^t \mathbf{A}^n_{\pi, 0 \to t} u(\boldsymbol{c}(z^n_t, p^n_t)) \right]$$

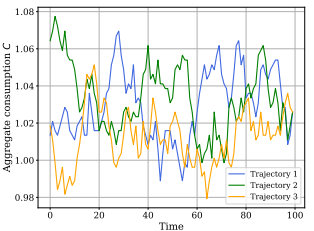
with $N$ price trajectories $p^n_t$ sample from interacting with environment (rollout):

$$p^n_t = \text{sg}\left[P^*(\mathbf{g}^n_t, z^n_t)\right], \quad \mathbf{g}^n_{t+1} = \mathbf{A}^\top_{\pi, z^n_t, p^n_t} \mathbf{g}^n_t, \quad z^n_{t+1} \sim \mathcal{T}_z(\cdot | z^n_t), \quad t = 0, 1, \ldots$$

with $\mathbf{g}^n_0 \sim \psi_g(\cdot)$, $z^n_0 \sim \psi_z(\cdot)$ and $\mathbf{A}^n_{\pi, 0 \to t} = \mathbf{A}_{\pi, z^n_0, p^n_0} \cdots \mathbf{A}_{\pi, z^n_{t-1}, p^n_{t-1}}$



# Structural policy gradient (SPG) method for maximizing $\mathbf{v}_\pi$

Find optimal policy $\boldsymbol{\pi}(z, p)$ that maximizes estimate of $\mathbb{E}_{z_0\sim\psi_z, p_0\sim\psi_p}[\mathbf{v}_\pi(z_0, p_0)]$:

$$\widehat{\mathbf{v}}_\pi = \frac{1}{N}\sum_{n=1}^{N}\left[\sum_{t=0}^{T}\beta^t \mathbf{A}^n_{\pi,0\to t}\, u(\boldsymbol{c}(z^n_t, p^n_t))\right]$$

with $N$ price trajectories $p^n_t$ sample from interacting with environment (rollout):

$$p^n_t = \text{sg}\left[P^*(\mathbf{g}^n_t, z^n_t)\right], \quad \mathbf{g}^n_{t+1} = \mathbf{A}^\top_{\pi,z^n_t,p^n_t}\mathbf{g}^n_t, \quad z^n_{t+1}\sim\mathcal{T}_z(\cdot|z^n_t), \quad t = 0, 1, \ldots$$

with $\mathbf{g}^n_0 \sim \psi_g(\cdot)$, $z^n_0 \sim \psi_z(\cdot)$ and $\mathbf{A}^n_{\pi,0\to t} = \mathbf{A}_{\pi,z^n_0,p^n_0}\cdots\mathbf{A}_{\pi,z^n_{t-1},p^n_{t-1}}$

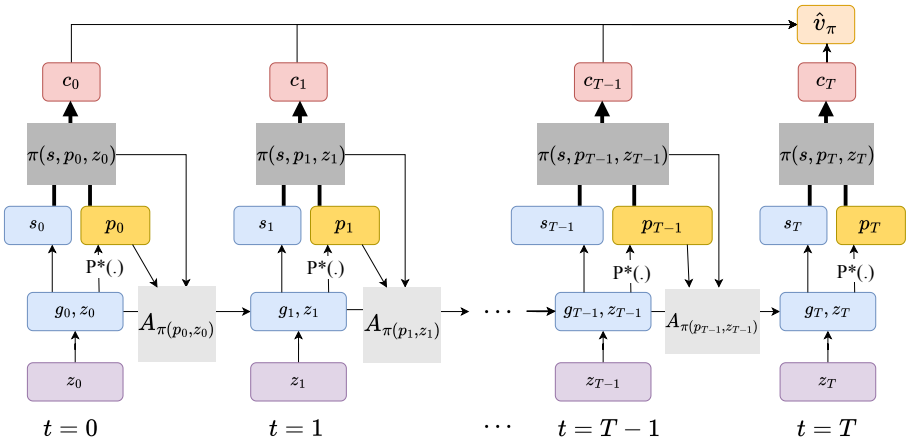
In practice, maximize scalar objective using gradient ascent:

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbf{d}_0^\top\widehat{\mathbf{v}}_\pi = \sum_{j=1}^{J} d_0(s_j)\widehat{v}_\pi(s_j), \quad d_0(s) = \text{uniform dist. over } s$$

Policy either grid based $\boldsymbol{\theta} = [\boldsymbol{\pi}(z_1, p_1), \ldots, \boldsymbol{\pi}(z_K, p_L)]$ or neural net $\pi(s, z, p; \boldsymbol{\theta})$

- low dim. $\Rightarrow$ grid-based method works well, no need for neural nets!

