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

Mahdi Ebrahimi Kahou* Jesús Fernández-Villaverde† Sebastián Gómez-Cardona*
Jesse Perla* Jan Rosa*

October 18, 2023

Abstract

In the long run, we are all dead. Nonetheless, even when investigating short-run dynamics, models require boundary conditions on long-run, forward-looking behavior (e.g., transversality and no-bubble conditions). In this paper, we show how deep learning approximations can automatically fulfill these conditions despite not directly calculating the steady-state, balanced growth path, or ergodic distribution. The main implication is that we can solve for transition dynamics with forward-looking agents, confident that long-run boundary conditions will implicitly discipline the short-run decisions, even converging towards the correct equilibria in cases with steady-state multiplicity. While this paper analyzes benchmarks such as the neoclassical growth model, the results suggest deep learning may let us calculate accurate transition dynamics with high-dimensional state spaces, and without directly solving for long-run behavior.

Keywords: Dynamic programming; deep learning; machine learning; turnpike theory; stability; economic growth; transitional dynamics

*University of British Columbia, Vancouver School of Economics; and †University of Pennsylvania.

‡ We would like to thank Marlon Azinovic, Jess Benhabib, Doga Bilgin, David Childers, Hiro Kasahara, Vadim Marmer, Kevin Song, Michael Peters, and seminar participants at the Society for Economic Dynamics 2022 Meeting.

1 Introduction

> But this *long run* is a misleading guide to current affairs. *In the long run* we are all dead—*J.M. Keynes, A Tract on Monetary Reform (1923), p. 65.*

Computing the solution of dynamic equilibrium models usually requires ensuring that some economic assumption that rules out an explosive solution (e.g., transversality or no-bubble conditions) holds. These assumptions are variations of "boundary conditions" in ordinary and partial differential equations but motivated by the forward-looking behavior of economic agents. How these conditions appear and their concrete form varies from model to model. Still, they are always present, whether we are solving the deterministic or stochastic version of the model or formulating the problem in a sequential or recursive structure. Without these conditions, dynamic optimization problems are not well-posed and have a multiplicity of solutions, many of which imply unplausible descriptions of economic behavior.

Unfortunately, these forward-looking boundary conditions are a fundamental limitation when we want to increase the model's dimensionality (e.g., raising the number of state variables). Without forward-looking boundary conditions, we could compute large models written in a sequential setup by solving the associated initial value problem or simulating backward-looking recursive stochastic difference equations. In fact, researchers in natural sciences and engineering routinely solve stochastic differential equations with initial values in the millions of equations.

Why do boundary conditions make life difficult? Because they force us to solve the model over a wide range of possible values of the state variables. For example, conditions for recursive formulations manifest as requiring accurate solutions for arbitrary values of the state variables, even though one may only care about the solution from a single initial condition.[1]

But the fact that we might only care about the dynamics of the economy given some initial conditions suggests an alternative: can we avoid calculating the steady-state, balanced growth path (BGP), or stationary distribution of the model, which are never reached, and still have accurate short- and medium-run dynamics disciplined, in their economic behavior, by the boundary conditions? If the answer is yes, we can bypass the most complicated part of solving a dynamic model, which opens the door to the solution of vastly more complicated macroeconomic models with transitional dynamics, endogenous growth, aggregate shocks, and agent distributions.

The contribution of our paper is to show how, why, and when deep learning solutions to macroeconomic models let us achieve this tradeoff. More precisely, we will document numerically and provide theoretical explanations of how the minimum-norm solutions delivered by deep learning automatically fulfill the long-run boundary conditions for forward-looking agents required in many dynamic models. In other words, deep learning allows us to emphasize accurate short- and

[1] Recall that, in a recursive formulation, we compute the value and policy functions for all values of the state variables, regardless of whether or not the equilibrium dynamics of the economy will travel through those values.

medium-run dynamics and worry less about precision in the long run, while still ensuring that agent decisions are disciplined asymptotically.

Since we want to understand when and why this happens, we investigate this tradeoff with standard benchmark models in macroeconomics, the linear asset pricing and the neoclassical growth models, rather than jumping to complicated models where the state-of-the-art algorithms are heuristic. Both the linear asset pricing and the neoclassical growth models are well understood, can be analyzed theoretically, and have benchmark numerical solutions. Furthermore, we can study cases with multiple steady states in a variation of the neoclassical growth model. Even in the presence of this multiplicity in long-run boundaries, deep learning will impose the correct short-run behavior without requiring any information about multiplicity. Finally, we show how our theoretical understanding of why these techniques work helps in designing deep learning architectures that solve for the short-term dynamics in the presence of a long-run BGP, despite neither providing any information to the architecture to suggest a BGP exists nor training it on grid points that would make it clear the geometric convergence to the BGP.

The "inductive bias" of deep learning. Collocation/interpolation-style solution methods for dynamic equilibrium models, the canonical global solution methods in macroeconomics, follow a straightforward template. First, the researcher chooses a parametric class of approximating functions on some compact region of the state space and a corresponding set of points one wishes to fit the approximation exactly (e.g., Chebyshev polynomials for the investment policy and Chebyshev collocation nodes as the grid points). Next, the researcher solves a system of equations for all the model equations (e.g., optimality conditions, budget and resource constraints, etc.), usually organized as residuals, at the grid points to find the coefficients for the approximating function. When there are more coefficients than grid points, the problem is underdetermined. Still, one can numerically minimize the sum of the squared residuals by interpolating (i.e., achieve zero residuals at all grid points). In those cases, many combinations of parameters yield zero residuals and interpolate the grid points.[2]

The methods that we employ in this paper follow that same approach, but rather than approximating with polynomials, we use flexible "deep learning"-style functional forms (e.g., neural networks).[3] The first defining feature of deep learning is massive over-parameterization, where the number of coefficients of the class of approximations is often orders of magnitude more than the number of grid points.

[2]In the case of sequential formulations, the grid points could be points in time. In continuous time, there is a tight connection between finite-difference solutions to differential equations, interpolation, and quadrature.

[3]In this paper, we do not need to radically change the algorithm from the familiar collocation template because we are analyzing low-dimensional models. Consequently, we can choose grid points without concerns about whether we are covering relevant regions of the state space. High-dimensional models require algorithms to find appropriate grid points for the fitting process. See Ebrahimi Kahou et al. (2021) for one variation that uses symmetry to simplify the number of required grid points in high dimensions.

The second defining feature of deep learning, especially important in higher dimensions, is that the functional approximations are organized into nested layers (i.e., "deep") rather than the flat structure of polynomials or splines. Given the models in this paper are low-dimensional, the details of the approximation class are less important (unlike in Ebrahimi Kahou et al., 2021, where a core contribution is a design that enforces symmetry and enables high-dimensional expectations). While "neural networks" is a broad term encompassing everything from linear functions to complicated functions used in computer vision, experience shows they are usually better with more structure, nested layers of function composition, and many parameters.[4] Deep nesting allows for exponential growth in the representations of the inputs to the network while having only a linear growth in computational cost.

While the algorithms used in collocation and deep learning might seem narrowly focused on fitting the function at the grid points, this is just a means to an end. The goal is not interpolation at the grid points (which are usually a measure-zero subset of the state space) but rather *generalization* performance of the function across much larger regions of the state space (e.g., some expected value of residuals over the domain).

To understand why deep learning generalizes well, one must look beyond the structure of the approximation class (i.e., the nested layers of activation functions). Instead, the key behind good generalization performance lies in the optimizers that minimize residuals. Computer science researchers have shown that an essential ingredient responsible for the empirical success of deep learning is its use of gradient-based optimization algorithms.[5] These include a variety of methods familiar to economists (e.g., LBFGS and gradient descent) as well as algorithms essential when there are a huge number of grid points (e.g., stochastic gradient descent and ADAM).

Why does combining a massive number of parameters and a class of gradient-based optimization lead to good generalization performance rather than overfitting and multiplicity? Because, in this situation, theory and practice show us that we have an "inductive" (or "implicit") bias towards simple smooth interpolating solutions (see Belkin, 2021, for details and a discussion of how an appropriate sense of functional smoothness means the "simplest" function). Furthermore, in many cases, the algorithm will reliably converge to a unique solution.[6] In other words, this (roughly) unique solution satisfies has a built-in *Occam's Razor*: it is the interpolating solution

[4] An example for approximating $f : \mathbb{R} \to \mathbb{R}$ would be a collection of coefficients $\theta \equiv \{b_1, W_1, b_2, W_2, b_3, W_3, b_4\}$ and $\hat{f}(x;\theta) = \tanh(\tanh(\tanh(W_1 \cdot x + b_1) \cdot W_2 + b_2) \cdot W_3 + b_3) + b_4$ where $\tanh(\cdot)$ is applied pointwise, $b_1 \in \mathbb{R}^N, W_1 \in \mathbb{R}^{N\times 1}, b_2 \in \mathbb{R}^N, W_2 \in \mathbb{R}^{N\times N}, W_3 \in \mathbb{R}^{1\times N}, b_3 \in R$, and $b_4 \in \mathbb{R}$. The number of nested functions (i.e., layers), the sizes of the vectors within those (i.e., nodes), and the particular nonlinear functions like tanh (i.e., what they call activation functions) are all flexible. Many variations exist that exploit problem structure (e.g., one may want to exponentiate the last layer to ensure it is always positive).

[5] The role of deep-learning frameworks, such as `JAX` or `PyTorch`, is to provide a flexible environment for specifying different approximations and compute gradients to aid optimization algorithms that minimize residuals (see Childers et al., 2022, for a broader discussion of automatic differentiation for economists).

[6] There is a continuum of possible coefficients that achieve the same interpolating solution up to numerical accuracy. Uniqueness here refers to the approximating functions using those coefficients, not the coefficients themselves.

that maximizes functional smoothness.

Formally, deep learning tends to choose the interpolating solution with the minimum norm, where the norm is such that the smoothest functions subject to interpolating the grids have the smallest norms.[7] Not only do these smooth solutions generalize much better, but they also minimize residuals faster and more reliably. We will explore these concepts more rigorously in Section 6.[8]

The inductive bias is the core of our argument, and it appears even if we have a massive degree of multiplicity in the possible functions that could interpolate the grid points. If one solves the problem with completely different initial conditions for the coefficients, the approximating function will end up often numerically indistinguishable, albeit with different coefficients. That is, deep learning usually converges to roughly the same approximation despite the risks of overfitting and multiplicity. Importantly, this approximation leads to good generalization within our models. Since the inductive bias of deep learning is frequently aligned with the functions that generalize best for models in macroeconomics (which also tend to be simple and smooth), we can beget accurate solutions across large regions of a state space with very few grid points.

At a fundamental level, nothing in the previous paragraphs should be mysterious to economists, since there is a deep connection between the inductive bias and a classic literature in economics: turnpike theory (McKenzie, 1976). We will argue now that the inductive bias gives a solution that satisfies a stronger turnpike result, an "on-ramp" result.

Turnpikes and on-ramps. Turnpike theorems teach us that models with finite but long horizons usually have the same properties as infinite-horizon models. Consider the problem of driving from the Vancouver School of Economics at the University of British Columbia to the department of economics at the University of Pennsylvania in Philadelphia. We can think about this trip as consisting of three segments: the leg from the origin to the on-ramp of the turnpike, the travel along the turnpike itself (where one would spend almost the entire drive because it is the lowest-cost path between the cities), and the choice of the off-ramp and travel to the destination. A critical insight of turnpike theory is that if we are primarily interested in the short- and medium-run dynamics of travel starting at Vancouver, we can ignore the details of what to do when we arrive in Philadelphia. Instead, first focus on finding what the solution would look like if we drove on the turnpike forever (i.e., calculate the stationary solution and policies for the infinite-horizon problem); next, solve the transition dynamics to get from the University of British Columbia to the turnpike (i.e., the transitional dynamics using the turnpike as a boundary condition), and

[7] This phenomenon also occurs with OLS whenever the model is underdetermined (see Liang and Rakhlin, 2020). The intuition in the case of OLS is that these methods choose among the interpolating solutions those with the minimum L^2 norm, which can be interpreted as finding the equivalent solution to a ridge regression with an asymptotically vanishing regularization penalty.

