Solving Equilibrium Models with Modern Machine Learning Methods

Mahdi Ebrahimi Kahou[1]

October 3, 2024

[1] University of British Columbia, Vancouver School of Economics

Articles

- **Exploiting Symmetry in High-Dimensional Dynamic Programming:** with Jesús Fernández-Villaverde, Jesse Perla, and Arnav Sood.
- **Spooky Boundaries at a Distance: Exploring Transversality and Stability with Deep Learning:** with Jesús Fernández-Villaverde, Sebastián Gómez-Cardona, Jesse Perla, and Jan Rosa.
- **Are Minimizing the Euler and Bellman Residuals Enough?**

Background: Deep learning for functional equations

Equilibrium conditions as functional equations

Most theoretical models in economics with equilibrium conditions can be written as functional equations:

- Take some function(s) $\psi \in \Psi$ where $\psi : \mathcal{X} \to \mathcal{Y}$ (e.g. asset price, investment choice, best-response).
- Domain $\mathcal{X}$ could be state (e.g. dividends, capital, opponents state) or time if sequential.
- The "model" is $\ell : \Psi \times \mathcal{X} \to \mathcal{R}$ (e.g., Euler and Bellman residuals, equilibrium FOCs).
- The solution is the root of the model (residuals operator), i.e., $\mathbf{0} \in \mathcal{R}$, at each $x \in \mathcal{X}$.

Then a **solution** is a $\psi^* \in \Psi$ where $\ell(\psi^*, x) = 0$ for all $x \in \mathcal{X}$. How do we find an approximate solution?

Example: recursive formulation of the neoclassical growth

An example of a recursive case:

- Domain: $x = \begin{bmatrix} k \end{bmatrix}$ and $\mathcal{X} = \mathbb{R}_+$.
- Solve for the optimal policy $k'(\cdot)$ and consumption function $c(\cdot)$: So $\psi : \mathbb{R} \to \mathbb{R}^2$ and $\mathcal{Y} = \mathbb{R}^2_+$.
- Residuals are the Euler equation and feasibility condition, so $\mathcal{R} = \mathbb{R}^2$:

$$\ell(\underbrace{\begin{bmatrix} k'(\cdot) & c(\cdot) \end{bmatrix}}_{\equiv \psi}, \underbrace{k}_{\equiv x}) = \underbrace{\begin{bmatrix} u'(c(k)) - \beta u'(c(k'(k)))\left(f'(k'(k)) + 1 - \delta\right) \\ f(k) - c(k) - k'(k) + (1-\delta)k \end{bmatrix}}_{\text{model}}$$

- Finally, $\psi^* = [k'(\cdot), c(\cdot)]$ is a solution if it has zero residuals on domain $\mathcal{X}$.

Classical solution method for functional equations

Quick **review** of collocation-like methods:

1. **Pick** finite set of D points $\hat{\mathcal{X}} \subset \mathcal{X}$ (e.g., a grid).
2. **Choose** approximation $\hat{\psi}(\cdot;\theta) \in \mathcal{H}(\Theta)$ with coefficients $\Theta \subseteq \mathbb{R}^M$ (e.g., Chebyshev polynomials).
3. **Fit** with nonlinear least-squares

   $$\min_{\theta \in \Theta} \sum_{x \in \hat{\mathcal{X}}} \ell(\hat{\psi}(\cdot;\theta), x)^2$$

   If $\theta \in \Theta$ is such that $\ell(\hat{\psi}(\cdot;\theta), x) = 0$ for all $x \in \hat{\mathcal{X}}$ we say $\hat{\psi}(\cdot;\theta)$ **interpolates** $\hat{\mathcal{X}}$.
4. The goal is to have good **generalization**:
   - The approximate function is close to the solution outside of $\hat{\mathcal{X}}$.
   - In high dimensions this becomes very important.

A deep learning approach: I

Recall we need a parametric function $\hat{\psi}(\cdot;\theta) \in \mathcal{H}(\Theta)$:

- **Deep neural networks** are **highly-overparameterized** functions designed for good generalization.
- Number of coefficients much larger than the grid points ($M \gg N$).
- Example: one layer neural network, $\hat{\psi} : \mathbb{R}^Q \to \mathbb{R}$:

$$\hat{\psi}(x;\theta) = W_2 \cdot \sigma\left(W_1 \cdot x + b_1\right) + b_2$$

- $W_1 \in \mathbb{R}^{P\times Q}$, $b_1 \in \mathbb{R}^{P\times 1}$, $W_2 \in \mathbb{R}^{1\times P}$, and $b_2 \in \mathbb{R}$.
- $\sigma(\cdot)$ is a nonlinear function applied element-wise (e.g., $\max\{\cdot, 0\}$, Sigmoid, Tanh,...).

A deep learning approach: II

- $\Theta \equiv \{b_1, W_1, b_2, W_2\}$ are the coefficients, in this example $M = PQ + P + P + 1$.
- Making it "deeper" by adding another "layer":

$$\hat{\psi}(x;\theta) \equiv W_3 \cdot \sigma(W_2 \cdot \sigma(W_1 \cdot x + b_1) + b_2) + b_3.$$

- Think of deep neural networks as **parametric functions**.
- Architecture of the neural networks can be flexibly informed by the economic insight and theory.
  - The **Symmetry** paper heavily relies on this.

Concerns regarding over-parameterization

> *"I remember my friend Johnny von Neumann used to say, with four parameters I can fit an elephant, and with five I can make him wiggle his trunk."*
>
> *Enrico Fermi*

> *"The best way to solve the problem from practical standpoint is you build a very big system ... basically you want to make sure you hit the zero training error."*
>
> *Ruslan Salakhutdinov, SI 2017*

- If the number of parameters is much larger than the grid points (i.e. $M \gg N$), there might be many interpolating solutions.
- So **which solution(s)** are we going to find? .
  - I will come back to this in the **Spooky** paper.

Exploiting Symmetry in High-Dimensional Dynamic Programming

Motivation

- Most dynamic models in macro (and other fields) deal with either:
  - Representative agent or few agents.
  - A continuum of agents.
- However, many models of interest in macro (IO and trade) deal with **finite** (but large) number of agents and idiosyncratic/aggregate uncertainty:
  - Industry dynamics with many firms, agents and industries, even models with networks.
  - Heterogeneous agent labor models (e.g., overlapping generations, different types).
- These models are becoming increasingly popular, **but**:
  - They pose computational challenges as we add more agents.
  - **No (non-heuristic)** algorithm exists providing **global** solutions in the presence of aggregate uncertainty.

Challenges: the curse of dimensionality in equilibrium models

Three components to the curse of dimensionality with many agents (Bellman, 1958, p. IX)

1. The cardinality of the state space is enormous.
   - With 266 state variables, with 2 values per state (zero and one), we have more arrangements (2^{266}) than the estimated number of protons in the universe.
2. With idiosyncratic and aggregate shocks we need to calculate high-dimensional conditional expectations.
3. Finding equilibrium paths to the steady-state (ergodic distributions) are extremely hard in high-dimensions.

Contribution

Inspired by economic theory, providing novel method for **globally** solving high-dimensional heterogeneous agent models with **aggregate shocks** which relies on:

1. A **symmetry** present in many heterogeneous agent models, i.e., **exchangeability** of agents.
    - Example: In general equilibrium models the **Walrasian** auctioneer removes indices.
2. **Concentration of measures**, something that resembles the law of large numbers to deal with conditional expectations (very fast).
    - More agents makes it easier to forecast the evolution of distributions.
3. Show how to implement the symmetry when using **deep neural networks**.

With these we **globally** solve a model with **10,000** (and even more) agents which was **not possible** before.

Literature Review

- Deep learning as a functional approximation: Maliar et al. (2019), Fernández-Villaverde et al. (2022), Duarte (2018), Azinovic et al. (2022), Han et al. (2021) (a mean-field approach).
- Symmetry in statistics and machine learning: Bloem-Reddy and Teh (2020), Zaheer et al. (2017), and Yarotsky (2018).
- Symmetry in computer science (MDP/RL): Ravindran and Barto (2001) and Narayanamurthy and Ravindran (2008), van der Pol et al. (2020).
- Symmetry in micro and games: Jovanovic and Rosenthal (1988), Hartford et al. (2016)

Application

How do we pick our application to show how all this works?