# Computational graph for construction of $\widehat{\mathbf{v}}_\pi$



# Computational experiments

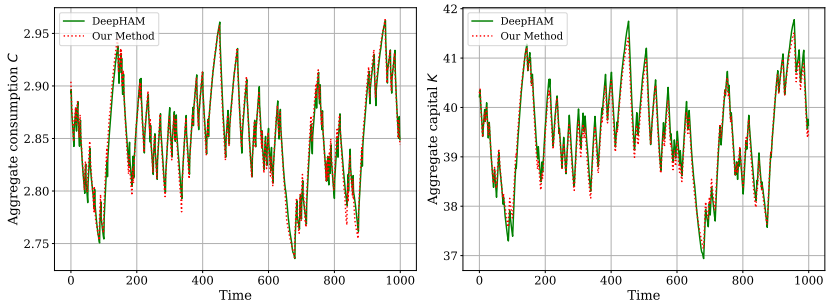
# Runtimes

- 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 | 462.3 | 10 | 36.77 |
| Huggett with agg. shocks | 573.0 | 10 | 45.91 |
| Portfolio choice (KS 1997) | 637.5 | 10 | 84.3 |
| HANK with agg. shocks | 707.5 | 10 | 246.49 |
| Partial Equilibrium (Huggett) | 355.8 | 10 | 24.50 |

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

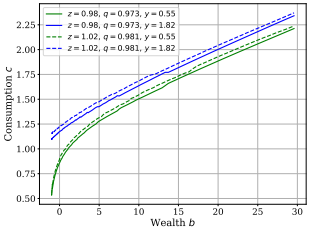
# KS model: Structural RL method recovers RE solutions

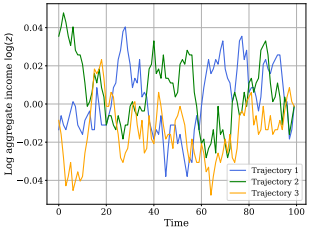


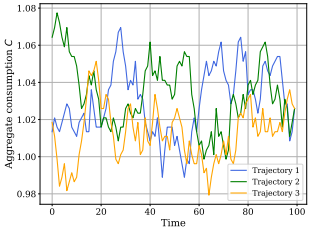

RE solutions are obtained with a deep learning based method (DeepHAM).

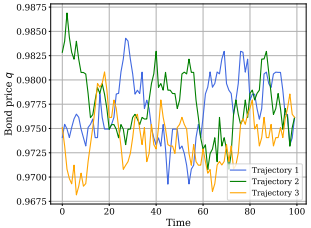
details

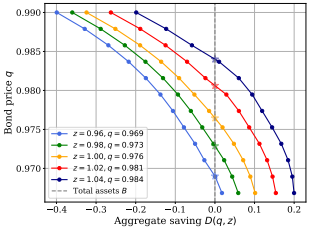
# Huggett: Simulated trajectories under optimal policy

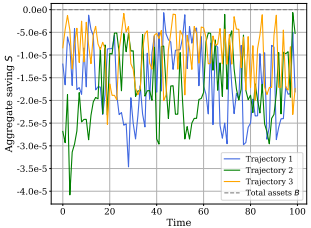


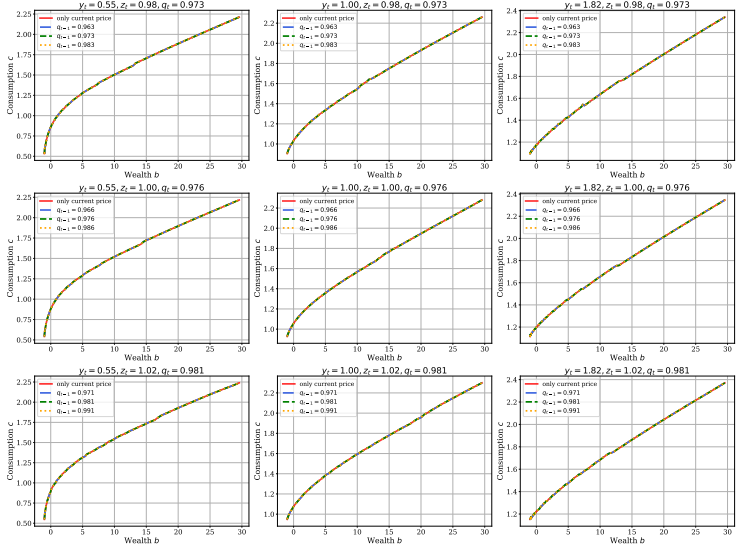




Euler equation error in relative consumption units: 0.096%.