[8] See Belkin et al. (2019) for a readable survey of the "double-descent" phenomenon and how it corrects the classic variance-bias tradeoff in statistics.

ignore the destination dynamics completely. Hence, we have simplified the problem by avoiding the need to work through the details of which off-ramp and local roads to take in Philadelphia by solving an infinite-horizon problem where we drive on the turnpike forever. While it is often proven with deterministic, representative agent models, the basic intuition of turnpike theory is robust: the present discounted value of optimal behavior at the end of the problem is trivially small when the time horizon is sufficiently long and, thus, "getting it right at the end" is close to irrelevant.[9] Expanding on the turnpike narrative: even if you need to switch drivers when they are tired, change lanes when there are random traffic arrivals, and occasional pit stops—most driving will be local to a tight ergodic distribution.[10] Similarly, even if the turnpike is a hill with exponentially growing altitude (i.e., a BGP in a growth model), one needs to stay on it.

But while turnpike theory simplifies models by avoiding solving for the dynamics from the off-ramp to the destination, it still requires calculating the steady-state, BGP, or ergodic distribution of the infinite-horizon problem to find the dynamics local to the origin. Also, one must solve for the policies and decisions along the turnpike. One needs to characterize the direction one drives, the policy for changing lanes in traffic, and the frequency of coffee breaks and ensure that the policy does not accidentally explode off of the turnpike. Even after characterizing this fragile steady-state, BGP, or ergodic distribution, one needs to use it as a boundary condition to solve the transitional dynamics of the first leg of the journey, from the origin to the on-ramp.

Returning to our paper, our main contribution is to take the spirit of the turnpike theory one step further. Rather than just ignoring the dynamics local to the destination (i.e., the off-ramp), deep learning allows us to avoid characterizing the travel or the location of the turnpike itself and solely focus on the dynamics starting at the origin (i.e., solve models anticipating the on-ramp will correctly lead to the turnpike). Loosely speaking, we have an "on-ramp result" instead of a "turnpike" one.

Deep learning makes this "on-ramp result" possible because the location of the turnpike is tightly connected with it being the non-explosive solution to the agents' optimization problems. As we argued above, deep learning solutions have an inductive bias toward choosing simple and smooth solutions, which are precisely the non-explosive solutions that satisfy the boundary conditions disciplining forward-looking behavior. Hence, the solutions from deep learning are consistent by construction with the features of the turnpike. Deep learning methods also pick which turnpike to go to in the case of multiplicity because, if there are two turnpikes it could go

[9] Indeed, finding the optimal path through the local roads in Philadelphia has next-to-no impact on the total driving time between the University of British Columbia and the University of Pennsylvania. According to Google Maps, local driving time in Philadelphia once off the highway only accounts for 5 minutes of a total of 45 hours of driving.

[10] For instance, see Marimón (1989) for establishing a turnpike theorem in stochastic equilibrium growth models. More recently, Maliar et al. (2020) use turnpike theorems to propose a method for solving non-stationary Markov models such as the stochastic neoclassical growth model with labor-augmenting technological progress, a stochastic version of the non-stationary model we solve later in this paper. Also, many methods for solving OLG models have a turnpike flavor, such as in Auerbach et al. (1987).

to, the roads to the "closest" turnpike are also the shortest.[11] Deep learning finds the appropriate roads to a BGP turnpike if one sets up the approximation to consider exponential scaling: the smoothest path is the one with the correct BGP rate and the smoothest detrended trajectories. The wrong BGP would lead to explosive trajectories that are eliminated by the inductive bias.

Literature. While there are many uses of machine learning in economics, such as in creating new data sources from text and images (e.g., Shen et al., 2021, and Gentzkow et al., 2019) and econometrics (e.g., Athey and Imbens, 2019), in this paper, we narrowly focus on solutions to standard macroeconomic models in the most traditional sense and without any behavioral approximations (e.g., not using Krusell and Smith, 1998,-style approximations or and model-free reinforcement learning such as in Calvano et al., 2020).

Within macroeconomics, there is an active literature solving dynamic models with machine learning that can be traced back to Duffy and McNelis (2001). Some more recent papers that solve the full model rather than a behavioral approximation of it are Azinovic et al. (2022), Ebrahimi Kahou et al. (2021), Duarte (2018), and some of the examples in Maliar et al. (2021). Other papers make tradeoffs between solving the full model and making behavioral assumptions, such as Fernández-Villaverde et al. (2019), which uses deep learning with a Krusell and Smith (1998)-style behavioral assumption, and some of the examples in Maliar et al. (2021) which use a large number of discrete agents to approximate the evolution equation of the distribution. None of these papers directly impose transversality conditions, nor do they calculate the stationary distribution analytically. Since transversality conditions are necessary for these models to be well-posed, it is unclear which interpolating solution each algorithm would converge to. In particular, many solutions have Euler or Bellman residuals approximately zero and do not fulfill long-run forward-looking behavior and stability (see Section 5 for some examples). Our paper provides insights on which interpolating solution those papers might converge towards and how to design the deep learning approximations to ensure the correct solution is attained. Our results can help explain why the solutions find the correct solution despite not directly imposing transversality in high-dimensional models with closed-form baselines, such as Ebrahimi Kahou et al. (2021).

Outline. The rest of the paper is organized as follows. Section 2 provides an introductory example with the canonical linear asset pricing, showing both numerically and theoretically why deep learning methods converge to the solution fulfilling the no-bubble condition even when ill-posed. Section 3 continues these examples by analyzing the sequential formulation of the neoclassical growth model, where forward-looking behavior requires a transversality condition to be well-specified. There, we show how deep learning leads to solutions that automatically fulfill

[11] However, unlike models calculating eigenvalues that can sharply characterize the basins of attraction for each turnpike, when starting at a point where the two highways are close to equidistant, these methods may become confused and take some wrong turns before ending up at the wrong steady-state.

transversality despite not directly solving for the steady-state, even in cases with steady-state multiplicity and non-stationarity. The case with saddle paths is analyzed in Section 4, which deals with the recursive formulation of the neoclassical growth model (with clear extensions to the stochastic case). We show in Section 5 how transversality conditions may be violated even with low Euler or Bellman residuals. We wrap up with Section 6 elaborating on the theoretical connections to the deep learning literature and Section 7 offering some final remarks.

2 Linear asset pricing

We start our investigation with a linear asset pricing model in sequential form. We choose this model because it clearly illustrates the connection between the implicit bias of over-parameterized functions and forward-looking boundary conditions.

Consider an asset that pays a dividend stream with a linear form:

$$\begin{aligned} y(t+1) &= c + (1+g)y(t) \\ y_0 &\text{ given,} \end{aligned}$$

where $c \geq 0$, $g \geq -1$ and $t = 0, \ldots, \infty$.

Although this problem is discrete in nature, we use a function notation (e.g., $y(t)$ instead of y_t). In order to solve sequential models of this sort we embed the discrete solution within a continuous function. For instance, we use deep neural networks to represent $p : \mathbb{R}_+ \to \mathbb{R}$ instead of $p : \mathbb{N} \to \mathbb{R}$. However, we still evaluate $p(t)$ at integer values. This approach is desirable for a couple of reasons. First, from a practical perspective, deep neural networks are defined on a continuous domain. Second, it enables us to use sparse grids, which can be used to assess the interpolation and extrapolation performance of the solutions.

We rewrite this problem as a linear space model:

$$\begin{aligned} x(t+1) &= Ax(t) \\ y(t) &= Gx(t), \end{aligned}$$

where:

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

The price $p_f(t)$, based on the fundamentals, is the present discounted value of the dividend stream:

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

where β is the discount factor, and $\mathbf{I}$ is a two-dimensional identity matrix. Here, we assume $\beta \in (0,1)$ and $\beta(1+g) < 1$. The second assumption ensures the convergence of the infinite sum in equation (1). We can also write the evolution of the price in a recursive way:

$$p(t) = Gx(t) + \beta p(t+1) \tag{2}$$

$$x(t+1) = Ax(t) \tag{3}$$

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

Equation (2) tell us that the price of an asset that pays a stream of dividends $\{y(t+j)\}_{j=0}^{\infty}$ is equal to the dividend the agent receives at time t plus the discounted price of owning the asset at time $t+1$. Although for a given initial condition x_0, the price based on the fundamentals $p_f(t)$ satisfies equations (2)-(4), this system of equations do not form a well-posed problem and there are infinitely many solutions of the form:

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

See Brunnermeier (2017) for a derivation of these solutions and their interpretation. The initial condition x_0, uniquely determines $p_f(t)$. However, ζ is not uniquely determined. In order to have a well-posed problem, we require a no-bubble condition that eliminates any solution that grows faster or at the same rate as β^{-t}. With this condition, the well-posed problem becomes:

$$p(t) = Gx(t) + \beta p(t+1) \tag{6}$$

$$x(t+1) = Ax(t) \tag{7}$$

$$0 = \lim_{T \to \infty} \beta^T p(T) \tag{8}$$

$$x_0 \text{ given,} \tag{9}$$

where equation (8) is the no-bubble condition.[12]

2.1 Interpolating solution

In this setup, prices are a function of time. Hence, their is defined as $\mathcal{X} \equiv \mathbb{R}_+$. As a generalization of the collocation approach one can pick a space of parametric functions $\mathcal{H}(\Theta)$,

[12] Sometimes the no-bubble condition is referred to as the transversality condition, see Brunnermeier (2017). However, to avoid confusion we reserve that term for the neoclassical growth model.

where $\Theta \equiv \{\theta_1, \ldots, \theta_M\}$ represents the set of parameters. For instance, in a standard collocation method $\mathcal{H}(\Theta)$ can be the space of Chebyshev polynomials of degree M. The interpolating function $p(\cdot;\theta) : \mathcal{X} \to \mathbb{R}$ belongs to $\mathcal{H}(\Theta)$. In order to represent the domain $\mathcal{X}$, one can pick a finite set of grid points $\mathcal{X}_{\text{train}} \subset \mathcal{X}$. For instance, in a standard Chebyshev collocation, $\mathcal{X}_{\text{train}}$ is the set of Chebyshev nodes. Generally, for sequential models such as this, the grid is a set of points in time:

$$\mathcal{X}_{\text{train}} \equiv \{t_1, \ldots, t_N\}.$$

The no-bubble condition is a boundary condition at infinity. One can approximate the infinity by choosing a very large point in time, denoted by T. Usually, T is much larger than the largest point in $\mathcal{X}_{\text{train}}$ (i.e. $T \gg t_N$). Given a space of parametric functions $\mathcal{H}(\Theta)$, a grid $\mathcal{X}_{\text{train}}$, and a very large point in time T, one can find the interpolating price function by solving:

$$\min_{\theta \in \Theta} \left[\frac{1}{|\mathcal{X}_{\text{train}}|} \sum_{t \in \mathcal{X}_{\text{train}}} \left(p(t;\theta) - Gx(t) - \beta p(t+1;\theta) \right)^2 + \underbrace{\left(\beta^T p(T;\theta) \right)^2}_{\text{Is this necessary?}} \right], \tag{10}$$

where $x(t)$ is evaluated by the law of motion for x

$$x(t) = A^t x_0 \quad \text{for } t \in \mathcal{X}_{\text{train}}. \tag{11}$$

The first term in the minimization represents the residuals of the recursive formulation for the prices (i.e., equation 6) and the second term is a finite approximation of the no-bubble condition (i.e., equation 8).