- In terms of application, there are two routes:

  1. Introducing a sophisticated application where the method “shines”.
  2. Or, applying it to a well-known example.

- If I tell you about a sophisticated application, how do we know our “solution” method works?
- So we study a well-known example (with a twist).
- Study the more sophisticated applications in future projects.

Our application

A variation of the Lucas and Prescott (1971) model of investment under uncertainty with N firms.

Why?

1. Ljungqvist and Sargent (2018), pp. 226-228, use it to introduce recursive competitive equilibria.
2. Simple model that fits in one slide.
3. Under one parameterization, the model has a known Linear-Quadratic (LQ) solution, which gives us an exact benchmark.
4. By changing one parameter, the model is nonlinear, with no known solution. Our method handles the nonlinear case as easily as the LQ case with high accuracy.

Investment under uncertainty

- Industry consisting of $N > 1$ firms, each producing the same good.
- Firm of interest produces output x (x units of capital).
- Thus, the vector $X \equiv [X_1, \ldots X_N]^\top$ is the production (or capital) of the whole industry.
- The inverse demand function for the industry is, for some $\nu \geq 1$ (this is our twist):

$$p(X) = 1 - \frac{1}{N} \sum_{i=1}^{N} X_i^\nu$$

- The firm does not consider the impact of its individual decisions on $p(X)$.
- Adjustment cost: investing u has a cost $\frac{\gamma}{2} u^2$.
- Law of motion for capital $x' = (1 - \delta)x + u + \sigma w + \eta \omega$ where $w \sim \mathcal{N}(0, 1)$ an i.i.d. idiosyncratic shock, and $\omega \sim \mathcal{N}(0, 1)$ an i.i.d. aggregate shock, common to all firms.
- The firm chooses u to maximize $\mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t \left(p(X)x - \frac{\gamma}{2} u^2\right)\right]$.

Recursive problem

The recursive problem of the firm taking the exogenous policy $\hat{u}(\cdot, X)$ for all other firms as given is:

$$v(x, X) = \max_u \left\{ p(X)x - \frac{\gamma}{2}u^2 + \beta\mathbb{E}\left[v(x', X')\right] \right\}$$

$$\text{s.t. } x' = (1-\delta)x + u + \sigma w + \eta\omega$$

$$X_i' = (1-\delta)X_i + \hat{u}(X_i, X) + \sigma W_i + \eta\omega, \quad \text{for } i \in \{1, ..., N\}$$

First order conditions + **symmetric equilibrium**

$$\boxed{\gamma u(x, X) = \beta\mathbb{E}\left[p(X') + \gamma(1-\delta)u(x', X')\right]}$$

Goal: Using economic theory to

Design $\mathcal{H}(\Theta)$ class for approximating $u(x, X)$?

General class of problems: A “big X, little x” dynamic programming

$$
\begin{aligned}
v(x, X) &= \max_u \left\{ r\left(x, u, X\right) + \beta \mathbb{E}\left[v(x', X')\right] \right\} \\
\text{s.t. } x' &= g(x, u) + \sigma w + \eta \omega \\
X' &= G(X) + \Omega W + \eta \omega \mathbf{1}_N
\end{aligned}
$$

1. x is the individual state of the agent.

2. X is a vector stacking the individual states of all of the N agents in the economy.

3. u is the control variable.

4. w is random innovation to the individual state, stacked in $W \sim \mathcal{N}(\mathbf{0}_N, \mathbf{I}_N)$ and where, w.l.o.g., $w = W_1$.

5. $\omega \sim \mathcal{N}(0, 1)$ is a random aggregate innovation to all the individual states.

Permutation Groups

- A permutation matrix is a square matrix with a single 1 in each row and column and zeros everywhere else.
- Let $\mathcal{S}_N$ be the set of all $n!$ permutation matrices of size $N \times N$. For example:

$$S_2 = \left\{ \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \right\}$$

- Multiplying vector $v \in \mathbb{R}^N$ by $\pi \in S_N$ reorders elements of v
- (If you know about this): $\mathcal{S}_N$ is the *symmetric group* under matrix multiplication.

Permutation-invariant dynamic programming

Definition

A 'big X, little x' dynamic programming problem is a **permutation-invariant dynamic programming problem** if, for all $(x, X) \in \mathbb{R}^{N+1}$ and all permutations $\pi \in \mathcal{S}_N$

1. The reward function r is **permutation invariant**:

$$r(x, u, \pi X) = r(x, u, X)$$

2. The deterministic component of the law of motion for X is **permutation equivariant**:

$$G(\pi X) = \pi G(X)$$

3. The covariance matrix of the idiosyncratic shocks satisfies

$$\pi\Omega = \Omega\pi$$

Main results I: Permutation invariance of the optimal solution

Proposition

The optimal solution of a permutation-invariant dynamic programming problem is permutation invariant. That is, for all $\pi \in \mathcal{S}_N$:

$$u(x, \pi X) = u(x, X)$$

and:

$$v(x, \pi X) = v(x, X)$$

Can $u(x, X)$ permutation invariance guide $\mathcal{H}(\Theta)$ choice?

Curse of dimensionality in this example

Recall there are three separate sources of the "curse" here as we increase the number of agents:

1. Can we approximate $u(x, X)$ for high dimensional $X \in \mathbb{R}^N$ without massive increases in the $\hat{\mathcal{X}}$ grid?
2. Given intuition that individual $X_i \in X$ have limited effect on $u(x, X)$, how to calculate $\mathbb{E}\left[u(x', X')\right]$?
   - Look at $\mathbb{E}_W[u(x', X')|w, \omega]$ to condition on firm's idiosyncratic w aggregate shock ω.
   - Why conditioning on these two? They matter a lot. Now, can something similar to the law of large numbers happen?
3. What about the stationary solutions and transversality condition?
   - Euler equation have multiple solutions, some leading to non-stationary paths (I will come back to this).

Representation of permutation-invariant functions

Proposition

(based on Wagstaff et al., 2019) Let $f : \mathbb{R}^{N+1} \to \mathbb{R}$ *be a continuous permutation-invariant function under* $\mathcal{S}_N$*, i.e., for all* $(x, X) \in \mathbb{R}^{N+1}$ *and all* $\pi \in \mathcal{S}_N$*:*

$$f(x, \pi X) = f(x, X)$$

Then, there exist a latent dimension $L \leq N$ *and continuous functions* $\rho : \mathbb{R}^{L+1} \to \mathbb{R}$ *and* $\phi : \mathbb{R} \to \mathbb{R}^L$ *such that:*

$$f(x, X) = \rho\left(x, \frac{1}{N}\sum_{i=1}^{N} \phi(X_i)\right)$$

Representation of permutation-invariant functions: Discussion and intuition

$$u(x, X) = \rho\left(x, \frac{1}{N}\sum_{i=1}^{N}\phi(X_i)\right)$$

- This proposition should remind you of Krusell-Smith (1998), $L = 1$, $\phi(X_i) = X_i$.
- Key benefit for approximation is the **representation** (ρ, ϕ), **not explicit** dimensionality reduction.
- Fitting a ρ and ϕ rather than f directly leads to **far better generalization** on $\mathcal{X}$. Why?:
  - Imposing structure on $\mathcal{H}(\Theta)$, functions that know a lot about the economic problem.
- In practice: $L \ll N$ generalizes very well.

Representation of permutation-invariant functions: Discussion and intuition

- We have seen a **variation** of this in IO.
  - Exit/Entry problems, $X_i \in \{0, 1\}$.
- Remember the example with 266 states, binary values (zeros and ones)
  - $f(x, X) : 2^{N+1} \to \mathbb{R}$.
  - If permutation invariant: I only care about the **number of ones**.
  - The dimensionality goes from 2^{N+1} to $N + 2$.

$$f(x, X) = \hat{f}(x, \frac{1}{N} \sum_{i=1}^{N} X_i)$$

But in this paper X_is are continuous variables.

