Topics in Heterogeneous Agent Macro: Sequence-Space Methods

Lecture 9

Andreas Schaab

Outline

1. Model
2. Sequence Space
3. Fake-News Algorithm

Model

Summary of Equilibrium Conditions

$$\left\{c_t(a,z), V_t(a,z), g_t(a,z), Y_t, N_t, r_t, i_t, w_t, \pi_t, \pi_t^w\right\},$$

Micro block:

$$\rho V_t(a,z) = U(c_t, N_t) + \partial_t V_t(a,z) + \mathcal{A}_t V_t(a,z) \tag{1}$$

$$u'(c_t(a,z)) = \partial_a V_t(a,z) \tag{2}$$

$$\partial_t g_t(a,z) = \mathcal{A}_t^* g_t(a,z), \tag{3}$$

where, using $s_t(a,z) = r_t a + z w_t N_t - c_t(a,z)$:

$$\mathcal{A}_t f_t(a,z) = s_t(a,z)\partial_a f_t(a,z) + \mathcal{A}^z f_t(a,z)$$

$$\mathcal{A}_t^* g_t(a,z) = -\partial_a\left[s_t(a,z) g_t(a,z)\right] - \mathcal{A}^{z,*} g_t(a,z)$$

Macro block: given $g_0(a,z)$ and MIT shock $\{A_t\}_{t\geq 0}$

$$Y_t = A_t N_t \tag{4}$$

$$\dot{\pi}_t^w = \rho \pi_t^w + \frac{\epsilon}{\delta} \iint N_t \left(\frac{\epsilon - 1}{\epsilon} (1 + \tau^L) w_t z u'(c_t(a,z)) - v'(N_t) \right) g_t(a,z) \, da \, dz \tag{5}$$

$$r_t = i_t - \pi_t \tag{6}$$

$$w_t = A_t \tag{7}$$

$$i_t = r_{ss} + \lambda_\pi \pi_t + \lambda_Y \frac{Y_t - Y_{ss}}{Y_{ss}} \tag{8}$$

$$\pi_t = \pi_t^w - \frac{\dot{A}_t}{A_t} \tag{9}$$

$$0 = \iint s_t(a,z) \, g_t(a,z) \, da \, dz \tag{10}$$

Drop goods market clearing condition by Walras' law

Lemma (Implementability). Sequences

$$\left\{c_t(a,z), V_t(a,z), g_t(a,z), N_t, \pi_t^w\right\}$$

form part of a competitive equilibrium if and only if micro block conditions are satisfied, as well as

$$\dot{\pi}_t^w = \rho\pi_t^w + \frac{\epsilon}{\delta}\iint N_t\left(\frac{\epsilon-1}{\epsilon}(1+\tau^L)A_t z u'(c_t(a,z)) - v'(N_t)\right)g_t(a,z)\,da\,dz$$

$$0 = \iint s_t(a,z)\,g_t(a,z)\,da\,dz$$

Sequence Space

Finite-Difference Discretization

- Computational implementation of differential equations: finite-difference methods

  Achdou et al. (2022), Schaab-Zhang (2022)

- Time grid with $N+1$ discrete points: $t_0 = 0$ and $t_N = T$ (truncation horizon) with a step size $dt = \frac{T}{N-1}$, we have $t_n = dt(n-1)$

- Discretize individual states with J points (2 for earnings and $J/2$ for wealth)

- Discretization: approximate $c_t(a, z)$ at discrete points in time and space by vector $\boldsymbol{c}_n$

- $\boldsymbol{c}_n = (c_{1,n}, \dots, c_{J,n})'$, with $c_{i,n} = c_{t_n}(a_i, z_i)$