In a standard collocation method, where the number of parameters is equal to the number of grid points, this problem can be solved exactly as a system of equations with a boundary condition at T. Since the interpolating solution depends on T, the solutions of the minimization problem described in (10) defines a sequence of solutions indexed by T.

Choice of $\mathcal{X}_{\text{train}}$: in high-dimensional cases the choice of $\mathcal{X}_{\text{train}}$ is very crucial, however in this paper we focus on low-dimensional cases. Although this choice is not very crucial for this study, we provide the results for different grids. In this paper we are interested in the generalization power of the solutions. Therefore, we also evaluate the solutions outside of $\mathcal{X}_{\text{train}}$. For instance, in sequential models we are interested in the behavior of the solutions for $t > t_N$.

Is the no-bubble condition necessary? As discussed before, without the no-bubble condition the linear asset pricing model has infinitely many solutions of the form expressed in equation (5). Each solution corresponds to a different value for ζ. Therefore, in the absence of the second term (finite approximation of the no-bubble condition), the optimization problem described in (10), has infinitely many solutions that achieve the zero of the objective function.

In the next section we establish how using over-parameterized functions, where the number of parameters is larger than the number of grid points, yields convergence to the solution based on the fundamentals without worrying about the no-bubble condition.

Therefore, using over-parameterized functions we can drop the approximate no-bubble condition and solve

$$\min_{\theta\in\Theta} \frac{1}{|\mathcal{X}_{\text{train}}|} \sum_{t\in\mathcal{X}_{\text{train}}} \left(p(t;\theta) - Gx(t) - \beta p(t+1;\theta)\right)^2, \tag{12}$$

to obtain an interpolating solution that satisfies the no-bubble condition.

2.2 Minimum-norm interpretation and no-bubble condition

In this paper we use a space of over-parameterized functions (i.e., deep neural networks) to solve the minimization problem described in (12). An over-parameterized function $p(t;\theta)$ is characterized by M parameters $\{\theta_1, \ldots, \theta_M\}$ where M is larger then the number of grid points, i.e., $M > N$. In practice the number of parameters is much larger than the number of grid points. Since $M \gg N$, over-parameterized functions and their corresponding optimization algorithms can provide exact interpolating solutions. For these solutions the objective function attains a zero value.

Since the number of parameters is much larger than the number of grid points, there are too many degrees of freedom. Therefore, there are many arrangements of the parameters that can achieve an exact interpolation. It has been observed and in some cases proved that the optimization algorithms used in over-parameterized interpolation problems have an implicit bias toward a specific class of functions. Recently, it has been proposed that this property can be interpreted as minimizing a new objective function on the space of parametric functions $\mathcal{H}(\Theta)$ subject to exact interpolation at all the points on the grid. We call this minimum-norm interpretation of the solutions of an over-parameterized interpolation. Section 6 provides a comprehensive background and discussion on this robust phenomena.

The minimum-norm interpretation of the solutions of the minimization problem described in (10) for the liming case of $T \to \infty$ can be written as

$$\min_{p(\cdot;\theta)\in\mathcal{H}(\Theta)} \quad \|p(\cdot;\theta)\|_S \tag{13}$$

$$\text{s.t. } p(t;\theta) - Gx(t;\theta) - \beta p(t+1;\theta) = 0 \quad \text{for } t \in \mathcal{X}_{\text{train}} \tag{14}$$

$$0 = \lim_{T\to\infty} \beta^T p(T;\theta) \tag{15}$$

where $\mathcal{H}(\Theta)$ is a space of over-parameterized functions, $\mathcal{X}_{\text{train}}$ is a set of points in time (i.e., $\mathcal{X}_{\text{train}} = \{t_1, \ldots, t_N\}$), and $x(t)$ is evaluated by the exogenous law of motion $x(t+1) = Ax(t)$,

given the initial condition x_0.

What is the this new objective function? The new objective function $\|\cdot\|_S$ is a semi-norm[13] from a space functions (for this case $\mathcal{H}(\Theta)$) to real numbers satisfying the following assumption:

Assumption 1. *Let Ψ_1 and Ψ_2 be two differentiable function from a compact space $\mathcal{X}$ in $\mathbb{R}$ to $\mathbb{R}$ such that:*

$$\int_{\mathcal{X}} \left|\frac{d\Psi_1}{ds}\right|^2 ds > \int_{\mathcal{X}} \left|\frac{d\Psi_2}{ds}\right|^2 ds,$$

then:

$$\|\Psi_1\|_S > \|\Psi_2\|_S.$$

Moreover, since $\|\cdot\|_S$ is a semi-norm, it satisfies the triangle inequality:

$$\|\Psi_1 + \Psi_2\|_S \leq \|\Psi_1\|_S + \|\Psi_2\|_S.$$

This is an assumption on the implicit bias of the over-parameterized interpolation and their optimization algorithms. Intuitively, this assumption states that among two interpolating solutions on a domain $\mathcal{X}$, the bias is toward the one with smaller derivatives. In other words, the bias is toward the smoother one. Through the rest of the paper we assume Assumption 1 holds. See Section 6 for a discussion of the validity of this assumption and recent advances in the statistics and optimization theory regarding this assumption.

As shown in equation (5) any solution that violates the no-bubble condition is associated with a non-zero ζ. In the following proposition we establish that minimizing $\|\cdot\|_S$ automatically makes the no-bubble condition redundant. Intuitively, due to the explosive term β^{-t} $(0 < \beta < 1)$, all the solutions that contain a bubble term $(\zeta > 0)$ have bigger derivatives than the fundamental price $p_f(t)$. More formally for any compact interval of the form $[0, T]$:

$$\int_0^T \left|\frac{dp}{dt}\right|^2 dt > \int_0^T \left|\frac{dp_f}{dt}\right|^2 dt. \tag{16}$$

We only focus on prices that are positive for all t, this excludes solutions with $\zeta < 0$.

[13] For a given space of function $\mathcal{H}$, a semi-norm $\|\cdot\|_S : \mathcal{H} \to \mathbb{R}$ is a mapping that for all functions ψ_1 and ψ_2 in $\mathcal{H}$ satisfies

1. Triangle inequality: $\|\psi_1 + \psi_2\|_S \leq \|\psi_1\|_S + \|\psi_2\|_S$, and
2. Absolute homogeneity: $\|\gamma\psi_1\|_S = |\gamma|\|\psi_1\|_S$ for all $\gamma \in \mathbb{R}$.

Proposition 1. *Let $\mathcal{P}$ be the set of all the solutions of equations (6) and (7) for a given x_0:*

$$\mathcal{P} \equiv \{p(t) : p(t) = p_f(t) + \zeta\beta^{-t}, \forall\zeta \in \mathbb{R}_+\},$$

where $p_f(t)$ is defined by equation (1). Let $\|\cdot\|_S$ be a norm that satisfies Assumption 1, then

$$\operatorname*{argmin}_{p\in\mathcal{P}} \|p\|_S = p_f.$$

In other words the minimum of $\|p\|_S$ is attained when $\zeta = 0$.

Proof. Since $\zeta \geq 0$, then by the triangle inequality and Assumption 1:

$$\|p_f\|_S \leq \|p\|_S \leq \|p_f\|_S + \zeta\|\beta^{-t}\|_S.$$

□

This proposition establishes that the price based on the fundamentals has the lowest semi-norm.[14] Therefore, over-parameterized interpolating solutions have a bias toward $p_f(t)$ on $\mathcal{X}_{\text{train}}$ and automatically satisfy the no-bubble condition. Therefore, the optimization problem described in (12) is enough to find the price based on the fundamentals without imposing any explicit regularity regarding the long-run behavior.

2.3 Results

Figure 1 shows the result of the minimization problem described in (12) for $\beta = 0.9$, $c = 0.01$, $g = -0.1$, and $y_0 = 0.08$.

In this experiment $\mathcal{X}_{\text{train}} = \{0, 1, 2, \ldots, 29\}$ and $\mathcal{X}_{\text{test}} = \{0, 1, 2, ..., 49\}$.

We approximate the price $p(t;\theta)$ using a deep neural network with four hidden layers, each with 128 nodes. Each hidden layer uses Tanh and the output layer uses Sofplus as activation functions. The dashed vertical lines separate the interpolation from the extrapolation region. To be more specific, the interpolation region is defined as $\mathcal{X}_{\text{train}}$ and the extrapolation as the points in $\mathcal{X}_{\text{test}}$ that are not in $\mathcal{X}_{\text{train}}$. The left panel shows the price paths. The price based on the fundamentals, denoted by $p_f(t)$, is calculated using equation (1), and $\hat{p}(t)$ shows the approximate path (or approximate solution). More specifically, $\hat{p}(t) \equiv p(t;\theta^*)$, where

$$\theta^* \equiv \operatorname*{argmin}_{\theta\in\Theta} \sum_{t\in\mathcal{X}_{\text{train}}} \left(p(t;\theta) - Gx(t) - \beta p(t+1;\theta)\right)^2.$$

[14] What is required in this proof is triangle inequality and an innocuous assumption that if $p(t) > p_f(t)$ for all $t \in \mathcal{X}$ then

$$\|p\|_S > \|p_f\|_S.$$

Therefore, Assumption 1 is too strong for this proof and can be relaxed.

The right panel shows the relative error between the approximate price and price base on the fundamentals defined as:

$$\varepsilon_p(t) \equiv \frac{\hat{p}(t) - p_f(t)}{p_f(t)} \quad \text{for } t \in \mathcal{X}_{\text{test}}.$$

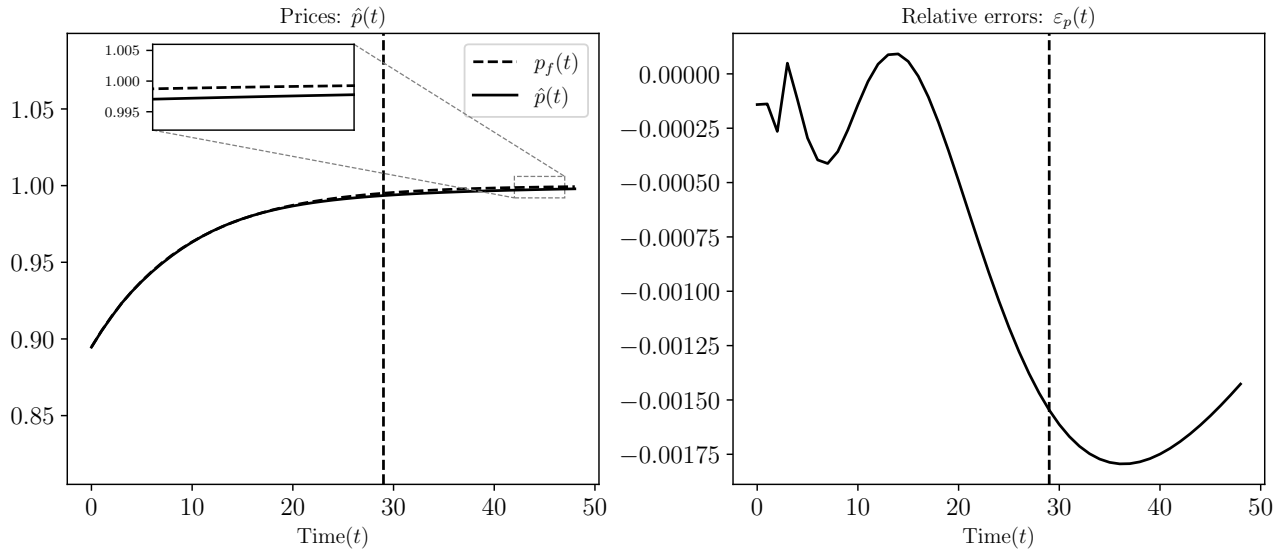

Figure 1: Comparison between the price based on the fundamentals and the price approximated by a deep neural network for the sequential linear asset pricing model. In the left panel, the solid curve shows the approximate price and the dashed curve shows the price based on the fundamentals. The right panel shows the relative errors between the approximate price path and the price based on the fundamentals. The dashed vertical lines separate the interpolation from the extrapolation region.