Expected gradient bounded in N

We need to focus on specific functions to deal with high-dimensional expectations:

Definition (Expected gradient bounded in N)

Let $f : \mathbb{R}^N \to \mathbb{R}$ be a bounded function in N and $z \sim \mathcal{N}(\mathbf{0}_N, \mathbf{I}_N)$ be a normalized Gaussian random vector. The function f has its expected gradient bounded in N if there exists a C such that:

$$\mathbb{E}\left[\|\nabla f(z)\|^2\right] \leq \frac{C}{N},$$

where C does not depend on N.

$$\mathbb{E}_W\left[\|\nabla u(x', X')\|^2\right] \leq \frac{C}{N}$$

- W: the idiosyncratic shocks of the rest of the agents in the economy.
- The policy to be well-behaved (non-explosive gradients).
- Other agent's influence vanishes as the economy grows.

Main result II: Concentration of measure

Proposition

Suppose $z \sim \mathcal{N}(\mathbf{0}_N, \Sigma)$, where the spectral radius of Σ, denoted by $\rho(\Sigma)$, is independent of N, z^1 a draw from z, and $f : \mathbb{R}^N \to \mathbb{R}$ is a function with expected gradient bounded in N. Then:

$$\mathbb{P}\left(\left|f(z^1) - \mathbb{E}\left[f(z)\right]\right| \geq \epsilon\right) \leq \frac{\rho(\Sigma)C}{\epsilon^2}\frac{1}{N}$$

- As Ledoux (2001) puts it: "A random variable that depends in a Lipschitz way on many independent variables (but not too much on any of them) is essentially constant."
- With concentration of measure, dimensionality is not a curse; it is a blessing.

Implication: We can calculate $\mathbb{E}_W[u(x', X')|w, \omega]$ with a *single draw* of idiosyncratic shocks W:

- $\mathbb{E}_W[u(x', X')|w, \omega] \approx u(x', X')|w, \omega$.
- Reducing an $N+1$-dimensional conditional expectation to a 2-D one (with good approximation).

Summarizing the results

- The structure symmetry imposes on the functions leads to better **generalization**
  - Functions extrapolate better outside of the grid points $\hat{\mathcal{X}}$.
- Concentration of measures provides a fast method for calculating the conditional expectations.
  - Calculate with **one draw** of the idiosyncratic shocks (conditional on the aggregate shock).
- **No non-heuristic** algorithm exists to solve this problem.

Solving the Model

Design of $\mathcal{H}(\Theta)$: Deep learning architectures

$$\boxed{u(x, X) = \rho\left(x, \frac{1}{N}\sum_{i=1}^{N}\phi(X_i)\right)}$$

Three cases for ϕ:

1. Identity function: One moment → ϕ(Identity).
2. Up to degree four polynomials: 4 moments → ϕ(Moments).
3. A deep neural network for ϕ, with $L = 4$ → ϕ(ReLU).

If polynomials for ϕ: A finite set of moments à la Krusell-Smith.

- In all cases, ρ is a highly over-parameterized neural network with four layers.
- The baseline ϕ(Identity), ϕ(Moments), and ϕ(ReLU) have $49.4K$, $50.3K$, and $199.6K$ coefficients.

Solution method follows “interpolation” methods

1. **Pick**: $\hat{\mathcal{X}}$ as simulated trajectories from X_0:
   - Only need 100 to 1000 points regardless of dimensionality of the state space N.
   - Using economic insight (i.e., symmetry) gives us good generalization.
2. **Choose**: Design the $\mathcal{H}(\Theta)$ with ρ and ϕ as discussed:
   - ϕ(Identity), ϕ(Moments), and ϕ(ReLU).

Applying concentration of measures:

- **One** draw $\hat{W} = \{\hat{W}_1, \ldots, \hat{W}_N\}$ of the idiosyncratic shocks. For a given $u(\cdot;\theta)$, and aggregate shock ω calculate:

$$X_i' = (1-\delta)X_i + u(X) + \sigma\hat{W}_i + \eta\omega, \quad \text{for } i \in \{1, ..., N\}.$$

Solution method follows “interpolation” methods

- Approximate the Euler residuals

$$\varepsilon\left(X; u(\cdot;\theta)\right) \equiv \gamma u(X;\theta) - \beta\mathbb{E}\left[P(X') + \gamma(1-\delta)u(X';\theta)\right]$$

using concentration of measures (one draw of W in X').

⏩ error analysis in N

3. **Fit**: The residuals $\varepsilon\left(X; u(\cdot;\theta)\right)$, that is the “model” i.e., ℓ.

$$\min_{\theta\in\Theta} \sum_{X\in\hat{\mathcal{X}}} \varepsilon\left(X; \hat{u}(\cdot;\theta)\right)^2$$

4. **How to Verify/Test**: Given the **approximate** solution simulate **new paths** from X_0 and check the Euler residuals (ε).

Study two cases: linear ($\nu = 1$) and nonlinear ($\nu > 1$) demand functions

Case 1: Linear to verify algorithms and methods

- With $\nu = 1$, we have a linear demand function: $p(X) = 1 - \frac{1}{N}\sum_{i=1}^{N} X_i$.
- It generates a Linear-Quadratic (LQ) dynamic programming problem (only the mean of X_i matters).
- We can find the exact $u(x, X)$, LQ has algebraic solutions.
- The LQ solution gives us a benchmark against which we can compare our deep learning solution.
- The neural network figures learns the true solution, $u(x, X) = H_0 + H_1 \frac{1}{N}\sum_{i=1}^{N} X_i$, very quickly.

Euler residuals: Linear case

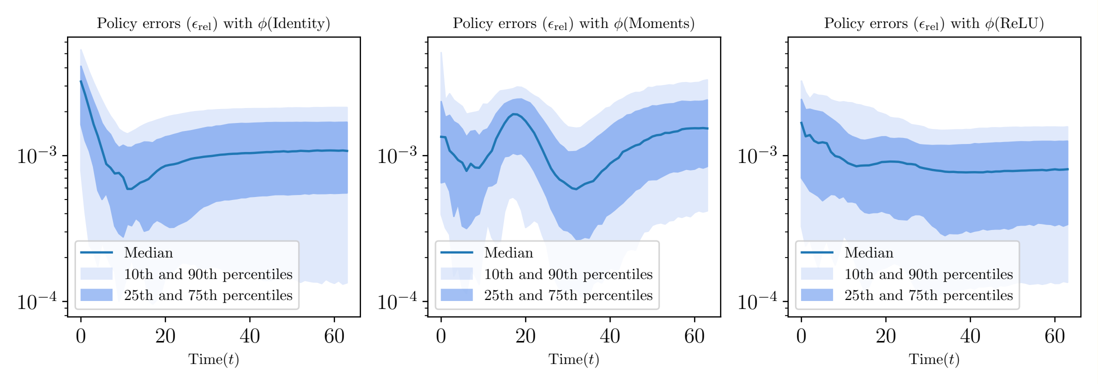


Figure 1: The absolute relative errors for $\nu = 1$ and $N = 128$ for ϕ(Identity), ϕ(Moments), and ϕ(ReLU). The dark blue curve shows the median errors along equilibrium paths for 100 **seeds** and 32 different trajectories.

$$\varepsilon \equiv \left| \frac{u(X) - \hat{u}(X)}{u(X)} \right|$$

Equilibrium Paths: Linear case

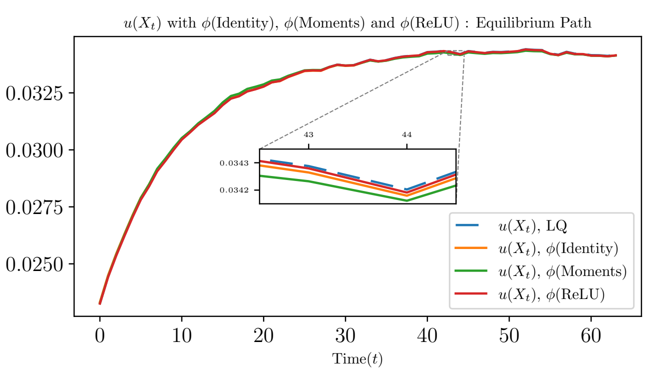


Figure 2: Comparison between baseline approximate solutions and the LQ solution for the case with $\nu = 1$ and $N = 128$.

