Using the Sequence-Space Jacobian to Solve and Estimate Heterogeneous-Agent Models

Adrien Auclert, Bence Bardóczy, Matt Rognlie, Ludwig Straub

May 2019

This paper

Q: How should we solve heterogeneous agent (HA) models with aggregate shocks in discrete time?

- Standard approach [Krusell-Smith, ...]: **state space**
  - wealth distribution as a state
- Alternative [Boppart-Krusell-Mitman, ...]: **sequence space**
  - paths for aggregates with perfect foresight ("M.I.T shocks")

This paper

Q: How should we solve heterogeneous agent (HA) models with aggregate shocks in discrete time?

- Standard approach [Krusell-Smith, ...]: **state space**
  - wealth distribution as a state
- Alternative [Boppart-Krusell-Mitman, ...]: **sequence space**
  - paths for aggregates with perfect foresight ("M.I.T shocks")

- Leading state space methods assume linear policy functions w.r.t. aggregate variables [Reiter, Ahn et al., ...]

  $\Rightarrow$ certainty equivalence [Theil 1957, Boppart-Krusell-Mitman 2018]

  $\Rightarrow$ solution in sequence space = solution in state space

This paper

Q: How should we solve heterogeneous agent (HA) models with aggregate shocks in discrete time?

- Standard approach [Krusell-Smith, ...]: **state space**
  - wealth distribution as a state
- Alternative [Boppart-Krusell-Mitman, ...]: **sequence space**
  - paths for aggregates with perfect foresight (“M.I.T shocks”)

- Leading state space methods assume linear policy functions w.r.t. aggregate variables [Reiter, Ahn et al., ...]

  $\Rightarrow$ certainty equivalence [Theil 1957, Boppart-Krusell-Mitman 2018]

  $\Rightarrow$ solution in sequence space = solution in state space

This paper: how to compute **linear sequence space solution**

- Comes with extensively documented code & notebooks
- Computation time $< \frac{1}{100}$ of existing methods
- Apply to: bus. cycle stats, estimation, nonlin. transitions

Key idea behind our method: the Jacobian is all you need

- Start from model in nonlinear sequence space:

$$H(\mathbf{X}, \mathbf{Z}) = 0$$

$\mathbf{X} \equiv$paths of endog. outcomes, $\mathbf{Z} \equiv$exog. shocks/params

Key idea behind our method: the Jacobian is all you need

- Start from model in nonlinear sequence space:

$$H(\mathbf{X}, \mathbf{Z}) = 0$$

$\mathbf{X} \equiv$paths of endog. outcomes, $\mathbf{Z} \equiv$exog. shocks/params

- **Get impulse response**: linear solution around st. state is

$$d\mathbf{X} = -\mathbf{H}_X^{-1}\mathbf{H}_Z d\mathbf{Z}$$

Key idea behind our method: the Jacobian is all you need

- Start from model in nonlinear sequence space:

$$H(\mathbf{X}, \mathbf{Z}) = 0$$

$\mathbf{X}$ $\equiv$paths of endog. outcomes, $\mathbf{Z}$ $\equiv$exog. shocks/params

- **Get impulse response**: linear solution around st. state is

$$d\mathbf{X} = -\mathbf{H}_X^{-1}\mathbf{H}_Z d\mathbf{Z}$$

1. Represent model as **Directed Acyclic Graph**, nodes either
   (a) "Recursive blocks" with analytical Jacobian
   (b) "Heterogeneous-agent blocks" $\rightarrow$ new formula for Jacobian!
2. Follow DAG to compose (a) and (b) $\rightarrow$ $\mathbf{H}_X$, $\mathbf{H}_Z$ $\rightarrow$ $d\mathbf{X}$

Key idea behind our method: the Jacobian is all you need

- Start from model in nonlinear sequence space:

$$H(\mathbf{X}, \mathbf{Z}) = 0$$

  $\mathbf{X}$ $\equiv$paths of endog. outcomes, $\mathbf{Z}$ $\equiv$exog. shocks/params

- **Get impulse response**: linear solution around st. state is

$$d\mathbf{X} = -\mathbf{H}_X^{-1}\mathbf{H}_Z d\mathbf{Z}$$

  1. Represent model as **Directed Acyclic Graph**, nodes either
     (a) "Recursive blocks" with analytical Jacobian
     (b) "Heterogeneous-agent blocks" $\to$ new formula for Jacobian!
  2. Follow DAG to compose (a) and (b) $\to$ $\mathbf{H}_X$, $\mathbf{H}_Z$ $\to$ $d\mathbf{X}$

- **Get second moments** from MA(∞) representation

Key idea behind our method: the Jacobian is all you need

- Start from model in nonlinear sequence space:

$$H(\mathbf{X}, \mathbf{Z}) = 0$$

$\mathbf{X} \equiv$paths of endog. outcomes, $\mathbf{Z} \equiv$exog. shocks/params

- **Get impulse response**: linear solution around st. state is

$$d\mathbf{X} = -\mathbf{H}_X^{-1}\mathbf{H}_Z d\mathbf{Z}$$

1. Represent model as **Directed Acyclic Graph**, nodes either
   (a) "Recursive blocks" with analytical Jacobian
   (b) "Heterogeneous-agent blocks" $\to$ new formula for Jacobian!
2. Follow DAG to compose (a) and (b) $\to$ $\mathbf{H}_X$, $\mathbf{H}_Z$ $\to$ $d\mathbf{X}$

- **Get second moments** from MA(∞) representation
- **Get likelihood** from 2nd moments $\to$ **estimate the model**

Key idea behind our method: the Jacobian is all you need

- Start from model in nonlinear sequence space:

$$H(\mathbf{X}, \mathbf{Z}) = 0$$

$\mathbf{X} \equiv$paths of endog. outcomes, $\mathbf{Z} \equiv$exog. shocks/params

- **Get impulse response**: linear solution around st. state is

$$d\mathbf{X} = -\mathbf{H}_X^{-1}\mathbf{H}_Z d\mathbf{Z}$$

1. Represent model as **Directed Acyclic Graph**, nodes either
   (a) "Recursive blocks" with analytical Jacobian
   (b) "Heterogeneous-agent blocks" $\to$ new formula for Jacobian!
2. Follow DAG to compose (a) and (b) $\to$ $\mathbf{H}_X$, $\mathbf{H}_Z$ $\to$ $d\mathbf{X}$

- **Get second moments** from MA(∞) representation
- **Get likelihood** from 2nd moments $\to$ **estimate the model**
- "All in one go" computation $\Rightarrow$ large speed gain!

Literature on dynamic GE with idiosyncratic & aggregate risk

- **State space methods**
  - **nonlinear:** Krusell-Smith 1998, Khan-Thomas 2008, Algan, Allais-Den Haan 2010, Fernandez-Villaverde-Nuño-Hurtado 2019...
  - **linear:** Reiter 2009, McKay-Reis 2016, Winberry 2018, Ahn-Kaplan-Moll-Winberry-Wolf 2018, Bayer-Luetticke 2018...
- **Sequence space methods**
  - **nonlinear**: Conesa-Krueger 1999, ..., McKay-Nakamura-Steinsson 2016, Guerrieri-Lorenzoni 2017, Farhi-Werning 2017, Kaplan-Moll-Violante 2018, Auclert-Rognlie 2018, Straub 2018, Hagedorn-Manovskii-Mitman 2019, ...
  - **linear**: Boppart-Krusell-Mitman 2018
- **Sufficient statistics for GE**: Auclert-Rognlie 2018, Auclert-Rognlie-Straub 2018, Guren-McKay-Nakamura-Steinsson 2019, Wolf 2019

Outline

All you need is the Jacobian

A first example: Krusell Smith (1998)

- For concreteness we present our methods with examples
- We start with a canonical HA model with aggreg. shocks
    - Krusell-Smith (1998): “RBC + HA”
- We set up the model in the sequence space
    - Assume perfect foresight w.r.t. aggregate vars
    - Idiosyncratic states: capital k_{-}, idiosyncratic skill e
    - Aggregate state variable: time $t = 0, 1, \ldots$

Krusell-Smith model: setup

- Mass 1 of het. households, $u(c) = \frac{c^{1-\sigma}}{1-\sigma}$, exogenous labor
- Idiosyncratic ability e, transition $\mathcal{P}(e, e')$

$$\begin{aligned} V_t(e, k_-) = \quad \max_{c,k} \quad & u(c) + \beta \sum_{e'} V_{t+1}\left(e', k\right) \mathcal{P}\left(e, e'\right) \\ \text{s.t.} \quad & c + k = (1 + r_t) k_- + w_t e l \\ & k \geq 0 \end{aligned}$$

