Universität Zürich[UZH]

Essays in finance

Luca Gaegauf | luca.gaegauf@bf.uzh.ch

Chapters

1) Deep Equilibrium Nets
2) A comprehensive machine learning framework for dynamic portfolio choice with transaction costs
3) Empirical causal asset pricing with trading costs

Universität Zürich[UZH]

Deep Equilibrium Nets

Marlon Azinovic
University of Pennsylvania

Luca Gaegauf
University of Zurich

Simon Scheidegger
University of Lausanne

Agenda

1) Motivation
2) Economic model
3) Algorithm
4) Evaluation
5) Economic results

1) Motivation

Computationally challenging model features

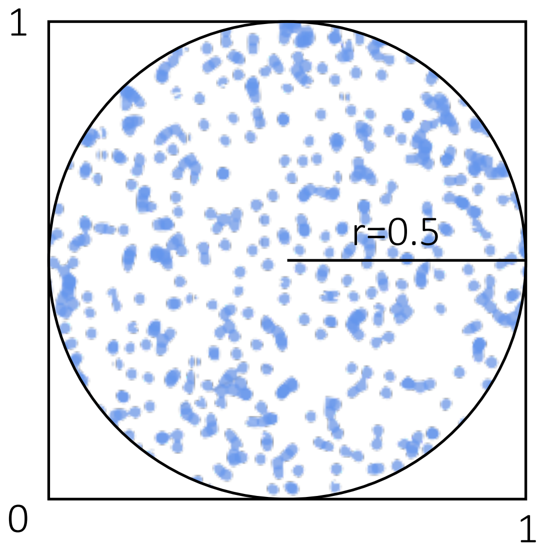
Stochasticity	Irregular geometry of state space	Strong nonlinearities and/or kinks in equilibrium functions	High-dimensional state space

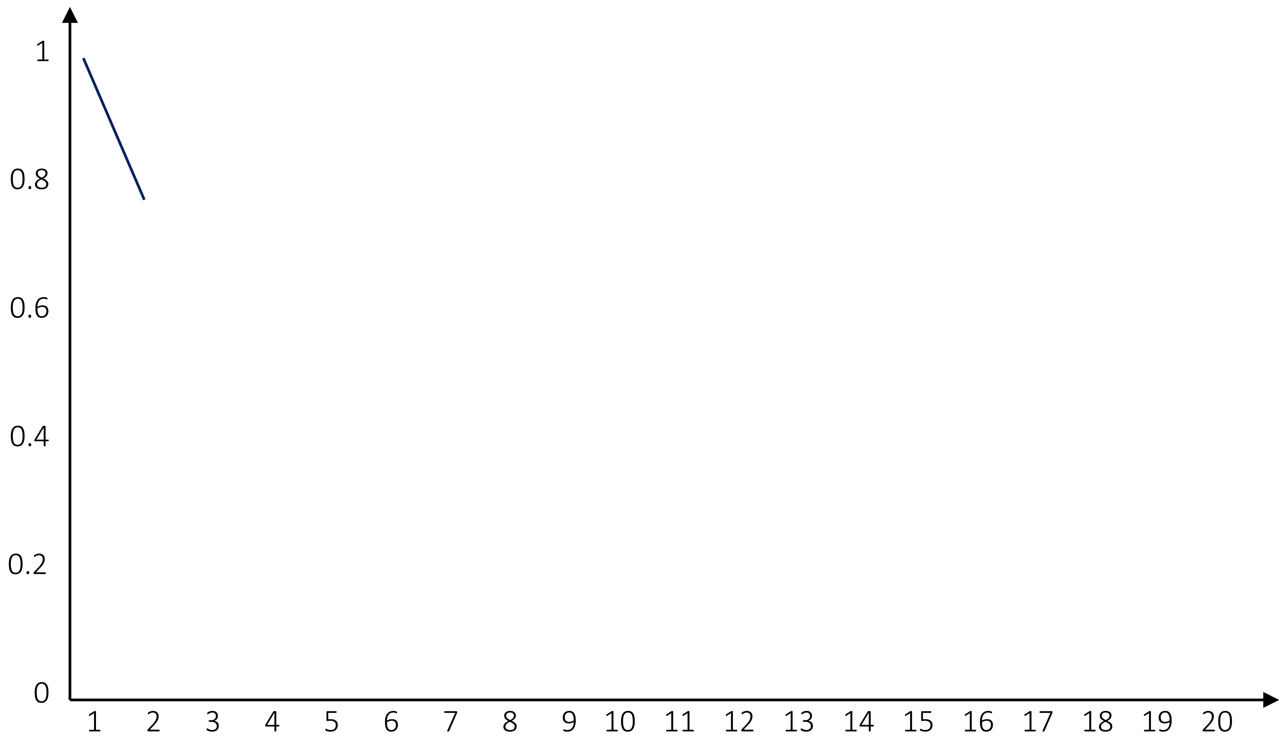
1) Motivation
Computationally challenging model features

Stochasticity	Irregular geometry of state space	Strong nonlinearities and/or kinks in equilibrium functions	High-dimensional state space

1) Motivation
High-dimensional irregular state space geometries

D = 2





Stochasticity

Irregular geometry of state space

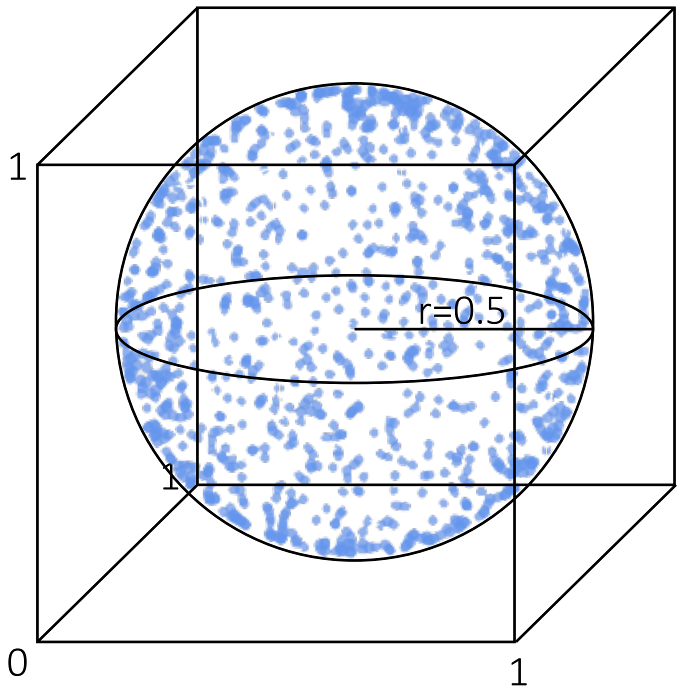
Strong nonlinearities and/or kinks in equilibrium functions

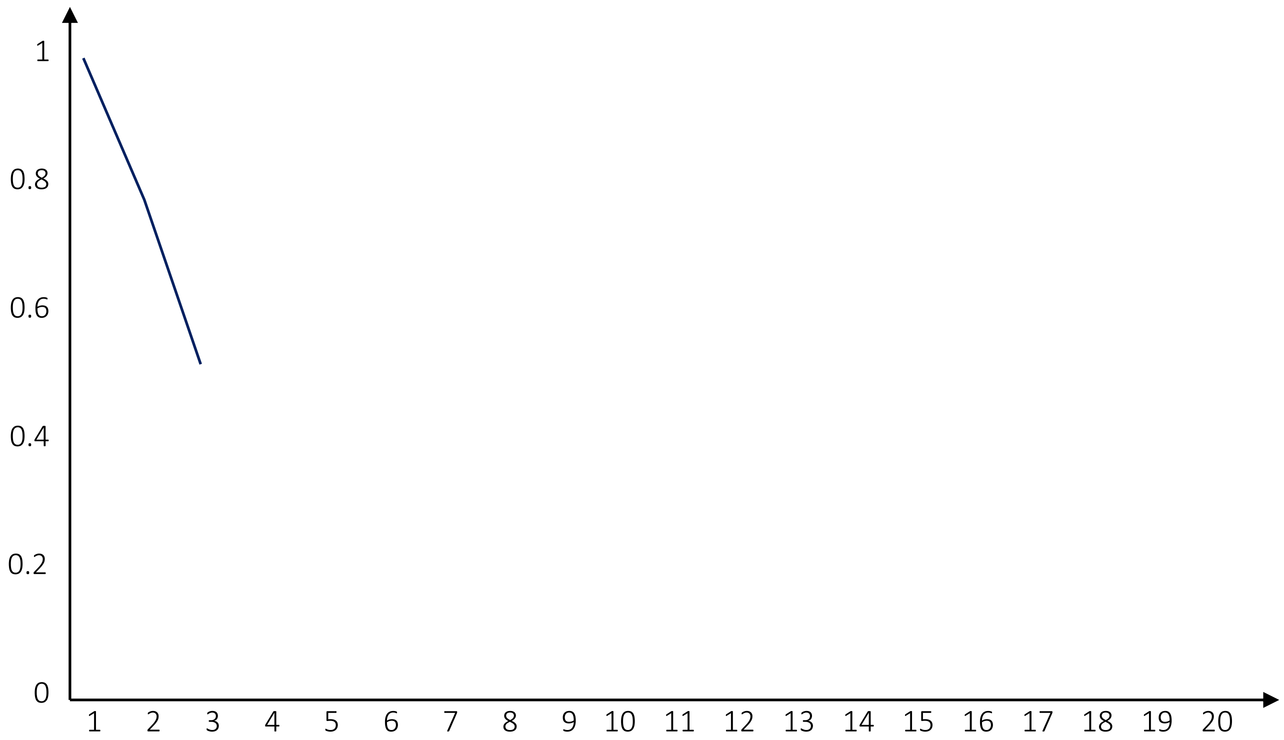
High-dimensional state space

1) Motivation

High-dimensional irregular state space geometries

D = 3



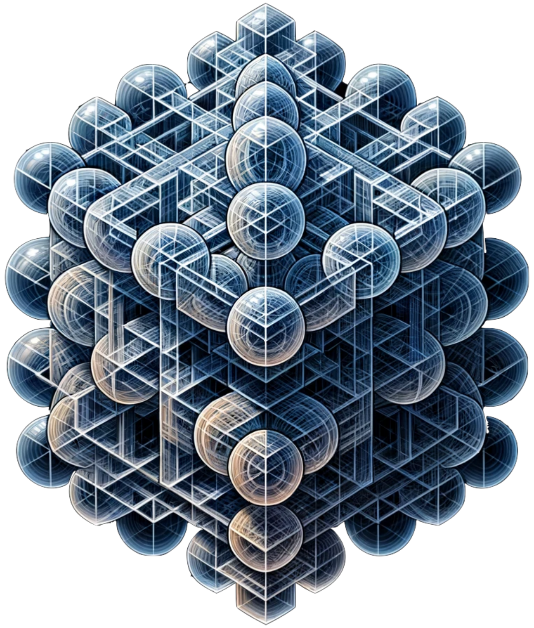


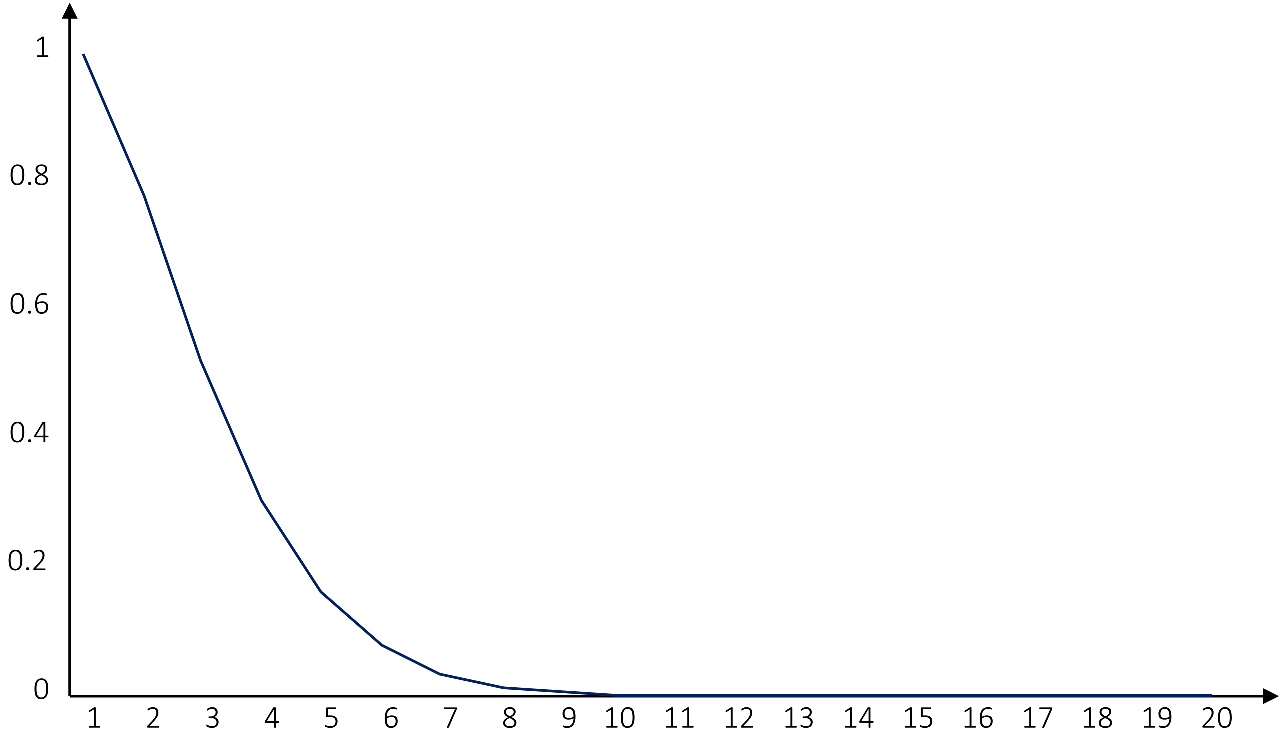
Stochasticity	Irregular geometry of state space	Strong nonlinearities and/or kinks in equilibrium functions	High-dimensional state space

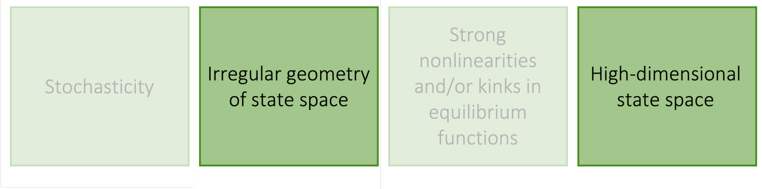
Luca Gaegauf | luca.gaegauf@bf.uzh.ch

1) Motivation
High-dimensional irregular state space geometries

D ≥ 4







DALL-E generated image

1) Motivation
Existing methods can deal with subsets…

- (Smolyak) sparse grids: Krueger & Kubler (2004), Winschel & Krätzig (2010)
- Adaptive sparse grids: Brumm, et al. (2017), Brumm & Scheidegger (2017), Schober, et al. (2020)
- Machine learning methods: e.g., Scheidegger & Bilionis (2019)
- Honorable mentions: Maliar & Maliar (2015), Muthuraman and Zha (2008)

	High-dimensional input	Can resolve local features accurately	Irregularly shaped domains	Large amount of data
Polynomials	✓	✗	✓	✓
Splines	✗	✓	✗	✓
Adaptive sparse grids	✓	✓	✗	✓
Gaussian processes	✓	✓	✓	✗

2) Benchmark economic model

OLG model based on Krueger and Kübler (2004)

Stochasticity	Irregular geometry of state space	Strong nonlinearities and/or kinks in equilibrium functions	High-dimensional state space
Stochastic production with discrete TFP and depreciation shocks	Ergodic set is not hypercube	Occasionally binding constraints and adjustment costs	State space increases linearly with the number of age groups

2) Benchmark economic model
Households

Households maximize:

Agent age s lives for N − s periods

Agent's time discount factor ($\beta < 1$)

$$(c_t^s\, a_t^s\, d_t^s) \in \arg\max_{c_t^s\, a_t^s\, d_t^s} \sum_{i=0}^{N-s} \mathrm{E}_t[\beta^i u(c_{t+i}^{s+i})]$$

Agent's utility function (standard assumptions)

2) Benchmark economic model
Households

Households maximize:

$$(c_t^s\, a_t^s\, d_t^s) \in \arg\max_{c_t^s\, a_t^s\, d_t^s} \sum_{i=0}^{N-s} \mathrm{E}_t\left[\beta^i u(c_{t+i}^{s+i})\right]$$

subject to:

$$c_t^s + p_t d_t^s + a_t^s + \underbrace{\frac{\zeta}{2}(a_t^s - k_t^s r_t)^2}_{\text{Adjustment cost}} = l^s w_t + b_t^s + r_t k_t^s$$

Illiquid risky capital

$$a_t^s = k_{t+1}^{s+1}$$

$$a_t^s \geq \underline{a}$$

Borrowing constraint

Liquid one-period bond

$$b_{t+1}^{s+1} = d_t^s$$

$$\kappa d_t^s + a_t^s \geq 0 \qquad \forall t, \quad \forall s \in \{1, \ldots, N-1\}$$