- Associate $i = 1$ with low-earnings types at borrowing constraint

Lemma. Consistent finite-difference discretization of equilibrium conditions:

$$\rho \boldsymbol{V}_n = \frac{\boldsymbol{V}_{n+1} - \boldsymbol{V}_n}{dt} + u(\boldsymbol{c}_n) - v(N_n) - \frac{\delta}{2}(\pi_n^w)^2 + \boldsymbol{s}_n \cdot \frac{\boldsymbol{D}_a}{da}\boldsymbol{V}_n + \boldsymbol{A}^z \boldsymbol{V}_n \tag{HJB}$$

$$u'(\boldsymbol{c}_{n,[2:J]}) = \left(\frac{\boldsymbol{D}_a}{da}\boldsymbol{V}_{n+1}\right)_{[2:J]} \tag{FOC}$$

$$\frac{\boldsymbol{g}_{n+1} - \boldsymbol{g}_n}{dt} = (\boldsymbol{A}^z)'\boldsymbol{g}_n + \frac{\boldsymbol{D}_a'}{da}\left[\boldsymbol{s}_n \cdot \boldsymbol{g}_n\right] \tag{KFE}$$

$$0 = \boldsymbol{s}_n' \boldsymbol{g}_n d\boldsymbol{x} \tag{BC}$$

$$\frac{\pi_{n+1}^w - \pi_n^w}{dt} = \rho \pi_n^w + \frac{\epsilon}{\delta}\left[\frac{\epsilon - 1}{\epsilon}(1 + \tau^L)A_n(\boldsymbol{z} \cdot u'(\boldsymbol{c}_n))'\boldsymbol{g}_n d\boldsymbol{x} - v'(N_n)\right]N_n \tag{NKPC}$$

and we have already used $c_{n,1} = i_n a_1 - \pi_n^w a_1 + \frac{A_{n+1} - A_n}{dt A_n} a_1 + z_1 A_n N_n$.

Remarks.

- We *assume* shocks small enough so upwind scheme $\boldsymbol{D}_a$ does not change
- Directly encode boundary condition for HTM $i = 1$ (Achdou et al. 2022)

$$\boldsymbol{s}_n = \begin{pmatrix} 0 \\ i_n \boldsymbol{a}_{[2:J]} - \pi_n^w \boldsymbol{a}_{[2:J]} + \frac{A_{n+1} - A_n}{dt A_n} \boldsymbol{a}_{[2:J]} + \boldsymbol{z}_{[2:J]} A_n N_n - \boldsymbol{c}_{n,[2:J]} \end{pmatrix}.$$

Sequence Space: Overview

What is the "sequence space approach"? Boppart et al. (2018), Auclert et al. (2021)

- Denote $\boldsymbol{X} = \{N_n, \pi_n^w\}_{n=0}^N$ and $\boldsymbol{Z} = \{A_n\}_{n=0}^N$
- What HANK literature (without optimal policy) has been doing for years:

$$\mathcal{H}(\boldsymbol{X}, \boldsymbol{Z}) = 0 \qquad \Longrightarrow \quad \boldsymbol{X}(\boldsymbol{Z}) \qquad \text{(Equilibrium Map)}$$

- Linearized dynamics around steady state:

$$d\boldsymbol{X} = \underbrace{-\mathcal{H}_{\boldsymbol{X}}^{-1}\mathcal{H}_{\boldsymbol{Z}}}_{\text{Sequence Space Jacobians}} \; d\boldsymbol{Z}$$

Sequence Space: Overview

What is the "sequence space approach"? Boppart et al. (2018), Auclert et al. (2021)

- Denote $\boldsymbol{X} = \{N_n, \pi_n^w\}_{n=0}^N$ and $\boldsymbol{Z} = \{A_n\}_{n=0}^N$
- What HANK literature (without optimal policy) has been doing for years:

$$\mathcal{H}(\boldsymbol{X}, \boldsymbol{Z}) = 0 \qquad \Longrightarrow \quad \boldsymbol{X}(\boldsymbol{Z}) \qquad \text{(Equilibrium Map)}$$

- Linearized dynamics around steady state:

$$d\boldsymbol{X} = \underbrace{-\mathcal{H}_{\boldsymbol{X}}^{-1}\mathcal{H}_{\boldsymbol{Z}}}_{\text{Sequence Space Jacobians}} \; d\boldsymbol{Z}$$

- Key: $\mathcal{H}(\cdot)$ incorporates all heterogeneity and SSJs are sufficient statistics for heterogeneity

Sequence Space: Equilibrium Map

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

Sequence Space: Equilibrium Map

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

Finite-difference discretization of **micro block**:

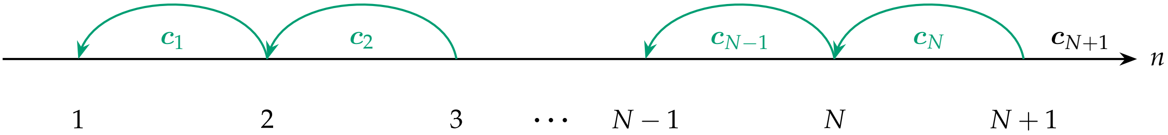
$$\boldsymbol{c}_n = \mathcal{C}(\boldsymbol{c}_{n+1}, X_n, Z_n)$$

Sequence Space: Equilibrium Map

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

Finite-difference discretization of **micro block**:

$$\boldsymbol{c}_n = \mathcal{C}(\boldsymbol{c}_{n+1}, X_n, Z_n)$$



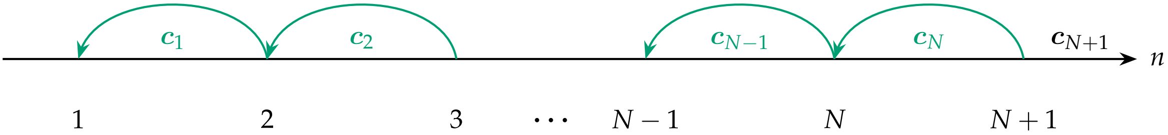
Sequence Space: Equilibrium Map

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

Finite-difference discretization of **micro block**:

$$\boldsymbol{c}_n = \mathcal{C}(\boldsymbol{c}_{n+1}, X_n, Z_n)$$

$$\boldsymbol{g}_{n+1} = \Lambda(\boldsymbol{c}_n, X_n, Z_n)\, \boldsymbol{g}_n$$



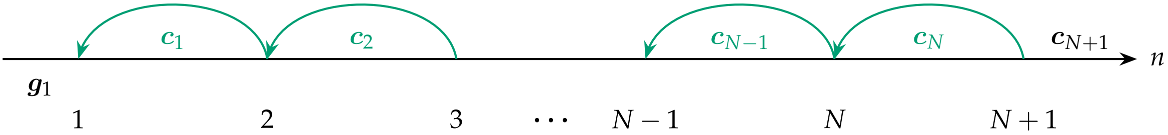
Sequence Space: Equilibrium Map

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

Finite-difference discretization of **micro block**:

$$\boldsymbol{c}_n = \mathcal{C}(\boldsymbol{c}_{n+1}, X_n, Z_n)$$

$$\boldsymbol{g}_{n+1} = \Lambda(\boldsymbol{c}_n, X_n, Z_n)\, \boldsymbol{g}_n$$



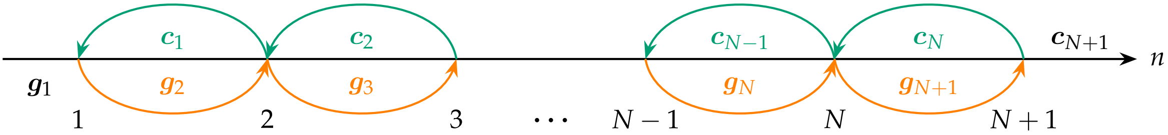
Sequence Space: Equilibrium Map

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

Finite-difference discretization of **micro block**:

$$\boldsymbol{c}_n = \mathcal{C}(\boldsymbol{c}_{n+1}, X_n, Z_n)$$

$$\boldsymbol{g}_{n+1} = \Lambda(\boldsymbol{c}_n, X_n, Z_n)\, \boldsymbol{g}_n$$



Sequence Space: Equilibrium Map

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

Finite-difference discretization of **micro block**:

$$\boldsymbol{c}_n = \mathcal{C}(\boldsymbol{c}_{n+1}, X_n, Z_n)$$

$$\boldsymbol{g}_{n+1} = \Lambda(\boldsymbol{c}_n, X_n, Z_n)\, \boldsymbol{g}_n$$