# Adding one lagged price $p_{t-1}$ into state space



# Portfolio choice: Huggett with bond and risky stock

- Two-asset Huggett model: risk-free bond + risky stock
- Three individual states $(b_{i,t}, e_{i,t}, y_{i,t})$, two prices $(q_t^b, q_t^e)$
- Households choose consumption $c_{i,t}$, bond $b_{i,t+1}$, and equity $e_{i,t+1}$ to maximize

$$v_{i,0} = \max_{\{c_{i,t}, b_{i,t+1}, e_{i,t+1}\}} \mathbb{E}_0 \sum_{t=0}^{\infty} \beta^t u(c_{i,t}) \quad \text{subject to}$$

$$c_{i,t} + q_t^b b_{i,t+1} + q_t^e e_{i,t+1} = b_{i,t} + [q_t^e + d(z_t)] e_{i,t} + y_{i,t}, \quad y_{i,t+1} \sim \mathcal{T}_y(\cdot|y_{i,t})$$

with $b_{i,t+1} \geq \underline{b}$, $e_{i,t+1} \geq 0$, dividend $d(z_t) = \bar{d} z_t$, $z_{t+1} \sim \mathcal{T}_z(\cdot|z_t)$

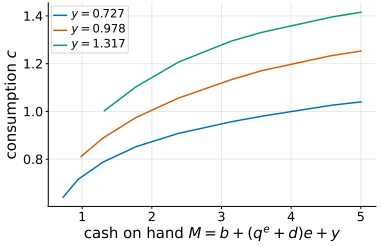
- Two asset prices clear two asset markets: prices $(q_t^b, q_t^e)$ such that

$$\int b_{t+1}(b, e, y)\, dG_t(b, e, y) = \bar{B}, \quad \int e_{t+1}(b, e, y)\, dG_t(b, e, y) = 1, \quad \text{all } t$$

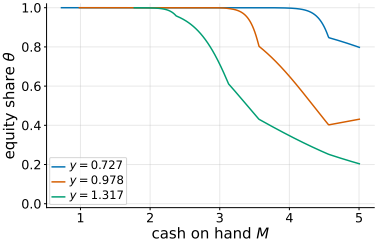
▸ Krusell-Smith 1997 (Easier problem with only one non-trivial market clearing)