Collateral constraint

2) Benchmark economic model
The firm

Households maximize:

$$(c_t^s\, a_t^s\, d_t^s) \in \arg\max_{c_t^s\, a_t^s\, d_t^s} \sum_{i=0}^{N-s} \mathrm{E}_t\left[\beta^i u(c_{t+i}^{s+i})\right]$$

subject to:

$$c_t^s + p_t d_t^s + a_t^s + \frac{\zeta}{2}(a_t^s - k_t^s r_t)^2 = l^s w_t + b_t^s + r_t k_t^s$$

$$a_t^s = k_{t+1}^{s+1}$$

$$a_t^s \geq \underline{a}$$

$$b_{t+1}^{s+1} = d_t^s$$

$$\kappa d_t^s + a_t^s \geq 0 \qquad \forall t, \quad \forall s \in \{1, \ldots, N-1\}$$

The firm maximizes profits:

$$(K_t, L_t) \in \arg\max_{K_t, L_t \geq 0} f(K_t, L_t, z_t) - r_t K_t - w_t L_t.$$

where:

Single representative Cobb-Douglas firm

$$f(K_t, L_t, z_t) = \eta_t K_t^{\alpha} L_t^{1-\alpha} + K_t(1-\delta_t)$$

with implied rates:

TFP and depreciation are exogenous shocks

$$w_t = (1-\alpha)\eta_t K_t^{\alpha} L_t^{-\alpha},$$

$$r_t = \alpha \eta_t K_t^{\alpha-1} L_t^{1-\alpha} + (1-\delta_t)$$

2) Benchmark economic model
Competitive equilibrium

Households maximize:

$$(c_t^s\, a_t^s\, d_t^s) \in \arg\max_{c_t^s\, a_t^s\, d_t^s} \sum_{i=0}^{N-s} \mathrm{E}_t\left[\beta^i u(c_{t+i}^{s+i})\right]$$

subject to:

$$c_t^s + p_t d_t^s + a_t^s + \frac{\zeta}{2}(a_t^s - k_t^s r_t)^2 = l^s w_t + b_t^s + r_t k_t^s$$

$$a_t^s = k_{t+1}^{s+1}$$

$$a_t^s \geq \underline{a}$$

$$b_{t+1}^{s+1} = d_t^s$$

$$\kappa d_t^s + a_t^s \geq 0 \qquad \forall t, \quad \forall s \in \{1, \ldots, N-1\}$$

The firm maximizes profits:

$$(K_t, L_t) \in \arg\max_{K_t, L_t \geq 0} f(K_t, L_t, z_t) - r_t K_t - w_t L_t.$$

where:

$$f(K_t, L_t, z_t) = \eta_t K_t^\alpha L_t^{1-\alpha} + K_t(1-\delta_t)$$

with implied rates:

$$w_t = (1-\alpha)\eta_t K_t^\alpha L_t^{-\alpha},$$

$$r_t = \alpha \eta_t K_t^{\alpha-1} L_t^{1-\alpha} + (1-\delta_t)$$

Markets clear:

$$L_t = \sum_{s=1}^{N} l_t^s, \quad K_t = \sum_{s=1}^{N} k_t^s, \quad 0 = \sum_{s=1}^{N} b_t^s$$

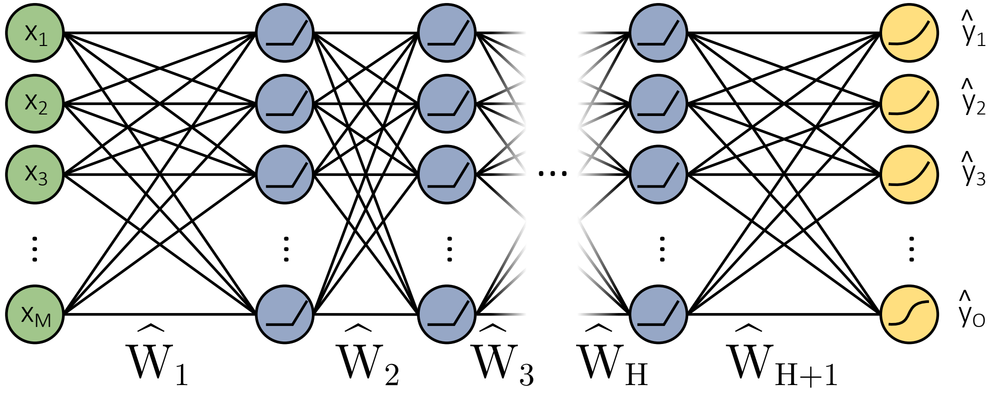
3) Algorithm

Main DEQN components

Approximate	Optimize	Data
Neural networks	Errors in the equilibrium conditions	Simulation

3) Algorithm
Function approximator: neural networks

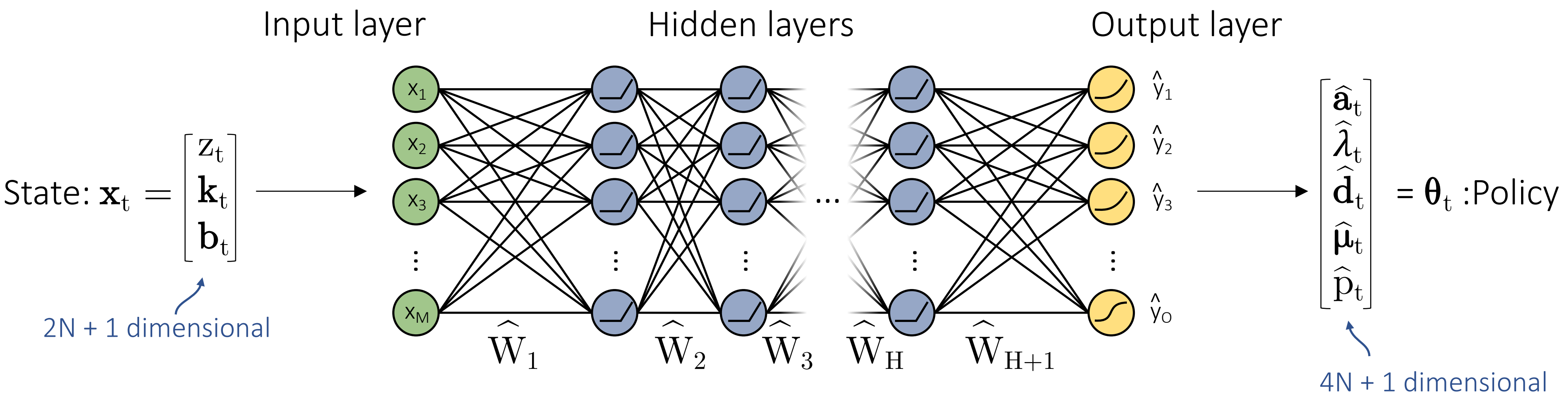
- Neural networks are non-linear function approximators that can handle large datasets.



$$\hat{y} = f_o(f_h(\ldots f_h(f_h(X\widehat{W}_1)\widehat{W}_2)\ldots\widehat{W}_H)\widehat{W}_{H+1})$$

3) Algorithm
Function approximator: neural networks

- Neural networks are non-linear function approximators that can handle large datasets.



$$\hat{y} = f_o(f_h(\ldots f_h(f_h(X\widehat{W}_1)\widehat{W}_2)\ldots\widehat{W}_H)\widehat{W}_{H+1})$$

3) Algorithm

Loss function: minimize errors in equilibrium functions

- First-order conditions:

$$(1+\zeta\,\Delta_t^s)u'(c_t^s) = \beta \mathrm{E}_t\big[u'(c_{t+1}^{s+1})r_{t+1}\big(1+\zeta\,\Delta_{t+1}^{s+1}\big)\big] + \lambda_t^s + \mu_t^s,$$

$$p_t u'(c_t^s) = \beta \mathrm{E}_t\big[u'(c_{t+1}^{s+1})\big] + \kappa\,\mu_t^s,$$

- KKT conditions:

$$\begin{aligned} 0 &= \lambda_t^s \cdot (a_t^s - \underline{a}) \\ 0 &= \mu_t^s \cdot (a_t^s + \kappa\, d_t^s) \end{aligned}$$

- Market clearing:

$$0 = \sum_{s=1}^{N} b_t^s$$

3) Algorithm

Loss function: minimize errors in equilibrium functions

- First-order conditions:

$$0 = \frac{u'^{-1}\left(\frac{\beta \mathrm{E}_t\left[u'(\hat{c}_{t+1}^{s+1}) r_{t+1}\left(1+\zeta\,\hat{\Delta}_{t+1}^{s+1}\right)\right] + \hat{\lambda}_t^s + \hat{\mu}_t^s}{(1+\zeta\,\hat{\Delta}_t^s)}\right)}{\hat{c}_t^s} - 1$$

$$0 = \frac{u'^{-1}\left(\frac{\beta \mathrm{E}_t\left[u'(\hat{c}_{t+1}^{s+1})\right] + \kappa\hat{\mu}_t^s}{\hat{p}_t}\right)}{\hat{c}_t^s} - 1$$

- KKT conditions:

$$0 = \hat{\lambda}_t^s \cdot (\hat{a}_t^s - \underline{a})$$

$$0 = \hat{\mu}_t^s \cdot (\hat{a}_t^s + \kappa\hat{d}_t^s)$$

- Market clearing:

$$0 = \sum_{s=1}^{N} \hat{d}_t^s$$

3) Algorithm
Loss function: minimize errors in equilibrium functions

- First-order conditions:

$$0 = \frac{u'^{-1}\left(\frac{\beta \mathrm{E}_t\left[u'(\hat{c}_{t+1}^{s+1}) r_{t+1}(1+\zeta \hat{\Delta}_{t+1}^{s+1})\right] + \hat{\lambda}_t^s + \hat{\mu}_t^s}{(1+\zeta \hat{\Delta}_t^s)}\right)}{\hat{c}_t^s} - 1$$

$$0 = \frac{u'^{-1}\left(\frac{\beta \mathrm{E}_t\left[u'(\hat{c}_{t+1}^{s+1})\right] + \kappa \hat{\mu}_t^s}{\hat{p}_t}\right)}{\hat{c}_t^s} - 1$$

- KKT conditions:

$$0 = \hat{\lambda}_t^s \cdot (\hat{a}_t^s - \underline{a})$$

$$0 = \hat{\mu}_t^s \cdot (\hat{a}_t^s + \kappa \hat{d}_t^s)$$

- Market clearing:

$$0 = \sum_{s=1}^{N} \hat{d}_t^s$$

⇒ Loss function:

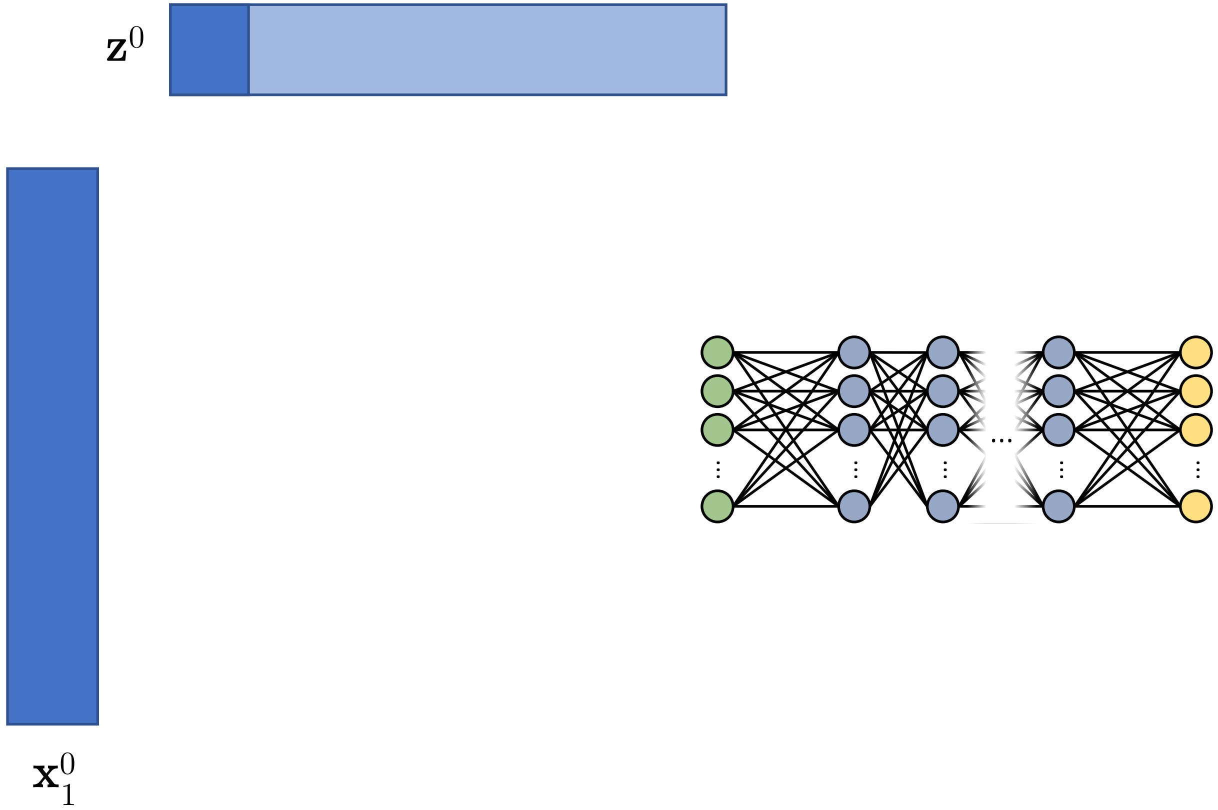
$$\boxed{\frac{1}{|\mathcal{D}_{\text{train}}|} \sum_{\mathbf{x}_j \in \mathcal{D}_{\text{train}}} \left(\frac{1}{N-1} \sum_{i=1}^{N-1} \left(\hat{e}_{\mathbf{x}_j}^{i,\text{REE cap}}\right)^2 + \left(\hat{e}_{\mathbf{x}_j}^{i,\text{REE bond}}\right)^2 + \left(\hat{e}_{\mathbf{x}_j}^{i,\text{KKT cap}}\right)^2 + \left(\hat{e}_{\mathbf{x}_j}^{i,\text{KKT bond}}\right)^2 \right) + \left(\hat{e}_{\mathbf{x}_j}^{\text{mc}}\right)^2}$$

3) Algorithm

Data sampling: simulating on the path of an economy

- Simulate a series of T-1 exogenous shocks.
- Sample M starting states.
- Use the neural network approximated policy function to simulate agents’ choices.

3) Algorithm
Final algorithm pipeline: simulation

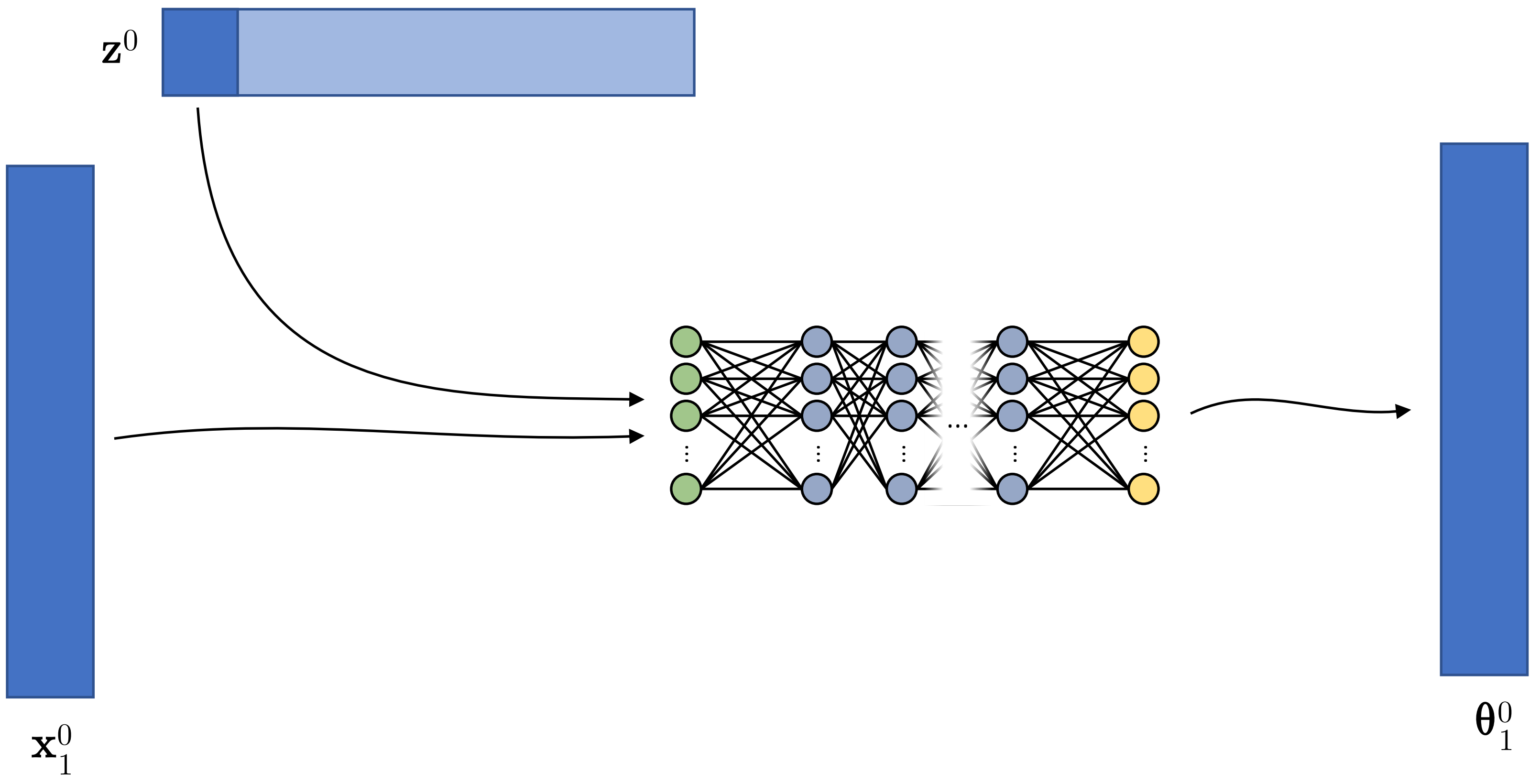


State: $\mathbf{x}_{\mathrm{t}}^0 = (\mathrm{z}_{\mathrm{t}}^0, \mathbf{k}_{\mathrm{t}}^0, \mathbf{b}_{\mathrm{t}}^0)$