Macro block:

$$\mathcal{H}_n(\boldsymbol{X}, \boldsymbol{Z}) = 0 = \begin{pmatrix} \boldsymbol{s}_n' \boldsymbol{g}_n \\ -\frac{\pi_{n+1}^w - \pi_n^w}{dt} + \rho_n \pi_n^w + \frac{\epsilon_n}{\delta}\left(\frac{\epsilon_n - 1}{\epsilon_n}(1+\tau^L) A_n (\boldsymbol{z} \cdot u'(\boldsymbol{c}_n))' \boldsymbol{g}_n - v'(N_n)\right) N_n \end{pmatrix}$$

Sequence Space: Equilibrium Map

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

Finite-difference discretization of **micro block**:

$$\boldsymbol{c}_n = \mathcal{C}(\boldsymbol{c}_{n+1}, X_n, Z_n)$$

$$\boldsymbol{g}_{n+1} = \Lambda(\boldsymbol{c}_n, X_n, Z_n)\, \boldsymbol{g}_n$$

Macro block:

$$\mathcal{H}_n(\boldsymbol{X}, \boldsymbol{Z}) = 0 = \begin{pmatrix} \boldsymbol{s}_n' \boldsymbol{g}_n \\ -\frac{\pi^w_{n+1} - \pi^w_n}{dt} + \rho_n \pi^w_n + \frac{\epsilon_n}{\delta}\left(\frac{\epsilon_n - 1}{\epsilon_n}(1 + \tau^L) A_n (\boldsymbol{z} \cdot u'(\boldsymbol{c}_n))' \boldsymbol{g}_n - v'(N_n)\right) N_n \end{pmatrix}$$

Algorithm: Take $\boldsymbol{Z}$ and $(\boldsymbol{c}_{N+1}, \pi_{N+1}, \boldsymbol{g}_1)$ as given, guess $\boldsymbol{X}$, iterate until $\boldsymbol{H} = [\mathcal{H}_n] = 0$

Sequence Space: Algorithm

- Compute stationary equilibrium denoted $_{ss}$
- Initialize: $(\boldsymbol{c}_{N+1}, \pi_{N+1}, \boldsymbol{g}_1) = (\boldsymbol{c}_{ss}, \pi_{ss}, \boldsymbol{g}_{ss})$
- Take as given $\boldsymbol{Z}$ and guess $\boldsymbol{X}^0$
- Solve HJB and KFE (illustrated previous slide)
- Compute $\{\mathcal{H}_n\}_{n=1}^{N}$ and construct $2N \times 1$ vector $\boldsymbol{H} = [\mathcal{H}_n]$
- Use “gap” in $\boldsymbol{H} \neq \boldsymbol{0}$ to update $\boldsymbol{X}^1$ and repeat

Sequence Space: Algorithm

How to do this in practice?

Sequence Space: Algorithm

How to do this in practice?

Two approaches: linear and non-linear (i.e., non-linear MIT shock transition)

Sequence Space: Algorithm

How to do this in practice?

Two approaches: linear and non-linear (i.e., non-linear MIT shock transition)

Linear:

$$\mathcal{H}(\boldsymbol{X}, \boldsymbol{Z}) \approx \mathcal{H}(\boldsymbol{X}_{ss}, \boldsymbol{Z}_{ss}) + \mathcal{H}_{\boldsymbol{X}}(\boldsymbol{X} - \boldsymbol{X}_{ss}) + \mathcal{H}_{\boldsymbol{Z}}(\boldsymbol{Z} - \boldsymbol{Z}_{ss})$$

Sequence Space: Algorithm

How to do this in practice?

Two approaches: linear and non-linear (i.e., non-linear MIT shock transition)

Linear:

$$0 \approx \mathcal{H}(\boldsymbol{X}_{ss}, \boldsymbol{Z}_{ss}) + \mathcal{H}_{\boldsymbol{X}}(\boldsymbol{X} - \boldsymbol{X}_{ss}) + \mathcal{H}_{\boldsymbol{Z}}(\boldsymbol{Z} - \boldsymbol{Z}_{ss})$$

Sequence Space: Algorithm

How to do this in practice?

Two approaches: linear and non-linear (i.e., non-linear MIT shock transition)

Linear:

$$0 \approx 0 + \mathcal{H}_{\boldsymbol{X}}(\boldsymbol{X} - \boldsymbol{X}_{ss}) + \mathcal{H}_{\boldsymbol{Z}}(\boldsymbol{Z} - \boldsymbol{Z}_{ss})$$

Sequence Space: Algorithm

How to do this in practice?