# Two-asset Huggett: policy functions and agent distribution

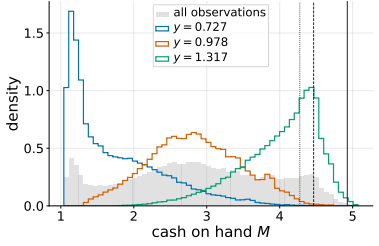
Consumption policy




Equity-share policy $\theta := q^e e' / (q^e e' + q^b b')$

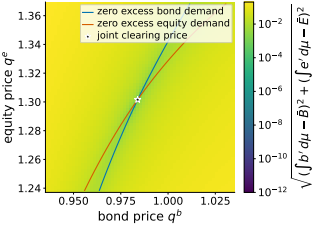



Cash-on-hand density

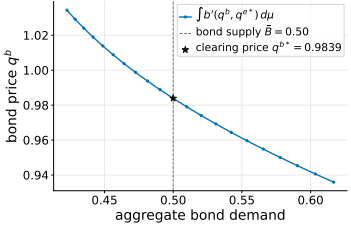


# Two-asset Huggett: asset prices and market clearing

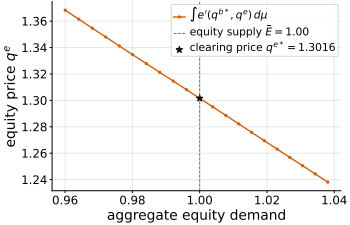
Two prices clear two markets



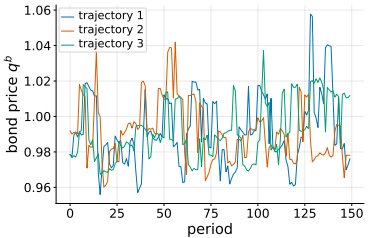

Bond market clearing



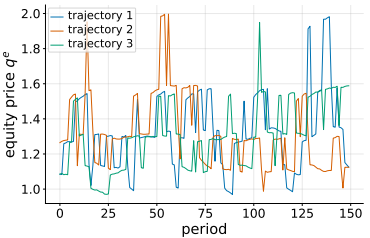

Equity market clearing




Bond price $q_t^b$




Equity price $q_t^e$



# HANK: policy gradient method for both households & firms

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

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

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

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

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

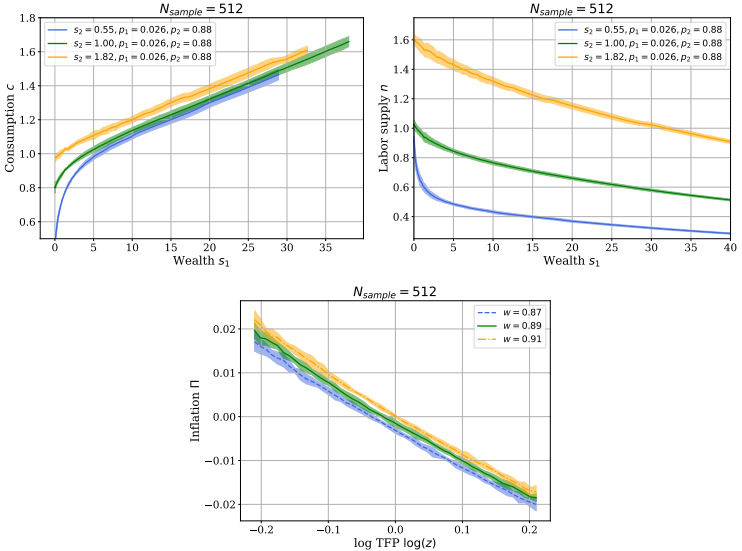
Our solution: solve firm price-setting problem using policy gradient method

Symmetric treatment of firms and households, update policies simultaneously

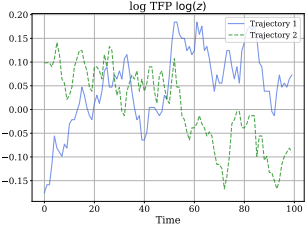
In practice: good convergence properties

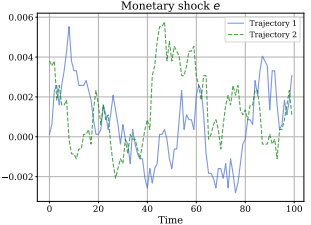
▸ details

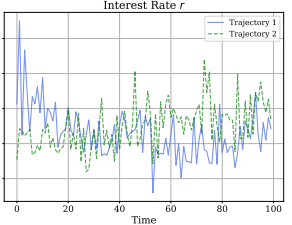
# HANK: Household and firm policy functions

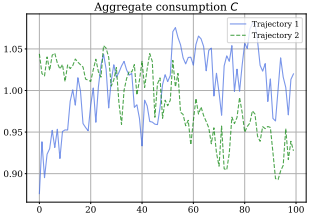


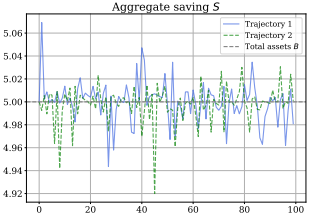
# HANK simulations

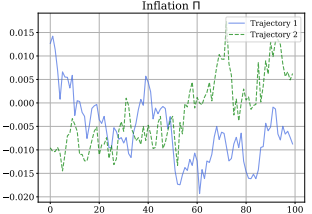


# Summary

Efficient and flexible global solution method for HA models with aggregate risk

"Structural RL": hybrid RL method that exploits known model structure

- RL about eqm 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

# Summary

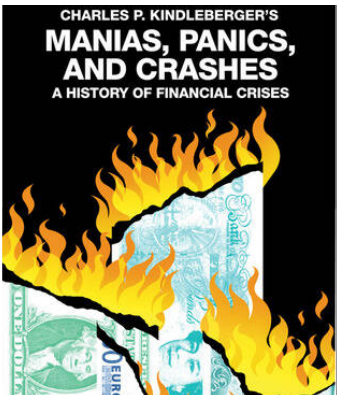
Efficient and flexible global solution method for HA models with aggregate risk