Policy: $\boldsymbol{\theta}_{\mathrm{t}}^0 = (\widehat{\mathbf{a}}_{\mathrm{t}}^0, \widehat{\boldsymbol{\lambda}}_{\mathrm{t}}^0, \widehat{\mathbf{d}}_{\mathrm{t}}^0, \widehat{\boldsymbol{\mu}}_{\mathrm{t}}^0, \widehat{\mathrm{p}}_{\mathrm{t}}^0)$

3) Algorithm

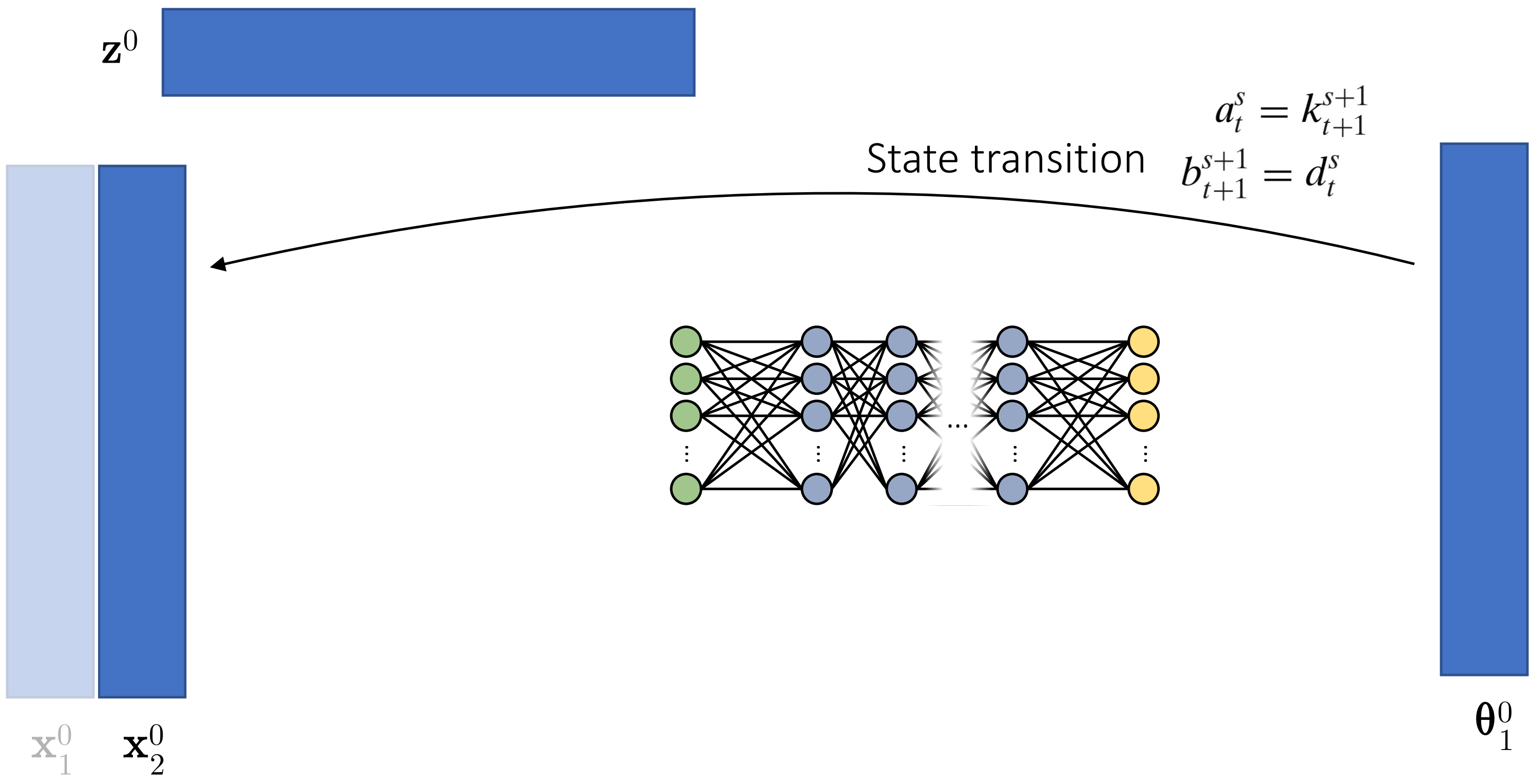
Final algorithm pipeline: simulation



State: $\mathbf{x}_t^0 = (z_t^0, \mathbf{k}_t^0, \mathbf{b}_t^0)$

Policy: $\boldsymbol{\theta}_t^0 = (\widehat{\mathbf{a}}_t^0, \widehat{\boldsymbol{\lambda}}_t^0, \widehat{\mathbf{d}}_t^0, \widehat{\boldsymbol{\mu}}_t^0, \widehat{\mathrm{p}}_t^0)$

3) Algorithm
Final algorithm pipeline: simulation

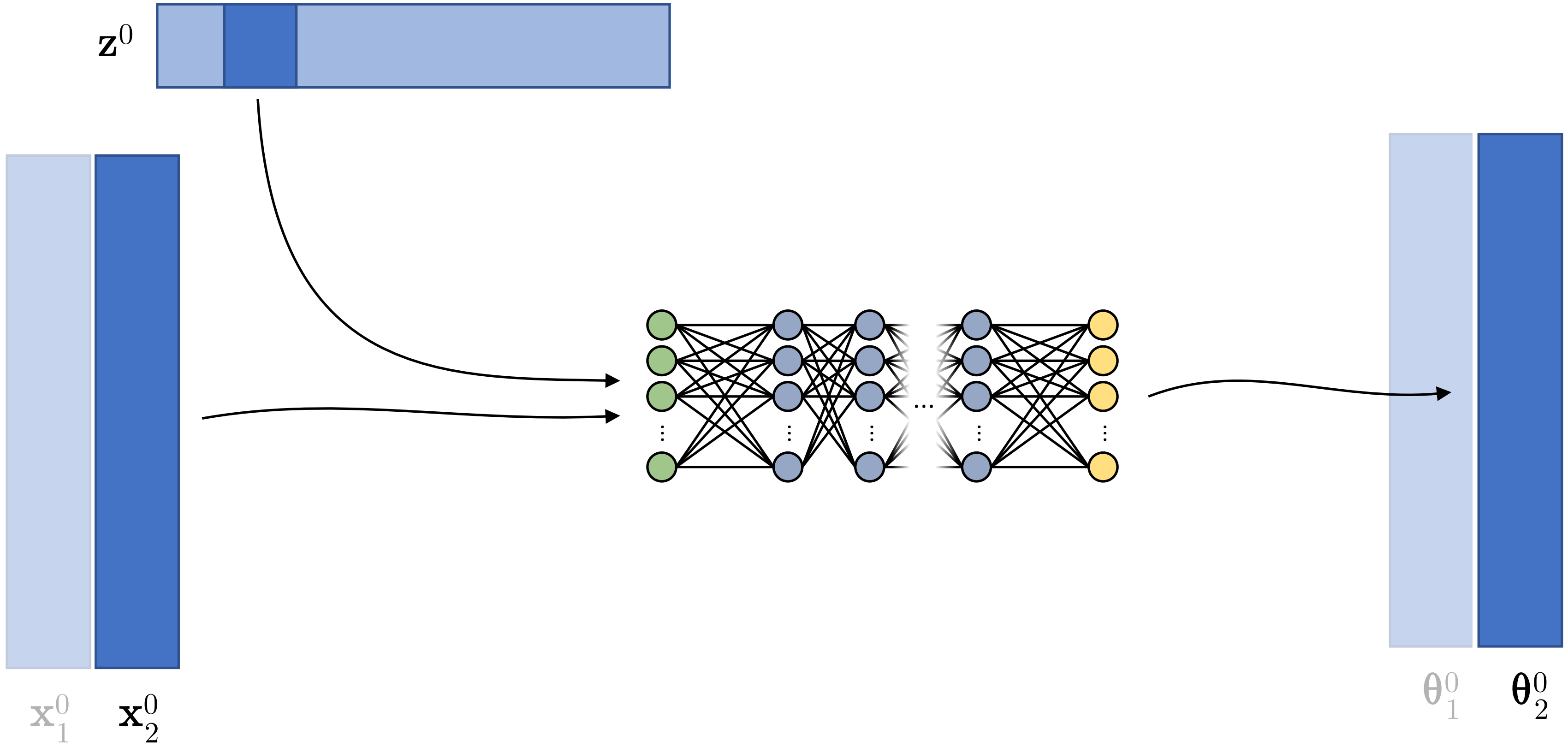


State: $\mathbf{x}_{\mathrm{t}}^0 = (\mathrm{z}_{\mathrm{t}}^0, \mathbf{k}_{\mathrm{t}}^0, \mathbf{b}_{\mathrm{t}}^0)$

Policy: $\boldsymbol{\theta}_{\mathrm{t}}^0 = (\widehat{\mathbf{a}}_{\mathrm{t}}^0, \widehat{\boldsymbol{\lambda}}_{\mathrm{t}}^0, \widehat{\mathbf{d}}_{\mathrm{t}}^0, \widehat{\boldsymbol{\mu}}_{\mathrm{t}}^0, \widehat{\mathrm{p}}_{\mathrm{t}}^0)$

3) Algorithm

Final algorithm pipeline: simulation

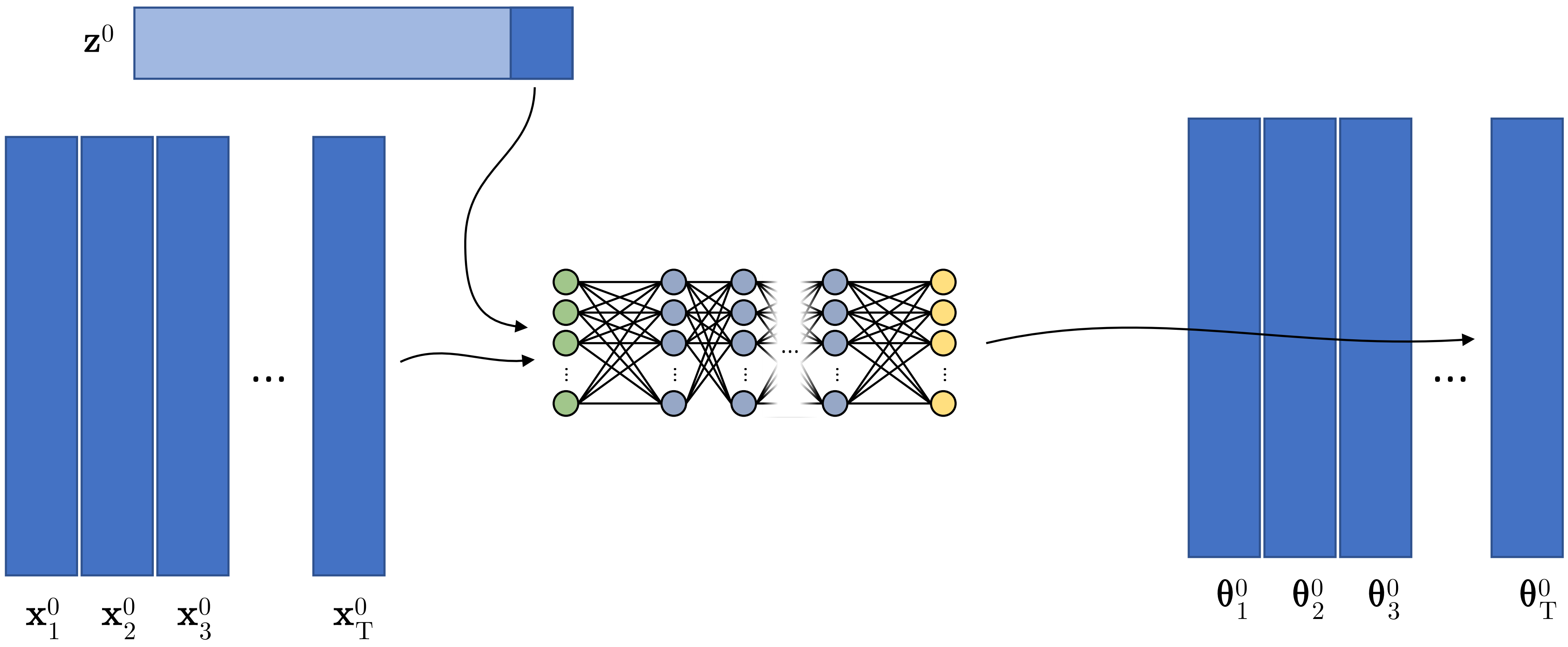


State: $\mathbf{x}_t^0 = (z_t^0, \mathbf{k}_t^0, \mathbf{b}_t^0)$

Policy: $\boldsymbol{\theta}_t^0 = (\widehat{\mathbf{a}}_t^0, \widehat{\boldsymbol{\lambda}}_t^0, \widehat{\mathbf{d}}_t^0, \widehat{\boldsymbol{\mu}}_t^0, \widehat{\mathrm{p}}_t^0)$

3) Algorithm

Final algorithm pipeline: simulation

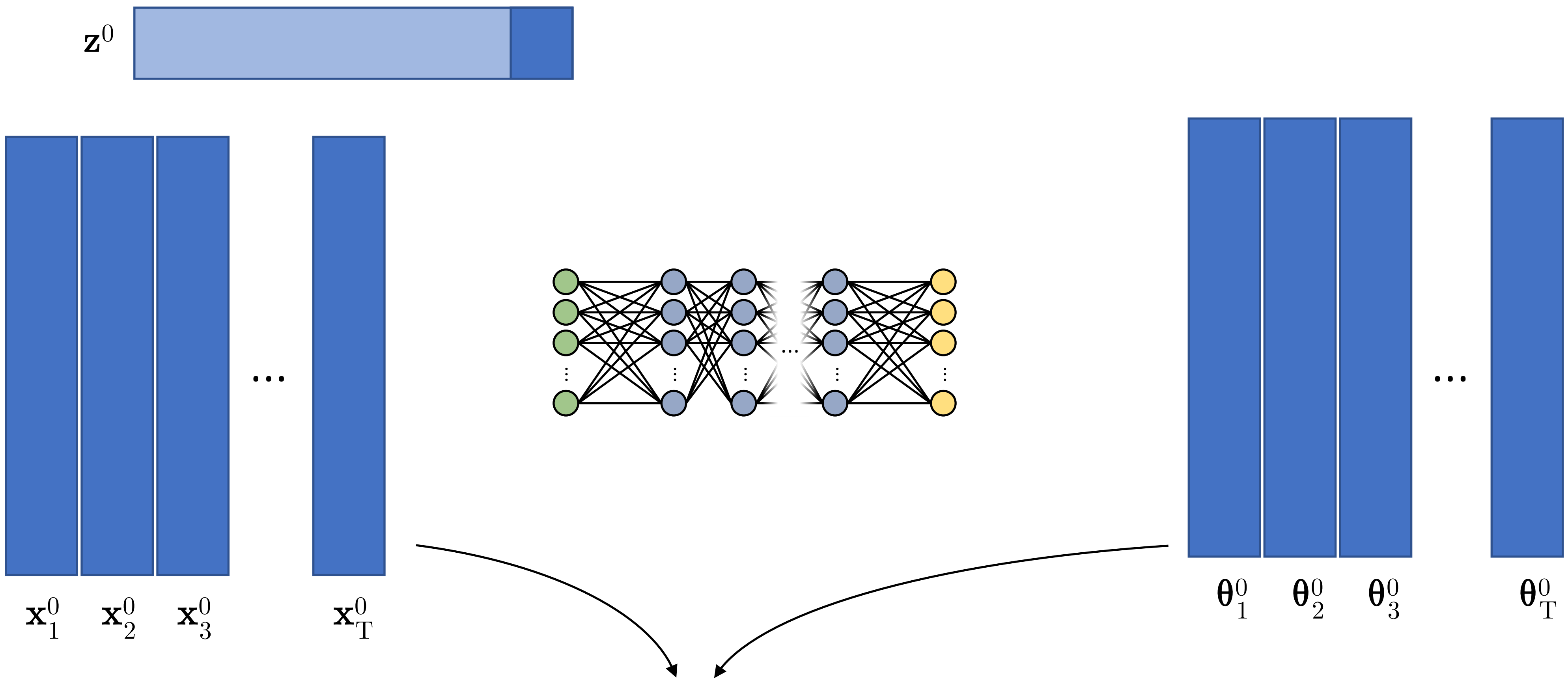


State: $\mathbf{x}_t^0 = (z_t^0, \mathbf{k}_t^0, \mathbf{b}_t^0)$

Policy: $\boldsymbol{\theta}_t^0 = (\widehat{\mathbf{a}}_t^0, \widehat{\boldsymbol{\lambda}}_t^0, \widehat{\mathbf{d}}_t^0, \widehat{\boldsymbol{\mu}}_t^0, \widehat{p}_t^0)$