Computation time: Linear case

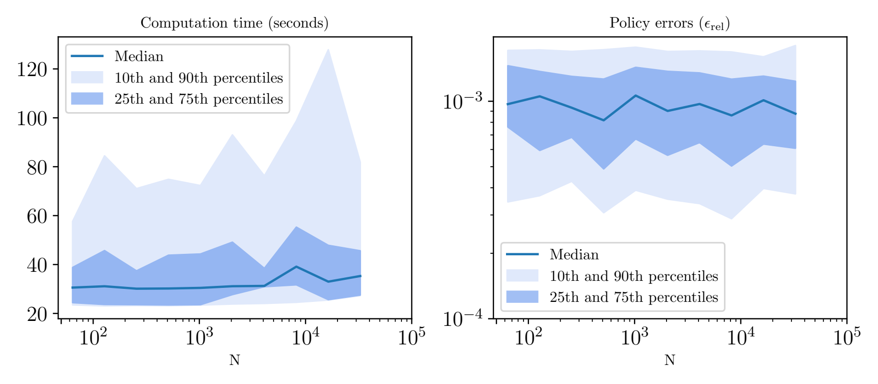


Figure 3: Performance of the ϕ(ReLU) for different N (median value of 100 trials).

Case 2: Nonlinear case with no “closed-form” solution

- With $\nu > 1$, we have a nonlinear demand function: $p(X) = 1 - \frac{1}{N}\sum_{i=1}^{N} X_i^{\nu}$.
- Notice how, now, the whole distribution of X_i matters.
- But we can still find the solution to this nonlinear case using exactly the same functional approximation and algorithm as before.
- We do not need change anything in the code except the value of ν.
- Since the LQ solution no longer holds, we do not have an exact solution to use as a benchmark, but can check residuals.
- Same model and method. Computation time by N nearly the same to linear case

Euler residuals: Nonlinear case

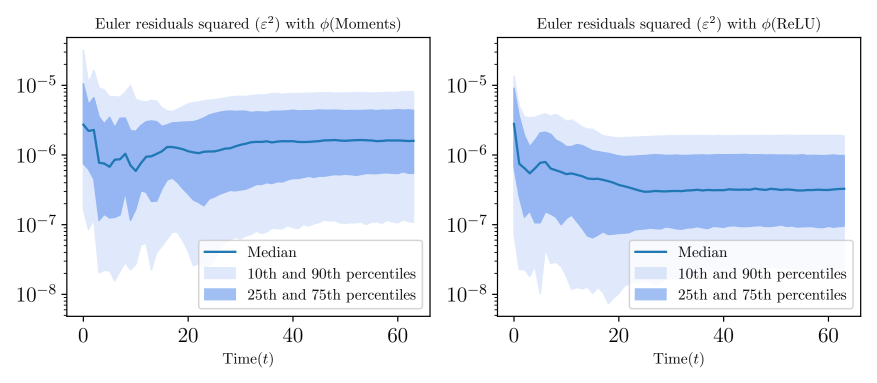


Figure 4: The Euler residuals for $\nu = 1.5$ and $N = 128$ for ϕ(Moments) and ϕ(ReLU). The dark blue curve shows the average residuals along equilibrium paths for **100** seeds and 32 different trajectories.

Equilibrium paths: Nonlinear case

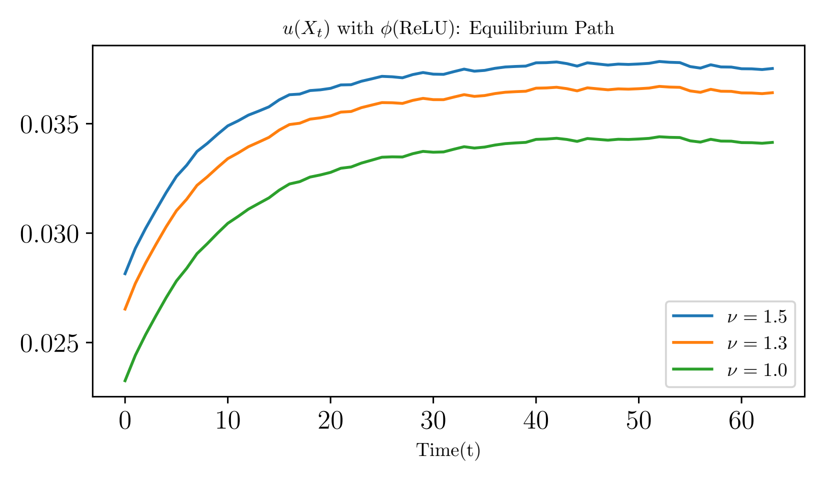


Figure 5: The optimal policy u along the equilibrium paths for $\nu = [1.0, 1.05, 1.1, 1.5]$ and $N = 128$. Each path shows the optimal policy for a single trajectory.

Some challenging question: Generalization puzzle

Question I: Generalization

- From statistical learning and numerical analysis we know:
  - More coefficients in the family of parametric functions $\mathcal{H}(\Theta)$ leads to over-fitting and poor generalization (bias-variance trade-off).
  - We have $\approx 200K$ parameters, and $< 1K$ grid points.
  - The results indicate the opposite: More coefficients $\rightarrow$ better generalization.

How come we achieve great generalization?

Convergence results: Transversality condition and stationarity

Table 1: Simulating multiple seeds for different $\mathcal{H}(\Theta)$ with $\nu = 1$

		Success (%)	Early stopping failure (%)	Violation of transversality (%)	Overfitting (%)
Group	Description				
Identity	Baseline	48%	2%	50%	0%
Moments	Baseline	59%	2%	37%	2%
Deep Sets	Baseline	97%	0%	3%	0%

Some challenging questions: Multiplicity and transversality puzzle

Question II: Multiplicity and transversality

$$\boxed{\begin{aligned} \gamma u(X) &= \beta \mathbb{E}\left[p(X') + \gamma(1-\delta)u(X')\right] \\ X_i' &= (1-\delta)X_i + u(X) + \sigma W_i + \eta\omega, \quad \text{for } i \in \{1, ..., N\} \end{aligned}}$$

with linear prices. Guess and verify with $u(X) \equiv H_0 + \frac{1}{N}H_1 \sum_{i=1}^{N} X_i$

- The Euler equation is quadratic → **two** solutions: $(H_0^-, H_1^-), (H_0^+, H_1^+)$:
  - $H_1^- < 0 \rightarrow$ **stationary** solution, $H_1^+ > 0 \rightarrow$ **non-stationary** solution.
  - We have **no explicit** device in our algorithm to weed out the second solution.

How come there is a strong bias towards the stationary solution

Understanding the **implicit bias** of deep neural networks answers both questions.

Summarizing the contribution

- **Method** for solving **high-dimensional** dynamic programming problems and competitive equilibria with idiosyncratic and aggregate shocks relying
  - Symmetry.
  - Concentration of measures: Dimensionality is a **blessing** not a curse.
- Using **economic theory** (i.e., exchangeability) and **deep learning** for function approximation with a huge # of parameters ($\gg$ grid points)
  - Achieve great generalization: key to alleviate the curse of dimensionality.
- Implementation
  - Can deal with 10000+ agents.
  - Can deal with 10000+ dimensional expectations with one Monte-carlo draw.

Spooky Boundaries at a Distance: Exploring Transversality and Stability with Deep Learning

Motivation

- Dynamic models usually require **economic conditions** eliminating explosive solutions (e.g., transversality or no-bubble).
    - These are variations of “boundary conditions” in ODEs and PDEs on **forward-looking** behavior.
    - Deterministic, stochastic, sequential, recursive formulations all require conditions in some form.
- These forward-looking boundary conditions are the key limitation on increasing dimensionality:
    - Otherwise, in sequential setups, we can easily solve high-dimensional initial value problems.
    - In recursive models accurate solutions are required for arbitrary values of the state variables.
- **Question:** Can we avoid precisely calculating steady-state, BGP, and stationary distribution, which are never reached, and still have accurate short/medium-run dynamics disciplined by these boundary conditions?