"Structural RL": hybrid RL method that exploits known model structure

- RL about eqm 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($\infty$)

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

Step 2: if VMA($\infty$) is invertible, it can be expressed as a VAR($\infty$) 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 $p_{t+1} \sim \mathcal{T}_p(\cdot|p_t)$

# Key difficulty: equilibrium prices are not Markov

- Equilibrium prices satisfy back

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

- 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

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

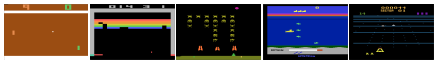
- Without Markov transition prob's: cannot even write Bellman equation!
- But what if there was a way to approximate value and policy functions with $p_t$ process for which there are no Markov transition probabilities?

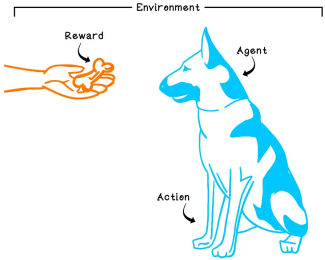
# Brief primer on reinforcement learning

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 $\{x_1, x_2, \ldots, x_N\}$

$$\mathbb{E}[x] \approx \bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i$$

# 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):

$$\bar{x}_k = \frac{1}{k} \sum_{i=1}^{k} x_i \quad \text{satisfies} \quad \bar{x}_k = \bar{x}_{k-1} + \frac{1}{k} (x_k - \bar{x}_{k-1}), \quad \frac{1}{k} = \text{“learning rate”}$$

# 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 $\left\{p_t^i\right\}_{t=0}^{T}$

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

# Computing a value function

For now: eliminate actions and individual states

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

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 $\left\{p_t^i\right\}_{t=0}^{T}$

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

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

# Portfolio Choice: Krusell-Smith 1997

# Portfolio choice: Krusell-Smith 1997 back

- RBC model with household portfolio choice
- Three individual states $(b_{i,t}, k_{i,t}, y_{i,t})$, three prices $(q_t, R_t, w_t)$
- Households choose consumption $c_{i,t}$ to maximize

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

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

- Rental rate and wage

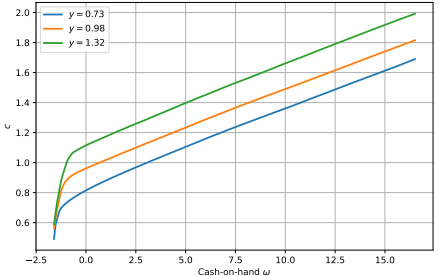
$$R_t = \alpha z_t K_t^{\alpha-1} + 1 - \delta, \quad w_t = (1-\alpha) z_t K_t^{\alpha}, \quad K_t = \int k \; dG_t(b,k,y)$$

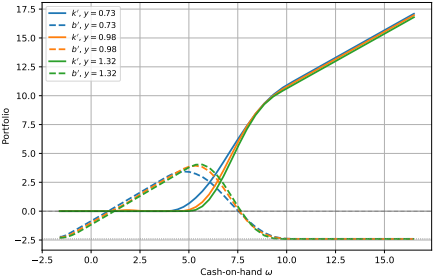
- Bond price $q_t$ such that

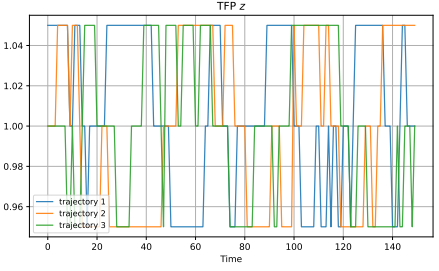
$$\int b_t'(b,k,y) \; dG_t(b,k,y) = 0, \quad \text{all } t$$

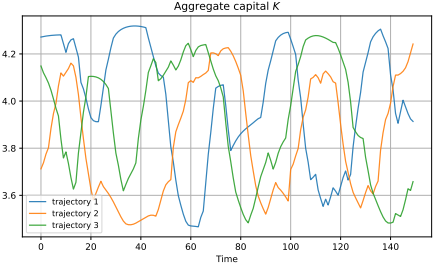
# Some simulated trajectories under the optimal policy