3) Algorithm

Final algorithm pipeline: training



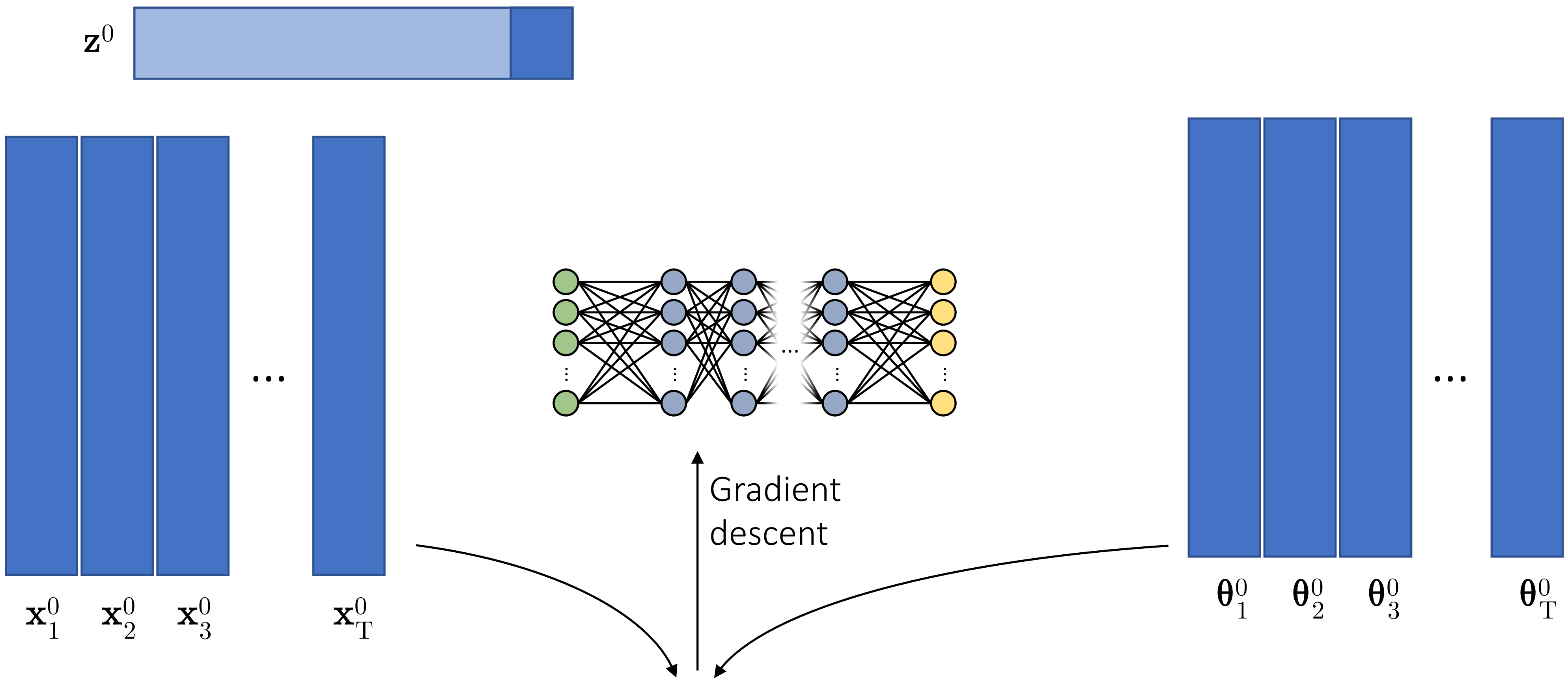
State: $\mathbf{x}_t^0 = (z_t^0, \mathbf{k}_t^0, \mathbf{b}_t^0)$

$\mathrm{Loss}(\{\mathbf{x}_t^0\}_{t=1}^T, \{\boldsymbol{\theta}_t^0\}_{t=1}^T)$

Policy: $\boldsymbol{\theta}_t^0 = (\widehat{\mathbf{a}}_t^0, \widehat{\boldsymbol{\lambda}}_t^0, \widehat{\mathbf{d}}_t^0, \widehat{\boldsymbol{\mu}}_t^0, \widehat{\mathrm{p}}_t^0)$

3) Algorithm

Final algorithm pipeline: training



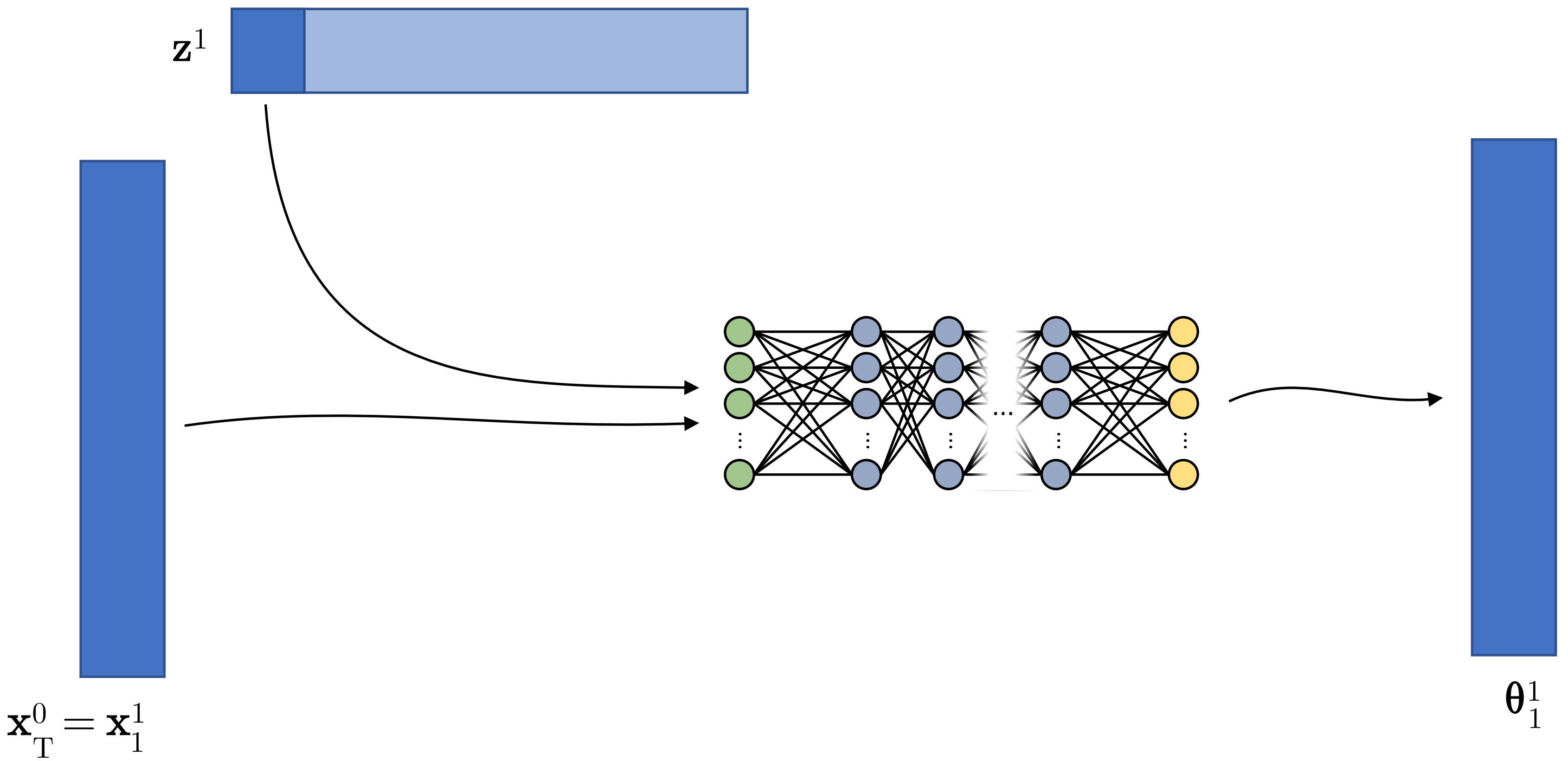
State: $\mathbf{x}_\mathrm{t}^0 = (\mathrm{z}_\mathrm{t}^0, \mathbf{k}_\mathrm{t}^0, \mathbf{b}_\mathrm{t}^0)$

$\mathrm{Loss}(\{\mathbf{x}_\mathrm{t}^0\}_{\mathrm{t}=1}^{\mathrm{T}}, \{\boldsymbol{\theta}_\mathrm{t}^0\}_{\mathrm{t}=1}^{\mathrm{T}})$

Policy: $\boldsymbol{\theta}_\mathrm{t}^0 = (\widehat{\mathbf{a}}_\mathrm{t}^0, \widehat{\boldsymbol{\lambda}}_\mathrm{t}^0, \widehat{\mathbf{d}}_\mathrm{t}^0, \widehat{\boldsymbol{\mu}}_\mathrm{t}^0, \widehat{\mathrm{p}}_\mathrm{t}^0)$

3) Algorithm

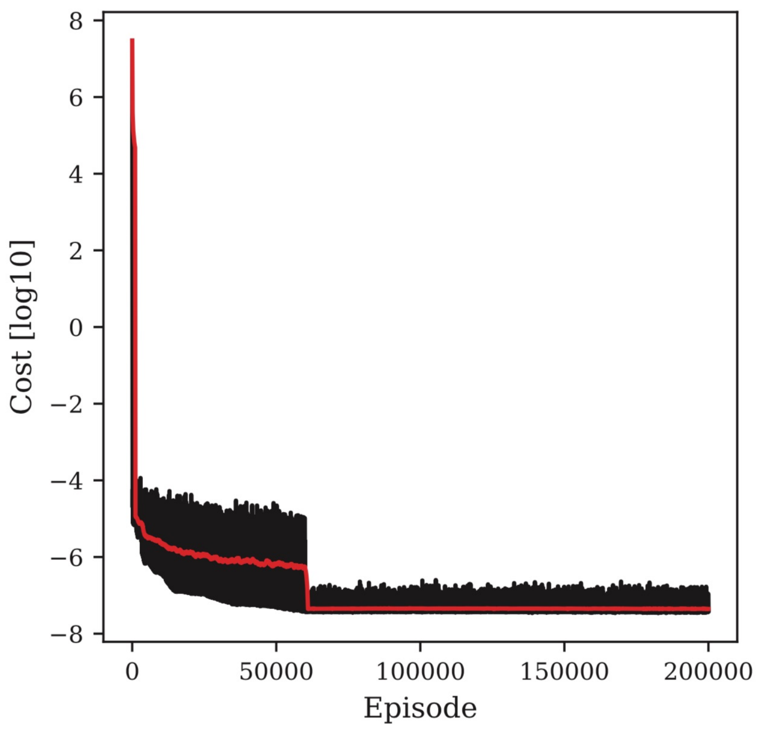
Final algorithm pipeline: simulation



State: $\mathbf{x}^1_t = (z^1_t, \mathbf{k}^1_t, \mathbf{b}^1_t)$

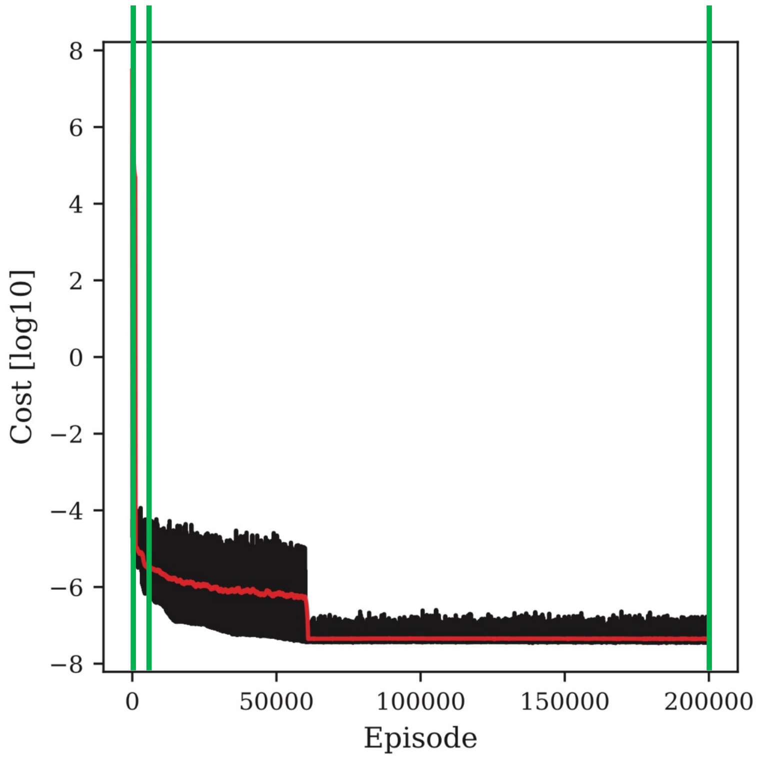
Policy: $\boldsymbol{\theta}^1_t = (\widehat{\mathbf{a}}^1_t, \widehat{\lambda}^1_t, \widehat{\mathbf{d}}^1_t, \widehat{\boldsymbol{\mu}}^1_t, \widehat{p}^1_t)$