$\Rightarrow$ policies $c_t(e, k_-)$ and $k_t(e, k_-)$

Krusell-Smith model: setup

- Mass 1 of het. households, $u(c) = \frac{c^{1-\sigma}}{1-\sigma}$, exogenous labor
- Idiosyncratic ability e, transition $\mathcal{P}(e, e')$

$$
\begin{aligned}
V_t(e, k_-) = \quad \max_{c,k} \quad & u(c) + \beta \sum_{e'} V_{t+1}(e', k)\, \mathcal{P}(e, e') \\
\text{s.t.} \quad & c + k = (1 + r_t)\, k_- + w_t e l \\
& k \geq 0
\end{aligned}
$$

$\Rightarrow$ policies $c_t(e, k_-)$ and $k_t(e, k_-)$

- Distrib. at start of t: $\Psi_t(e, k_-) = \Pr(e_t = e, k_{t-1} \leq k_-)$
- Law of motion:

$$
\Psi_{t+1}(e', k) = \sum_e \Psi_t\left(e', [k_t]^{-1}(k, e)\right) \mathcal{P}(e, e')
$$

Krusell-Smith model: setup

- Mass 1 of het. households, $u(c) = \frac{c^{1-\sigma}}{1-\sigma}$, exogenous labor
- Idiosyncratic ability e, transition $\mathcal{P}(e, e')$

$$\begin{aligned} V_t(e, k_-) = \quad & \max_{c,k} \quad u(c) + \beta \sum_{e'} V_{t+1}(e', k) \mathcal{P}(e, e') \\ & \text{s.t.} \qquad c + k = (1 + r_t) k_- + w_t e l \\ & \qquad\qquad k \geq 0 \end{aligned}$$

$\Rightarrow$ policies $c_t(e, k_-)$ and $k_t(e, k_-)$

- Distrib. at start of t: $\Psi_t(e, k_-) = \Pr(e_t = e, k_{t-1} \leq k_-)$
- Law of motion:

$$\Psi_{t+1}(e', k) = \sum_e \Psi_t\left(e', [k_t]^{-1}(k, e)\right) \mathcal{P}(e, e')$$

- p. comp. firms: $Y_t = Z_t K_{t-1}^{\alpha} L_t^{1-\alpha}$, $r_t + \delta = \frac{\partial Y_t}{\partial K_{t-1}}$, $w_t = \frac{\partial Y_t}{\partial L_t}$

Krusell-Smith model: solution

- Given $\Psi_0(\cdot,\cdot)$, aggregate capital holdings at end of t are

$$\mathcal{K}_t(\{r_s, w_s\}) = \int k_t(e, k_-)\, d\Psi_t(e, k_-)$$

capital function, depends on entire sequence $\{r_s, w_s\}_{s\geq 0}$

[Farhi-Werning, Kaplan-Moll-Violante, Auclert-Rognlie-Straub]

Krusell-Smith model: solution

- Given $\Psi_0(\cdot,\cdot)$, aggregate capital holdings at end of t are

$$\mathcal{K}_t\left(\{r_s, w_s\}\right) = \int k_t\left(e, k_-\right) d\Psi_t\left(e, k_-\right)$$

capital function, depends on entire sequence $\{r_s, w_s\}_{s\geq 0}$

[Farhi-Werning, Kaplan-Moll-Violante, Auclert-Rognlie-Straub]

- **General equilibrium** characterized by
  - Labor market clearing $L_t = \pi l$
  - Factor prices $r_t + \delta = \alpha Z_t\left(\frac{K_{t-1}}{L_t}\right)^{\alpha-1}$, $w_t = (1-\alpha) Z_t\left(\frac{K_{t-1}}{L_t}\right)^{\alpha}$
  - Capital market clearing

$$K_t = \mathcal{K}_t\left(\{r_s, w_s\}\right) \quad \forall t$$

  - Walras's law $\rightarrow$ goods market clearing is redundant

Krusell-Smith model: linear sequence space solution

- Let $\mathbf{Z} \equiv (Z_0, Z_1, \ldots)'$, $\mathbf{K} \equiv (K_0, K_1, \ldots)'$, $\mathbf{H} \equiv (H_0, H_1, \ldots)'$ with

$$H_t(\mathbf{K}, \mathbf{Z}) \equiv \mathcal{K}_t\left(\left\{\alpha Z_s\left(\frac{K_{s-1}}{\pi l}\right)^{\alpha-1} - \delta, (1-\alpha) Z_s\left(\frac{K_{s-1}}{\pi l}\right)^{\alpha}\right\}\right) - K_t$$

Krusell-Smith model: linear sequence space solution

- Let $\mathbf{Z} \equiv (Z_0, Z_1, \ldots)'$, $\mathbf{K} \equiv (K_0, K_1, \ldots)'$, $\mathbf{H} \equiv (H_0, H_1, \ldots)'$ with

$$H_t(\mathbf{K}, \mathbf{Z}) \equiv \mathcal{K}_t\left(\left\{\alpha Z_s \left(\frac{K_{s-1}}{\pi l}\right)^{\alpha-1} - \delta, (1-\alpha) Z_s \left(\frac{K_{s-1}}{\pi l}\right)^{\alpha}\right\}\right) - K_t$$

- Given K_{-1} and $\mathbf{Z}$, nonlinear seq. space solution is $\mathbf{K}$ s.t.

$$\mathbf{H}(\mathbf{K}, \mathbf{Z}) = 0 \tag{1}$$

- Let $\mathbf{Z} \equiv (Z_0, Z_1, \ldots)'$, $\mathbf{K} \equiv (K_0, K_1, \ldots)'$, $\mathbf{H} \equiv (H_0, H_1, \ldots)'$ with

$$H_t(\mathbf{K}, \mathbf{Z}) \equiv \mathcal{K}_t\left(\left\{\alpha Z_s\left(\frac{K_{s-1}}{\pi l}\right)^{\alpha-1} - \delta, (1-\alpha) Z_s\left(\frac{K_{s-1}}{\pi l}\right)^{\alpha}\right\}\right) - K_t$$

- Given K_{-1} and $\mathbf{Z}$, nonlinear seq. space solution is $\mathbf{K}$ s.t.

$$\mathbf{H}(\mathbf{K}, \mathbf{Z}) = 0 \tag{1}$$

- First-order solution around steady state $(\mathbf{K}_{ss}, \mathbf{Z}_{ss})$:

$$\mathbf{H}_K(\mathbf{K}_{ss}, \mathbf{Z}_{ss})\, d\mathbf{K} = -\mathbf{H}_Z(\mathbf{K}_{ss}, \mathbf{Z}_{ss})\, d\mathbf{Z}$$

Krusell-Smith model: linear sequence space solution

- Let $\mathbf{Z} \equiv (Z_0, Z_1, \ldots)'$, $\mathbf{K} \equiv (K_0, K_1, \ldots)'$, $\mathbf{H} \equiv (H_0, H_1, \ldots)'$ with

$$H_t(\mathbf{K}, \mathbf{Z}) \equiv \mathcal{K}_t\left(\left\{\alpha Z_s\left(\frac{K_{s-1}}{\pi l}\right)^{\alpha-1} - \delta, (1-\alpha) Z_s\left(\frac{K_{s-1}}{\pi l}\right)^{\alpha}\right\}\right) - K_t$$

- Given K_{-1} and $\mathbf{Z}$, nonlinear seq. space solution is $\mathbf{K}$ s.t.

$$\mathbf{H}(\mathbf{K}, \mathbf{Z}) = 0 \tag{1}$$

- First-order solution around steady state $(\mathbf{K}_{ss}, \mathbf{Z}_{ss})$:

$$\mathbf{H}_K(\mathbf{K}_{ss}, \mathbf{Z}_{ss})\, d\mathbf{K} = -\mathbf{H}_Z(\mathbf{K}_{ss}, \mathbf{Z}_{ss})\, d\mathbf{Z}$$

- To get $d\mathbf{K}$, all we need are Jacobians $\mathbf{H}_K$ and $\mathbf{H}_Z$, where e.g.

$$(\mathbf{H}_K)_{t,s} = \frac{\partial H_t}{\partial K_s}(\mathbf{K}_{ss}, \mathbf{Z}_{ss}) = \frac{\partial \mathcal{K}_t}{\partial r_{s+1}}\frac{\partial r_{s+1}}{\partial K_s} + \frac{\partial \mathcal{K}_t}{\partial w_{s+1}}\frac{\partial w_{s+1}}{\partial K_s} - 1_{\{s=t\}}$$