Two approaches: linear and non-linear (i.e., non-linear MIT shock transition)

Linear:

$$\boldsymbol{X} \approx \boldsymbol{X}_{ss} - \underbrace{\mathcal{H}_{\boldsymbol{X}}^{-1}\mathcal{H}_{\boldsymbol{Z}}}_{\text{Sequence Space Jacobians of } \mathcal{H}} (\boldsymbol{Z} - \boldsymbol{Z}_{ss})$$

Sequence Space: First-Order Perturbation

- In other words, since $\mathcal{H}(\boldsymbol{X}, \boldsymbol{Z}) \implies \boldsymbol{X}(\boldsymbol{Z})$:

$$d\boldsymbol{X} = \boldsymbol{X}_{ss} + \boldsymbol{X}_{\boldsymbol{Z}} d\boldsymbol{Z},$$

- Here $d\boldsymbol{Z} = \boldsymbol{Z} - \boldsymbol{Z}_{ss}$ and key object to compute is $\boldsymbol{X}_{\boldsymbol{Z}}$
- By the implicit function theorem, we have $H_{\boldsymbol{X}} \boldsymbol{X}_{\boldsymbol{Z}} + H_{\boldsymbol{Z}} = 0$, or simply

$$\boldsymbol{X}_{\boldsymbol{Z}} = -H_{\boldsymbol{X}}^{-1} H_{\boldsymbol{Z}},$$

- Let K be $N \cdot \#$ of macro guesses (here $K = 2N$)
- $H_{\boldsymbol{Z}}$ is a $K \times N$ and $H_{\boldsymbol{X}}$ a $K \times K$, whose ijth elements are $\frac{\partial H_i}{\partial \boldsymbol{Z}_j}$ and $\frac{\partial H_i}{\partial \boldsymbol{X}_j}$
- **Conclusion**: first-order perturbation requires Sequence-Space Jacobians $(\boldsymbol{H}_{\boldsymbol{X}}, \boldsymbol{H}_{\boldsymbol{Z}})$

Sequence Space: Non-Linear MIT Shock

- Alternatively, use **Newton** class of algorithms to solve $\mathcal{H}(\boldsymbol{X}, \boldsymbol{Z}) = 0$
- Most powerful approach in practice: quasi-Newton (Broyden)
- Why? Compute Jacobians **once**, then iterate recursively
- **Conclusion**: once $\boldsymbol{H_X}$ and $\boldsymbol{H_Z}$ are computed for first-order perturbation, non-linear MIT dynamcis come for free via quasi-Newton
- Feel free to use my custom quasi-Newton algorithm! (`SparseEcon`)

Fake News Algorithm

Overview

- Computing $\boldsymbol{H_X}$ requires $2N$ function evaluations of $\mathcal{H}(\cdot)$

$$\frac{\partial H_i}{\partial \boldsymbol{X}_j} \approx \frac{\mathcal{H}_i(\boldsymbol{X} + \boldsymbol{\epsilon}_j, \boldsymbol{Z}) - \mathcal{H}_i(\boldsymbol{X}, \boldsymbol{Z})}{h}$$

- This is very costly: function eval of $\mathcal{H}(\cdot)$ requires solving HJB backwards (N linear systems) and KFE forwards (N linear systems)
- Fake-news algorithm: turns out 1 function evaluation is enough (per # of guesses $\frac{K}{N}$)
- No longer scales with N!!

Policy Functions

- Let $\theta_n \in \{N_n, \pi_n^w, A_n\}$
- Abuse notation for clarity: use t instead of n
- Recall: $\boldsymbol{c}_t = \mathcal{C}(X_t, Z_t, \boldsymbol{c}_{t+1})$

Lemma. Auclert et al. (2021)

$$\partial \boldsymbol{c}_t^s = \frac{\partial \boldsymbol{c}_t}{\partial \theta_s} = \frac{\partial \boldsymbol{c}_{t-k}}{\partial \theta_{s-k}}.$$

- Why?
- **Remark:** Any (forward-looking) *backward* equation satisfies this property (not just backward equations that emerge from dynamic programming)

Policy Functions

Lemma. Suppose $t < s$, then to first order

$$\partial \boldsymbol{c}_t^s = \frac{\partial \boldsymbol{c}_t}{\partial \theta_s} = \mathcal{C}_{\boldsymbol{c}} \frac{\partial \boldsymbol{c}_{t+1}}{\partial \theta_s} = \mathcal{C}_{\boldsymbol{c}}^{s-t} \mathcal{C}_\theta,$$

where $\mathcal{C}_{\boldsymbol{c}}$ and $\mathcal{C}_\theta$ are evaluated at steady state.