The results of this experiment are multi-fold. First, we can achieve an accurate solution based on the fundamentals and eliminate bubble solutions simply by relying on the implicit bias of deep neural networks. Second, within the grid points $\mathcal{X}_{\text{train}}$, the short- and medium-run dynamics are accurate and it is not impaired by the long-run errors (at most -0.05% relative error). Third, the extrapolation errors are very small which can be explained by the smoothness imposed by deep neural networks and the choice of $\mathcal{X}_{\text{train}}$. Here, we used a large time horizon in $\mathcal{X}_{\text{train}}$, i.e., $t_N = 29$. The prices in the last few points of the grid are very close to the asymptotic value. Therefore, the deep neural network learns that for large values of t the price is almost a constant and stays very close to that value even outside of $\mathcal{X}_{\text{train}}$.

Contiguous vs. Sparse Grid: there is a common belief that deep neural networks are only suitable for high-dimensional environments with an abundance of data (grid points). In this experiment we show that they can also be very useful when the grid contains few points.

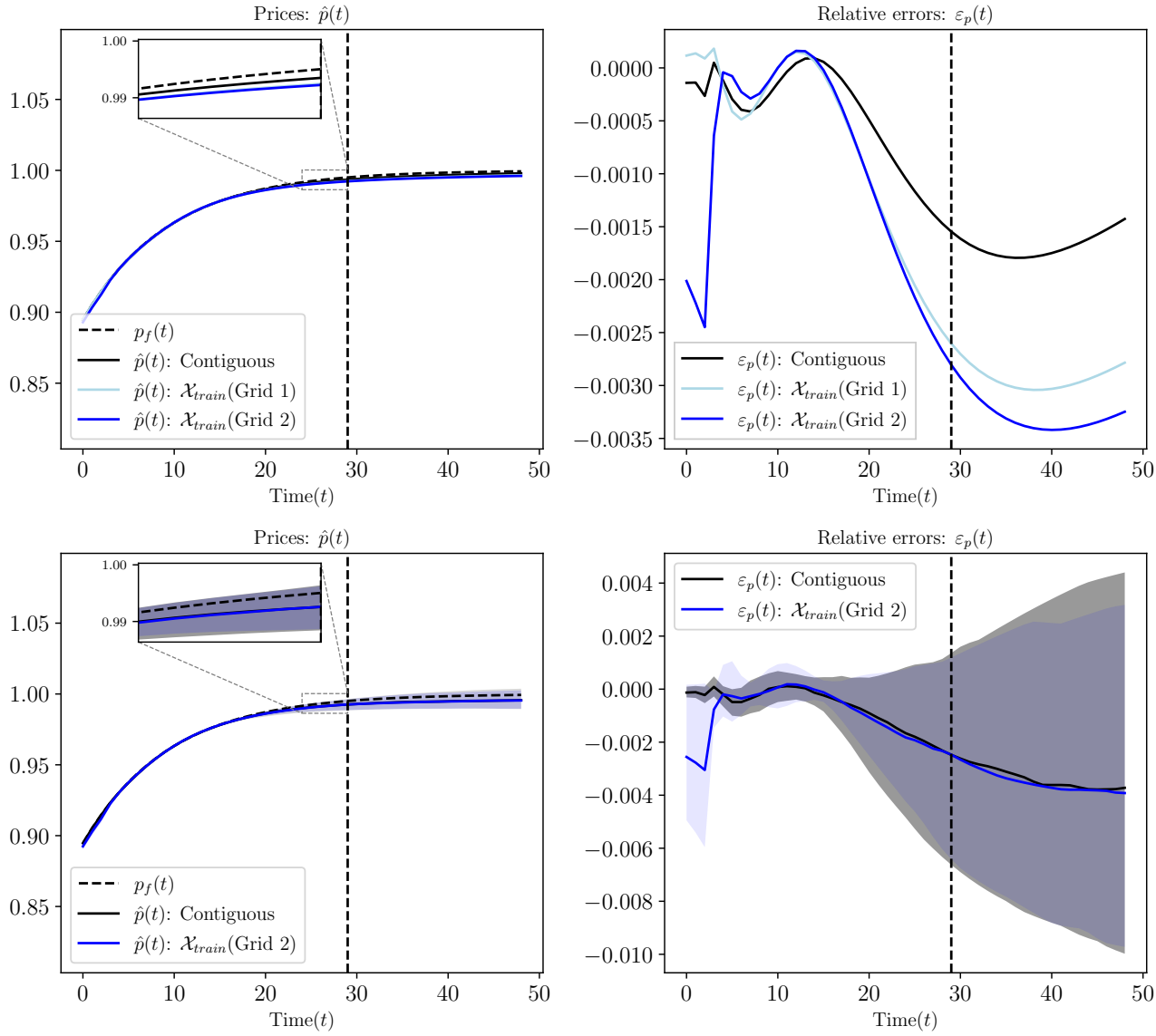

Figure 2: Comparison between the contiguous grid and two sparse grids for the sequential linear asset pricing model. The first sparse grid is defined as $\mathcal{X}_{\text{train}}(\text{Grid 1}) \equiv \{0, 1, 2, 4, 6, 8, 12, 16, 20, 24, 29\}$ and the second sparse grid is defined as $\mathcal{X}_{\text{train}}(\text{Grid 2}) \equiv \{0, 1, 4, 8, 12, 16, 20, 24, 29\}$. The top panels show the results for price paths for one random initialization (one seed). The bottom panels show the median of the approximate price paths and the relative errors for the contiguous grid and $\mathcal{X}_{\text{train}}(\text{Grid 2})$. The shaded regions show the 10th and 90th percentiles. The left panels provide a comparison between the approximate price paths and the price based on the fundamentals. The right panels show the relative errors between the approximate price paths and the price based on the fundamentals. The dashed vertical lines separate the interpolation from the extrapolation region.

Figure 2 shows the results of previous experiment with two different sparse grids for $\mathcal{X}_{\text{train}}$. The first grid contains 11 points and is defined as $\mathcal{X}_{\text{train}}(\text{Grid 1}) \equiv \{0, 1, 2, 4, 6, 8, 12, 16, 20, 24, 29\}$. The second grid contains 9 points and is defined as $\mathcal{X}_{\text{train}}(\text{Grid 2}) \equiv \{0, 1, 4, 8, 12, 16, 20, 24, 29\}$.

The contiguous grid is the same as the previous experiment, $\mathcal{X}_{\text{train}} = \{0, 1, 2, \ldots, 29\}$ denoted by Contiguous. The top-left panel shows the price paths for these grids. $\hat{p}(t)$: Contiguous shows the approximate price path for $\mathcal{X}_{\text{train}}$, $\hat{p}(t)$: $\mathcal{X}_{\text{train}}$(Grid 1) shows the approximate price path for $\mathcal{X}_{\text{train}}$(Grid 1), and $\hat{p}(t)$: $\mathcal{X}_{\text{train}}$(Grid 2) shows the approximate price path for $\mathcal{X}_{\text{train}}$(Grid 2). The top-right panel shows the relative errors between the approximate price paths and the price based on the fundamentals. $\varepsilon_p(t)$: Contiguous shows the relative errors for the contiguous grid, $\varepsilon_p(t)$: $\mathcal{X}_{\text{train}}$(Grid 1) shows the relative errors for $\mathcal{X}_{\text{train}}$(Grid 1), and $\varepsilon_p(t)$: $\mathcal{X}_{\text{train}}$(Grid 2) shows the relative errors for $\mathcal{X}_{\text{train}}$(Grid 2). In the top panels we use only one random initialization (one random seed) of the deep neural network to generate the results.

We solve the minimization problem described in (12) for 100 times. Each time with a different random initialization of the parameters of the deep neural network. We report the median, the 10th, and the 90th percentiles of the results.

The solid curves in the bottom-left panel show the median of price paths for the contiguous grid and $\mathcal{X}_{\text{train}}$(Grid 2), the shaded regions show the 10th and 90th percentiles of the approximate price paths. The solid curves in the bottom-right panel show the median of the relative errors for the contiguous grid and $\mathcal{X}_{\text{train}}$(Grid 2), the shaded regions show the 10th and 90th percentiles of the relative errors. The dashed vertical lines separate the interpolation from the extrapolation region.

These results show that the contiguous grid outperforms both sparse grids. However, the results for both sparse grids are very accurate (at most relative error of -0.35 %). As expected, the most sparse grid has higher relative errors. It is worth mentioning that convergence to the price based on the fundamentals is not sensitive to the grid size. Therefore, it can be a promising avenue for methods of adaptive sample selection in high-dimensional problems. These results also show robustness to random initialization of the deep neural networks for both contiguous and sparse grid. This confirms the implicit bias in the deep neural networks, which consistently favors prices based on fundamentals despite variations in parameters initialization.

Positive growth in dividends ($g > 0$) : in the last two experiments we assumed that the dividends follow a stationary process. In this experiment we show that exploiting a priori economic knowledge combined with the implicit bias, our method can deal with cases where no stationary solution exists. In this case we know the price based on the fundamentals grows exponentially, this economic knowledge can be flexibly implemented in the design of the function space $\mathcal{H}(\Theta)$. In the case of $g > 0$ and $c = 0$, the fundamental price grows with the rate g. In this experiment we use this information to construct an approximating function of the form:

$$\hat{p}(t; \theta) = e^{\phi t} NN(t; \theta_1),$$

where $\theta \equiv \{\phi, \theta_1\}$, $NN(\cdot; \theta_1)$ is a deep neural network, and ϕ is a single parameter that needs to be found in the optimization process.

Again for this experiment, we solve the minimization problem described in (12) for 100 times. Each time with a different random initialization of the parameters of the deep neural network. We report the median, the 10th, and the 90th percentiles of the results.

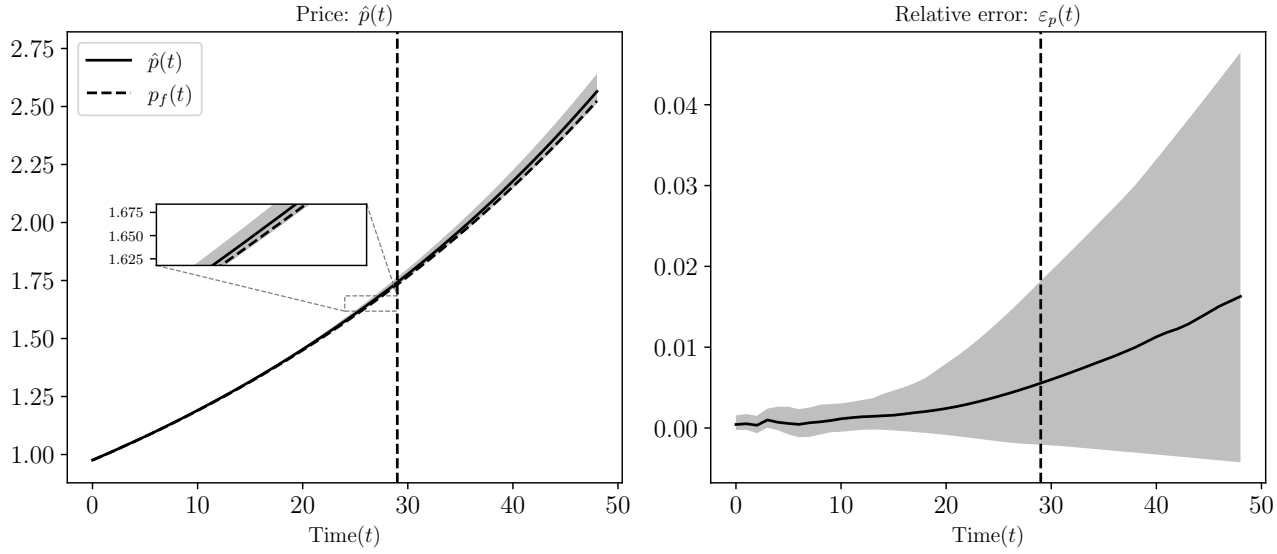

Figure 3: Comparison between the price based on the fundamentals and the price approximated by a deep neural network for the sequential linear asset pricing model with growing dividends ($g = 0.02$). The solid curve in the left panel shows the median of the approximate price paths, the dashed curve shows the price based on the fundamentals. The solid curve in the right panel shows the median of the relative errors between the approximate price paths and the price based on the fundamentals. The shaded regions show the 10th and 90th percentiles. The dashed vertical lines separate the interpolation from the extrapolation region.