Capital function Jacobians

- Can compute analytically

$$\frac{\partial r_{s+1}}{\partial K_s} = \alpha\left(\alpha - 1\right) Z_{ss}\left(\frac{K_{ss}}{\pi l}\right)^{\alpha-2}$$

$$\frac{\partial w_{s+1}}{\partial K_s} = \left(1 - \alpha\right)\alpha Z_{ss}\left(\frac{K_{ss}}{\pi l}\right)^{\alpha-1}$$

and similarly for derivatives w.r.t. Z

Capital function Jacobians

- Can compute analytically

$$\frac{\partial r_{s+1}}{\partial K_s} = \alpha\left(\alpha - 1\right) Z_{ss}\left(\frac{K_{ss}}{\pi l}\right)^{\alpha-2}$$

$$\frac{\partial w_{s+1}}{\partial K_s} = \left(1 - \alpha\right)\alpha Z_{ss}\left(\frac{K_{ss}}{\pi l}\right)^{\alpha-1}$$

  and similarly for derivatives w.r.t. Z

- So all we *really* need are Jacobians of the capital function

$$J^{\mathcal{K},r}_{t,s} \equiv \frac{\partial \mathcal{K}_t}{\partial r_s} \quad J^{\mathcal{K},w}_{t,s} \equiv \frac{\partial \mathcal{K}_t}{\partial w_s}$$

- Then, get $d\mathbf{K} = -\left(\mathbf{H}_K\right)^{-1}\mathbf{H}_Z d\mathbf{Z}$ with one matrix inversion

Capital function Jacobians

- Can compute analytically

$$\frac{\partial r_{s+1}}{\partial K_s} = \alpha (\alpha - 1) Z_{ss} \left(\frac{K_{ss}}{\pi l}\right)^{\alpha-2}$$

$$\frac{\partial w_{s+1}}{\partial K_s} = (1 - \alpha) \alpha Z_{ss} \left(\frac{K_{ss}}{\pi l}\right)^{\alpha-1}$$

  and similarly for derivatives w.r.t. Z
- So all we *really* need are Jacobians of the capital function

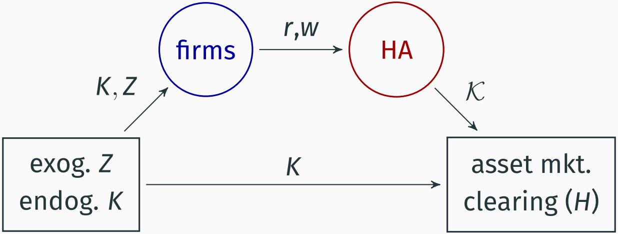
$$J^{\mathcal{K},r}_{t,s} \equiv \frac{\partial \mathcal{K}_t}{\partial r_s} \quad J^{\mathcal{K},w}_{t,s} \equiv \frac{\partial \mathcal{K}_t}{\partial w_s}$$

- Then, get $d\mathbf{K} = -(\mathbf{H}_K)^{-1} \mathbf{H}_Z d\mathbf{Z}$ with one matrix inversion
- **Next**:
  1. Generalize this procedure with DAG representation
  2. New formula to obtain terms such as $J^{\mathcal{K},r}$, $J^{\mathcal{K},w}$ “in one go”

Equilibrium as a Directed Acyclic Graph (DAG)

Representing equilibrium as a graph

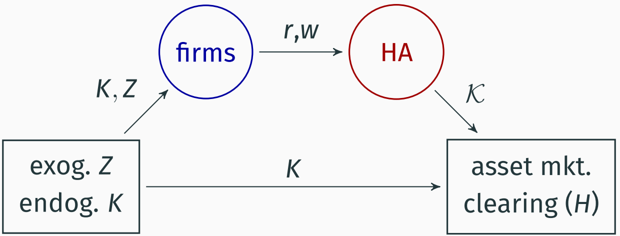
- We found a simple graph representation of the KS model:



- Each node ("block") takes sequences as inputs & outputs
  - Inputs of later nodes are outputs of earlier nodes
  - No. of equations in H = no. of "outside" endog. variables

Representing equilibrium as a graph

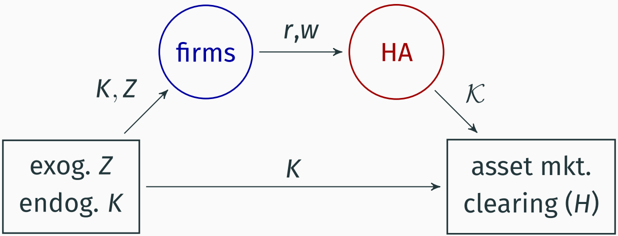
- We found a simple graph representation of the KS model:



- Each node ("block") takes sequences as inputs & outputs
    - Inputs of later nodes are outputs of earlier nodes
    - No. of equations in H = no. of "outside" endog. variables
- A very large class of models can be represented this way:
    - Equilibrium = Directed Acyclic Graph (DAG)
    - Blocks are either *recursive blocks* or *HA blocks*

Representing equilibrium as a graph

- We found a simple graph representation of the KS model:

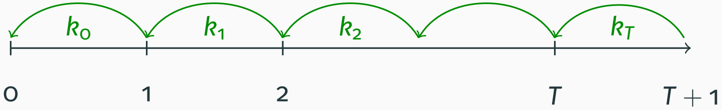


- Each node ("block") takes sequences as inputs & outputs
    - Inputs of later nodes are outputs of earlier nodes
    - No. of equations in H = no. of "outside" endog. variables
- A very large class of models can be represented this way:
    - Equilibrium = Directed Acyclic Graph (DAG)
    - Blocks are either *recursive blocks* or *HA blocks*
- Traditional approach to computation: guess & iterate

Traditional approach

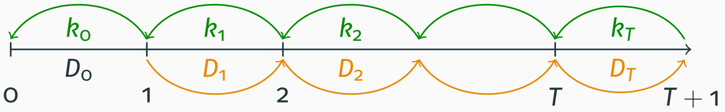
1. Truncate at T and discretize state space with n points
2. Guess a sequence $K_0, K_1, \ldots, K_{T+1} = K_{ss}$
    - Compute implied w_t and r_t at $t = 0 \ldots T+1$

Traditional approach



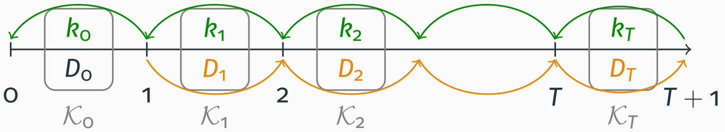
1. Truncate at T and discretize state space with n points
2. Guess a sequence $K_0, K_1, \ldots, K_{T+1} = K_{ss}$
   - Compute implied w_t and r_t at $t = 0 \ldots T+1$
3. Compute policy functions $k_t \in \mathbb{R}^n$ from $t = T \ldots 0$
   - "backward iteration" (use endogenous grids [Carroll '06])

Traditional approach



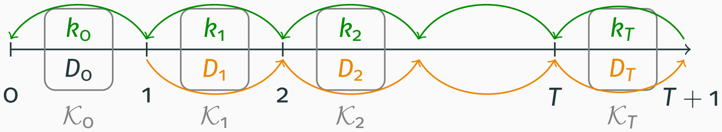
1. Truncate at T and discretize state space with n points
2. Guess a sequence $K_0, K_1, \ldots, K_{T+1} = K_{ss}$
    - Compute implied w_t and r_t at $t = 0 \ldots T+1$
3. Compute policy functions $k_t \in \mathbb{R}^n$ from $t = T \ldots 0$
    - "backward iteration" (use endogenous grids [Carroll '06])
4. Compute distributions $D_t \in \mathbb{R}^n$ from $t = 1 \ldots T$
    - "forward iteration" (use gridpoint lotteries [Young '10])

Traditional approach



1. Truncate at T and discretize state space with n points
2. Guess a sequence $K_0, K_1, \ldots, K_{T+1} = K_{ss}$
   - Compute implied w_t and r_t at $t = 0 \ldots T+1$
3. Compute policy functions $k_t \in \mathbb{R}^n$ from $t = T \ldots 0$
   - "backward iteration" (use endogenous grids [Carroll '06])
4. Compute distributions $D_t \in \mathbb{R}^n$ from $t = 1 \ldots T$
   - "forward iteration" (use gridpoint lotteries [Young '10])
5. Form capital demand $\mathcal{K}_t = k_t' D_t$ at every $t = 0 \ldots T$

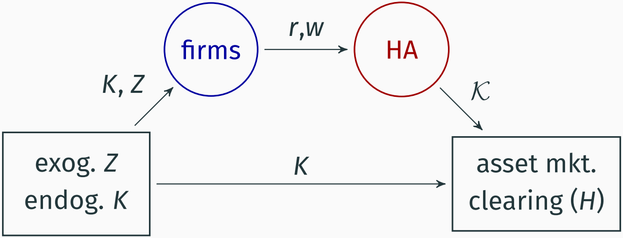
Traditional approach



1. Truncate at T and discretize state space with n points
2. Guess a sequence $K_0, K_1, \ldots, K_{T+1} = K_{ss}$
   - Compute implied w_t and r_t at $t = 0 \ldots T+1$
3. Compute policy functions $k_t \in \mathbb{R}^n$ from $t = T \ldots 0$
   - "backward iteration" (use endogenous grids [Carroll '06])
4. Compute distributions $D_t \in \mathbb{R}^n$ from $t = 1 \ldots T$
   - "forward iteration" (use gridpoint lotteries [Young '10])
5. Form capital demand $\mathcal{K}_t = k_t{}' D_t$ at every $t = 0 \ldots T$
6. If $\max_t |\mathcal{K}_t - K_t| \neq 0$, update K_t with ad-hoc rule, go to 2.