Contribution

- Show that **deep learning** solutions to many dynamic forward-looking models automatically fulfill the long-run boundary conditions we need (transversality and no-bubble).
  - We show how to design the approximation using economic insight.
- Solve classic models with known solutions (asset pricing and neoclassical growth) and show excellent short/medium term dynamics –even when **non-stationary** or with **steady state multiplicity**.
- Suggests these methods may solve high-dimensional problems while avoiding the key computational limitation.
  - We have to understand low-dimensional problems first.
- **Intuition**: DL has an "implicit bias" toward smooth and simple functions. Explosive solutions are not smooth.

So what is this **implicit bias**?

Deep learning optimizes in a space of functions

Remember

$$\min_{\theta\in\Theta}\sum_{x\in\hat{\mathcal{X}}}\ell(\hat{\psi}(\cdot;\theta),x)^2$$

- Deep learning: number of coefficients is much larger than the number of grid points.
- Since $M \gg D$, it is possible for $\hat{\psi}$ to interpolate and the objective value will be ≈ 0.
- Since $M \gg D$ there are many solutions (e.g., θ_1 and θ_2),
  - Agree on the grid points: $\hat{\psi}(x;\theta_1) \approx \hat{\psi}(x;\theta_2)$ for $x \in \hat{\mathcal{X}}$.
- Since individual θ are irrelevant it is helpful to think of optimization directly within $\mathcal{H}$

$$\boxed{\min_{\hat{\psi}\in\mathcal{H}}\sum_{x\in\hat{\mathcal{X}}}\ell(\hat{\psi},x)^2}$$

But which $\hat{\psi}$?

Deep learning and interpolation

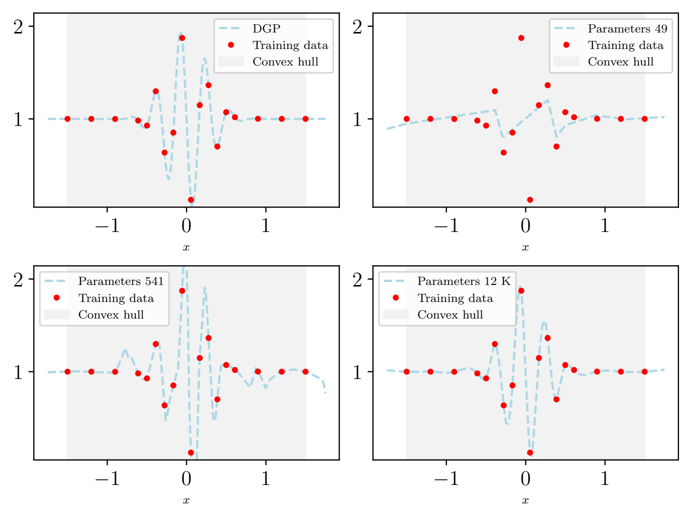
- For M large enough, optimizers **tend to** converge to **unique "simple"** $\hat{\psi}$ (w.r.t to some norm $\|\cdot\|_S$). Unique both in $\hat{\mathcal{X}}$ and $\mathcal{X}$. There is a **bias** toward a specific class of solutions.
- **How to interpret:** interpolating solutions for some functional norm $\|\cdot\|_S$

$$\min_{\hat{\psi}\in\mathcal{H}} ||\hat{\psi}||_S$$
$$\text{s.t. } \ell(\hat{\psi}, x) = 0, \quad \text{for } x \in \hat{\mathcal{X}}$$

  - Comp Sci literature refers to this as the **inductive bias** or **implicit bias**: optimization process is biased toward particular $\hat{\psi}$.
  - Small values of $\|\cdot\|_S$ corresponds to **flat** solutions with **small gradients** (w.r.t. input).
  - Characterizing $\|\cdot\|_S$ is an active research area in CS at the heart of deep learning theory.

⏩ Sobolev

Flat and smooth interpolation: Illustration



Deep learning and interpolation in practice

Reminder: in practice we solve

$$\boxed{\min_{\theta\in\Theta}\sum_{x\in\hat{\mathcal{X}}}\ell\left(\hat{\psi}(\cdot;\theta),x\right)^2}$$

- The smooth interpolation is imposed **implicitly** through the optimization process.
- No explicit norm minimization or penalization is required.

In this paper: we describe how (and when) the $\min_{\hat{\psi}\in\mathcal{H}}||\hat{\psi}||_S$ solutions are also the ones which automatically fulfill transversality and no-bubble conditions.

- They are disciplined by long-run boundary conditions. Therefore, we can obtain accurate short/medium-run dynamics.

Outline

To explore how we can have accurate short-run dynamics, we show deep learning solutions to

1. Classic linear-asset pricing model.
2. Sequential formulation of the neoclassical growth model.
3. Sequential neoclassical growth model with multiple steady states.
4. Recursive formulation of the neoclassical growth model.
5. Non-stationarity, such as balanced growth path.

Linear asset pricing

Sequential formulation

- Dividends, $y(t)$, y_0 as given, and follows the process:

$$y(t+1) = c + (1+g)y(t)$$

- Writing as a linear state-space model with $x(t+1) = Ax(t)$ and $y(t) = Gx(t)$ and

$$x(t) \equiv \begin{bmatrix} 1 & y(t) \end{bmatrix}^\top, A \equiv \begin{bmatrix} 1 & 0 \\ c & 1+g \end{bmatrix}, G \equiv \begin{bmatrix} 0 & 1 \end{bmatrix}$$

- "Fundamental" price given $x(t)$ is PDV with $\beta \in (0,1)$ and $\beta(1+g) < 1$

$$\boxed{p_f(t) \equiv \sum_{j=0}^{\infty} \beta^j y(t+j) = G(I - \beta A)^{-1} x(t).}$$

Recursive formulation

With standard transformation, all solutions $p_f(t)$ fulfill the recursive equations

$$p(t) = Gx(t) + \beta p(t+1) \tag{1}$$

$$x(t+1) = Ax(t) \tag{2}$$

$$0 = \lim_{T\to\infty} \beta^T p(T) \tag{3}$$

$$x_0 \text{ given} \tag{4}$$

That is, a system of two difference equations with one boundary and one initial condition.

- The boundary condition (3) is an **condition** necessary for the problem to be well-posed and have a unique solution.
- It ensures that $p(t) = p_f(t)$ by imposing long-run boundary condition.
- But without this assumption there can be "bubbles" with $p(t) \neq p_f(t)$, only fulfilling (1) and (2).
- Intuition: system of $\{p(t), x(t)\}$ difference equations requires total of two boundaries or initial values to have a unique solution.

Solutions without no-bubble condition

Without the no-bubble condition:

- Solutions in this deterministic asset pricing model are of the form:

$$\boxed{p(t) = p_f(t) + \zeta\, \beta^{-t}.} \tag{5}$$

- For any $\zeta \geq 0$. The initial condition $x(0)$ determines $p_f(t)$.
- There are infinitely many solutions.
- The no-bubble condition chooses $\zeta = 0$.

Interpolation problem: without no-bubble condition

- A set of points in time $\hat{\mathcal{X}} = \{t_1, \ldots, t_{\max}\}$.
- A family of over-parameterized functions $p(\cdot;\theta) \in \mathcal{H}(\Theta)$.
- Generate $x(t)$ using the law of motion and $x(0)$, equation (2).

  In practice we minimize the residuals of the recursive form for the price:

$$\boxed{\min_{\theta\in\Theta} \frac{1}{|\hat{\mathcal{X}}|} \sum_{t\in\hat{\mathcal{X}}} \left[p(t;\theta) - Gx(t) - \beta p(t+1;\theta)\right]^2} \tag{6}$$

- This minimization **does not contain** no-bubble condition. It has infinitely many minima.
- Does the implicit bias of over-parameterized interpolation weed out the bubbles? **Yes**.
- **Intuition**: bubble solutions are explosive, i.e., big functions with big derivatives.

  Let's analyze this more rigorously.

Interpolation formulation: min-norm mental model

The min-norm **interpretation** (mental model) is:

$$
\boxed{
\begin{aligned}
&\min_{p \in \mathcal{H}} && \|p\|_S && (7)\\
&\text{s.t.} && p(t) - Gx(t) - \beta p(t+1) = 0 \quad \text{for } t \in \hat{\mathcal{X}} && (8)\\
& && 0 = \lim_{T \to \infty} \beta^T p(T) && (9)
\end{aligned}
}
$$

Where $x(t)$ for $t \in \hat{\mathcal{X}}$ is defined by $x(0)$ initial condition and recurrence $x(t+1) = Ax(t)$ in (2)

- The minimization of norm $\|p\|_S$ is the "inductive bias" toward particular solutions for $t \in [0, \infty] \setminus \hat{\mathcal{X}}$.