4) Evaluation
Out-of-sample loss

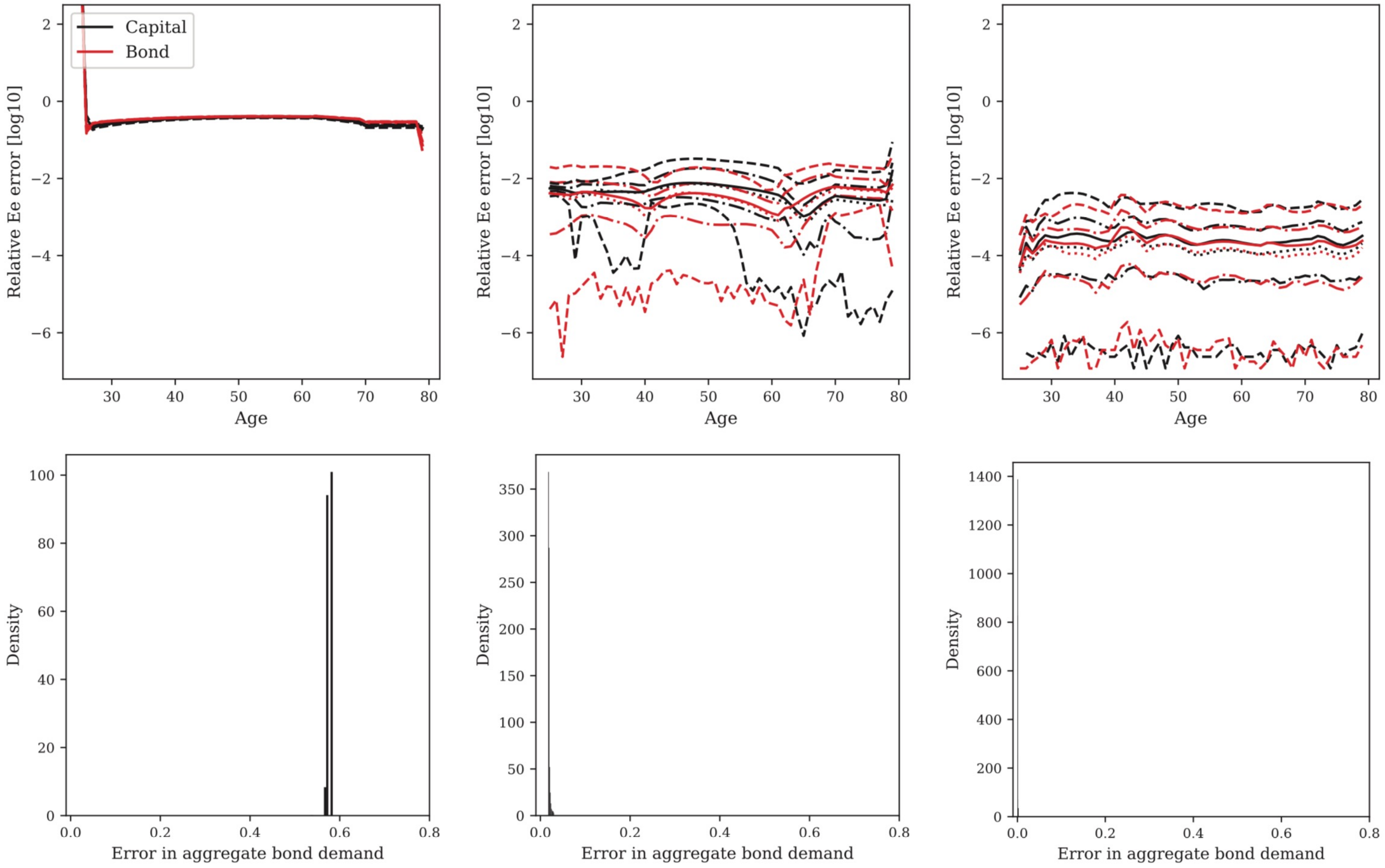


4) Evaluation

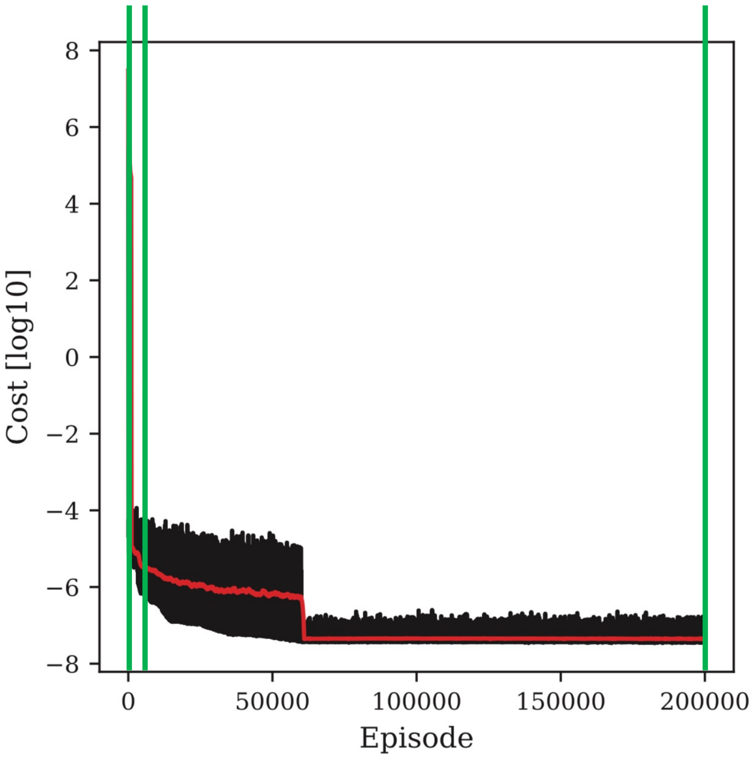
Euler and market clearing errors

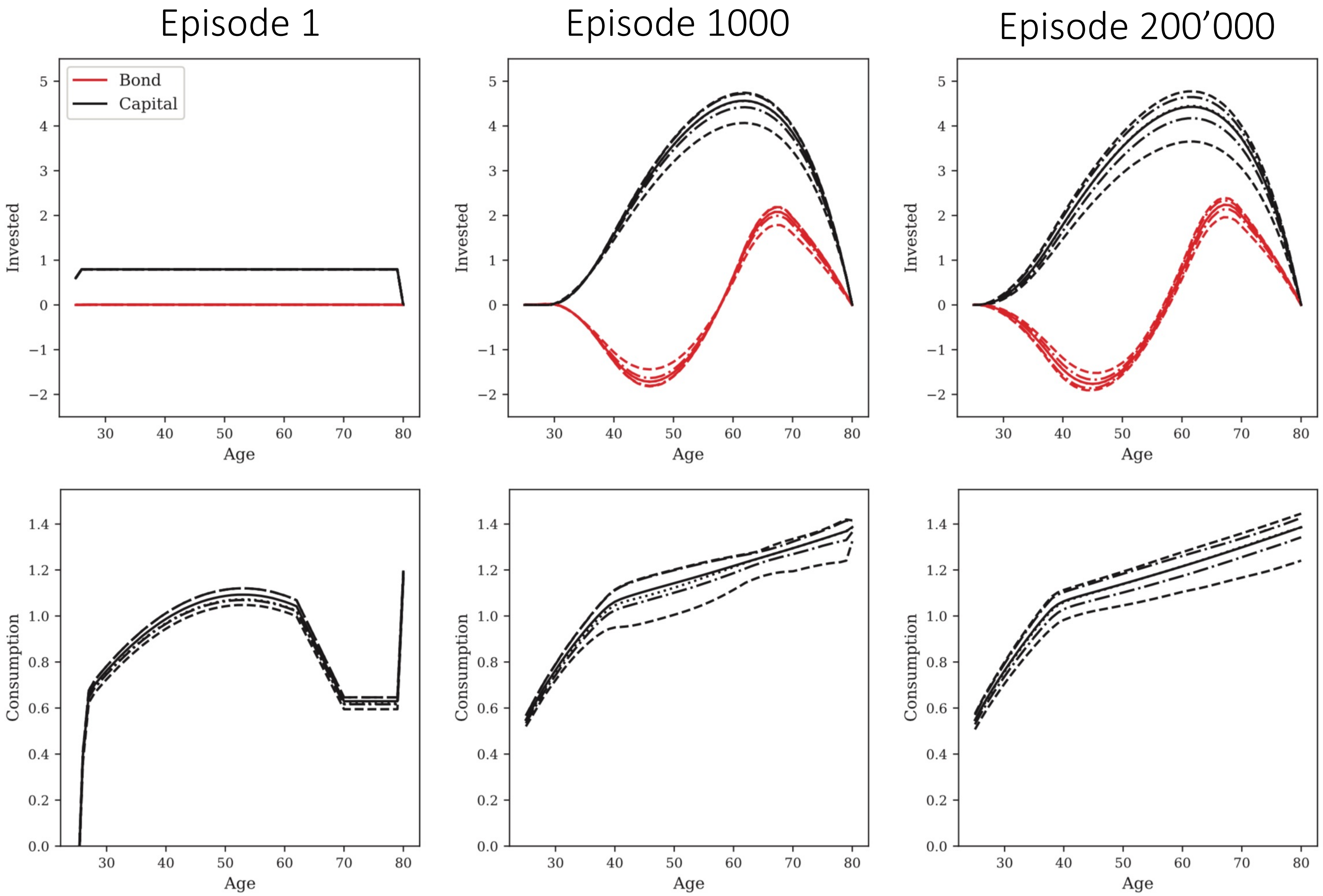


Episode 1 | Episode 1000 | Episode 200’000



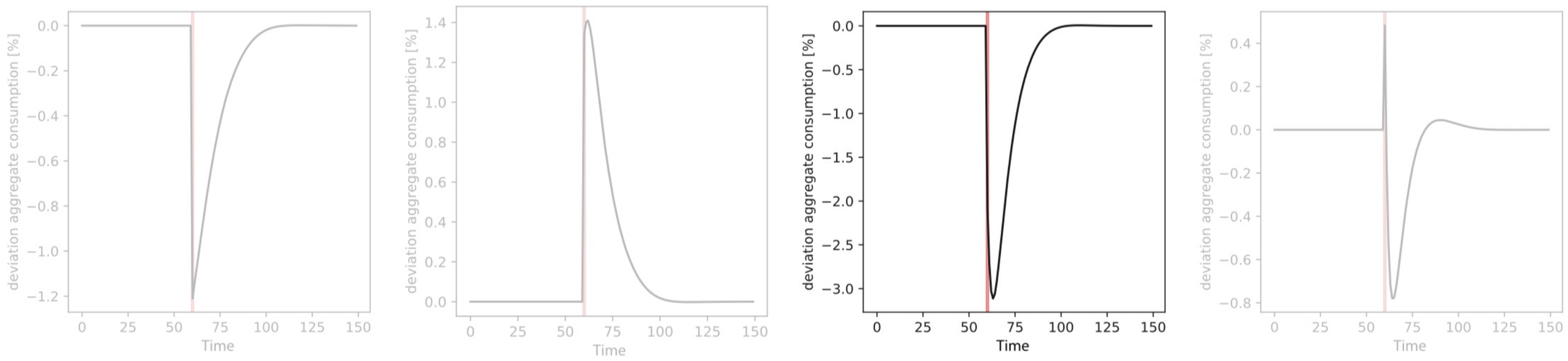
4) Evaluation
Policies at different training stages





5) Economic results

Impulse response of aggregate consumption



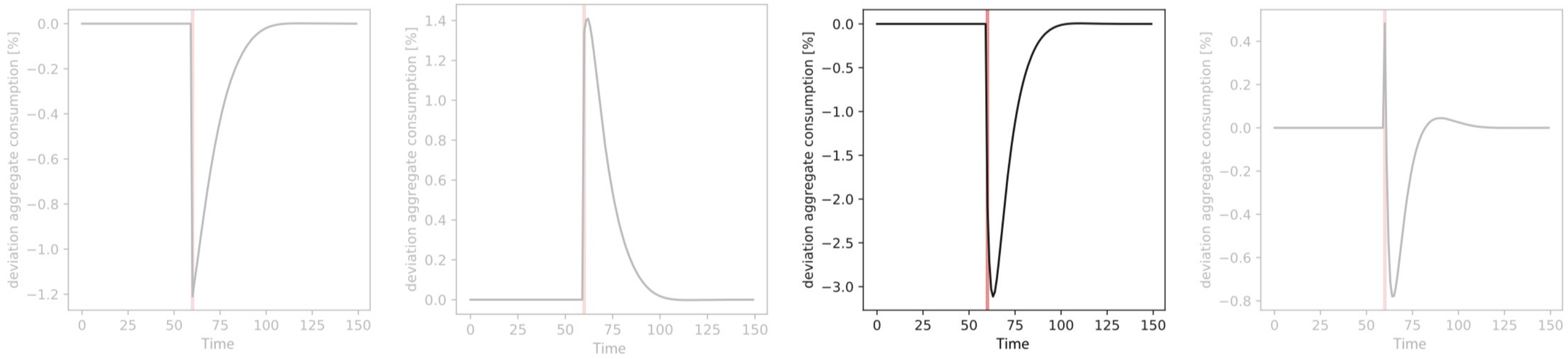
(a) Shock 1: $\eta = 0.978,\ \delta = 0.08$

(b) Shock 2: $\eta = 1.022,\ \delta = 0.08$

(c) Shock 3: $\eta = 0.978,\ \delta = 0.11$

(d) Shock 4: $\eta = 1.022,\ \delta = 0.11$

5) Economic results
Consumption response across age groups

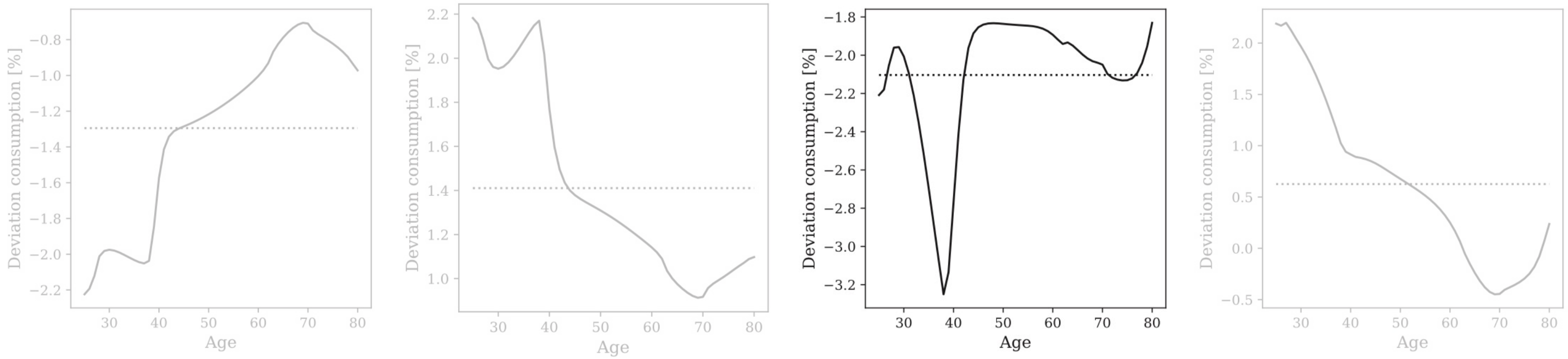


(a) Shock 1: $\eta = 0.978,\ \delta = 0.08$

(b) Shock 2: $\eta = 1.022,\ \delta = 0.08$

(c) Shock 3: $\eta = 0.978,\ \delta = 0.11$

(d) Shock 4: $\eta = 1.022,\ \delta = 0.11$



(a) Shock 1: $\eta = 0.978,\ \delta = 0.08$

(b) Shock 2: $\eta = 1.022,\ \delta = 0.08$

(c) Shock 3: $\eta = 0.978,\ \delta = 0.11$

(d) Shock 4: $\eta = 1.022,\ \delta = 0.11$

References

- Brumm, J., F. Kubler, and S. Scheidegger, “Computing Equilibria in Dynamic Stochastic Macro-Models with Heterogeneous Agents,” In *Advances in Economics and Econometrics: Volume 2: Eleventh World Congress*, Volume 59 (Cambridge, MA: Cambridge University Press, 2017), 185–230.
- Brumm, J., and S. Scheidegger, “Using Adaptive Sparse Grids to Solve High-Dimensional Dynamic Mod- els,” *Econometrica* 85 (2017), 1575–612.
- Krueger, D., and F. Kubler, “Computing Equilibrium in OLG Models with Stochastic Production,” *Journal of Economic Dynamics and Control* 28 (2004), 1411–36.
- Maliar, L., and S. Maliar, “Merging Simulation and Projection Approaches to Solve High-Dimensional Problems with an Application to a New Keynesian Model,” *Quantitative Economics* 6 (2015), 1–47.
- Muthuraman, K. and Zha, H. (2008). Simulation-based portfolio optimization for large portfolios with transaction costs. Mathematical Finance: An International Journal of Mathematics, Statistics and Financial Economics, 18(1):115–134.
- Scheidegger, S., and I. Bilionis, “Machine Learning for High-Dimensional Dynamic Stochastic Economies,” *Journal of Computational Science* 33 (2019), 68–82.
- Schober, P., Valentin, J., and Pflüger, D. (2022). Solving high-dimensional dynamic portfolio choice models with hierarchical b-splines on sparse grids. Computational Economics, 59(1):185–224.
- Winschel, V., and M. Krätzig, “Solving, Estimating, and Selecting Nonlinear Dynamic Models without the Curse of Dimensionality,” *Econometrica* 78 (2010), 803–21.

Appendix

Benchmark model optimality conditions

- w.r.t. capital holdings:

$$(1+\zeta\,\Delta_t^s)u'(c_t^s) = \beta \mathrm{E}_t\big[u'(c_{t+1}^{s+1})r_{t+1}\big(1+\zeta\,\Delta_{t+1}^{s+1}\big)\big] + \lambda_t^s + \mu_t^s,$$

$$\lambda_t^s \cdot (a_t^s - \underline{a}) = 0,$$

$$a_t^s - \underline{a} \geq 0,$$

$$\lambda_t^s \geq 0,$$

- w.r.t. bond holdings:

$$p_t u'(c_t^s) = \beta \mathrm{E}_t\big[u'(c_{t+1}^{s+1})\big] + \kappa\mu_t^s,$$

$$\mu_t^s \cdot (a_t^s + \kappa d_t^s) = 0,$$

$$a_t^s + \kappa d_t^s \geq 0,$$

$$\mu_t^s \geq 0.$$

Benchmark model optimality conditions
Undiscussed KKT conditions

- w.r.t. capital holdings:

$$(1+\zeta\Delta_t^s)u'(c_t^s) = \beta \mathrm{E}_t\big[u'(c_{t+1}^{s+1})r_{t+1}\big(1+\zeta\Delta_{t+1}^{s+1}\big)\big] + \lambda_t^s + \mu_t^s,$$

$$\lambda_t^s \cdot (a_t^s - \underline{a}) = 0,$$

$$\left.\begin{aligned} a_t^s - \underline{a} &\geq 0, \\ \lambda_t^s &\geq 0, \end{aligned}\right\}$$

Encoded directly in neural network output layer (softplus activation

- w.r.t. bond holdings:

$$p_t u'(c_t^s) = \beta \mathrm{E}_t\big[u'(c_{t+1}^{s+1})\big] + \kappa\mu_t^s,$$

$$\mu_t^s \cdot (a_t^s + \kappa d_t^s) = 0,$$

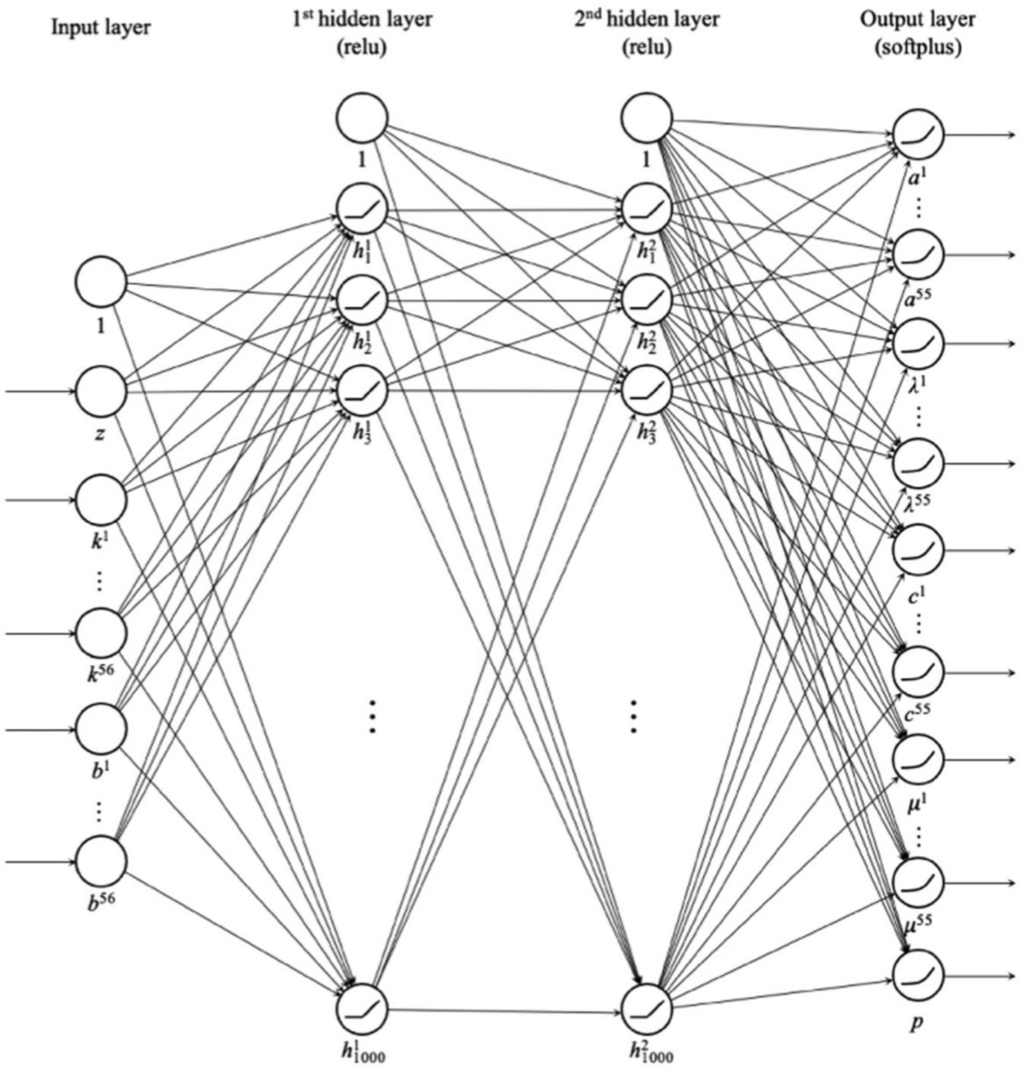
$$\left.\begin{aligned} a_t^s + \kappa d_t^s &\geq 0, \\ \mu_t^s &\geq 0. \end{aligned}\right\}$$