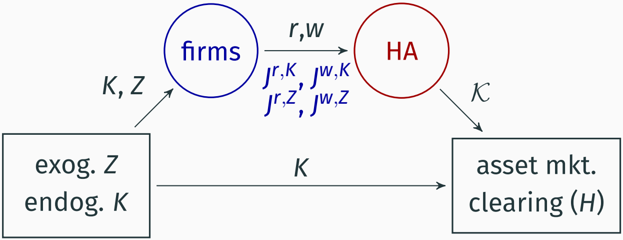
$\rightarrow$ Need as many backward&forward iterations as no. of loops

Our approach: compute Jacobians along computational graph



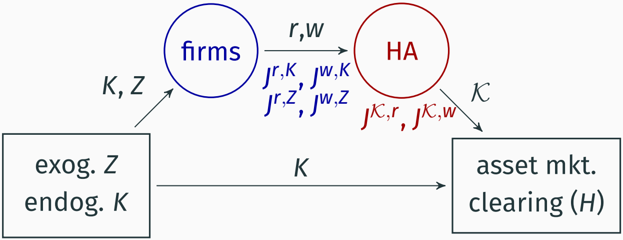
1. For each block, obtain Jacobian $J^{o,i}$ of output o wrt input i

Our approach: compute Jacobians along computational graph



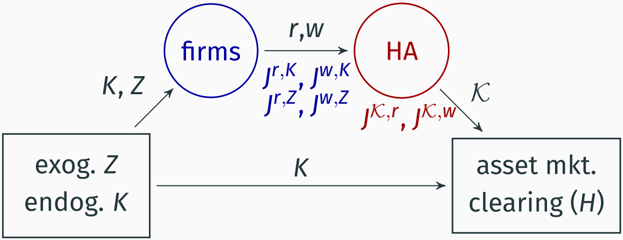
1. For each block, obtain Jacobian $J^{o,i}$ of output o wrt input i
   - For recursive blocks : automatically (J very sparse!)

Our approach: compute Jacobians along computational graph



1. For each block, obtain Jacobian $J^{o,i}$ of output o wrt input i
    - For recursive blocks : automatically (J very sparse!)
    - For HA blocks : using “fake news shocks” (up next!)

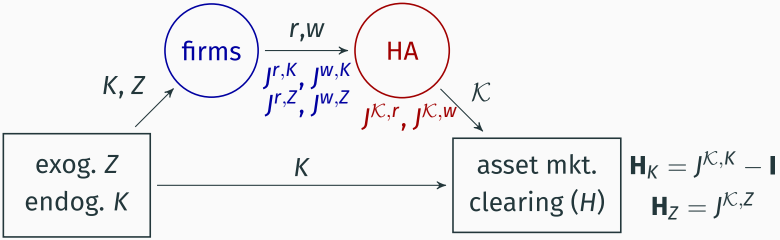
Our approach: compute Jacobians along computational graph



$$\mathbf{H}_K = J^{\mathcal{K},K} - \mathbf{I}$$
$$\mathbf{H}_Z = J^{\mathcal{K},Z}$$

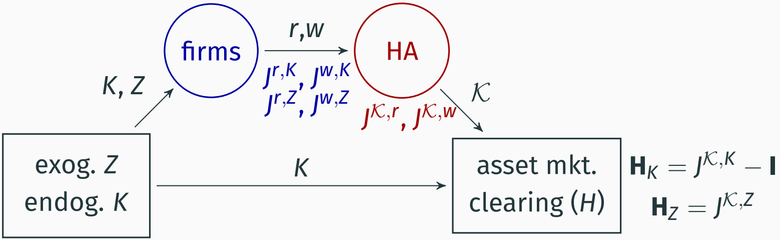
1. For each block, obtain Jacobian $J^{o,i}$ of output o wrt input i
    - For recursive blocks : automatically (J very sparse!)
    - For HA blocks : using "fake news shocks" (up next!)
2. Compose Jacobians along computational graph $\rightarrow$ $\mathbf{H}_K$, $\mathbf{H}_Z$
    - Here: $J^{\mathcal{K},K} = J^{\mathcal{K},r} J^{r,K} + J^{\mathcal{K},w} J^{w,K}$ (same for Z)

Our approach: compute Jacobians along computational graph



1. For each block, obtain Jacobian $J^{o,i}$ of output o wrt input i
    - For recursive blocks : automatically (J very sparse!)
    - For HA blocks : using "fake news shocks" (up next!)
2. Compose Jacobians along computational graph $\rightarrow \mathbf{H}_K, \mathbf{H}_Z$
    - Here: $J^{\mathcal{K},K} = J^{\mathcal{K},r}J^{r,K} + J^{\mathcal{K},w}J^{w,K}$ (same for Z)
3. Invert $\mathbf{H}_K$ to obtain impulse response $d\mathbf{K} = -\mathbf{H}_K^{-1}\mathbf{H}_Z d\mathbf{Z}$

Our approach: compute Jacobians along computational graph



1. For each block, obtain Jacobian $J^{o,i}$ of output o wrt input i
   - For recursive blocks : automatically (J very sparse!)
   - For HA blocks : using "fake news shocks" (up next!)
2. Compose Jacobians along computational graph $\to$ $\mathbf{H}_K$, $\mathbf{H}_Z$
   - Here: $J^{\mathcal{K},K} = J^{\mathcal{K},r}J^{r,K} + J^{\mathcal{K},w}J^{w,K}$ (same for Z)
3. Invert $\mathbf{H}_K$ to obtain impulse response $d\mathbf{K} = -\mathbf{H}_K^{-1}\mathbf{H}_Z d\mathbf{Z}$
4. Get other impulses from J's, e.g. $d\mathbf{r} = \underbrace{\left(J^{r,Z} - J^{r,K}\mathbf{H}_K^{-1}\mathbf{H}_Z\right)}_{\mathbf{G}^r} d\mathbf{Z}$

Main advantages of our approach

1. Computes first order linear impulse response *directly*
    - For many questions this is the object of interest!
    - Can use Newton's method for nonlinear impulse response
    - No approximation to distn, no pbm handling constraints!
2. Alternative shocks **do not require recomputing $\mathbf{H}_K$**
    - "Multiplier" $\mathbf{H}_K^{-1}$ is invariant to shocks
    - General equilibrium sufficient statistic [Auclert-Rognlie-Straub]
3. Parameters often only affect Jacobians of one/few blocks
    - In particular **may not change HA block** Jacobians
        - eg aggregate frictions (adj costs, sticky prices...) & shocks
    - Use this insight to speed up estimation even more!