back



# Krusell-Smith model: details

# Computational experiments: Krusell-Smith model

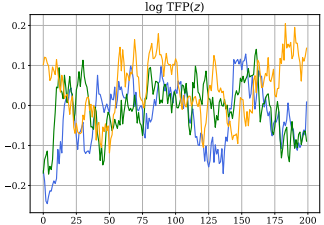
back

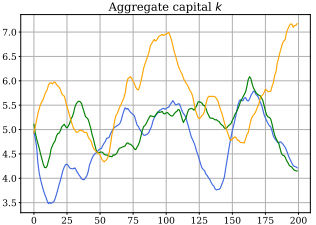
| Parameter | Description | Value |
|---|---|---|
| $\alpha$ | Capital share | 0.36 |
| $\delta$ | Capital depreciation rate | 0.08 |
| $\gamma$ | Discount factor | 0.95 |
| $\sigma$ | Coefficient of relative risk aversion | 3 |
| $\rho_z$ | Persistence of AR(1) for $z_t$ (log TFP) | 0.9 |
| $\nu_z$ | Volatility of AR(1) for $z_t$ (log TFP) | 0.03 |

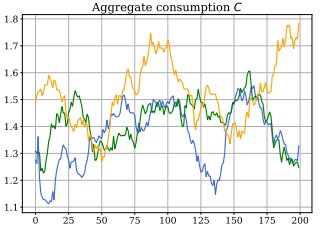
| Hyperparameter | Description | Value |
|---|---|---|
| $J_{s_1}$ | Number of $s_1$ (wealth) grid points | 200 |
| $J_{s_2}$ | Number of $s_2$ (income) states | 3 |
| $J_{p_1}$ | Number of $p_1$ (rental rate) grid points | 50 |
| $J_{p_2}$ | Number of $p_2$ (wage) grid points | 70 |
| $N$ | Sample size = number of $p$ trajectories | 32,128,512,... |
| $T$ | Time truncation s.t. $\beta^T < 0.001$ | 135 |
| $\epsilon$ | Convergence criterion on $\widehat{\mathbf{v}}_\pi$ | $1 \times 10^{-4}$ |
| $\eta_{\texttt{ini}}$ | Initial learning rate | $5 \times 10^{-4}$ |
| $\eta_{\texttt{decay}}$ | Learning rate decay (exponential) | 0.5 |

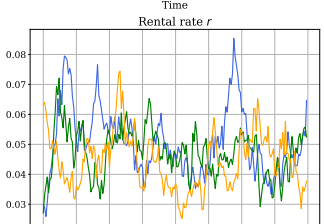
# KS model: simulated trajectories under the optimal policy

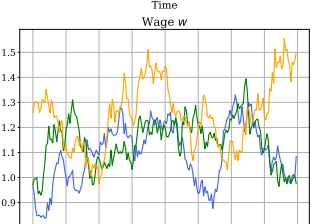
back

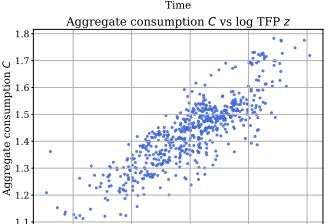


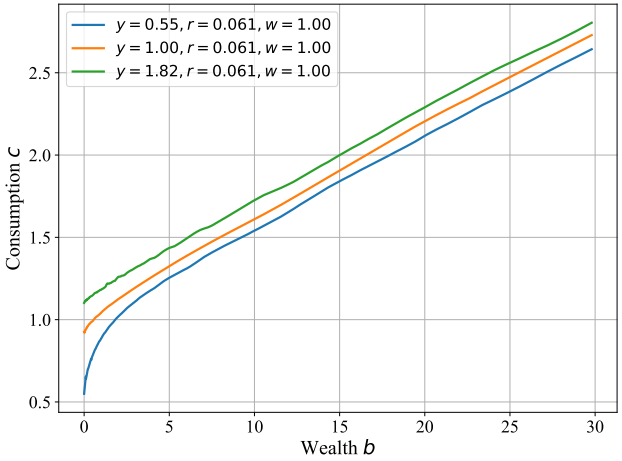



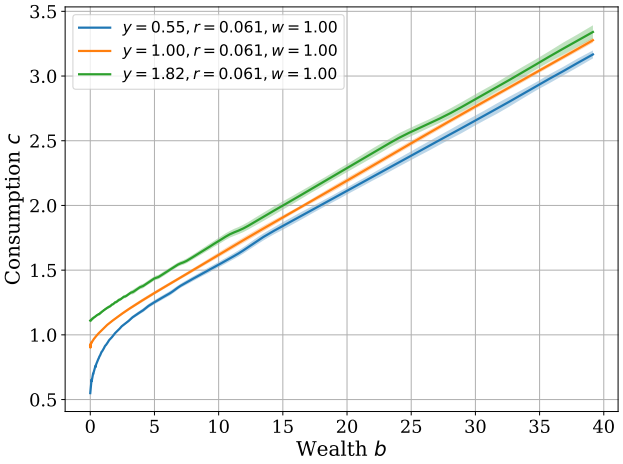
# Consumption policy function: single run with $N_{\text{sample}} = 128$