Figure 3 shows the results for sequential linear asset pricing model with growing dividends (i.e., $c = 0$, $g = 0.02$). We use the same deep neural network we used in the first experiment. The only difference is the additional exponential term. In the left panel, the solid curve, denoted by $\hat{p}(t)$, shows the median of the approximate price paths, the dashed curve, denoted by $p_f(t)$, shows the solution based on the fundamentals. The shaded region shows the 10th and 90th percentiles of the approximate price paths. The right panel shows the median of the relative errors between the approximate price paths and the price based on the fundamentals. The shaded region shows the 10th and 90th percentiles of the the relative errors. The dashed vertical lines separate the interpolation from the extrapolation region.

These results show that the long-run errors do not impair the short- and medium-run accuracy even in the presence of growing dividends. It is worth noting if interpolation is the object of interest (left of the dashed lines) the same result can be obtained without building the exponential term in the approximating function. However, if extrapolation (right of the dashed lines) is the

object of interest, using economic knowledge can dramatically enhance the generalization power (less than 0.5% relative error after 20 periods). Moreover, the algorithm learns the growth rate accurately. See Figure 22 and the discussion in Appendix A.1 for a detailed treatment of the approximate growth rate.

In this case, we used the correct information about the functional form of the growth in the price path. However, in more complicated dividend processes, when the growth in prices is unknown, it is possible to misspecify the functional form of the growth. Figure 23 in Appendix A.2 confirms that even in the presence of functional misspecification, long-run errors do not impair the accuracy of short- and medium-run dynamics.

3 Neoclassical growth model: sequential form

Another form of forward-looking boundary condition at infinity that appears frequently in dynamic economic models is the transversality condition. In these models the transversality condition is a necessary condition for optimality (see Ekeland and Scheinkman, 1986, and Kamihigashi, 2005). In this section we focus on the sequential version of the neoclassical growth model.

Setup: consider an agent maximizing the discounted utility of consumption $\sum_{t=0}^{\infty} \beta^t u\big(c(t)\big)$. The agent has access to a production technology $z^{1-\alpha} f(k)$, where z is the total factor productivity and k is the capital. Capital depreciates at rate $\delta \in [0,1]$. The agent's optimization problem can be written as:

$$\max_{c(t),k(t+1)} \sum_{t=0}^{\infty} \beta^t u\big(c(t)\big) \tag{17}$$

$$\text{s.t. } k(t+1) = z(t)^{1-\alpha} f\big(k(t)\big) + (1-\delta)k(t) - c(t) \tag{18}$$

$$z(t+1) = (1+g)z(t) \tag{19}$$

$$k(t) \geq 0 \tag{20}$$

$$0 = \lim_{T\to\infty} \beta^T u'\big(c(T)\big)k(T+1) \tag{21}$$

$$k_0, z_0 \text{ given } . \tag{22}$$

The problem described above is the standard neoclassical growth problem. Here we focus on the constant relative risk aversion utility functions $u(c) = \frac{c^{1-\sigma}}{1-\sigma}$, and production function of the form $f(k) = k^{\alpha}$. The first order conditions for this problem can be written as:

$$u'\big(c(t)\big) = \beta u'\big(c(t+1)\big)\big[z(t+1)^{1-\alpha} f'\big(k(t+1)\big) + (1-\delta)\big] \tag{23}$$

$$k(t+1) = f\big(k(t)\big) + (1-\delta)k(t) - c(t), \tag{24}$$

where the first equation is the Euler equation and the second one is the feasibility condition. Equations (23) and (24) do not form a well-posed problem. There are infinitely many solutions $\{c(t), k(t)\}_{t=0}^{\infty}$ that satisfy these equations. The transversality condition, described in (21), uniquely determines the optimal solution.

Saddle path nature of the optimal solution: The optimal solution of this problem is a saddle path and the steady-state of the optimal solution is a saddle point. The saddle path nature of the optimal solution makes the optimal path $\{k(t), c(t)\}$ very sensitive to the choice of the initial consumption, i.e., $c(0)$. Therefore, a small error in the choice of consumption at time zero can lead to a solution that violate the transversality condition.

It is also worth noting that if an economy starts with zero capital (i.e. $k = 0$) this economy stays at zero level of capital forever, therefore, $k = 0$ is a fixed point of this economy. However, it is a repulsive fixed point. Therefore, for an economy that starts with any non-zero level of capital there is no optimal path that takes the economy to $k = 0$. See the discussion and Figure 24 in Appendix B.1 for analysis of our solution method for very small values of $k(0)$.

3.1 Interpolating solution

In this setup, capital (also consumption) is a function of time. Therefore, their domain is defined as $\mathcal{X} \equiv \mathbb{R}_+$. As a generalization of the collocation approach one can pick a space of functions $\mathcal{H}(\Theta)$, where $\Theta \equiv \{\theta_1, \ldots, \theta_M\}$ represents the set of parameters. The interpolating function $k(\cdot;\theta) : \mathcal{X} \to \mathbb{R}$ belongs to $\mathcal{H}(\Theta)$. In order to represent the domain $\mathcal{X}$, one can pick a finite set of grid points $\mathcal{X}_{\text{train}} \subset X$. Since this is a sequential model, similar to the linear asset pricing model:

$$\mathcal{X}_{\text{train}} \equiv \{t_1, \ldots, t_N\}.$$

The Euler equation (i.e., equation 23), the feasibility condition (i.e., equation 24), the law of motion for total factor productivity (i.e., equation (19)), and the transversality condition (i.e., 21) combined with the initial conditions for capital and productivity form a system of equations. However, the transversality condition is a boundary condition at infinity. One can approximate the infinity by choosing a very large point in time, denoted by T. Usually, T is much larger than the largest point in $\mathcal{X}_{\text{train}}$ (i.e., $T \gg t_N$).

For a given capital function $k(t;\theta) \in \mathcal{H}(\Theta)$ we define the consumption function $c(t; k(\cdot;\theta))$ through the feasibility condition:

$$c(t;k(\cdot;\theta)) \equiv f\big(k(t;\theta)\big) + (1-\delta)k(t;\theta) - k(t+1;\theta). \tag{25}$$

Given a space of parametric functions $\mathcal{H}(\Theta)$, a grid $\mathcal{X}_{\text{train}}$, and a very large point in time T, one can find the interpolating capital function by solving the following optimization problem

$$\min_{\theta\in\Theta}\left[\frac{1}{|\mathcal{X}_{\text{train}}|}\sum_{t\in\mathcal{X}_{\text{train}}}\left(\frac{u'\big(c(t;k(\cdot;\theta))\big)}{u'\big(c(t+1;k(\cdot;\theta))\big)} - \beta\big[z(t+1)^{1-\alpha}f'\big(k(t+1;\theta)\big) + (1-\delta)\big]\right)^2 + \right.$$
$$\left.(k(0;\theta)-k_0)^2 + \underbrace{\left(\beta^T u'\Big(c\big(T;k(\cdot;\theta)\big)\Big)k(T+1;\theta)\right)^2}_{\text{Is this necessary?}}\right], \tag{26}$$

where $z(t)$ is evaluated by the law of motion for z

$$z(t) = (1+g)^t z_0 \quad \text{for } t \in \mathcal{X}_{\text{train}}. \tag{27}$$

The first term in the optimization problem represents the Euler residuals and the second term is the residual for the initial condition. The second term ensures that the capital function at time zero matches the initial level of capital. The third term is a finite approximation of the transversality condition. The non-negativity of capital, $k(t;\theta) \geq 0$, can be built into $\mathcal{H}(\Theta)$.

In a standard collocation method, where the number of parameters is equal to the number of grid points, this problem can be solved exactly as a system of equations with a boundary condition at T. It is worth noting that the interpolating solution depend on T. Therefore, the solutions of the minimization problem described in (26) defines a sequence of solutions indexed by T.

Here, we only provide a framework for interpolation of the capital function. It is also possible to interpolate both capital and consumption functions simultaneously. See Appendix B.3 for more details.

The approach we introduced for solving sequential models in this paper has some similarities with parametric path methods. These methods approximate the equilibrium paths with a family of parametric functions, see Judd (2002). In these methods the convergence to a solution that does not violate the transversality condition is baked into the approximating functions as an a priori economic knowledge. For instance, by design the approximating function approaches the desired steady-state as time goes to infinity. Therefore, these methods require calculating the steady-state for finding the approximate solution. However, ours does not require any knowledge

about the steady-state.[15]

Choice of $\mathcal{X}_{\text{train}}$: similar to the case of linear asset pricing model, the choice of $\mathcal{X}_{\text{train}}$ is not very crucial. However, we provide the results for different grids. We also evaluate the approximate capital function outside of $\mathcal{X}_{\text{train}}$ to study the generalization performance of the solutions.

Is the transversality condition necessary? Without the transversality condition the neoclassical growth problem has infinitely many solutions. Equations (19),(23), and (24) form a three-dimensional dynamical system with two initial conditions k_0 and z_0[16]. Therefore, in the absence of the third term (finite approximation of the transversality condition) the minimization problem described in (26) has infinitely many solutions that minimize the objective function.

In Section 3.3 we establish how using over-parameterized functions (when $M \gg N$) and their optimization processes yield convergence to a solution that does not violate the transversality condition. Intuitively, the solutions that violate the transversality condition have bigger derivatives than the optimal solution. Therefore, using over-parameterized functions we can drop the transversality condition and solve:

$$\min_{\theta\in\Theta}\left[\frac{1}{|\mathcal{X}_{\text{train}}|}\sum_{t\in\mathcal{X}_{\text{train}}}\left(\frac{u'\big(c(t;k(\cdot;\theta))\big)}{u'\big(c(t+1;k(\cdot;\theta))\big)}-\beta\left[z(t+1)^{1-\alpha}f'\big(k(t+1;\theta)\big)+(1-\delta)\right]\right)^2+\big(k(0;\theta)-k_0\big)^2\right], \tag{28}$$

to obtain an interpolating solution that does not violate the transversality condition.[17]

3.2 Results

Figure 4 shows the result of the minimization problem described in (28) for $\beta = 0.9$, $\alpha = 0.33$, $\delta = 0.1$, $g = 0$, $k_0 = 0.4$, and $z_0 = 1$.

[15] Judd (2002) uses Laguerre polynomials instead of the deep neural networks. The author uses an approximating function with two terms. The first term is the parametric form to deal with the short- and medium-run transition. The second term, by design, becomes the dominant term in the long-run and approaches the steady-state.