Encoded directly in neural network output layer (softplus activation

Softplus σ(x) = ln(1 + e_x)

Neural network architecture

$$\mathbf{x}^* = \begin{bmatrix} \mathrm{z} \\ \mathbf{k} \\ \mathrm{b} \end{bmatrix}$$



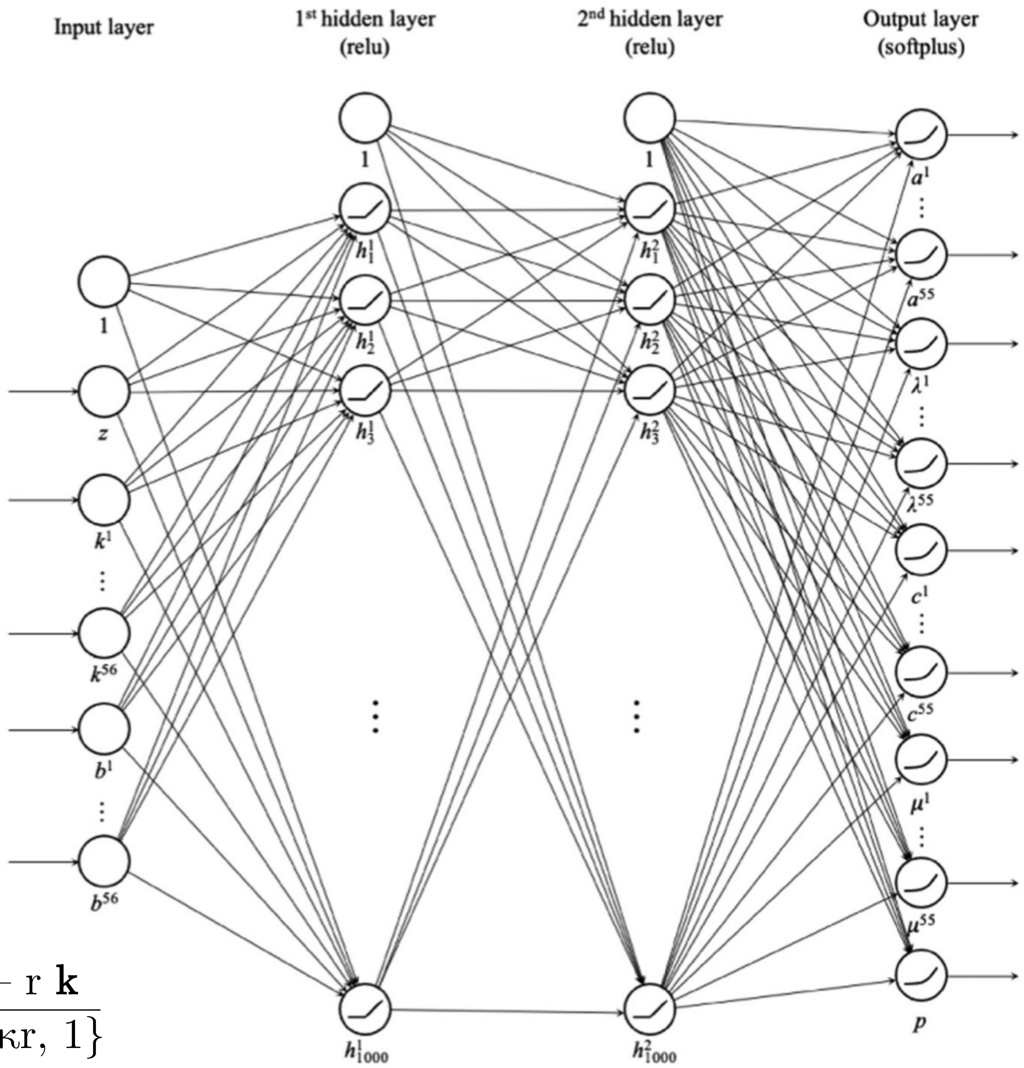
$$N_\rho(\mathbf{x}) = \begin{bmatrix} \boldsymbol{\theta}_{\mathrm{a}}(\mathbf{x}) \\ \boldsymbol{\theta}_{\lambda}(\mathbf{x}) \\ \boldsymbol{\theta}_{\mathrm{col}}(\mathbf{x}) \\ \boldsymbol{\theta}_{\mu}(\mathbf{x}) \\ \theta_{\mathrm{p}}(\mathbf{x}) \end{bmatrix}$$

where $\boldsymbol{\theta}_{\mathrm{d}}(\mathbf{x}) = (\boldsymbol{\theta}_{\mathrm{col}}(\mathbf{x}) - \boldsymbol{\theta}_{\mathrm{a}}(\mathbf{x}))/\kappa$

Passing redundant information to the DEQN

$$\mathbf{x}^* = \begin{bmatrix} \mathrm{z} \\ \eta(\mathrm{z}) \\ \delta(\mathrm{z}) \\ \mathrm{K} \\ \mathrm{L} \\ \mathrm{r} \\ \mathrm{w} \\ \mathrm{Y} \\ \mathbf{k} \\ \mathbf{i}_{\mathrm{fin}} \\ \mathbf{i}_{\mathrm{labor}} \\ \mathbf{i}_{\mathrm{total}} \\ \Pi[\mathrm{z}] \end{bmatrix}$$

where $\mathbf{b} = \dfrac{\mathbf{i}_{\mathrm{fin}} - \mathrm{r}\,\mathbf{k}}{\min\{\kappa \mathrm{r},\, 1\}}$



$$N_\rho(\mathbf{x}) = \begin{bmatrix} \theta_{\mathrm{a}}(\mathbf{x}) \\ \theta_{\lambda}(\mathbf{x}) \\ \theta_{\mathrm{col}}(\mathbf{x}) \\ \theta_{\mu}(\mathbf{x}) \\ \theta_{\mathrm{p}}(\mathbf{x}) \end{bmatrix}$$

where $\theta_{\mathrm{d}}(\mathbf{x}) = (\theta_{\mathrm{col}}(\mathbf{x}) - \theta_{\mathrm{a}}(\mathbf{x}))/\kappa$

Loss penalty to deal with infeasible predictions at the beginning of training

At the beginning of training, neural network parameters are randomly initialized and hence the neural network predicts random policies.

If predicted policies are infeasible:

- Set a punishment parameter: $\varepsilon_{\text{punish}} = 10^{-5}$.
- If predicted policy is infeasible (*e.g.*, consumption is negative), replace it with $\varepsilon_{\text{punish}}$:
$$\text{pol}_{\text{adj}}(\mathbf{x}) = \max(\text{pol}(\mathbf{x}),\, \varepsilon_{\text{punish}}).$$
- Add punishment to loss function: $(1/\varepsilon_{\text{punish}} \max(\text{-pol}(\mathbf{x}),\, 0))^2$.

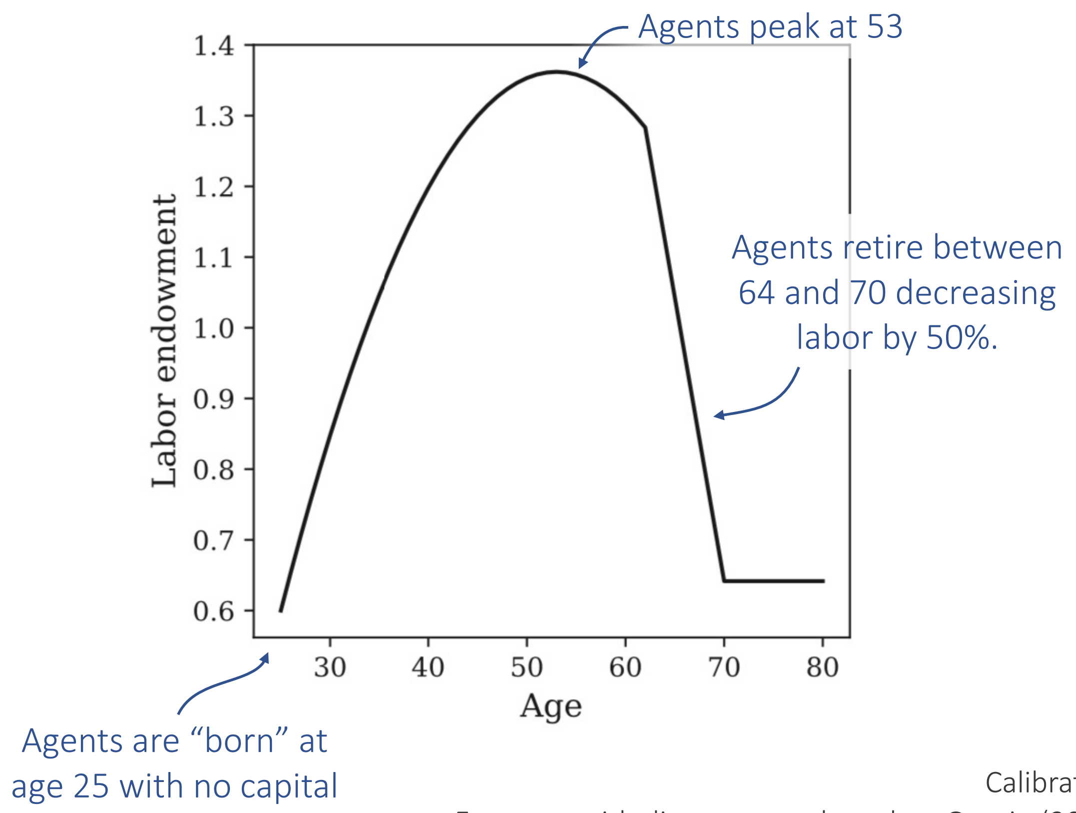
Benchmark model parameterization

Table 1.1: Parameterization of the benchmark model

Discount factor β	0.95
Relative risk aversion γ	2
Capital share α	0.3
TFP η	$\{0.978, 1.022\}$
Borrowing constraint $\underline{a}$	0.0
Adjustment cost capital ζ	0.5
Collateral requirement bond κ	1.1236
Persistence TFP:	
$P(\eta_{t+1} = 1.022 \mid \eta_t = 1.022)$	0.905
$P(\eta_{t+1} = 0.978 \mid \eta_t = 0.978)$	0.905
Persistence depreciation:	
$P(\delta_{t+1} = 0.08 \mid \delta_t = 0.08)$	0.972
$P(\delta_{t+1} = 0.11 \mid \delta_t = 0.11)$	0.700

FIGURE A.2

LABOR ENDOWMENT OVER THE LIFE CYCLE IN THE BENCHMARK MODEL



Calibration:
Economy with disaster state based on Gourio (2012),
Labor based on Guvenen et al. (2021) and Brumm et al. (2017),
Parameters based on Krueger and Kübler (2004)

DEQN algorithm

Algorithm 1. Algorithm for training deep equilibrium nets

Data:
T (length of an episode), N^{epochs} (number of epochs on each episode), N^{iter} (maximum number of iterations),
τ^{mean}, τ^{max} (desired threshold for mean and max error, respectively),
$\epsilon^{\text{mean}} = \infty$, $\epsilon^{\text{max}} = \infty$ (starting value for current mean and max error, respectively),
$\boldsymbol{\rho}^0$ (initial parameters of the neural network), $\mathbf{x}_1^0$ (initial state to start simulations from), $i = 0$ (set counter),
α^{learn} (learning rate)
Result:
success (boolean if thresholds were reached)
$\boldsymbol{\rho}^{\text{final}}$ (final neural network parameters)
while $((i < N^{iter}) \wedge ((\epsilon^{mean} \geq \tau^{mean}) \vee (\epsilon^{max} \geq \tau^{max})))$ **do**
  $\mathcal{D}_{\text{train}}^i \leftarrow \{\mathbf{x}_1^i, \mathbf{x}_2^i, \ldots, \mathbf{x}_T^i\}$ (generate new training data)
  $\mathbf{x}_0^{i+1} \leftarrow \mathbf{x}_T^i$ (set new starting point)
  $\epsilon_{\max} \leftarrow \max\left\{\max_{\mathbf{x} \in \mathcal{D}_{\text{train}}^i} |e_{\mathbf{x}}^{\cdot\cdot}(\boldsymbol{\rho})|\right\}$, $\epsilon_{\text{mean}} \leftarrow \max\left\{\frac{1}{T}\sum_{\mathbf{x} \in \mathcal{D}_{\text{train}}^i} |e_{\mathbf{x}}^{\cdot\cdot}(\boldsymbol{\rho})|\right\}$ (update errors)
  for $j \in [1, ..., N^{epochs}]$ **do**
    (learn N^{epochs} on data)
    for $k \in [1, ..., \mathit{length}(\boldsymbol{\rho})]$ **do**

$$\rho_k^{i+1} = \rho_k^i - \alpha^{\text{learn}} \frac{\partial \ell_{\mathcal{D}_{\text{train}}^i}(\boldsymbol{\rho}^i)}{\partial \rho_k^i} \tag{50}$$

      (do a gradient descent step to update the network parameters)
    end
  end
  $i \leftarrow i + 1$ (update episode counter)
end
if $i = N^{iter}$ **then return** (success $\leftarrow \mathit{False}$, $\boldsymbol{\rho}^{\text{final}} \leftarrow \boldsymbol{\rho}^i$) ;
else return (success $\leftarrow \mathit{True}$, $\boldsymbol{\rho}^{\text{final}} \leftarrow \boldsymbol{\rho}^i$) ;

DEQN training hyperparameters

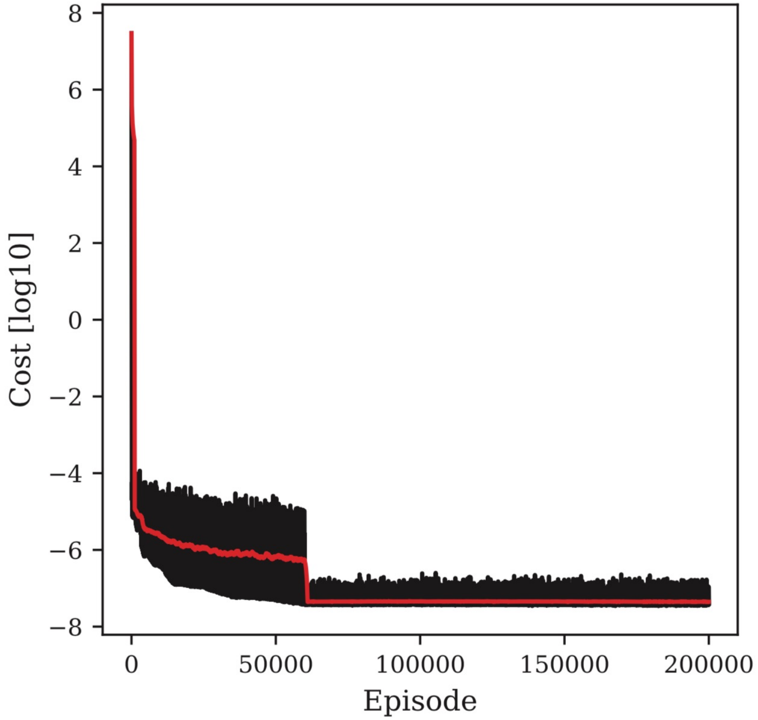


TABLE 2
HYPERPARAMETERS CHOSEN TO TRAIN THE DEEP EQUILIBRIUM NET FOR THE BENCHMARK MODEL

Episodes	Learning Rate α^{learn}	Periods per Episode T	Epochs per Episode N^{epochs}	Mini-Batch Size m	Nodes Hidden Layers	Activations Hidden Layers
					1,000	relu
1–60,000	1×10^{-5}	10,000	1	64	1,000	relu
					1,000	relu
60,000–200,000	1×10^{-6}	10,000	1	1,000	1,000	relu

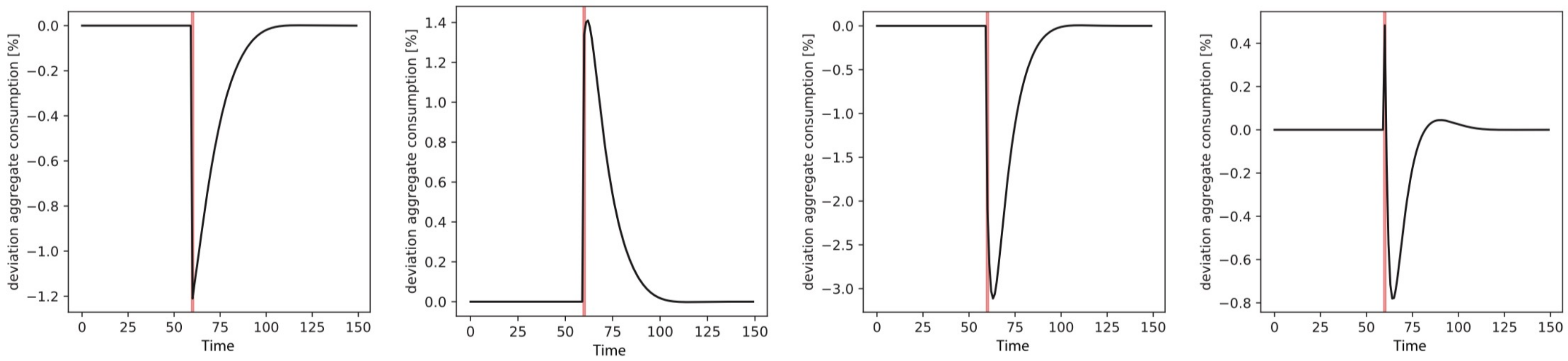
Benchmark model error statistics

TABLE 3
STATISTICS OF ERRORS IN THE EQUILIBRIUM CONDITIONS

	Mean	Max	0.1	10	50	90	99.9
Rel Ee capital [%]	0.024	0.528	0.000	0.002	0.013	0.060	0.324
Rel Ee bond [%]	0.021	0.608	0.000	0.002	0.012	0.051	0.255
KKT capital [$\times 10^{-2}$]	0.000	0.127	0.000	0.000	0.000	0.000	0.000
KKT bond [$\times 10^{-2}$]	0.005	0.104	0.000	0.000	0.000	0.000	0.021
Market clearing [$\times 10^{-2}$]	0.053	0.333	0.000	0.017	0.047	0.089	0.252
Market clearing (normalized) [$\times 10^{-2}$]	0.000	0.002	0.000	0.000	0.000	0.000	0.001

NOTE: The first two rows show the relative Euler equation errors (Rel Ee in %) on 10,000 simulated periods. Rows 3 and 4 show the errors in the KKT conditions ($\times 10^{-2}$). Rows 5 and 6 show the errors in the market clearing conditions ($\times 10^{-2}$). In row 6, the error in the market clearing condition for the bond is divided by aggregate production. The columns show the mean and the max errors as well as the 0.1st, 10th, 50th, 90th, and 99.9th percentile.