Heterogeneous-Agent Jacobian computation

Jacobians for aggregate outcomes

- We call the capital function

$$\mathcal{K}_t\left(\{r_s, w_s\}\right) = \int k_t\left(e, k_-\right) d\Psi_t\left(e, k_-\right)$$

  an *aggregate outcome*

  - *Individual outcome* k_t integrated over distribution
  - This is the prototypical output of a HA block
  - We need $\partial\mathcal{K}_t/\partial r_s$ and $\partial\mathcal{K}_t/\partial w_s$ for all $t, s \leq T$

Jacobians for aggregate outcomes

- We call the capital function

$$\mathcal{K}_t\left(\{r_s, w_s\}\right) = \int k_t\left(e, k_-\right) d\Psi_t\left(e, k_-\right)$$

  an *aggregate outcome*

  - *Individual outcome k_t* integrated over distribution
  - This is the prototypical output of a HA block
  - We need $\partial\mathcal{K}_t/\partial r_s$ and $\partial\mathcal{K}_t/\partial w_s$ for all $t, s \leq T$

- Our method covers the **generic problem:**

$$Y_t\left(\{Z_s\}\right) \equiv \int y_t\left(\omega\right) d\Psi_t\left(\omega\right)$$

  where Z_s is aggregate input, ω is state, $y_t(\omega)$ is individual outcome, Y_t is aggregate outcome and we need, for all t, s:

$$J_{t,s} = J_{t,s}^{Y,Z} \equiv \partial Y_t/\partial Z_s$$

Discretized problem

- As above, with discretized state space of n points,

$$Y_t = \int y_t(\omega)\, d\Psi_t(\omega)$$

becomes

$$Y_t = y_t' D_t$$

where $y_t \in \mathbb{R}^n$ is individual outcome and $D_t \in \mathbb{R}^n$ is distribution mass at each state

Discretized problem

- As above, with discretized state space of n points,

$$Y_t = \int y_t(\omega)\, d\Psi_t(\omega)$$

becomes

$$Y_t = y_t' D_t$$

where $y_t \in \mathbb{R}^n$ is individual outcome and $D_t \in \mathbb{R}^n$ is distribution mass at each state

- Distribution follows law of motion

$$D_t = \Lambda_{t-1}' D_{t-1}$$

where $\Lambda_{t-1} \in \mathbb{R}^{n \times n}$ is (usually sparse) Markov matrix giving transition probability between states in $t-1$ and t

  - Partly exogenous, partly from endog. decisions at $t-1$

Calculating Jacobian: naive vs. sophisticated

- The s^{th} column $J_{\cdot,s}$ is $\{dY_t\}$
  - Path of outcomes in response to infinitesimal shock dZ_s

Calculating Jacobian: naive vs. sophisticated

- The s^{th} column $J_{\cdot,s}$ is $\{dY_t\}$
  - Path of outcomes in response to infinitesimal shock dZ_s
- **Naive algorithm**: for each s, follow traditional approach to get $\{dY_t\}$ after perturbation $dZ_s = \epsilon$
  - Problem: requires separate backward iteration and forward iteration for each column $s = 0 \dots T \rightarrow$ **very costly**

Calculating Jacobian: naive vs. sophisticated

- The s^{th} column $J_{\cdot,s}$ is $\{dY_t\}$
  - Path of outcomes in response to infinitesimal shock dZ_s
- **Naive algorithm**: for each s, follow traditional approach to get $\{dY_t\}$ after perturbation $dZ_s = \epsilon$
  - Problem: requires separate backward iteration and forward iteration for each column $s = 0 \dots T \rightarrow$ **very costly**
- **Our approach**: exploit the fact that

$$dY_t = dy_t' D_{ss} + y_{ss}' dD_t$$

  - Use symmetry to relate dy_t' and dD_t for different s's
  - Get all in **single** backward iteration & forward iteration
  - Cost reduced by factor of T, usually **200x** or more

Time symmetry in backward iteration

- Outcomes dy_t, $d\Lambda_t$ of backward iteration from shock dZ_s:
    1. only depend on $s - t$
    2. are 0 if $s < t$

Time symmetry in backward iteration

- Outcomes dy_t, $d\Lambda_t$ of backward iteration from shock dZ_s:
    1. only depend on $s - t$
    2. are 0 if $s < t$

- **Q**: holding the date-t distribution fixed at $D_t = D_{ss}$, what is effect of dZ_s on aggregate dY_t and next-period dD_{t+1}?

$$dY_t = dy_t' D_{ss} \equiv \mathcal{Y}_{s-t} \in \mathbb{R}$$
$$dD_{t+1} = d\Lambda_t' D_{ss} \equiv \mathcal{D}_{s-t} \in \mathbb{R}^n$$

Time symmetry in backward iteration

- Outcomes dy_t, $d\Lambda_t$ of backward iteration from shock dZ_s:
  1. only depend on $s - t$
  2. are 0 if $s < t$

- **Q**: holding the date-t distribution fixed at $D_t = D_{ss}$, what is effect of dZ_s on aggregate dY_t and next-period dD_{t+1}?

$$dY_t = dy_t' D_{ss} \equiv \mathcal{Y}_{s-t} \in \mathbb{R}$$
$$dD_{t+1} = d\Lambda_t' D_{ss} \equiv \mathcal{D}_{s-t} \in \mathbb{R}^n$$

  This too *only depends* on $s - t$, and is 0 for $s < t$
  In particular: $\mathcal{Y}_u$, $\mathcal{D}_u$ are effects of shock at T on date $T - u$

$\Rightarrow$ can get $\mathcal{Y}_u$ and $\mathcal{D}_u$ for all u with a **single** **backward iteration**

Time symmetry in forward iteration

- **Q**: holding the date-t individual outcome fixed at $y_t = y_{ss}$, what is effect of a shock dD_s to distribution at date s on aggregate dY_t?

$$dY_t = \begin{cases} \underbrace{y_{ss}'(\Lambda_{ss}')^{t-s}}_{\equiv \mathcal{P}_{t-s}'} dD_s & s \leq t \\ 0 & s > t \end{cases}$$

Time symmetry in forward iteration

- **Q**: holding the date-t individual outcome fixed at $y_t = y_{ss}$, what is effect of a shock dD_s to distribution at date s on aggregate dY_t?

$$dY_t = \begin{cases} \underbrace{y_{ss}'(\Lambda_{ss}')^{t-s}}_{\equiv \mathcal{P}_{t-s}'} dD_s & s \leq t \\ 0 & s > t \end{cases}$$

- Only a function of **prediction vectors** $\mathcal{P}_u = \Lambda_{ss}^u y_{ss} \in \mathbb{R}^n$
  - Effect of change in date-0 distribution on aggregates at u

$\Rightarrow$ can get $\mathcal{P}_u$ for all u with **one “transpose” forward iteration**

Recursive expression for Jacobian

- It turns out that the full Jacobian is given by

$$J_{t,s} = \frac{dY_t}{dZ_s} = \begin{cases} \mathcal{Y}_s & t = 0 \\ \mathcal{P}'_{t-1}\mathcal{D}_0 & t \geq 1, s = 0 \\ \frac{dY_{t-1}}{dZ_{s-1}} + \mathcal{P}'_{t-1}\mathcal{D}_s & s, t \geq 1 \end{cases} \tag{2}$$

This uses *only* the $\mathcal{Y}$'s, $\mathcal{D}$'s, and $\mathcal{P}$'s

Recursive expression for Jacobian

- It turns out that the full Jacobian is given by

$$J_{t,s} = \frac{dY_t}{dZ_s} = \begin{cases} \mathcal{Y}_s & t = 0 \\ \mathcal{P}'_{t-1}\mathcal{D}_0 & t \geq 1, s = 0 \\ \frac{dY_{t-1}}{dZ_{s-1}} + \mathcal{P}'_{t-1}\mathcal{D}_s & s, t \geq 1 \end{cases} \tag{2}$$

  This uses *only* the $\mathcal{Y}$'s, $\mathcal{D}$'s, and $\mathcal{P}$'s

- **Why?** Time symmetry $\Rightarrow$ aggregate outcome at date t from input shock at date s equals:
  - outcome at date $t-1$ from shock at date $s-1$, plus
  - additional term $\mathcal{P}'_{t-1}\mathcal{D}_s$ from the persistence over $t-1$ periods of effect on period-1 distribution from s-period-ahead anticipation of shock at date 0

- Simpler way to construct (2):

1. Define the **fake news matrix** as

$$F_{t,s} = \begin{cases} \mathcal{Y}_s & \text{if } t = 0 \\ \mathcal{P}'_{t-1}\mathcal{D}_s & \text{if } t > 0 \end{cases}$$

2. Construct the Jacobian $J_{t,s} = \frac{dY_t}{dZ_s}$ recursively as

$$J_{t,s} = \begin{cases} F_{t,s} & \text{if } t = 0, s = 0 \\ F_{t,s} + J_{t-1,s-1} & \text{otherwise} \end{cases} \tag{3}$$

- Simpler way to construct (2):

  1. Define the **fake news matrix** as

$$F_{t,s} = \begin{cases} \mathcal{Y}_s & \text{if } t = 0 \\ \mathcal{P}'_{t-1}\mathcal{D}_s & \text{if } t > 0 \end{cases}$$

  2. Construct the Jacobian $J_{t,s} = \frac{dY_t}{dZ_s}$ recursively as

$$J_{t,s} = \begin{cases} F_{t,s} & \text{if } t = 0, s = 0 \\ F_{t,s} + J_{t-1,s-1} & \text{otherwise} \end{cases} \tag{3}$$

- Interpretation of the fake news matrix:
  - At $t = 0$, news of shock at date $s \geq 0$
    - Effect on aggregate outcome is $\mathcal{Y}_s$
  - At $t \geq 1$, news of shock disappears (it was “fake news”)
    - Effect $\mathcal{P}'_{t-1}\mathcal{D}_s$ from persistence via distribution

Overview of generalized algorithm

- Typical HA block: several outcomes $o \in \mathcal{O}$ and inputs $i \in \mathcal{I}$
  - How to efficiently calculate Jacobians $J^{o,i}$?