Distribution

- Letting $\boldsymbol{\Lambda}_t = 1 + \boldsymbol{A}_t' dt$, discretized KFE takes form: $\boldsymbol{g}_{t+1} = \boldsymbol{\Lambda}_t \boldsymbol{g}_t$
- Differentiating, we have to first order around the steady state

$$\frac{\partial \boldsymbol{g}_{t+1}}{\partial \theta_s} = \frac{\partial \boldsymbol{\Lambda}_t}{\partial \theta_s} \boldsymbol{g}_{ss} + \boldsymbol{\Lambda}_{ss} \frac{\partial \boldsymbol{g}_t}{\partial \theta_s}$$

Lemma. Sequence space derivatives of the KF matrix (adjoint) $\boldsymbol{\Lambda}_t$ satisfy

$$\frac{\partial \boldsymbol{\Lambda}_t}{\partial \theta_s} = \frac{\partial \boldsymbol{\Lambda}_0}{\partial \theta_{s-t}}$$

to first order around steady state. Implementation relevant representation is

$$\frac{\partial \boldsymbol{\Lambda}_t}{\partial \theta_s} = \begin{cases} \frac{\partial \boldsymbol{\Lambda}_{t+(N-s)}}{\partial \theta_N} & \text{if } s \geq t \\ 0 & \text{if } s < t \end{cases}$$

Distribution

Lemma. We have

(a)

$$\frac{\partial \boldsymbol{g}_t}{\partial \theta_s} = \sum_{k=0}^{t-1} \boldsymbol{\Lambda}_{ss}^{k} \frac{\partial \boldsymbol{\Lambda}_{t-1-k}}{\partial \theta_s} \boldsymbol{g}^{\text{ss}} + \boldsymbol{\Lambda}_{ss}^{t-1} \frac{\partial \boldsymbol{g}_1}{\partial \theta_s}$$

(b)

$$\frac{\partial \boldsymbol{g}_t}{\partial \theta_s} = \sum_{k=1}^{t} \boldsymbol{\Lambda}_{ss}^{k-1} \frac{\partial \boldsymbol{g}_1}{\partial \theta_{s-t+k}}$$

(c)

$$\frac{\partial \boldsymbol{g}_t}{\partial \theta_s} = \frac{\partial \boldsymbol{\Lambda}_0}{\partial \theta_{s-(t-1)}} \boldsymbol{g}_{ss} + \boldsymbol{\Lambda}_{ss} \frac{\partial \boldsymbol{g}_{t-1}}{\partial \theta_s}$$

Derivation

We have

$$
\begin{aligned}
d\boldsymbol{g}_t^s &= d((1+\boldsymbol{A}_{t-1})\boldsymbol{g}_{t-1}) \\
&= (1+d\boldsymbol{A}_{t-1}^s)\boldsymbol{g}_{t-1} + (1+\boldsymbol{A}_{t-1})d\boldsymbol{g}_{t-1}^s \\
&= (1+d\boldsymbol{A}_{t-1}^s)\boldsymbol{g}_{t-1} + (1+\boldsymbol{A}_{t-1})\Big((1+d\boldsymbol{A}_{t-2}^s)\boldsymbol{g}_{t-2} + (1+\boldsymbol{A}_{t-2})d\boldsymbol{g}_{t-2}^s\Big) \\
&= (1+d\boldsymbol{A}_{t-1}^s)\boldsymbol{g}_{t-1} + (1+\boldsymbol{A}_{t-1})(1+d\boldsymbol{A}_{t-2}^s)\boldsymbol{g}_{t-2} + (1+\boldsymbol{A}_{t-1})(1+\boldsymbol{A}_{t-2})\Big((1+d\boldsymbol{A}_{t-3}^s)\boldsymbol{g}_{t-3} + (1
\end{aligned}
$$