Is the no-bubble condition still necessary?

- To analyze, drop the no-bubble condition and examine the class of solutions.
- In this case, we know the interpolating solutions to (8) without imposing (9)

$$p(t) = p_f(t) + \zeta\beta^{-t} \tag{10}$$

- Applying the triangle inequality

$$\boxed{\|p_f\|_S \leq \|p\|_S \leq \|p_f\|_S + \zeta\ \|\beta^{-t}\|_S} \tag{11}$$

- Relative to classic methods the "deep learning" problem now has a new objective, minimizing $\|p\|_S$.
- The new objective of minimizing the norm, makes the no-bubble condition **redundant**.

Min-norm norm formulation: redundancy of no-bubble condition

Given the no-bubble condition is automatically fulfilled, could solve the following given some $\mathcal{H}$ and compare to $p_f(t)$

$$\min_{p\in\mathcal{H}} \quad \|p\|_S \tag{12}$$

$$\text{s.t.} \quad p(t) - Gx(t) - \beta p(t+1) = 0 \quad \text{for } t \in \hat{\mathcal{X}} \tag{13}$$

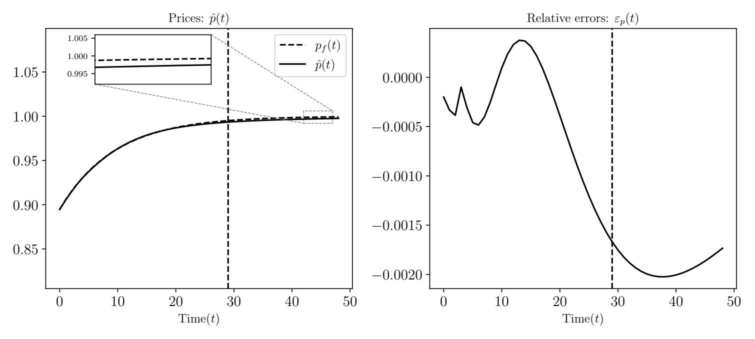
A reminder: in practice, given the $\hat{\mathcal{X}}$, we directly implement this as $p(\cdot;\theta) \in \mathcal{H}(\Theta)$ and fit with

$$\min_{\theta\in\Theta} \frac{1}{|\hat{\mathcal{X}}|} \sum_{t\in\hat{\mathcal{X}}} \left[p(t;\theta) - Gx(t) - \beta p(t+1;\theta)\right]^2 \tag{14}$$

Since law of motion is deterministic, given $x(0)$ we generate $x(t)$ with $x(t+1) = Ax(t)$ for $t \in \hat{\mathcal{X}}$

- The $\hat{\mathcal{X}}$ does not need to be contiguous and $|\hat{\mathcal{X}}|$ may be relatively small.
- Most important: no steady state calculated, nor large $T \in \hat{\mathcal{X}}$ required.

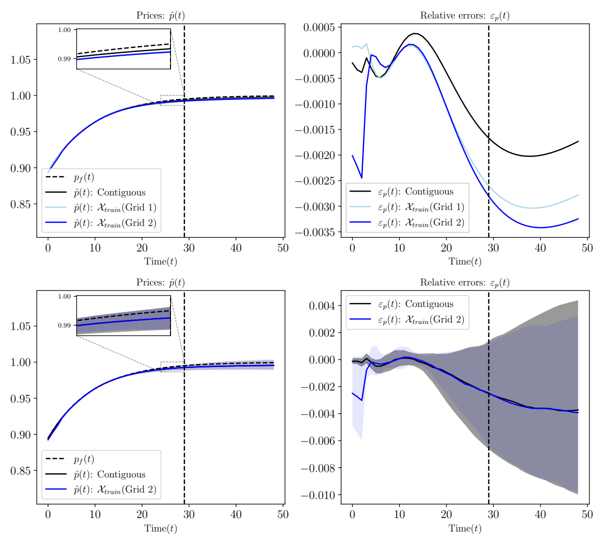
Results



1. **Pick** $\hat{\mathcal{X}} = \{0, 1, 2, ..., 29\}$ and $t > 29$ is "extrapolation" where $c = 0.01$, $g = -0.1$, and $y_0 = 0.8$.
2. **Choose** $p(t;\theta) = NN(t;\theta)$ where "NN" has 4 hidden layers of 128 nodes. $|\Theta| = 49.9K$ coefficients.
3. **Fit** using L-BFGS and PyTorch in just a **few seconds**. Could use Adam/SGD/etc.
4. Low generalization errors, even without imposing no-bubble condition.

$$\varepsilon_p(t) \equiv \frac{\hat{p}(t) - p(t)}{p(t)}.$$

Contiguous vs. sparse grid



- **Pick** $\hat{\mathcal{X}}(\text{Grid } 1) = \{0, 1, 2, 4, 6, 8, 12, 16, 20, 24, 29\}$ and $\hat{\mathcal{X}}(\text{Grid } 2) = \{0, 1, 4, 8, 12, 18, 24, 29\}$.
- Contrary to popular belief, can use **less grid points** relative to alternatives.
- Hypothesis verified, the solutions agree on the seen and **unseen** grid points.

Neoclassical growth in sequence space

Sequential formulation

$$\max_{\{c(t),k(t+1)\}_{t=0}^{\infty}} \quad \sum_{t=0}^{\infty} \beta^t u\left(c(t)\right) \tag{15}$$

$$\text{s.t.} \quad k(t+1) = z(t)^{1-\alpha} f\left(k(t)\right) + (1-\delta)k(t) - c(t) \tag{16}$$

$$z(t+1) = (1+g)z(t) \tag{17}$$

$$k(t) \geq 0 \tag{18}$$

$$0 = \lim_{T\to\infty} \beta^T u'\left(c(T)\right) k(T+1) \tag{19}$$

$$k_0, z_0 \text{ given} \tag{20}$$

- Preferences: $u(c) = \frac{c^{1-\sigma}-1}{1-\sigma}$, $\sigma > 0$, $\lim_{c\to 0} u'(c) = \infty$, and $\beta \in (0,1)$.
- Cobb-Douglas production function: $f(k) = k^{\alpha}$, $\alpha \in (0,1)$ before scaling by TFP z_t.
- Skip standard steps... Euler equation: $u'(c(t)) = \beta u'(c(t+1))\big[z(t+1)^{1-\alpha} f'(k(t+1)) + 1 - \delta\big]$.

Interpolation problem: without transversality condition

- A set of points in time $\hat{\mathcal{X}} = \{t_1, \ldots, t_{\max}\}$.
- A family of over-parameterized functions $k(\cdot;\theta) \in \mathcal{H}(\Theta)$.
- Generate $z(t)$ using the law of motion and $z(0)$, equations (17).
- Use the feasibility condition and define $c(t;k) \equiv z(t)^{1-\alpha} f\left(k(t)\right) + (1-\delta)k(t) - k(t+1)$.

In practice we minimize the Euler and initial conditions residuals:

$$\boxed{\min_{\theta\in\Theta}\left(\frac{1}{|\hat{\mathcal{X}}|}\sum_{t\in\hat{\mathcal{X}}}\lambda_1\Bigg[\underbrace{\frac{u'\big(c(t;k(\cdot,\theta))\big)}{u'\big(c(t+1;k(\cdot;\theta))\big)}-\beta\big[z(t+1)^{1-\alpha}f'(k(t+1;\theta))+1-\delta\big]}_{\text{Euler residuals}}\Bigg]^2 + \lambda_2\Big[\underbrace{k(0;\theta)-k_0}_{\text{Initial condition residuals}}\Big]^2\right)}$$

- λ_1 and λ_2 positive weights.

Interpolation problem: without transversality condition

- This minimization **does not contain** the transversality condition.
  - Without the transversality condition it has infinitely many minima.

- **No explicit** norm minimization.