Overview of generalized algorithm

- Typical HA block: several outcomes $o \in \mathcal{O}$ and inputs $i \in \mathcal{I}$
    - How to efficiently calculate Jacobians $J^{o,i}$?

- Four steps:

1. A backward iteration for each $i \in \mathcal{I}$ to get $\mathcal{Y}_u^{o,i}$ and $\mathcal{D}_u^i$
2. A (transpose) forward iteration for each $o \in \mathcal{O}$ to get $\mathcal{P}_u^o$
3. For each $o, i \in \mathcal{O} \times \mathcal{I}$, combine $\mathcal{Y}_u^{o,i}$ with matrix product of $(\mathcal{P}^o)'$ and $\mathcal{D}^i$ to get **fake news matrix** $F^{o,i}$
4. For each $o, i \in \mathcal{O} \times \mathcal{I}$, use (3) to convert $F^{o,i}$ to **Jacobian** $J^{o,i}$

Overview of generalized algorithm

- Typical HA block: several outcomes $o \in \mathcal{O}$ and inputs $i \in \mathcal{I}$
  - How to efficiently calculate Jacobians $J^{o,i}$?

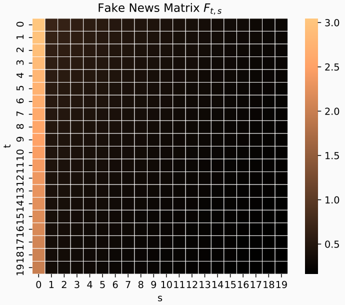
- Four steps:

  1. A backward iteration for each $i \in \mathcal{I}$ to get $\mathcal{Y}_u^{o,i}$ and $\mathcal{D}_u^i$
  2. A (transpose) forward iteration for each $o \in \mathcal{O}$ to get $\mathcal{P}_u^o$
  3. For each $o, i \in \mathcal{O} \times \mathcal{I}$, combine $\mathcal{Y}_u^{o,i}$ with matrix product of $(\mathcal{P}^o)'$ and $\mathcal{D}^i$ to get **fake news matrix** $F^{o,i}$
  4. For each $o, i \in \mathcal{O} \times \mathcal{I}$, use (3) to convert $F^{o,i}$ to **Jacobian** $J^{o,i}$

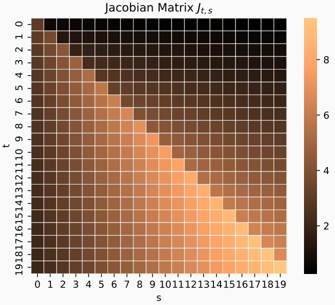
- 1. and 2. are only iterations required
- 3. and 4. are standard fast math, not model-specific

- For our calibration of the KS model, compute $\frac{\partial \mathcal{K}_t}{\partial r_s}$:

Fake news matrix F



Full Jacobian J



- For $T = 300$ and $n = 3500$, get F in 140ms, J in 1 extra ms

Example codes

- For this Krusell-Smith example, we provide files and notebook that
  1. calculate the steady state
  2. compute all Jacobians
  3. compute impulse responses (linear and nonlinear)
  4. compute business cycle statistics (second moments)
  5. compute model likelihood for estimation
- Very easy to adapt, especially if you are only changing or adding recursive blocks
  - changing HA blocks requires more coding
  - but we provide the routines to compute and compose all the Jacobians!

Our programming language: Python

- Our code is written in Python
  - Powerful general-purpose open-source language
  - Use NumPy library for numerical programming and Numba library for just-in-time compilation of our own routines
- Two steps for installation:
  1. Get Anaconda distribution of Python 3.7 at https://www.anaconda.com/distribution/
  2. Download our example files and notebooks at https://github.com/shade-econ/sequence-jacobian

Coding the KS model in practice

- See `demo_krusell_smith.ipynb` and `ks.py`. Key steps:

1. Define firm block

```
@recursive
def firm(K, L, Z, alpha, delta):
    r = alpha * Z * (K(-1) / L) ** (alpha-1) - delta
    w = (1 - alpha) * Z * (K(-1) / L) ** alpha
    Y = Z * K(-1) ** alpha * L ** (1 - alpha)
    return r, w, Y
```

2. Define HA block & get steady stte

```
""" (HA block defined in backward_iterate fun.) """
ss = household_ss(Pi, a_grid, s_grid, r, w, beta, eis)
```

Coding the KS model in practice (continued)

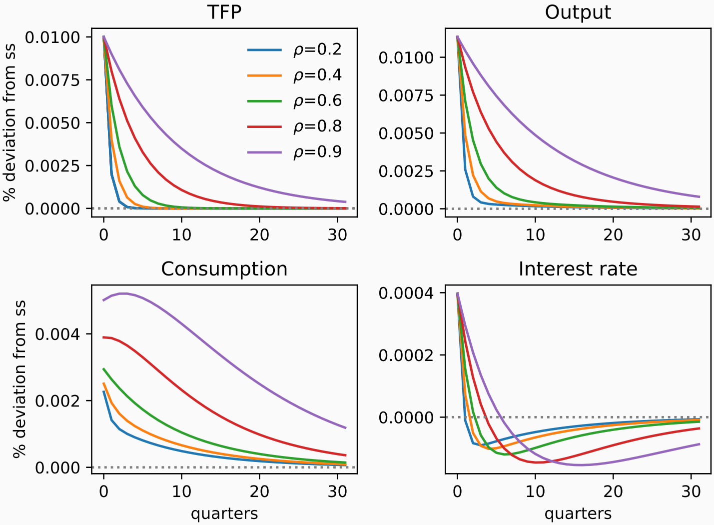
3. Find & compose Jacobians

```
J_firm = rec.all_Js(firm, ss, T, ['K', 'Z'])
J_ha = het.all_Js(backward_iterate, ss, T, {'r','w'})
# apply chain rule
J_comp = jac.compose_jacobians(J_firm, J_ha)
# obtain Jacobian
H_K = J_comp['curlyK']['K'] - np.eye(T)
H_Z = J_comp['curlyK']['Z']
G_Z = -np.linalg.solve(H_K, H_Z)
```

4. Get impulse response

```
dZ = 0.8**np.arange(T)
dK = G_Z @ dZ
```

Impulse response to TFP shocks of different persistence



- Given Jacobian, get these 5 impulse responses in 0.2 ms

SHADE models

SHADE models

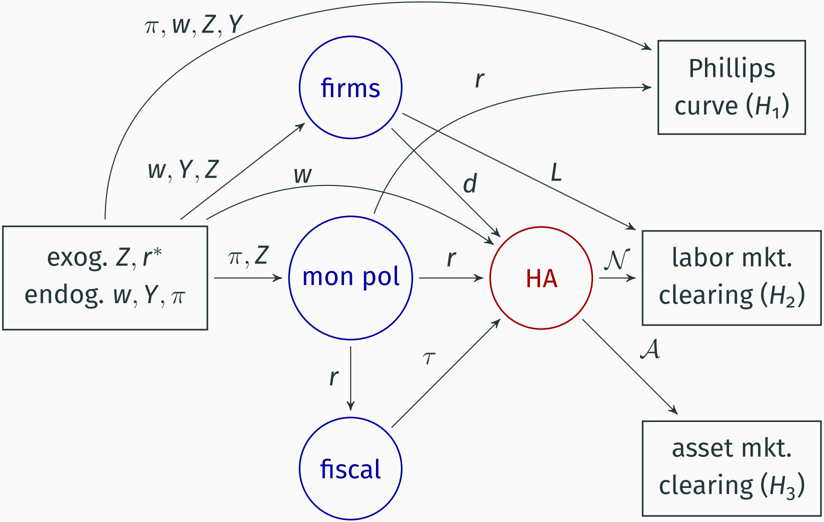
- Many models written in sequence space are made up of *some* combination of recursive blocks and HA blocks
- We call them **S**equence-space **H**eterogeneous **A**gents **D**ynamic **E**quilibrium models
  - **SHADE** models!
- Examples are models with:
  - Heterogeneous households w/ labor supply, 2 assets...
  - OLG + idiosyncratic risk [Conesa-Krueger, Hubbard-Skinner-Zeldes]
  - Household default [Livshits et al, Chatterjee et al]
  - Heterogeneous firms [Hopenhayn, Kahn-Thomas]
  - Pricing + idiosyncratic shocks [Golosov-Lucas]
  - ...
- Any SHADE model can be computed with our method

Another SHADE model: HANK

▸ Full model

- Another SHADE model example: a HANK model with aggregate shocks
  - No capital
  - Sticky prices à la Rotemberg, flexible wages
  - Monetary policy follows a Taylor rule
  - Very similar to McKay-Nakamura-Steinsson (2016)
- More complicated computational graph...
  - ... but solution follows the exact same steps!