and so we arrive at

$$
\begin{aligned}
d\boldsymbol{g}_t^s =& (1+d\boldsymbol{A}_{t-1}^s)\boldsymbol{g}_{t-1} + (1+\boldsymbol{A}_{t-1})(1+d\boldsymbol{A}_{t-2}^s)\boldsymbol{g}_{t-2} \\
& + (1+\boldsymbol{A}_{t-1})(1+\boldsymbol{A}_{t-2})(1+d\boldsymbol{A}_{t-3}^s)\boldsymbol{g}_{t-3} + (1+\boldsymbol{A}_{t-1})(1+\boldsymbol{A}_{t-2})(1+\boldsymbol{A}_{t-3})d\boldsymbol{g}_{t-3}^s \\
=& (1+d\boldsymbol{A}_{t-1}^s)\boldsymbol{g}^{\text{ss}} + (1+\boldsymbol{A}^{\text{ss}})(1+d\boldsymbol{A}_{t-2}^s)\boldsymbol{g}^{\text{ss}} \\
& + (1+\boldsymbol{A}^{\text{ss}})(1+\boldsymbol{A}^{\text{ss}})(1+d\boldsymbol{A}_{t-3}^s)\boldsymbol{g}_{t-3} + (1+\boldsymbol{A}^{\text{ss}})(1+\boldsymbol{A}^{\text{ss}})(1+\boldsymbol{A}^{\text{ss}})d\boldsymbol{g}_{t-3}^s
\end{aligned}
$$

By induction, representation (a) follows.

Derivation

Representations (b) and (c) follow from the following derivation. We know that

$$
\begin{aligned}
d\boldsymbol{g}_t^s &= (1 + d\boldsymbol{A}_{t-1}^s)\boldsymbol{g}^{\text{ss}} + (1 + \boldsymbol{A}^{\text{ss}})d\boldsymbol{g}_{t-1}^s \\
&= (1 + d\boldsymbol{A}_0^{s-(t-1)})\boldsymbol{g}^{\text{ss}} + (1 + \boldsymbol{A}^{\text{ss}})d\boldsymbol{g}_{t-1}^s \\
&= d\boldsymbol{g}_1^{s-(t-1)} + (1 + \boldsymbol{A}^{\text{ss}})d\boldsymbol{g}_{t-1}^s \\
&= d\boldsymbol{g}_1^{s-(t-1)} + (1 + \boldsymbol{A}^{\text{ss}})\left[d\boldsymbol{g}_1^{s-(t-2)} + (1 + \boldsymbol{A}^{\text{ss}})d\boldsymbol{g}_{t-2}^s\right] \\
&= \sum_{k=1}^{t}(1 + \boldsymbol{A}^{\text{ss}})^{k-1}d\boldsymbol{g}_1^{s-t+k}
\end{aligned}
$$

Adjoint

- KF matrix $\boldsymbol{\Lambda}_t = 1 + dt\,\boldsymbol{A}_t'$ has **special structure** in continuous time. Recall:

$$\frac{\boldsymbol{g}_{t+1} - \boldsymbol{g}_t}{dt} = (\boldsymbol{A}^z)'\boldsymbol{g}_t + \sum_{i\geq 2} \frac{\boldsymbol{D}'_{a,[i,:]}}{da}\left[\begin{pmatrix} 0 \\ r_t\boldsymbol{a}_{[2:J]} + \boldsymbol{z}_{[2:J]}w_t N_t - \boldsymbol{c}_{t,[2:J]} \end{pmatrix} \cdot \boldsymbol{g}_t\right]$$

- Or simply:

$$\boldsymbol{g}_{t+1} = \boldsymbol{g}_t + dt\left[(\boldsymbol{A}^z)'\boldsymbol{g}_t + \frac{1}{da}\boldsymbol{D}_a'(\boldsymbol{s}_t \cdot \boldsymbol{g}_t)\right].$$