- Does the implicit bias weed out the solutions that violate the transversality condition? **Yes**.

- **Intuition**: The solutions that violate the transversality condition are big functions with big derivatives.

  Let's analyze this more rigorously.

Interpolation formulation: min-norm mental model

$$\boxed{\begin{aligned} \min_{k \in \mathcal{H}} \quad & \|k\|_S && (21)\\ \text{s.t.} \quad & u'(c(t;k)) = \beta u'(c(t+1;k))\left[z(t+1)^{1-\alpha} f'(k(t+1)) + 1 - \delta\right] \quad \text{for } t \in \hat{\mathcal{X}} && (22)\\ & k(0) = k_0 && (23)\\ & 0 = \lim_{T\to\infty} \beta^T u'(c(T;k))k(T+1) && (24) \end{aligned}}$$

$$\boxed{c(t;k) \equiv z(t)^{1-\alpha} f\left(k(t)\right) + (1-\delta)k(t) - k(t+1)} \tag{25}$$

Where $z(t)$ for $t \in \hat{\mathcal{X}}$ is defined by $z(0)$ initial condition and recurrence $z(t+1) = (1+g)z(t)$.

Is the transversality condition still necessary? Case of $\mathbf{g = 0, z = 1}$

Sketch of the proof:

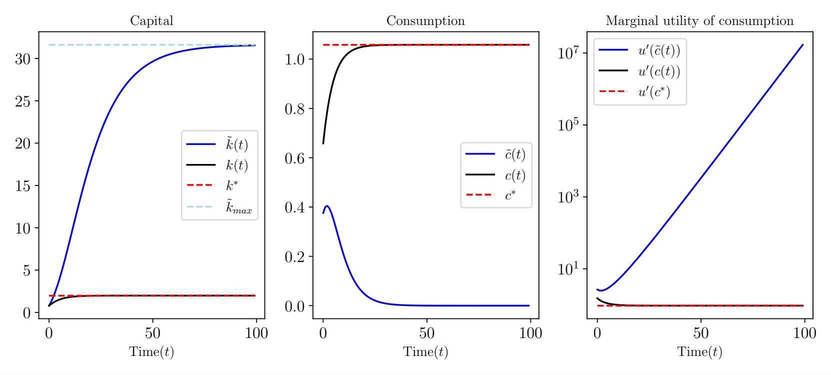
- Let $\{k(t), c(t)\}$ be the sequence of optimal solution.
- Let $\{\tilde{k}(t), \tilde{c}(t)\}$ be a sequence of solution that satisfy all the equations **except** transversality condition (24).

1. $\tilde{c}(t)$ approaches zero.
2. $\tilde{k}(t)$ approaches $\tilde{k}_{\text{max}} \equiv \delta^{\frac{1}{\alpha-1}}$, and $k(t)$ approaches $k^* \equiv \left(\frac{\beta^{-1}+\delta-1}{\alpha}\right)^{\frac{1}{\alpha-1}}$.
3. Both $\tilde{k}(t)$ and $k(t)$ are monotone. $\tilde{k}_{\text{max}} \gg k^*$. Therefore,

$$0 \leq \|k\|_S \leq \|\tilde{k}\|_S.$$

Is the transversality condition still necessary? Case of g = 0, z = 1

Example: the violation of the transversality condition.



- The solution that violate the transversality are associated with **"big"** capital path.
- The new objective of minimizing the norm, makes the transversality condition **redundant**.

Min-norm formulation: redundancy of transversality condition

Given the transversality condition is automatically fulfilled, one could solve

$$\begin{aligned} \min_{k\in\mathcal{H}} \quad & \|k\|_S \\ \text{s.t.} \quad & u'(c(t;k)) = \beta u'(c(t+1;k))\left[z(t+1)^{1-\alpha} f'(k(t+1)) + 1 - \delta\right] \quad \text{for } t \in \hat{\mathcal{X}} \\ & k(0) = k_0 \end{aligned}$$

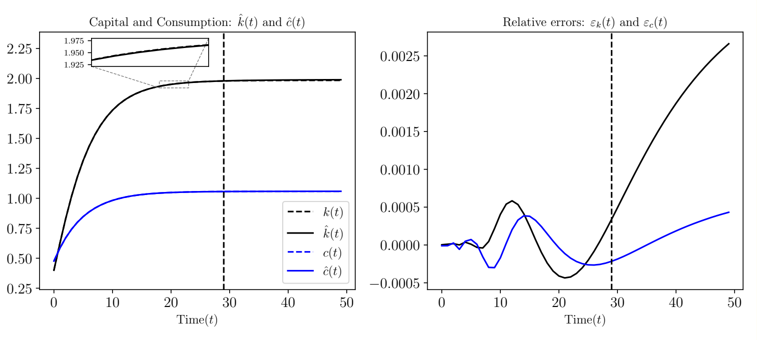
Reminder: in practice we solve

$$\min_{\theta\in\Theta} \left(\frac{1}{|\hat{\mathcal{X}}|} \sum_{t\in\hat{\mathcal{X}}} \lambda_1 \left[\frac{u'\big(c(t;k(\cdot,\theta))\big)}{u'\big(c(t+1;k(\cdot;\theta))\big)} - \beta\left[z(t+1)^{1-\alpha} f'(k(t+1;\theta)) + 1 - \delta\right] \right]^2 \right.$$

$$\left. + \lambda_2 \left[\underbrace{k(0;\theta) - k_0}_{\text{Initial condition residuals}} \right]^2 \right)$$

- $|\hat{\mathcal{X}}|$ may be relatively small, no steady state calculated, nor large $T \in \hat{\mathcal{X}}$ required.

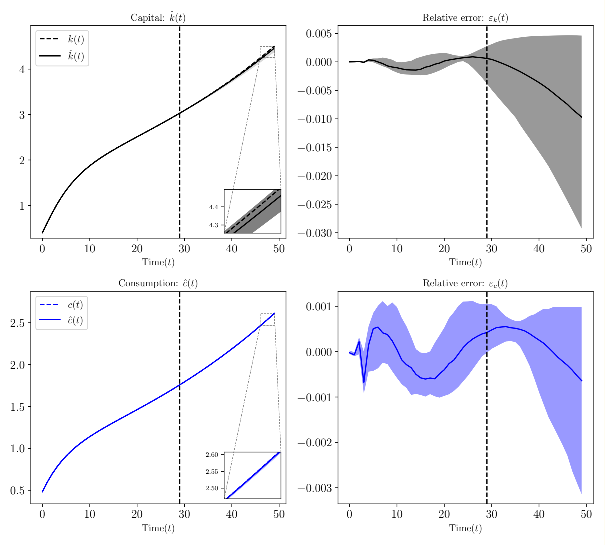
Results



1. **Pick** $\hat{\mathcal{X}} = \{0, 1, ..., 30\}$ and $t > 30$ is "extrapolation" $\alpha = \frac{1}{3}$, $\sigma = 1$, $\beta = 0.9$, $g = 0.0$, and $k_0 = 0.4$
2. **Choose** $k(t;\theta) = NN(t;\theta)$ where "NN" has 4 hidden layers of 128 nodes. $|\Theta| = 49.9K$ coefficients.
3. **Fit** using L-BFGS in just a **few seconds**. Comparing with value function iteration solution.
4. Low generalization errors, even without imposing the transversality condition.

Relative errors defined as $\varepsilon_c(t) \equiv \frac{\hat{c}(t)-c(t)}{c(t)}$, $\varepsilon_k(t) \equiv \frac{\hat{k}(t)-k(t)}{k(t)}$.

Growing TFP



- **Pick** same $\hat{\mathcal{X}}$ but now $g = 0.02$.
- **Choose** $k(t;\theta) = e^{\phi t} NN(t;\theta_{NN})$ where $\theta \equiv \{\phi, \theta_{NN}\} \in \Theta$ is the coefficient vector
  - Here we used economic intuition of problem to design the $\mathcal{H}(\Theta)$ to generalize better.
- Non-stationary but can figure out the BGP.
- Learns the growth rate: $\phi \approx \ln(1+g)$
- Economic insight leads to great extrapolation!