[16] Given the initial level of capital k_0, factor productivity z_0, and an arbitrary c_0 for the initial level of consumption we can generate a set of solutions $\{\tilde{k}(t), \tilde{c}(t), z(t)\}$ by iterating forward the Euler equation, feasibility condition, and the law of motion for total factor productivity. By construction this set of solutions is a solution for this system of equations. Therefore, there are infinitely many solutions each characterized by a different c_0.

[17] In practice one can solve:

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

where λ_1 and λ_2 are a pair of positive parameters that determine the importance of each residuals in the optimization process. In our experiments, the results are not sensitive to the choice of these parameters so we can choose equally weighted residuals (i.e., $\lambda_1 = \lambda_2 = 1$).

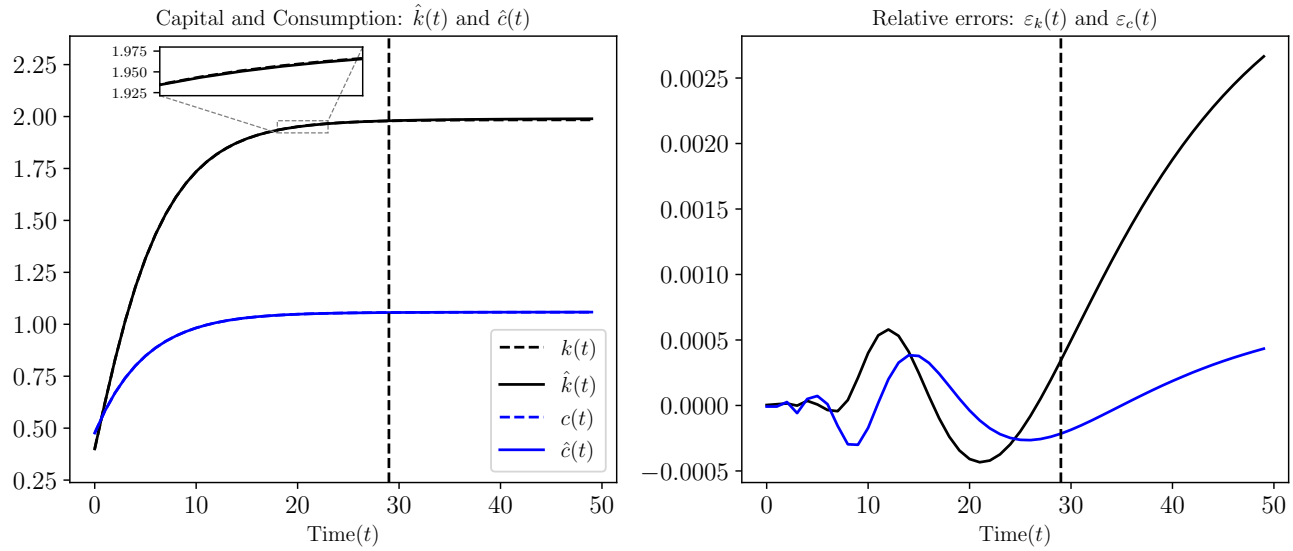

Figure 4: Comparison between the value function iteration and approximate solution using a deep neural network for the sequential neoclassical growth model. In the left panel the black solid line shows the approximate capital path and the blue solid line shows the consumption path. The dashed curves show the capital and consumption paths obtained by the value function iteration method. The right panel shows the relative errors for both capital and consumption paths. The dashed vertical lines separate the interpolation from the extrapolation region.

In this experiment $\mathcal{X}_{\text{train}} = \{0, 1, 2, \ldots, 29\}$ and $\mathcal{X}_{\text{test}} = \{0, 1, 2, ..., 49\}$. We approximate the capital path $k(t;\theta)$ using a deep neural network with four hidden layers, each with 128 nodes. Each hidden layer uses Tanh and the output layer uses Softplus as activation functions. The dashed vertical lines separate the interpolation from the extrapolation region. The left panel shows the capital and consumption paths. The benchmark solution is calculated using value function iteration method and are denoted by $k(t)$ and $c(t)$.[18] The approximate solutions are denoted by $\hat{k}(t)$, and $\hat{c}(t)$. More specifically, $\hat{k}(t) \equiv k(t;\theta^*)$, where

$$\theta^* \equiv \underset{\theta\in\Theta}{\operatorname{argmin}} \left[\frac{1}{|\mathcal{X}_{\text{train}}|} \sum_{t\in\mathcal{X}_{\text{train}}} \left(\frac{u'\big(c(t;k(\cdot;\theta))\big)}{u'\big(c(t+1;k(\cdot;\theta))\big)} - \beta\left[z(t+1)^{1-\alpha} f'\big(k(t+1;\theta)\big) + (1-\delta)\right] \right)^2 + \left(k(0;\theta) - k_0\right)^2 \right],$$

and $\hat{c}(t)$ is calculated using the consumption function defined in equation (25) for the capital function $k(t;\theta^*)$, i.e., $\hat{c}(t) \equiv c\left(t;k(\cdot;\theta^*)\right)$. The right panel shows the relative errors between the approximate solutions and the solutions obtained by the value function iteration method defined as:

[18] The value function iteration is also an approximation method, here we assume that value function iteration method yields more accurate results than our method.

$$\varepsilon_c(t) \equiv \frac{\hat{c}(t) - c(t)}{c(t)} \quad \text{for } t \in \mathcal{X}_{\text{test}} \tag{29}$$

$$\varepsilon_k(t) \equiv \frac{\hat{k}(t) - k(t)}{k(t)} \quad \text{for } t \in \mathcal{X}_{\text{test}}. \tag{30}$$

The results of this experiment are multi-fold. First, we can achieve an accurate solutions that do not violate the transversality condition without imposing any long-run constraint. Second, the short- and medium-run dynamics are accurate and they are not impaired by the long-run errors (at most -0.05% relative error for capital). Third, the extrapolation errors are very small which can be explained by the smoothness imposed by deep neural networks and the choice of $\mathcal{X}_{\text{train}}$. Here, we used a large time horizon in $\mathcal{X}_{\text{train}}$, i.e., $t_N = 29$. The capital path in the last few points of the grid are very close to the steady-state. Therefore, the deep neural network learns that for large values of t the capital is almost a constant and stays very close to that value even outside of $\mathcal{X}_{\text{train}}$.

Contiguous vs. Sparse Grid: Figure 5 shows the result of the previous experiment for two different sparse grids for $\mathcal{X}_{\text{train}}$. The first grid contains 11 points and is defined as $\mathcal{X}_{\text{train}}(\text{Grid 1}) \equiv \{0, 1, 2, 4, 6, 8, 12, 16, 20, 24, 29\}$. The second grid contains 9 points and is defined as $\mathcal{X}_{\text{train}}(\text{Grid 2}) \equiv \{0, 1, 4, 8, 12, 16, 20, 24, 29\}$. The contiguous grid is the same as the previous experiment $\mathcal{X}_{\text{train}} = \{0, 1, 2, \ldots, 29\}$, denoted by Contiguous. The dashed vertical lines separate the interpolation from the extrapolation region. The top-left panel shows the approximate capital paths for these grids, denoted by $\hat{k}(t)$, versus the capital path generated by value function iteration method, denoted by $k(t)$. The top-right panel shows the relative errors between the approximate capital paths and the capital path generated by the value function iteration method. In both top panels we only use one random initialization (one random seed) of the parameters of the deep neural network to generate the results.

We solve the minimization problem described in (28) for 100 times. Each time with a different random initialization of the parameters of the deep neural network. We report the median, the 10th, and the 90th percentiles of the results.

The bottom-left panel shows the capital paths obtained by value function iteration method, denoted by $k(t)$ and the median of the approximate capital paths for the contiguous grid and $\mathcal{X}_{\text{train}}(\text{Grid 2})$. The shaded regions show the 10th and 90th percentiles of the approximate capital paths. The bottom-right panel shows the median of relative errors between the approximate capital paths and the capital path obtained by value function iteration method for the contiguous grid and $\mathcal{X}_{\text{train}}(\text{Grid 2})$. The shaded regions show the 10th and 90th percentiles of the relative errors.

These results show that the contiguous grid outperforms both sparse grids. However, the

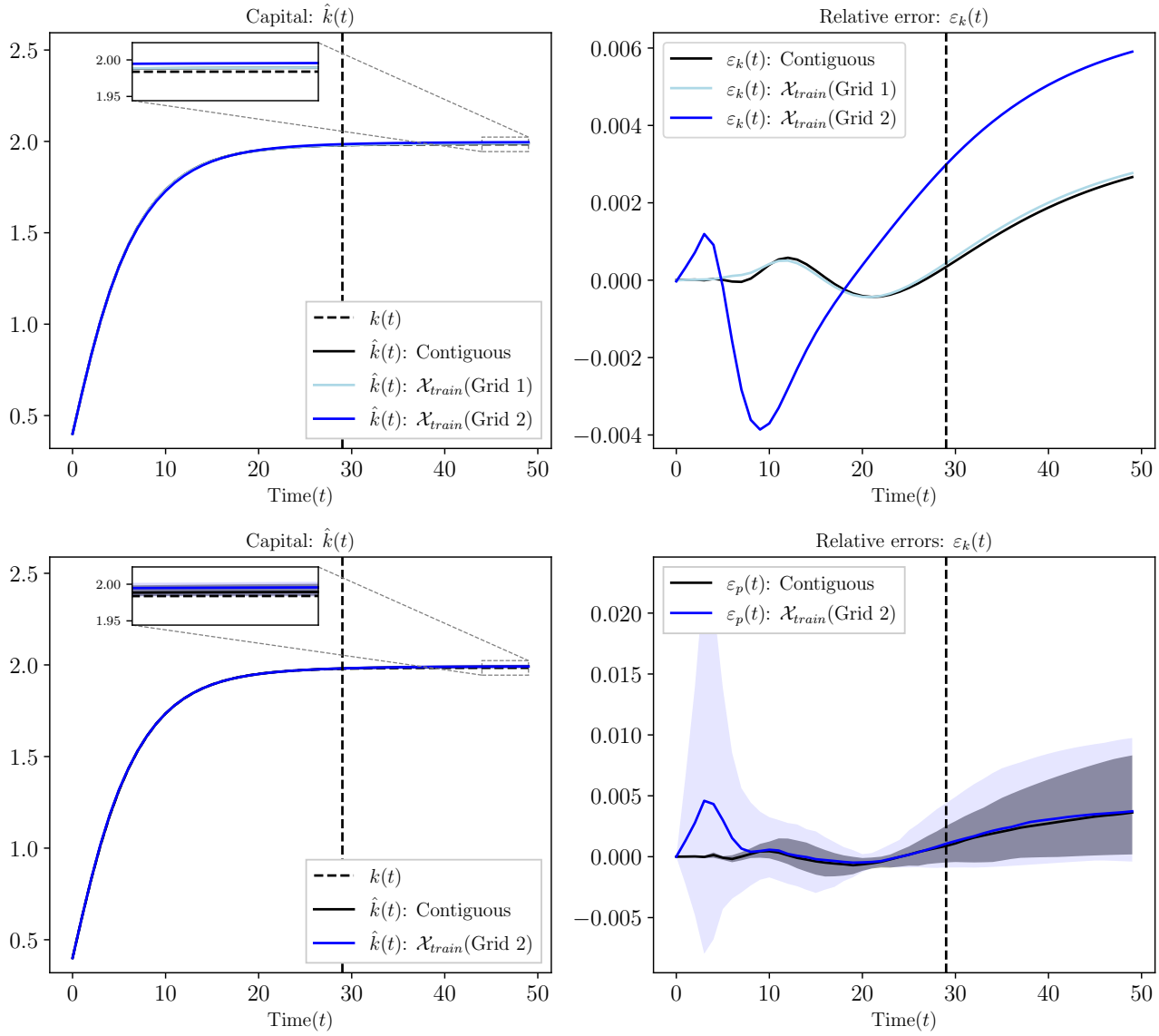

Figure 5: Comparison between the value function iteration and approximate solution using a deep neural network for the sequential neoclassical growth model. The first sparse grid is defined as $\mathcal{X}_{\text{train}}(\text{Grid 1}) \equiv \{0, 1, 2, 4, 6, 8, 12, 16, 20, 24, 29\}$ and the second sparse grid is defined as $\mathcal{X}_{\text{train}}(\text{Grid 2}) \equiv \{0, 1, 4, 8, 12, 16, 20, 24, 29\}$. The top panels show the approximate capital paths and the relative errors between the approximate capital path and the capital path obtained by the value function iteration method, for $\mathcal{X}_{\text{train}}(\text{Grid 1})$, $\mathcal{X}_{\text{train}}(\text{Grid 2})$, and the contiguous grid. The bottom panels show the median of the approximate capital paths and the median of relative errors for $\mathcal{X}_{\text{train}}(\text{Grid 1})$, $\mathcal{X}_{\text{train}}(\text{Grid 2})$, and the contiguous grid. The shaded regions show the 10th and 90th percentiles. The dashed curves, denoted by $k(t)$, show the capital path obtained by the value function iteration method. The dashed vertical line separates the interpolation from the extrapolation region.