Lemma. The adjoint operator of the Kolmogorov forward equation satisfies

$$\frac{\partial \boldsymbol{\Lambda}_t}{\partial \theta_s} = \frac{dt}{da}\left(\frac{\partial \boldsymbol{s}_t}{\partial \theta_s} \cdot \boldsymbol{D}_a\right)',$$

Distribution

- Adjoint satisfies: $\frac{\partial \boldsymbol{\Lambda}_t}{\partial \theta_s} = \frac{dt}{da}\left(\frac{\partial \boldsymbol{s}_t}{\partial \theta_s} \cdot \boldsymbol{D}_a\right)'$

- Using special structure of adjoint, we have

$$\frac{\partial \boldsymbol{g}_{t+1}}{\partial \theta_s} = \boldsymbol{\Lambda}_{ss} \frac{\partial \boldsymbol{g}_t}{\partial \theta_s} + \frac{dt}{da}\left(\frac{\partial \boldsymbol{s}_t}{\partial \theta_s} \cdot \boldsymbol{D}_a\right)' \boldsymbol{g}_{ss} = \boldsymbol{\Lambda}_{ss} \frac{\partial \boldsymbol{g}_t}{\partial \theta_s} + \frac{dt}{da}(\boldsymbol{g}_{ss} \cdot \boldsymbol{D}_a)' \frac{\partial \boldsymbol{s}_t}{\partial \theta_s}$$

Lemma. Solving the recursion, we can express sequence space derivatives of the distribution entirely in terms of derivatives of policy functions:

$$\frac{\partial \boldsymbol{g}_{t+1}}{\partial \theta_s} = \frac{dt}{da} \sum_{k=0}^{t} \boldsymbol{\Lambda}_{ss}^k (\boldsymbol{g}_{ss} \cdot \boldsymbol{D}_a)' \frac{\partial \boldsymbol{s}_{t-k}}{\partial \theta_s}$$

Sequence Space Jacobians of Equilibrium Map

- **We are done**! We can now compute $\boldsymbol{H_X}$ and $\boldsymbol{H_Z}$

- Consider bond market clearing condition (similar for NKPC):

$$H_t = \boldsymbol{s}_t' \boldsymbol{g}_t,$$

- This implies:

$$\frac{\partial H_t}{\partial \theta_s} = \boldsymbol{s}_{ss}' \frac{\partial \boldsymbol{g}_t}{\partial \theta_s} + \boldsymbol{g}_{ss}' \frac{\partial \boldsymbol{s}_t}{\partial \theta_s}$$

- We have $\frac{\partial \boldsymbol{s}_t}{\partial \theta_s}$ from a single function evaluation of $\mathcal{H}(\cdot)$ and we solve for $\frac{\partial \boldsymbol{g}_t}{\partial \theta_s}$ entirely in terms of policy function derivatives!!!

Implementation

- In practice, macro block has structure given by

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

where $\boldsymbol{C}$ is a vector of micro block aggregates: $\boldsymbol{C}_t = \boldsymbol{c}_t' \boldsymbol{g}_t$

- Right way to think about it is that $\boldsymbol{C} = \boldsymbol{C}(\boldsymbol{X}, \boldsymbol{Z})$ and:

$$H_{\boldsymbol{X}} d\boldsymbol{X} + H_{\boldsymbol{C}} \left(\boldsymbol{C}_{\boldsymbol{X}} d\boldsymbol{X} + \boldsymbol{C}_{\boldsymbol{Z}} d\boldsymbol{Z} \right) + H_{\boldsymbol{Z}} d\boldsymbol{Z} = 0,$$

- This implies: $d\boldsymbol{X} = -(H_{\boldsymbol{X}} + H_{\boldsymbol{C}} \boldsymbol{C}_{\boldsymbol{X}})^{-1} (H_{\boldsymbol{C}} \boldsymbol{C}_{\boldsymbol{Z}} + H_{\boldsymbol{Z}}) d\boldsymbol{Z}$

- And we obtain the only matrix that we don't have yet via:

$$(\boldsymbol{C}_{\boldsymbol{Z}})_{ts} = \boldsymbol{c}_{ss}' \frac{\partial \boldsymbol{g}_t}{\partial Z_s} + \boldsymbol{g}_{ss}' \frac{\partial \boldsymbol{c}_t}{\partial Z_s}$$