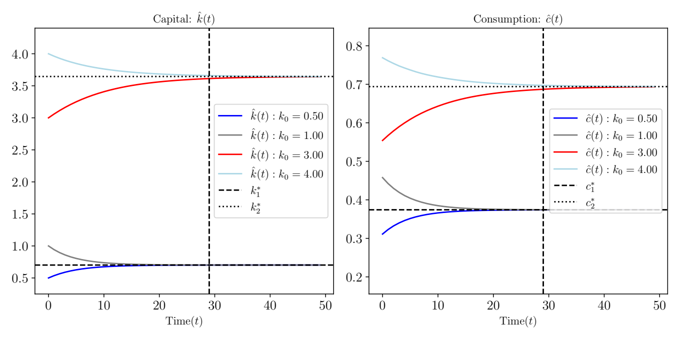
The neoclassical growth model with multiple steady states

Sequential formulation

$$
\begin{aligned}
\max_{\{c_t, k_{t+1}\}_{t=0}^{\infty}} \quad & \sum_{t=0}^{\infty} \beta^t u(c_t) \\
\text{s.t.} \quad & k_{t+1} = f(k_t) + (1-\delta)k_t - c_t \\
& k_t \geq 0 \\
& 0 = \lim_{T\to\infty} \beta^T u'(c_T) k_{T+1} \\
& k_0 \text{ given.}
\end{aligned}
$$

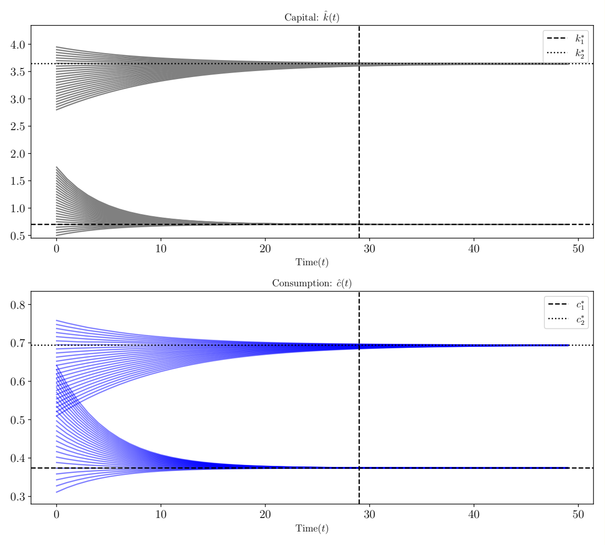
1. Preferences: $u(c) = \frac{c^{1-\sigma}-1}{1-\sigma}$, $\sigma > 0$, $\lim_{c\to 0} u'(c) = \infty$, and $\beta \in (0,1)$.
2. **"Butterfly production function"**: $f(k) = a \max\{k^{\alpha}, b_1 k^{\alpha} - b_2\}$, $\alpha \in (0,1)$:
   - There is a kink in the production function at $k^* \equiv \left(\frac{b_2}{b_1 - 1}\right)^{\frac{1}{\alpha}}$.
   - This problem has **two** steady states, k_1^* and k_2^* and their corresponding consumption levels c_1^* and c_2^*.

Results



1. **Pick** $\hat{\mathcal{X}} = \{0, \dots, 30\}$, $\alpha = \frac{1}{3}$, $\sigma = 1$, $\beta = 0.9$, $g = 0.0$, $a = 0.5$, $b_1 = 3$, $b_2 = 2.5$ and $k_0 \in \{0.5, 1.0, 3.0, 4.0\}$
2. **Choose** $k(t;\theta) = NN(t;\theta)$ where "NN" has 4 hidden layers of 128 nodes. $|\Theta| = 49.9K$ coefficients.
3. **Fit** using Adam optimizer.

Results: different initial conditions



- Different initial conditions in $k_0 \in [0.5, 1.75] \cup [2.75, 4]$.
- In the vicinity of k_1^* and k_2^* the paths converge to the right steady-states.
  - The implicit bias picks up the right path.
- Low generalization errors, even without imposing the transversality condition.

Conclusion

- Solving functional equations with deep learning is an extension of collocation/interpolation methods.
- With **massive over-parameterization**, optimizers tend to choose those interpolating functions which are not explosive and with smaller gradients (i.e., **inductive bias**).
- Over-parameterized solutions **automatically** fulfill **forward-looking** boundary conditions:
  - Shedding light on the convergence of deep learning based solutions in dynamic problems in macroeconomics.
- If we solve models with deep-learning without (directly) imposing long-run boundary conditions,
  - Short/medium-run errors are small, and long-run errors after **"we are all dead"** are even manageable.
  - Long-run errors do not affect transition dynamics even in the presence of **non-stationarity** and **steady-state multiplicity**.
  - Gives hope for solving high-dimensional models still disciplined by forward-looking economic assumptions.

Appendix

Definition (Bounded functions in N)

Let:

$$\mathcal{L}(M) \equiv \{y \in \mathbb{R}^N : |y_i| \leq M \ \forall i = 1, \ldots, N\}$$

be an N-dimensional hypercube in $\mathbb{R}^N$. A function $f : \mathbb{R}^N \to \mathbb{R}$ is bounded in N if for every M there exists K_M such that

$$\sup_{y \in \mathcal{L}(M)} |f(y)| < K_M,$$

where K_M is a constant that does not depend on N, but may depend on M.

- Example $f(y) = \frac{1}{N} \sum_{i=1}^N y_i \to \sup_{y \in \mathcal{L}(M)} |f(y)| < M$.
- To avoid $f(y) = \sum_{i=1}^N y_i \to \sup_{y \in \mathcal{L}(M)} |f(y)| < NM$.

⏩ back

Concentration of measure is the bless of dimensionality

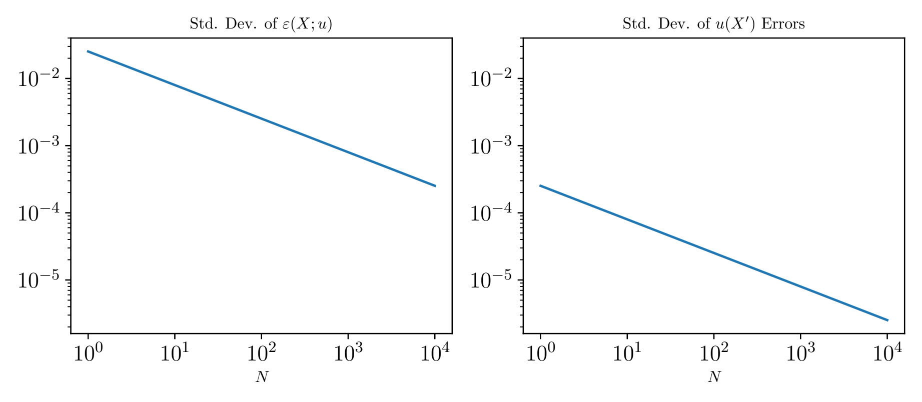
In the linear case we know the closed form solution for u

$$\boxed{\begin{aligned} \hat{\varepsilon}\left(X;u\right) - 0 &\sim \mathcal{N}\left(0, \frac{\sigma_\varepsilon^2}{N}\right) \\ u(\hat{X}') - \mathbb{E}\left[u(X') \mid \omega\right] &\sim \mathcal{N}\left(0, \frac{\sigma_u^2}{N}\right) \end{aligned}}$$

- Conditional expectation becomes constant as N gets large.
  - One **single Monte-carlo draw** of the idiosyncratic shocks is enough.

⏩ back

Analytic euler error due to the concentration of measure



Parameters

- $\gamma = 90$, $\beta = 0.95$, $\sigma = 0.005$, $\eta = 0.001$.

Implicit bias: More details

Let ψ_1 and ψ_2 be two differentiable function from a compact space $\mathcal{X}$ in $\mathbb{R}$ to $\mathbb{R}$ such that

$$\int_{\mathcal{X}} \left| \frac{d\psi_1}{dx} \right|^2 dx > \int_{\mathcal{X}} \left| \frac{d\psi_2}{dx} \right|^2 dx$$

then

$$\|\psi_1\|_S > \|\psi_2\|_S.$$

Recently shown the optimizers (first order e.g. SGD) regularize Sobolev semi-norms: Ma, Ying (2021).

⏩ Back