results for both sparse grids are very accurate (at most 0.6 % in relative errors). As expected the most sparse grid has higher relative errors. It is worth mentioning that convergence to the optimal

solution is not sensitive to the grid size. Therefore, it can be a promising avenue for adaptive optimization and sample selection in high-dimensional sequential problems. These results also show robustness to random initialization of the deep neural networks in the optimization process for both contiguous and sparse grid. This confirms the implicit bias in deep neural networks, which in this case the bias is toward the optimal path for capital.

Far from the steady-state: in the previous experiment we established that we can achieve accurate short- and medium-run accuracy on sparse grids. However, in both grids we used a large time horizon (i.e., $t_N = 29$). After 29 periods the capital and consumption paths are very close to their steady-states. In this experiment we study whether the approximate solution has accurate short-run dynamics when using a medium time horizon for $\mathcal{X}_{\text{train}}$ (e.g., $t_N \approx 10$).

Again for this experiment, we solve the minimization problem described in (28) for 100 times. Each time with a different random initialization of the parameters of the deep neural network. We report the median, the 10th, and the 90th percentiles of the results.

Figure 6 shows the results for the sequential neoclassical growth model with $\mathcal{X}_{\text{train}} = \{0, 1, \ldots, 9\}$. We used the same parameters and deep neural network as the previous experiments. The dashed vertical lines separate the interpolation from the extrapolation region. The solid curve in the top-left panel shows the median of the approximate capital paths, the dashed curve shows the capital path obtained by the value function iteration method, and the shaded region shows the 10th and 90th percentiles of the approximate capital paths. The solid curve in the top-right panel shows the median of the relative errors between the approximate capital paths and the capital path obtained by the value function iteration method, the shaded region shows the 10th and 90th percentiles of the relative errors. The solid curve in the bottom-left panel shows the median of the approximate consumption paths, the dashed curve shows the consumption path obtained by the value function iteration method, and the shaded region shows the 10th and 90th percentiles of the approximate consumption paths. The solid curve in the bottom-right panel shows the median of relative errors between the approximate consumption paths and the consumption path obtained by the value function iteration method, the shaded region shows the 10th and 90th percentiles of the relative errors.

These results show that we obtain solutions that do not violate the transversality condition even when we use medium time horizons in $\mathcal{X}_{\text{train}}$. Moreover, the long-run errors do not impair the accuracy of short-run dynamics (less than 0.5% after 5 periods). Therefore, we get accurate solutions when the capital and consumption paths are far from the steady-state. This result is robust to using very short time horizons in $\mathcal{X}_{\text{train}}$ (i.e., $t_N = 4$). See Figure 26 and the discussion in Appendix B.4 for more details.

Balanced growth path ($g > 0$): in the last experiments we assumed that the total factor productivity is constant and stationary. Here we show that exploiting a priori economic knowledge

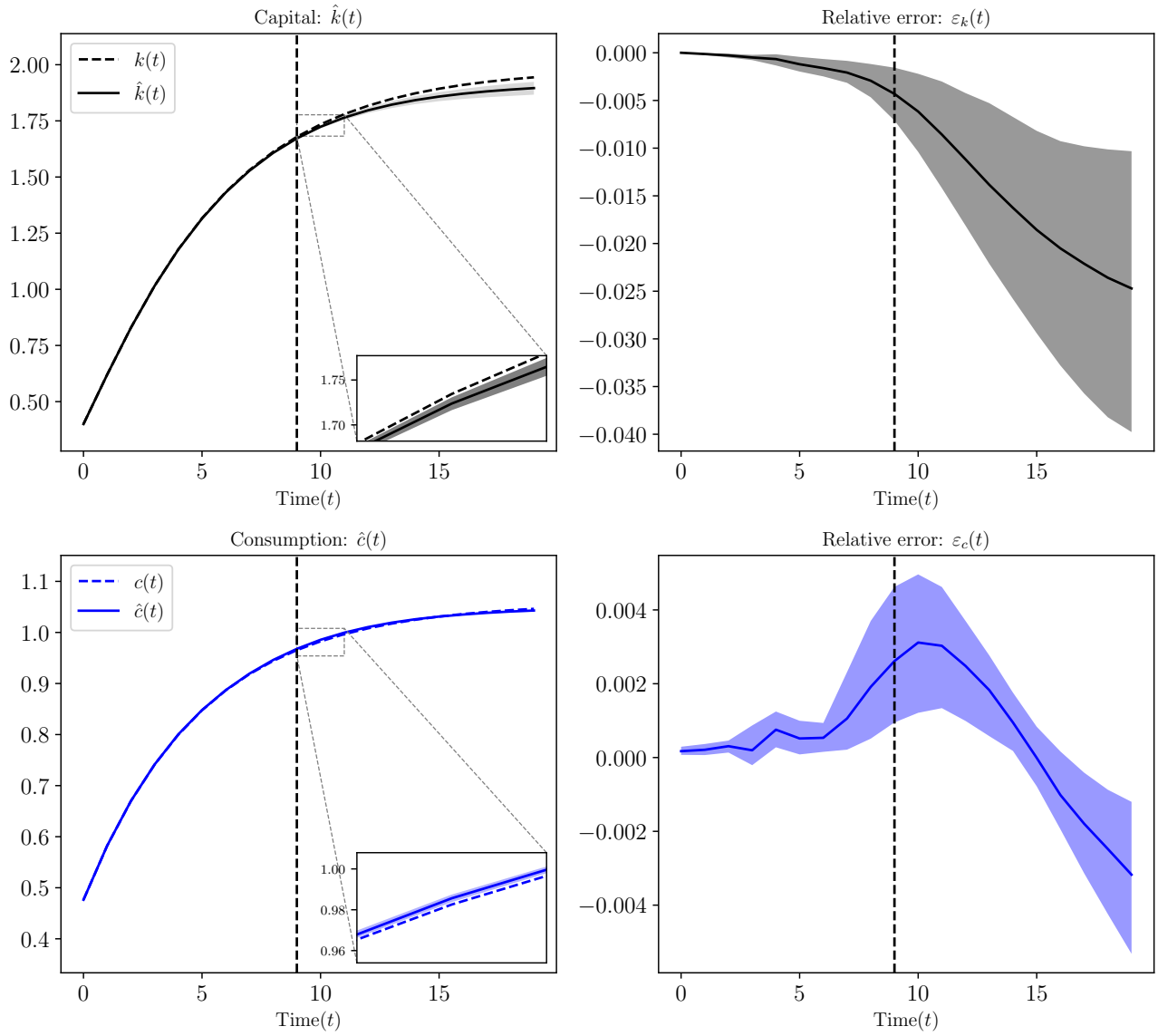

Figure 6: Comparison between the value function iteration and approximate solution using a deep neural network for the sequential neoclassical growth model with a medium time horizon (i.e., $\mathcal{X}_{\text{train}} = \{0, 1, \ldots, 9\}$). The solid curves in the left panels show the median of the approximate capital and consumption paths. The dashed curves show the capital and consumption paths obtained by the value function iteration method. The solid curves in the right panels show the median of the relative errors between the approximate solutions and solutions obtained by the value function iteration method. The shaded regions show the 10th and 90th percentiles. The dashed vertical lines separate the interpolation from the extrapolation region.

combined with the implicit bias of deep neural networks, our method can deal with models where no stationary solution exists. Since the production function is homogeneous of degree one, in the long-run the capital path has a growth rate of g. This economic knowledge can be implemented flexibly in the design of the function space $\mathcal{H}(\Theta)$. In this experiment we construct

an approximating function of the following form:

$$\hat{k}(t;\theta) = e^{\phi t} NN(t;\theta_1)$$

where $\theta \equiv \{\phi, \theta_1\}$, $NN(\cdot;\theta_1)$ is a deep neural network, and ϕ is a single parameter that needs to be found in the optimization process.

Again for this experiment, we solve the minimization problem described in (28) for 100 times. Each time with a different random initialization of the parameters of the deep neural network. We report the median, the 10th, and the 90th percentiles of the results.

Figure 7 shows the results for sequential neoclassical growth for non-stationary total factor productivity (i.e., $g = 0.02$). We use the same deep neural network as before for $NN(\cdot;\theta_1)$ and use the original grid, $\mathcal{X}_{\text{train}} = \{0, 1, 2, \ldots, 29\}$. The only difference is the additional exponential term. The solid curve in the top-left panel shows the median of the approximate capital paths and the dashed curve shows the capital path obtained by the value function iteration method. The shaded region shows the 10th and 90th percentiles of the approximate capital paths. The solid curve in the top-right panel shows the median of the relative errors between the approximate capital paths and the capital path obtained by the value function iteration method. The shaded region shows the 10th and 90th percentiles of the relative errors. The solid curve in the bottom-left panel shows the median of the approximate consumption paths, the dashed curve shows the consumption path obtained by the value function iteration method. In this case the 10th and 90th percentiles for the relative errors are so close to each other that they are not distinguishable in the plot. The solid curve in the bottom-right panel shows the median of the relative errors between the approximate consumption paths and the consumption path obtained by the value function iteration method, the shaded region shows the 10th and 90th percentiles of the relative errors.

These results show that the long-run errors do not impair the short- and medium-run accuracy even in the presence of balanced growth path. It is worth noting if interpolation is the object of interest (left of the dashed line) the same result can be obtained without building the exponential term in the approximating function. However, if extrapolation (right of the dashed line) is the object of interest, as shown in this results, using economic knowledge can dramatically enhance the generalization power (1% relative errors in capital after 49 periods).

Here we used the correct information about the functional form of the growth in capital and consumption paths. However, in cases where the production function is not homogeneous of degree one or the total factor productivity follows a non-linear process, it is possible to misspecify the functional form of the growth. Figure 27 in Appendix B.5 confirms that even in the presence of functional misspecification, the long-run errors do not impair the accuracy of the short- and medium-run dynamics. Moreover, the results shows that the solutions generalize well in the extrapolation region.