back



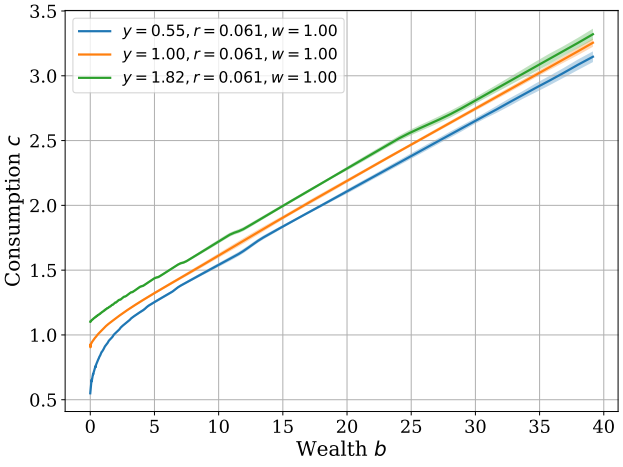
# Consumption policy: multiple runs with $N_{\text{sample}} = 128$

back



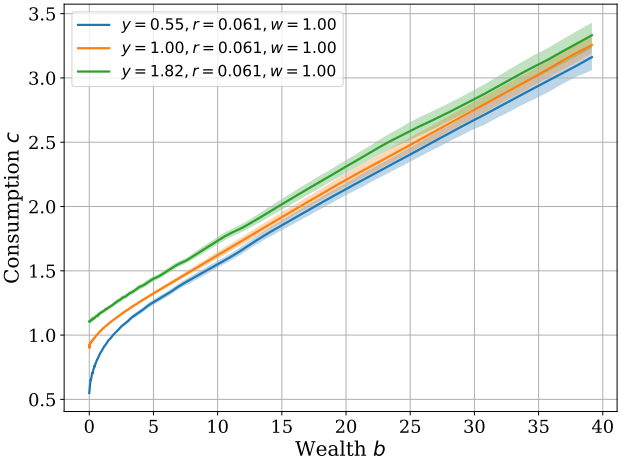
# Larger sample size $N = 512 \Rightarrow$ more precise estimate

back



# Smaller sample size $N = 32 \Rightarrow$ noisier estimate

back



# HANK: details

# HANK with forward-looking Phillips curve

back

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

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

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

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

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

# HANK with forward-looking Phillips curve

back

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

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

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

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

$$J_{j,0} = \max_{\{P_{j,t}\}} \mathbb{E}_0 \left[ \sum_{t=0}^{\infty} \Lambda_{0 \to t} \left\{ \text{Profits} \left( \frac{P_{j,t}}{P_t}, \frac{w_t}{z_t}, Y_t \right) - \frac{\theta}{2} \left( \frac{P_{j,t} - P_{j,t-1}}{P_{j,t-1}} \right)^2 \right\} \right]$$

# HANK with forward-looking Phillips curve

back

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

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

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

Or in terms of relative price $\phi_{j,t} = P_{j,t}/P_t$ and inflation $\Pi_{j,t} = (P_{j,t} - P_{j,t-1})/P_{j,t}$

$$J_{j,0} = \max_{\{\Pi_{j,t}\}} \mathbb{E}_0 \left[ \sum_{t=0}^{\infty} \Lambda_{0\to t} \left\{ \text{Profits} \left( \phi_{j,t}, \frac{w_t}{z_t}, Y_t \right) - \frac{\theta}{2} \left( \Pi_{j,t} \right)^2 \right\} \right], \quad \phi_{j,t} = \frac{1 + \Pi_{j,t}}{1 + \Pi_t} \phi_{j,t-1}$$

# HANK: formal firm block

back

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

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

Firm block: states $(z, Y)$ and prices $p = (q, w)$

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

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

Symmetric treatment of firms and households, update policies simultaneously

In practice: good convergence properties