Consumption response across age groups

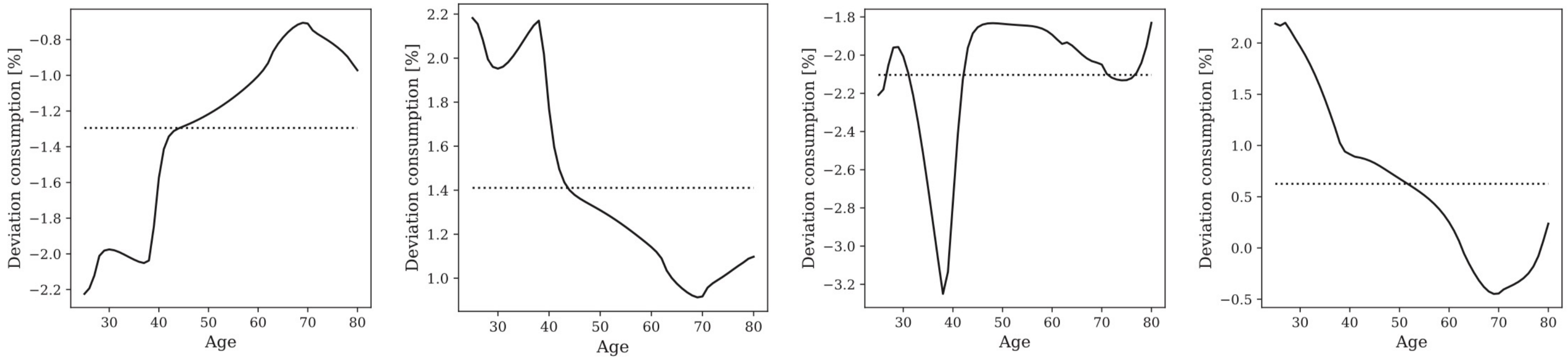


(a) Shock 1: $\eta = 0.978,\ \delta = 0.08$

(b) Shock 2: $\eta = 1.022,\ \delta = 0.08$

(c) Shock 3: $\eta = 0.978,\ \delta = 0.11$

(d) Shock 4: $\eta = 1.022,\ \delta = 0.11$



(a) Shock 1: $\eta = 0.978,\ \delta = 0.08$

(b) Shock 2: $\eta = 1.022,\ \delta = 0.08$

(c) Shock 3: $\eta = 0.978,\ \delta = 0.11$

(d) Shock 4: $\eta = 1.022,\ \delta = 0.11$

Model with two types of agents, OLG, and idiosyncratic health shocks

Health-shock model

- Two sources of idiosyncratic risk over the life-cycle:
  - Two types of agents (realized at “birth”) with different labor endowment trajectories over the life-cycle.
  - Health shock shortly before retiring.
- Sick agents have lower utility from consumption, causing them to rely on their savings, and hence enter retirement with reduced savings.
- We study the welfare gains from introducing health insurance:
  - An asset that pays out contingent on the realization of the health-shock.

Health-shock model
Utility function

Agents maximize expected lifetime utility, where per-period utility function is:

Agent type

Age

Health-shock index:
$\mathrm{h}=1$: no health shock;
$\mathrm{h}=2$: no health shock.

$$u\left(\frac{c_t^{(y,s,h)}}{g^{(y,s,h)}}\right)$$

Health-shock factor:
$\mathrm{g} \geq 1$ if agent $\mathrm{s} = \mathrm{s}^{\mathrm{h}}$
experiences health-shock ($\mathrm{h}=2$);
and $\mathrm{g} = 1$ otherwise.

Health-shock model
Optimality conditions

- w.r.t. capital holdings:

$$\frac{1+\zeta\Delta^{(y,s,h)}(\mathbf{x})}{g^{(y,s,h)}}u'\left(\frac{c^{(y,s,h)}(\mathbf{x})}{g^{(y,s,h)}}\right)=\beta\mathrm{E}_{z,h}\left[\frac{r(\mathbf{x}_+)\big(1+\zeta\Delta^{(y,s+1,h_+)}(\mathbf{x}_+)\big)}{g^{(y,s+1,h_+)}}u'\left(\frac{c^{(y,s+1,h_+)}(\mathbf{x}_+)}{g^{(y,s+1,h_+)}}\right)\right]+\lambda^{(y,s,h)}(\mathbf{x})+\mu^{(y,s,h)}(\mathbf{x}),$$

$$0=a^{(y,s,h)}(\mathbf{x})\cdot\lambda^{(y,s,h)}(\mathbf{x}),$$

$$0\leq a^{(y,s,h)}(\mathbf{x}),$$

$$0\leq\lambda^{(y,s,h)}(\mathbf{x}),$$

- w.r.t. bond holdings:

$$\frac{p(\mathbf{x})}{g^{(y,s,h)}}u'\left(\frac{c^{(y,s,h)}(\mathbf{x})}{g^{(y,s,h)}}\right)=\beta\mathrm{E}_{z,h}\left[\frac{1}{g^{(y,s+1,h_+)}}u'\left(\frac{c^{(y,s+1,h_+)}(\mathbf{x}_+)}{g^{(y,s+1,h_+)}}\right)\right]+\kappa\mu^{(y,s,h)}(\mathbf{x}),$$

$$0=\mu^{(y,s,h)}(\mathbf{x})\Big(a^{(y,s,h)}(\mathbf{x})+\kappa d^{(y,s,h)}(\mathbf{x})\Big),$$

$$0\leq a^{(y,s,h)}(\mathbf{x})+\kappa d^{(y,s,h)}(\mathbf{x}),$$

$$0\leq\mu^{(y,s,h)}(\mathbf{x}),$$

- Market clearing

Fraction of agents who receive health shock

$$K=\sum_{y=1,2}\left(\sum_{s=1}^{s^h-1}k^{(y,s,1)}(\mathbf{x})+\sum_{s=s^h}^{N}(1-\pi^y)k^{(y,s,1)}(\mathbf{x})+\pi^y k^{(y,s,2)}(\mathbf{x})\right),$$

$$L=\sum_{y=1,2}\left(\sum_{s=1}^{s^h-1}l^{(y,s,1)}(\mathbf{x})+\sum_{s=s^h}^{N}(1-\pi^y)l^{(y,s,1)}(\mathbf{x})+\pi^y l^{(y,s,2)}(\mathbf{x})\right).$$

$$0=\sum_{y=1,2}\left(\sum_{s=1}^{s^h-1}d^{(y,s,1)}(\mathbf{x})+\sum_{s=s^h}^{N-1}(1-\pi^y)d^{(y,s,1)}(\mathbf{x})+\pi^y d^{(y,s,2)}(\mathbf{x})\right)$$

Health-shock model

Insurance

Agents can purchase an asset before the health-shock age:

- Asset promises a payout of 1 if health shock realizes.
- Insurance price is pinned down by bond price:

$$p^y(\mathbf{x}) = \pi^y p(\mathbf{x})$$

Selling π^y units of insurance is same as selling 1 unit of bond.

- Market clearing:

Units of insurance purchased.

$$0 = \sum_{y=1,2} \left(\pi^y i^y(\mathbf{x}) + \sum_{s=1}^{s^h-1} d^{(y,s,1)}(\mathbf{x}) + \sum_{s=s^h}^{N-1} (1-\pi^y) d^{(y,s,1)}(\mathbf{x}) + \pi^y d^{(y,s,2)}(\mathbf{x}) \right)$$

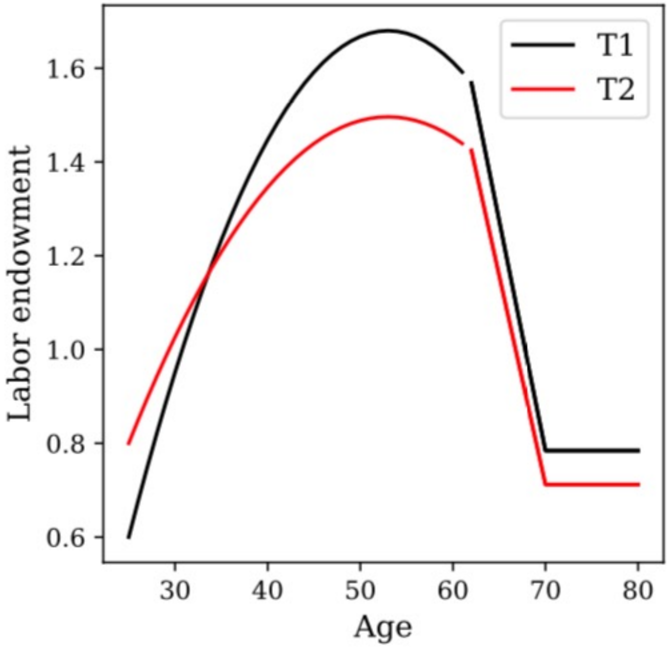
- First-order condition:

$$p^y(\mathbf{x}) u'\left(c^{(y,s^h-1,1)}(\mathbf{x})\right) = \beta \mathrm{E}_z \left[\pi^y \frac{1}{g^{(y,s^h,2)}} u'\left(\frac{c^{(y,s^h,2)}(\mathbf{x}_+)}{g^{(y,s^h,2)}} \right) \right]$$

Model parameterization

Discount factor β	0.95
Relative risk aversion γ	2
Capital share α	0.3
TFP η	$\{0.978, 1.022\}$
Depreciation δ	$\{0.08, 0.11\}$
Borrowing constraint $\underline{a}$	0.0
Adjustment cost capital ζ	0.5
Collateral requirement bond κ	1.1236
Timing of health shock s^h	38
Probability of health shock:	
π^1	0.1
π^2	0.3
Magnitude of health shock:	
$g^{(1,s^h,2)}$	2
$g^{(1,s^h,2)}$	3
Persistence TFP:	
$P(\eta_{t+1} = 1.022 \mid \eta_t = 1.022)$	0.905
$P(\eta_{t+1} = 0.978 \mid \eta_t = 0.978)$	0.905
Persistence depreciation:	
$P(\delta_{t+1} = 0.08 \mid \delta_t = 0.08)$	0.972
$P(\delta_{t+1} = 0.11 \mid \delta_t = 0.11)$	0.700

Labor endowment over the life-cycle for both types



Neural network parameterization

Table 1.D.2: Deep equilibrium net training hyper-parameters for the model with two types and health-shocks

Episodes	Learning rate α^{learn}	Periods per episode T	Epochs per episode N^{epochs}	Mini-batch size m	Nodes hidden layers	Activations hidden layers
1 – 20,000	1×10^{-5}	10,000	1	64	1,000 1,000	relu relu
20,001 – 100,000	1×10^{-6}	10,000	1	1,000	1,000 1,000	relu relu

Episodes	Learning rate α^{learn}	Periods per episode T	Epochs per episode N^{epochs}	Mini-batch size m	Nodes hidden layers	Activations hidden layers
1 – 18,500	1×10^{-5}	10,000	1	64	1,000 1,000	relu relu
18,501 – 100,000	1×10^{-6}	10,000	1	1,000	1,000 1,000	relu relu

NOTE: The top table shows the case without insurance, the bottom table shows the case with insurance.

Solution quality without insurance

Table 1.D.3: Errors in the equilibrium conditions for the model without health insurance

	mean	max	0.1	10	50	90	99.9
T1, Rel Ee capital [%]	0.025	0.533	0.000	0.002	0.013	0.060	0.310
T1, Rel Ee bond [%]	0.021	0.554	0.000	0.002	0.011	0.050	0.257
T1, KKT capital [$\times 10^{-2}$]	0.001	0.132	0.000	0.000	0.000	0.001	0.039
T1, KKT bond [$\times 10^{-2}$]	0.003	0.150	0.000	0.000	0.000	0.011	0.052
T2, Rel Ee capital [%]	0.028	0.844	0.000	0.002	0.014	0.068	0.345
T2, Rel Ee bond [%]	0.024	0.881	0.000	0.002	0.012	0.057	0.321
T2, KKT capital [$\times 10^{-2}$]	0.000	0.140	0.000	0.000	0.000	0.001	0.013
T2, KKT bond [$\times 10^{-2}$]	0.004	0.196	0.000	0.000	0.000	0.013	0.072
Market clearing [$\times 10^{-2}$]	0.193	1.679	0.001	0.056	0.169	0.362	1.003
Market clearing (normalized) [$\times 10^{-2}$]	0.000	0.004	0.000	0.000	0.000	0.001	0.002

NOTES: The first four rows show the equilibrium conditions for type 1 agents, rows 5 to 8 show the conditions for type 2 agents. The first two rows show the relative Euler equation errors (Rel Ee in %) on 10,000 simulated periods. The next two rows show the errors in the KKT conditions ($\times 10^{-2}$). Rows 9 and 10 show the errors in the market clearing conditions ($\times 10^{-2}$). In row 10, the error in the market clearing condition for the bond is divided by aggregate production. The columns show the mean and the max errors as well as the 0.1st, 10th, 50th, 90th, and 99.9th percentile.

Solution quality with insurance

Table 1.D.4: Errors in the equilibrium conditions for the model with health insurance

	mean	max	0.1	10	50	90	99.9
T1, Rel Ee capital [%]	0.027	0.522	0.000	0.002	0.014	0.066	0.319
T1, Rel Ee bond [%]	0.022	0.467	0.000	0.002	0.012	0.053	0.253
T1, Rel Ee insurance [%]	0.016	0.196	0.000	0.003	0.012	0.032	0.113
T1, KKT capital [$\times 10^{-2}$]	0.001	0.049	0.000	0.000	0.000	0.001	0.038
T1, KKT bond [$\times 10^{-2}$]	0.003	0.135	0.000	0.000	0.000	0.011	0.058
T2, Rel Ee capital [%]	0.029	0.820	0.000	0.003	0.016	0.070	0.376
T2, Rel Ee bond [%]	0.025	0.847	0.000	0.002	0.014	0.061	0.330
T2, Rel Ee insurance [%]	0.024	0.030	0.000	0.004	0.017	0.055	0.162
T2, KKT capital [$\times 10^{-2}$]	0.000	0.178	0.000	0.000	0.000	0.001	0.003
T2, KKT bond [$\times 10^{-2}$]	0.004	0.000	0.000	0.000	0.000	0.012	0.079
Market clearing [$\times 10^{-2}$]	0.723	2.303	0.022	0.561	0.731	0.880	1.666
Market clearing (normalized) [$\times 10^{-2}$]	0.002	0.005	0.000	0.001	0.002	0.002	0.004

NOTES: The first five rows show the equilibrium conditions for type 1 agents, rows 6 to 9 show the conditions for type 2 agents. The first three rows show the relative Euler equation errors (Rel Ee in %) on 10,000 simulated periods. The next two rows show the errors in the KKT conditions ($\times 10^{-2}$). Rows 11 and 12 show the errors in the market clearing conditions ($\times 10^{-2}$). In row 12, the error in the market clearing condition for the bond is divided by aggregate production. The columns show the mean and the max errors as well as the 0.1st, 10th, 50th, 90th, and 99.9th percentile.