HANK model: DAG representation



"Finding the nicest DAG of a SHADE model is an art"

Second moments

Computing second moments

- Second moments ("business cycle statistics") can be computed directly from the set of impulse responses
- Why? Because of certainty equivalence, impulse responses *are* the MA(∞) representation of the model
- If $\left\{\theta_k^{Y,s}\right\}_{k\geq 0}$ is impulse response of outcome Y_t to shock s, then value of Y_t after history of innovations $\left\{\epsilon_{t-k}^{s}\right\}_{k\geq 0}$ to shocks $s = 1 \dots S$ is

$$Y_t = \sum_{s=1}^{S} \sum_{k=0}^{\infty} \theta_k^{Y,s} \epsilon_{t-k}^{s} \tag{4}$$

Going from MA(∞) to second moments

- Assume white noise innovations:

$$\text{Cov}\left(\epsilon_t^s, \epsilon_{t-k}^{s'}\right) = \sigma_s^2 \cdot 1_{s=s'} \cdot 1_{k=0}$$

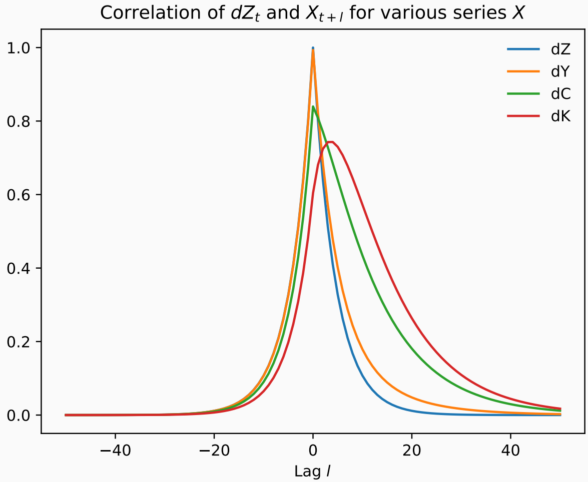
- Then, from (4), for any two outcomes Y, X:

$$\text{Cov}\left(Y_t, X_{t+l}\right) = \sum_{s=1}^{S} \sum_{k=0}^{\infty} \theta_k^{Y,s} \theta_{k+l}^{X,s} \sigma_s^2$$

  - Get this with matrix products or Fast Fourier Transform
  - No simulation required!

Example with AR(1) TFP shock

- Assume single AR(1) shock to Z, $\rho = 0.8$, $\sigma = 0.02$



- Given impulse responses, get all autocovs in 0.47 ms

Estimation

Computing the likelihood

- Standard way to estimate DSGE model: **state space**
- We use alternative **sequence space** method
  - Directly evaluate the log-likelihood, bypass Kalman filter
  - Use MA(∞) representation [Hamilton 1994, Mankiw-Reis 2007]

Computing the likelihood

- Standard way to estimate DSGE model: **state space**
- We use alternative **sequence space** method
    - Directly evaluate the log-likelihood, bypass Kalman filter
    - Use MA(∞) representation [Hamilton 1994, Mankiw-Reis 2007]
- Assume Gaussian shocks ϵ_{t-k}^{s} and measurement error $\mathbf{u}_t$
- Then observations w_t are also Gaussian:

$$\mathbf{w}_t = \mathbf{Y}_t + \mathbf{u}_t$$

- Log-likelihood of the parameters θ and the data $\mathbf{w}$ is

$$\mathcal{L}(\mathbf{w}, \theta) = -\frac{1}{2} \log \left(\det \mathbf{V}_w(\theta) \right) - \frac{1}{2} \mathbf{w}' \left[\mathbf{V}_w(\theta) \right]^{-1} \mathbf{w}$$

where $[\mathbf{V}_w(\theta)]_{t,s} \equiv \mathrm{Cov}(\mathbf{w}_t, \mathbf{w}_s) = \mathrm{Cov}(\mathbf{Y}_t, \mathbf{Y}_s) + \Sigma_u 1_{t=s}$

    - Computed in the previous step!

Computation of the likelihood in practice

- How to rapidly recompute $\mathbf{V}_w(\theta)$ for many θ's?
  - Recall $d\mathbf{X} = -\mathbf{H}_X^{-1}\mathbf{H}_Z d\mathbf{Z}$
  - Could start from scratch at each θ:
    - Jacobian $\rightarrow$ Impulses $\rightarrow$ Auto-covariances
  - **More efficient**: use invariant parts of Jacobian
    - When only shocks $d\mathbf{Z}$ change, then $\mathbf{H}_X$, $\mathbf{H}_Z$ are the same
    - When only aggregate frictions change, HA block Jacobians unchanged, cheap to recompute $\mathbf{H}_X$ and $\mathbf{H}_Z$

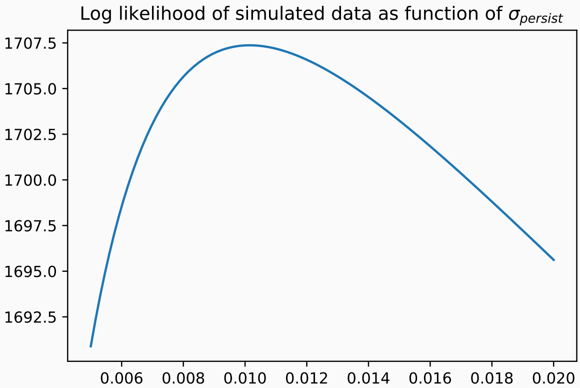
Computation of the likelihood in practice

- How to rapidly recompute $\mathbf{V}_w(\theta)$ for many θ's?
  - Recall $d\mathbf{X} = -\mathbf{H}_X^{-1}\mathbf{H}_Z d\mathbf{Z}$
  - Could start from scratch at each θ:
    - Jacobian $\rightarrow$ Impulses $\rightarrow$ Auto-covariances
  - **More efficient**: use invariant parts of Jacobian
    - When only shocks $d\mathbf{Z}$ change, then $\mathbf{H}_X$, $\mathbf{H}_Z$ are the same
    - When only aggregate frictions change, HA block Jacobians unchanged, cheap to recompute $\mathbf{H}_X$ and $\mathbf{H}_Z$
- How to rapidly compute $\det \mathbf{V}_w(\theta)$ and $\mathbf{w}' [\mathbf{V}_w(\theta)]^{-1} \mathbf{w}$?
  - Use Cholesky decomposition of $\mathbf{V}_w(\theta)$
  - Alternatively could use "Levinson recursion"

Example: recovering true parameters from simulated data

- Suppose we observe Z_t, Y_t, C_t, K_t with noise
  - Measurement error has variance σ_u^2 for each variable
- Assume these come from KS model with two AR(1) shocks:
  - $z_t^1 = \rho z_{t-1}^1 + \sigma_{persist} \cdot \epsilon_t^1$ and $z_t^2 = \sigma_{trans} \cdot \epsilon_t^2$
- Fix parameters at θ_0, draw observations **w** from model
- Then, evalutate log-likelihood for alternative $\sigma_{persist}$'s holding other parameters at θ_0

Log-likelihood plot



- This peaks near the true value of $\sigma_{persist} = 0.01$!
- 100 evaluations of the likelihood only took 1.44s
- Feasible to estimate HANK! see Auclert-Rognlie-Straub

Local determinacy

- Recall $d\mathbf{X} = -\mathbf{H}_X^{-1}\mathbf{H}_Z d\mathbf{Z}$
- Local determinacy $\leftrightarrow$ invertibility of $\mathbf{H}_X$
- This can be checked numerically, after computing $\mathbf{H}_X$, by using the singular value decomposition and seeing how close to 0 the smallest singular value is
- The Krusell-Smith model is always determinate
- HANK models can be indeterminate for insufficiently responsive monetary policy, especially when (say) income risk is countercyclical