Welfare analysis of insurance

Mean lifetime utility in the model where no insurance is available

	Unconditional	Cond. on type					
		T1	T2	Cond. on health shock			
				T1, sick	T1, healthy	T2, sick	T2, healthy
Mean lifetime utility	-18.206	-18.533	-17.878	-18.609	-18.524	-17.992	-17.830
Consumption factor [%] (compared to unconditional)	0.0	-1.8	+1.8	-2.2	-1.7	+1.2	+ 2.1

Mean lifetime utility in the model where insurance is available

	Unconditional	Conditional on type					
		T1	T2	Conditional on health shock			
				T1, sick	T1, healthy	T2, sick	T2, healthy
Mean lifetime utility	-18.204	-18.531	-17.877	-18.569	-18.527	-17.933	-17.852
Consumption factor [%] (compared to unconditional)	0.0	-1.8	+1.8	-2.0	-1.7	+1.5	+ 2.0

Analytical solution

Model with analytical solution
Krueger and Kübler (2004)

- Household problem is given by:

$$\max_{\{c^i_{t+i}, a^i_{t+i}\}_{i=0}^{N-1}} \mathrm{E}_t \left[\sum_{i=0}^{N-1} \log(c^i_{t+i}) \right]$$

subject to:

$$\begin{aligned} c_t^h + a_t^h &= r_t k_t^h + l_t^h w_t \\ k_{t+1}^{h+1} &= a_t^h \\ a_t^N &\geq 0, \end{aligned}$$

- Single representative firm with stochastic Cobb-Douglas production function:

$$f(K_t, L_t, z_t) = \eta_t K_t^\alpha L_t^{1-\alpha} + K_t(1 - \delta_t)$$

- We study the special case where $\mathrm{L_t} = \mathrm{l_t^1} = 1$.

Model with analytical solution
Optimality condition

First-order condition

$$u'(c^i(\mathbf{x})) = \beta E_z \left[r(\mathbf{x}_+) u'(c^{i+1}(\mathbf{x}_+)) \right],$$

where:

$$\begin{aligned}
\mathbf{x}_+ &= [z_+, 0, \boldsymbol{\theta}_a(\mathbf{x})^T]^T \\
r_t &= \alpha \eta_t K_t^{\alpha-1} L_t^{1-\alpha} + (1 - \delta_t) \\
w_t &= (1-\alpha) \eta_t K_t^{\alpha} L_t^{-\alpha}
\end{aligned}$$

$$c^i(\mathbf{x}) = \begin{cases} w(\mathbf{x}) - \theta_a(\mathbf{x})_i, & \mathit{for}\ i = 1, \\ r(\mathbf{x}) x_{1+i} - \theta_a(\mathbf{x})_i, & \mathit{for}\ i = 2, \ldots, N-1, \\ r(\mathbf{x}) x_{1+N} & \mathit{for}\ i = N. \end{cases}$$

Model with analytical solution

Exact solution

The exact solution:

$$\boldsymbol{\theta}_a(\mathbf{x}) = \beta \left[\frac{1-\beta^{N-1}}{1-\beta^N} w(\mathbf{x}), \frac{1-\beta^{N-2}}{1-\beta^{N-1}} r(\mathbf{x}) k_2, \frac{1-\beta^{N-3}}{1-\beta^{N-2}} r(\mathbf{x}) k_3, \ldots, \frac{1-\beta^1}{1-\beta^2} r(\mathbf{x}) k_{N-1} \right]^T$$

which implies the aggregate capital evolution:

$$K_t = \gamma_1 w(\mathbf{x}_{t-1}) + \gamma_2 w(\mathbf{x}_{t-2}) r(\mathbf{x}_{t-1}) + \cdots + \gamma_{N-1} w(\mathbf{x}_{t-(N-1)}) r(\mathbf{x}_{t-1}) r(\mathbf{x}_{t-2}) \ldots r(\mathbf{x}_{t-(N-2)})$$

where:

$$\gamma_i := (\beta^i - \beta^N)/(1 - \beta^N)$$

Model with analytical solution
Economic and algorithm parameterization

Economic parameters	
Number age-groups N	6
Discount factor β	0.7
Relative risk aversion γ	1
Capital share α	0.3
TFP η	$\{0.95, 1.05\}$
Depreciation δ	$\{0.5, 0.9\}$
Persistence TFP:	
$P(\eta_{t+1} = 1.05 \mid \eta_t = 1.05)$	0.5
$P(\eta_{t+1} = 0.95 \mid \eta_t = 0.95)$	0.5
Persistence depreciation:	
$P(\delta_{t+1} = 0.5 \mid \delta_t = 0.5)$	0.5
$P(\delta_{t+1} = 0.9 \mid \delta_t = 0.9)$	0.5
Algorithm parameters	
Learning rate α^{learn}	0.00001
Periods per episode T	12,800
Epochs per episode N^{epochs}	20
Mini-batch size m	640
Hidden layers:	
# Nodes	100, 50
Activation	relu, relu

Model with analytical solution
Euler errors

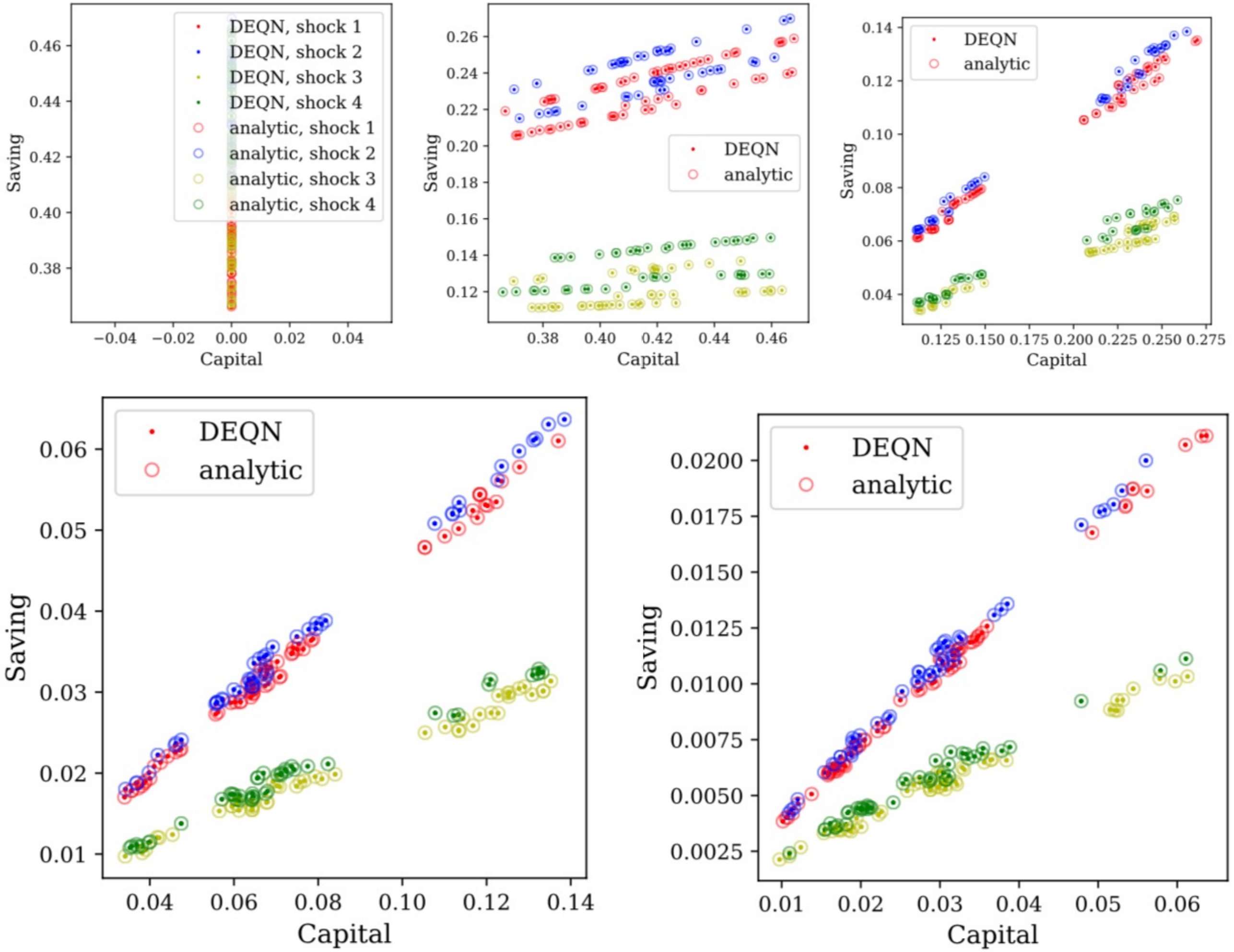
Relative Euler equation errors on 15,000 simulated periods

	mean	max	0.1	10	50	90	99.9
Rel Ee capital [log10]	-3.4	-2.4	-6.4	-4.4	-3.6	-3.0	-2.5

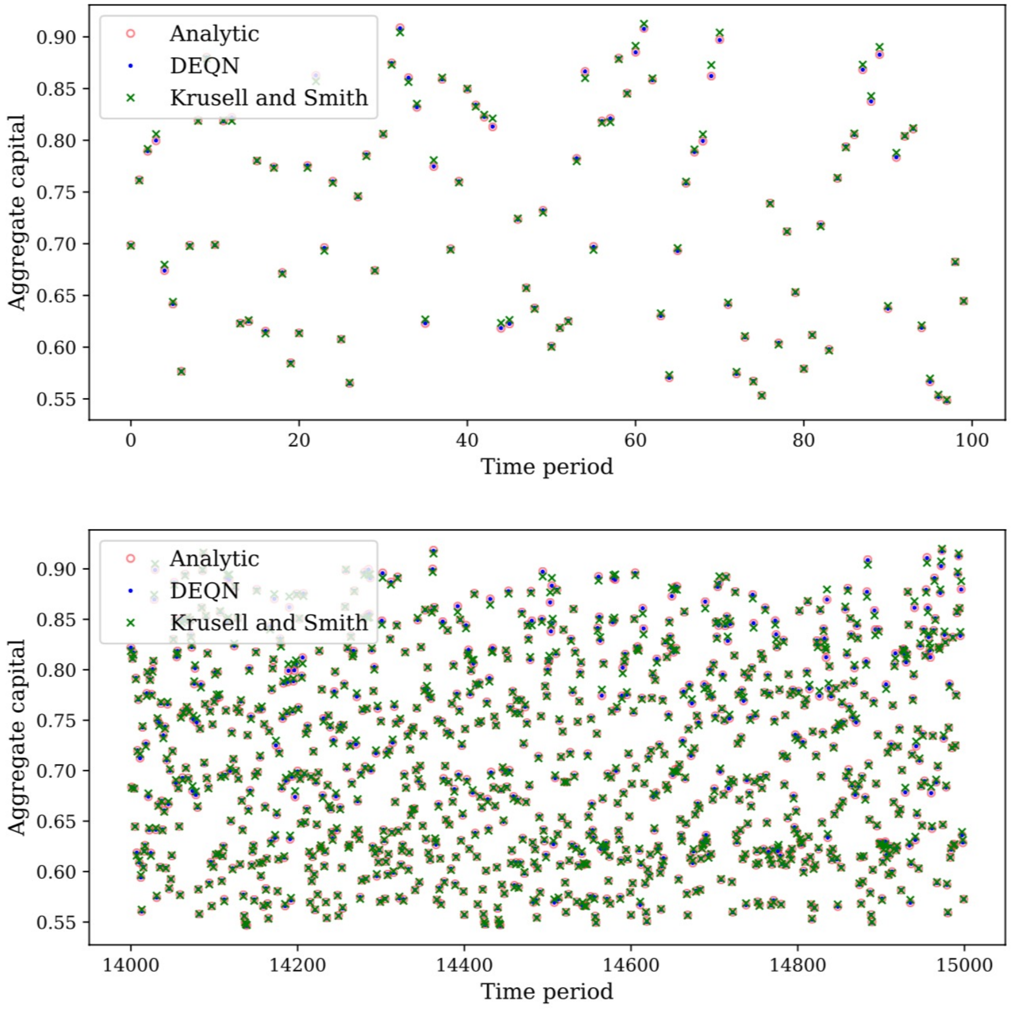
Statistics of the relative errors in the learned policy functions on 15,000 simulated periods

	mean	max	0.1	50	99.9
Relative policy errors age-group 1 [%]	0.03	0.14	0.00	0.03	0.13
Relative policy errors age-group 2 [%]	0.02	0.09	0.00	0.01	0.08
Relative policy errors age-group 3 [%]	0.02	0.10	0.00	0.02	0.09
Relative policy errors age-group 4 [%]	0.01	0.05	0.00	0.01	0.14
Relative policy errors age-group 5 [%]	0.01	0.06	0.00	0.01	0.04
Relative policy errors age-group 1 [log10]	-3.47	-2.85	-6.08	-3.54	-2.88
Relative policy errors age-group 2 [log10]	-3.82	-3.06	-6.72	-3.91	-3.10
Relative policy errors age-group 3 [log10]	-3.69	-2.99	-6.40	-3.75	-3.04
Relative policy errors age-group 4 [log10]	-4.09	-3.29	-7.11	-4.26	-3.37
Relative policy errors age-group 5 [log10]	-3.92	-3.33	-6.70	-4.00	-3.35

DEQN’s approximate versus exact saving decisions



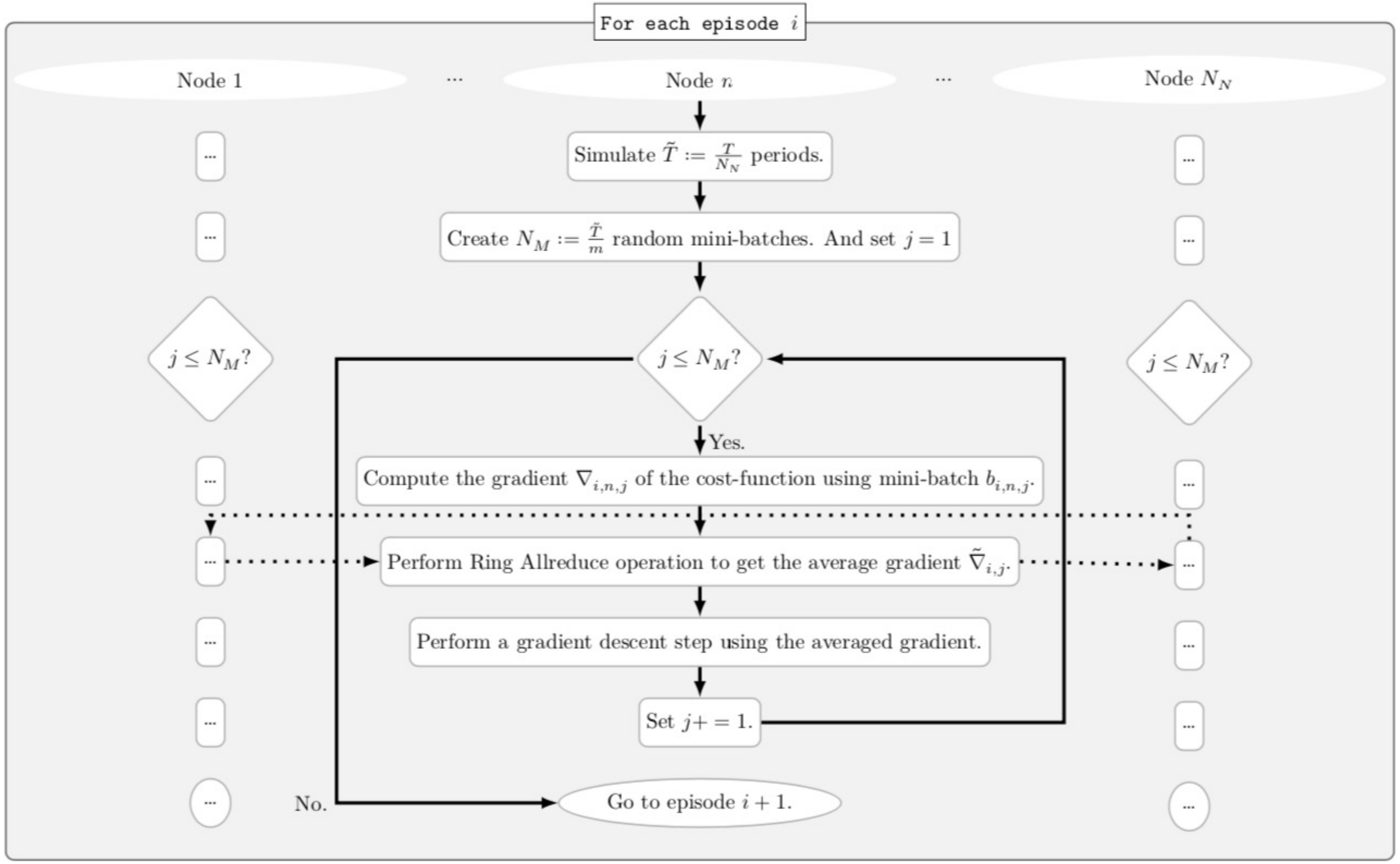
DEQN versus Krusell-Smith approximate aggregation



	mean	max
DEQN [%]	0.019	0.13
Log-linear forecast [%]	0.24	1.3
DEQN [log10]	-3.72	-2.88
Log-linear forecast [log10]	-2.62	-1.87

Parallelization

Schematic illustration of the parallelization scheme



Parallelization scheme

Simulation phase

- On each node n:
  - Draw a random sequence of exogenous shocks and simulate T periods using the policy encoded by the neural network,
  - Create random mini-batches.

Learning phase

- On each node n:
  - For each mini-batch:
    - Compute the gradient of the cost function with respect to the parameters of the neural network,
    - Perform a Ring Allreduce operation to obtain the average gradient across nodes,
    - Update the parameters of the neural network by doing a gradient descent steps as implied by the average gradient.

Strong scaling