An alternative determinacy criterion

- SHADE models have a simple asymptotic structure:

$$\mathbf{H}_X \simeq \begin{pmatrix} * & * & \ddots & \ddots & & \\ * & * & A_{-1} & A_{-2} & \ddots & \\ \ddots & A_1 & A_0 & A_{-1} & A_{-2} & \ddots \\ \ddots & A_2 & A_1 & A_0 & A_{-1} & \ddots \\ & \ddots & A_2 & A_1 & A_0 & \ddots \\ & & \ddots & \ddots & \ddots & \ddots \end{pmatrix}$$

  where $A_i \in \mathbb{R}^{m \times m}$ is response of the m targets at $t+i$ to change in the m endog vars in X at t

- Given this, we can devise a test for determinacy that only relies on the asympotics of $\mathbf{H}_X$, ie the matrices $\{A_i\}$

Winding number criterion

- Let

$$A(\lambda) \equiv \det \sum_{j=-\infty}^{\infty} A_j e^{ij\lambda}$$

then $\mathbf{H}_X$ is invertible if and only if

$$w_A \equiv \frac{1}{2\pi i} \int_0^{2\pi} \frac{A'(\lambda)}{A(\lambda)} d\lambda = 0 \tag{5}$$

- This is a **“winding number criterion”**
  - Equivalent to Blanchard-Kahn for SHADE models that only have recursive blocks [Onatski 2006]
  - Alternative, faster route to checking determinacy

Nonlinear solution

Computing nonlinear perfect foresight transitions

- Armed with Jacobian $\mathbf{H}_X$, can compute full nonlinear solution to

$$\mathbf{H}(\mathbf{X}, \mathbf{Z}) = 0$$

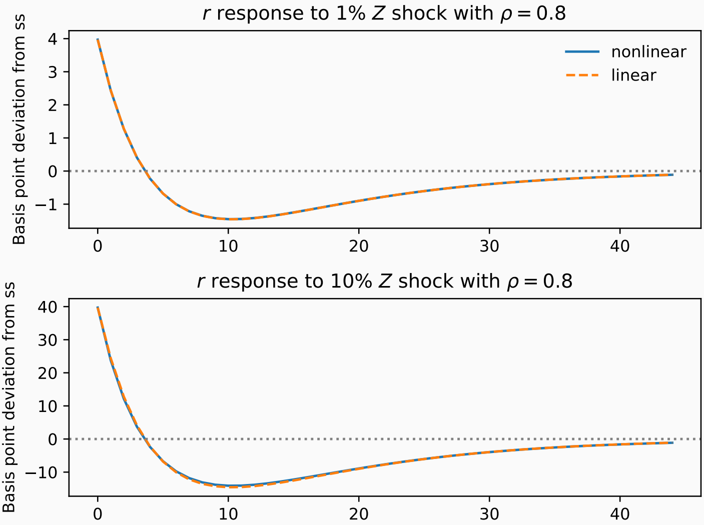
  - Require going back to MIT shock interpretation
  - Can explore asymmetries in response to shocks

- Idea: use (quasi)-**Newton's method**
  - Start from $\mathbf{X}^{(0)} = \mathbf{X}^{ss}$ and iterate using

$$\mathbf{X}^{(n)} = \mathbf{X}^{(n-1)} - [\mathbf{H}_X]^{-1} \mathbf{H}\left(\mathbf{X}^{(n-1)}, \mathbf{Z}\right)$$

  - Used for perfect foresight transitions in Dynare [Julliard 1996]
  - Previous HA applications used *approximations* to $\mathbf{H}_X$
    - Auxiliary model [Auclert and Rognlie 2018]
    - Interpolation [Straub 2018]

Do nonlinearities matter? Not unless shocks are very large!



- Computation time: nonlinear 233ms (linear 0.2ms)

Conclusion

- Many interesting macro models are SHADE models
- Linearizing wrt aggregate shocks usually works very well
- So the Jacobian is all you need, and it's very simple to get
- Calculation of impulse responses in a few milliseconds
  - Makes estimation of HA models feasible
  - Opens the door to optimal policy computation

Extra slides

Krusell-Smith model calibration

Back

- $\mathcal{P}(e, e')$ discretizes $\log e_t = \rho \log e_{t-1} + \sigma \epsilon_t$ where $\epsilon_t \sim \mathcal{N}(0, 1)$
- Use Rouwenhorst method

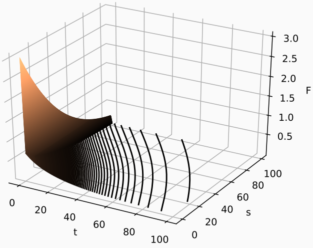
Parameter		Value
r	Real interest rate	0.01
σ	Risk aversion	1
α	Capital share	0.11
δ	Depreciation rate	0.025
ρ	Skill mean reversion	0.966
$\sigma/\sqrt{1-\rho^2}$	Sd of log e	0.5
n_e	Points in Markov chain for e	7
n_k	Points on asset grid	500

- Overall grid points $n = n_e \times n_k = 3500$

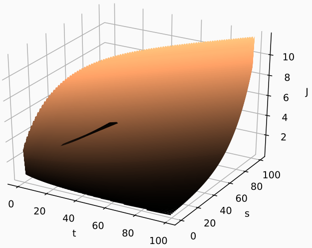
Jacobian computation: 3d version

▸ Back

Fake news matrix F



Full Jacobian J



HANK model: setup of HA block

- Endogenous labor: $u(c_t) - v(n_t)$ with $v(n) = b\frac{n^{1+\psi}}{1+\psi}$
- S idiosyncratic states $e \in \{e_1, \ldots, e_S\}$, transition $\mathcal{P}(e, e')$

$$
\begin{aligned}
V_t(e, a_-) = \quad \max_{c,n,a} \quad & u(c) - v(n) + \beta \sum_{e'} V_{t+1}(e', a)\, \mathcal{P}(e, e') \\
\text{s.t.} \quad & c + a = (1 + r_t)\, a_- + w_t e n_t - \overline{\tau}(e)\, \tau_t + d_t \\
& a \geq 0
\end{aligned}
$$

$\Rightarrow$ policies $c_t(e, a_-)$, $n_t(e, a_-)$ and $a_t(e, a_-)$

HANK model: setup of HA block

- Endogenous labor: $u(c_t) - v(n_t)$ with $v(n) = b\frac{n^{1+\psi}}{1+\psi}$
- S idiosyncratic states $e \in \{e_1, \ldots, e_S\}$, transition $\mathcal{P}(e, e')$

$$\begin{aligned} V_t(e, a_-) = \quad & \max_{c,n,a} \quad u(c) - v(n) + \beta \sum_{e'} V_{t+1}(e', a)\, \mathcal{P}(e, e') \\ & \text{s.t.} \qquad c + a = (1 + r_t)\, a_- + w_t e n_t - \overline{\tau}(e)\, \tau_t + d_t \\ & \qquad\qquad a \geq 0 \end{aligned}$$

$\Rightarrow$ policies $c_t(e, a_-)$, $n_t(e, a_-)$ and $a_t(e, a_-)$

- Aggregate labor and asset functions

$$\begin{aligned} \mathcal{N}_t\left(\{r_s, w_s, \tau_s, d_s\}_{s\geq 0}\right) &\equiv \int e n_t(e, a_-)\, d\Psi_t(e, a_-) \\ \mathcal{A}_t\left(\{r_s, w_s, \tau_s, d_s\}_{s\geq 0}\right) &\equiv \int a_t(e, a_-)\, d\Psi_t(e, a_-) \end{aligned}$$

HANK model: recursive blocks and equilibrium conditions ▸ Back

- Standard New-Keynesian production:
  - Monopolistic competition $Y_t = Z_t L_t$, $d_t = Y_t - w_t L_t$
  - Rotemberg pricing: dynamics for inflation π_t is

$$\log(1+\pi_t) = \kappa\left(\frac{w_t}{Z_t} - \frac{1}{\mu}\right) + \frac{1}{1+r_{t+1}}\frac{Y_{t+1}}{Y_t}\log(1+\pi_{t+1})$$

  - Flexible wages: $bn_t(a_-, e)^{\psi} c_t(a_-, e)^{\sigma} = w_t e$ for all (a_-, e)
- Government maintains constant debt B, adjusts τ_t:

$$r_t B = \tau_t \sum_e \pi(e)\,\overline{\tau}(e)$$

- Taylor rule monetary policy + Fisher equation:

$$1 + r_{t+1} = \frac{1+i_t}{1+\pi_{t+1}} = \frac{1 + r_t^* + \phi\pi_t}{1+\pi_{t+1}}$$

- Market clearing:

$$\mathcal{N}_t = L_t; \qquad \mathcal{A}_t = B$$