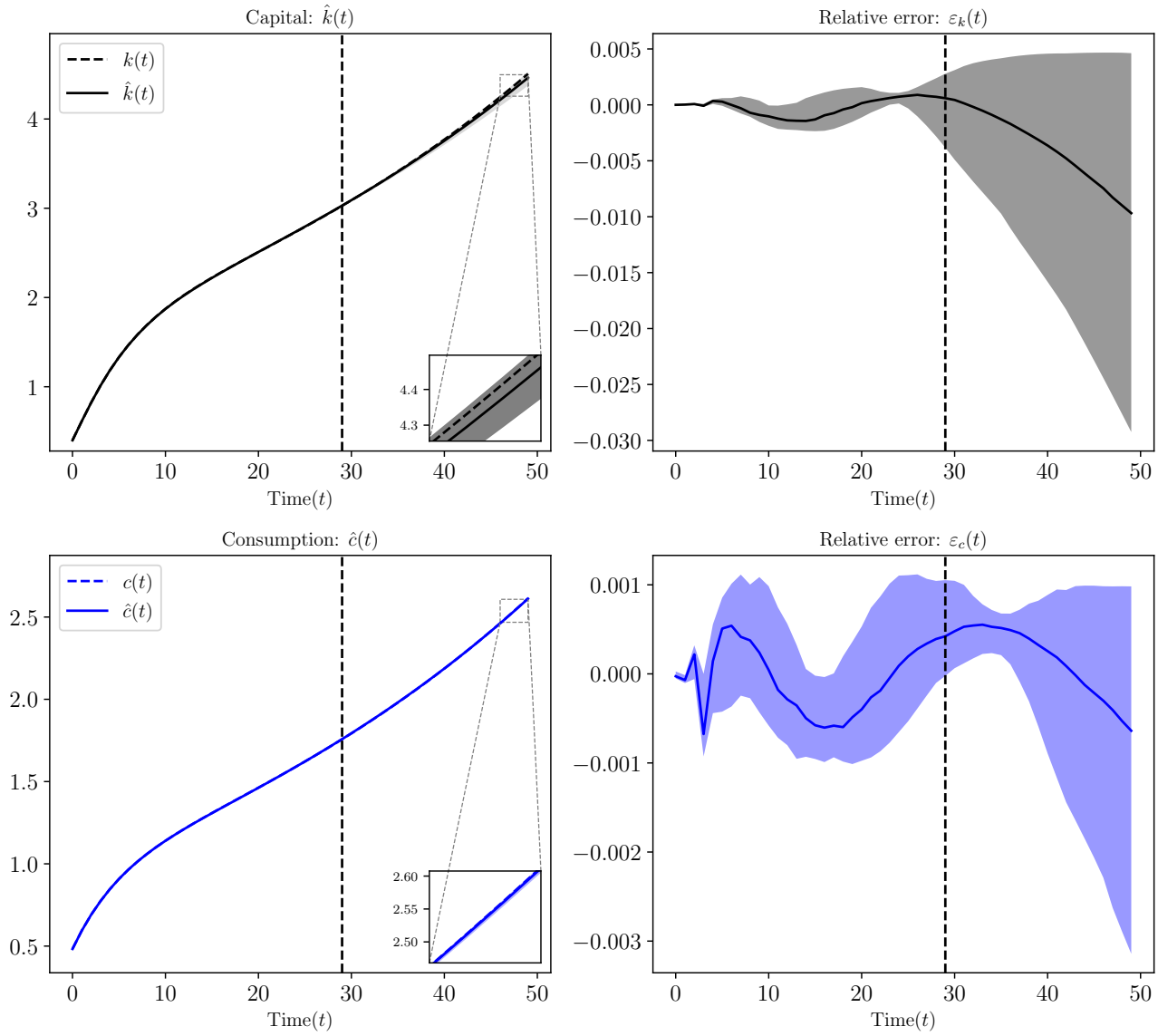

Figure 7: Comparison between the value function iteration and approximate solution using a deep neural network for the sequential neoclassical growth model with growth in total factor productivity (i.e., $g = 0.02$). The solid curves in the left panels show the median of the approximate capital and consumption paths. The dashed curves show the capital and consumption paths obtained by the value function iteration method. The solid curves in the right panels show the median of the relative errors between the approximate solutions and the solutions obtained by the value function iteration method. The shaded regions show the 10th and 90th percentiles. The dashed vertical lines separate the interpolation from the extrapolation region.

3.3 Minimum-norm interpretation and transversality condition

In this section we provide the connection between minimum-norm interpretation and the transversality condition. We focus on the case of zero total factor productivity growth (i.e., $g = 0$ and $z = 1$).

The solutions that satisfy the Euler equation and feasibility condition (i.e., equations 23-24) and violate the transversality condition can be characterized by their asymptotic behavior. The common feature between these solutions is that the consumption goes to zero over time. Let $\{\tilde{c}(t), \tilde{k}(t)\}_{t=0}^{\infty}$ be one of those solutions, then:

$$\lim_{t\to\infty} \tilde{c}(t) = 0$$
$$\lim_{t\to\infty} \tilde{k}(t) = \tilde{k}_{\max},$$

where $\tilde{k}_{\max}$ solves:

$$\delta\tilde{k}_{\max} = f\left(\tilde{k}_{\max}\right).$$

Since the production function f is increasing and strictly concave, $\tilde{k}_{\max}$ exists and is unique. Moreover, it is the stable fixed point for a dynamical system described by $k(t+1) = f\big(k(t)\big) + (1-\delta)k(t)$. For $f(k) = k^{\alpha}$;

$$\tilde{k}_{\max} = \delta^{\frac{1}{\alpha-1}}.$$

Comparing $\tilde{k}_{\max}$ with the steady-state of the capital $k^* \equiv \left(\frac{\beta^{-1}+\delta-1}{\alpha}\right)^{\frac{1}{\alpha-1}}$, one can establish that:

$$\tilde{k}_{\max} > k^*.$$

The blue curves in Figure 8 show a set of paths for capital, consumption and marginal utility of consumption (shadow prices), denoted by $\tilde{k}(t), \tilde{c}(t)$, and $u'(\tilde{c}(t))$, that satisfy the Euler equation and feasibility condition, and violate the transversality condition.[19] The black curves, denoted by $k(t)$, $c(t)$, and $u'(c(t))$, show the optimal paths for capital, consumption and marginal utility of consumption. These paths satisfy the Euler equation, feasibility condition and the transversality condition. The steady-states for capital and consumption in the optimal solution are denoted by k^* and c^*. It is important to note that for this set of parameters $\frac{\tilde{k}_{\max}}{k^*} \approx 15$.

The minimum-norm interpretation of the solutions of the minimization problem described in (26) for the limiting case of $T \to \infty$ can be written as:

$$\begin{aligned}
\min_{k(\cdot;\theta)\in\mathcal{H}(\Theta)} \quad & \|k(\cdot;\theta)\|_S \\
\text{s.t.} \quad & u'\Big(c\big(t;k(\cdot;\theta)\big)\Big) = \beta u'\Big(c\big(t+1;k(\cdot;\theta)\big)\Big)\Big[f'\big(k(t+1;\theta)\big) + 1 - \delta\Big] \quad \text{for } t \in \mathcal{X}_{\text{train}} \\
& k(0) = k_0 \\
& 0 = \lim_{T\to\infty} \beta^T u'(c(T;k(\cdot;\theta)))k(T+1),
\end{aligned}$$

where $\mathcal{H}(\Theta)$ is a space of over-parameterized functions, $\mathcal{X}_{\text{train}}$ is a set of points in time (i.e.,

[19] The parameters we used are $k_0 = 0.4$, $\beta = 0.9$, $\alpha = 0.33$, and $\delta = 0.1$.

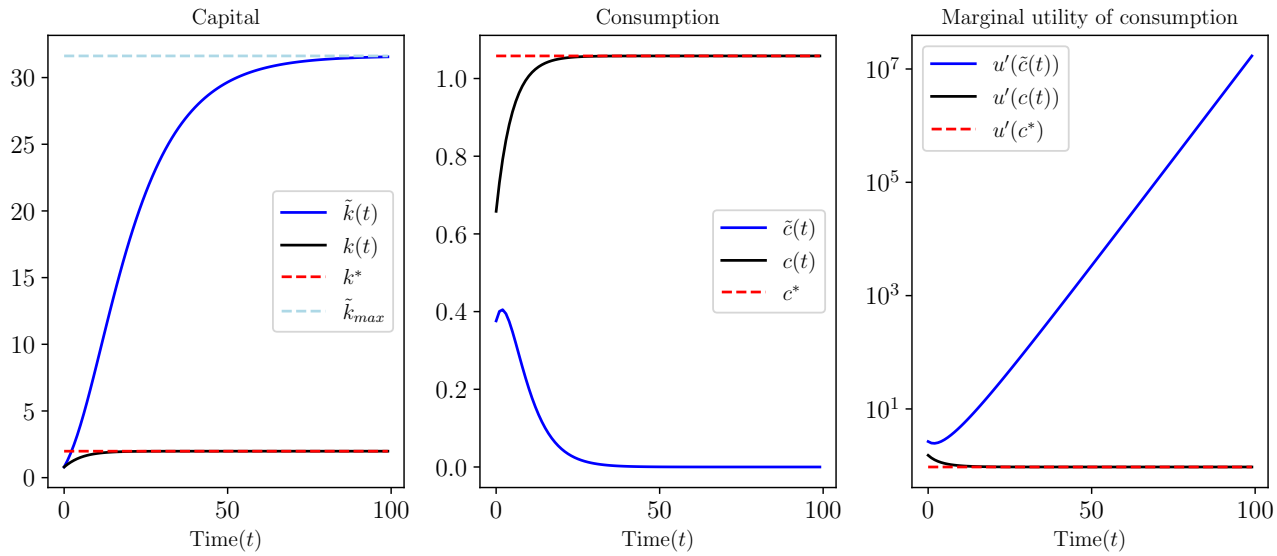

Figure 8: Comparison between the optimal solution and the solutions that violate the transversality condition for $k_0 < k^*$. The blue curves, denoted by $\tilde{k}(t)$, $\tilde{c}(t)$, and $u'(\tilde{c}(t))$, show a set of capital, consumption, and marginal utility of consumption paths that violate the transversality condition. The black curves denoted by $k(t)$, $c(t)$, and $u'(c(t))$ show the capital, consumption, and marginal utility of consumption paths for the optimal solution. The steady-states for capital an consumption are denoted by k^* and c^*.

$\mathcal{X}_{\text{train}} = \{t_1, \ldots, t_N\}$), the semi-norm $\|\cdot\|_S$ satisfies Assumption 1, and the consumption function $c\big((t; k(\cdot;\theta)\big)$ is defined in equation (25).

As evident in Figure 8, any capital sequence that starts from k_0 and violates the transversality condition converges to $\tilde{k}_{\max}$. Both capital paths $\tilde{k}(t)$ and $k(t)$ increase monotonically. Starting from k_0, $\tilde{k}(t)$ must have bigger derivatives than $k(t)$ to reach $\tilde{k}_{\max}$. Therefore, the solutions that violate the transversality condition have bigger gradients. More formally, let $\tilde{k}(t)$ be a capital path violating the transversality condition and $k(t)$ be the optimal solution, then in a compact space of the form $[0, T]$:

$$\int_0^T \left|\frac{d\tilde{k}}{dt}\right|^2 dt > \int_0^T \left|\frac{dk}{dt}\right|^2 dt.$$

Therefore, by Assumption 1:

$$\|k\|_S < \|\tilde{k}\|_S.$$

The solutions that violate the transversality condition have bigger semi-norms. In other words, over-parameterized interpolating solutions has a bias toward $k(t)$ and automatically satisfy the transversality condition. Therefore, the optimization described in (28) is enough to find the opti-

mal solution without imposing any explicit regularity condition regarding the long-run behavior.

It is important to note that when the transversality condition is violated the marginal utility of consumption (shadow price) grows boundlessly (order of 10^7 after 100 periods). Alternatively, one can use an over-parameterized function to approximate the shadow price. Since the optimal solution has bounded shadow prices, it is easy to establish that [20]:

$$\|u'(c)\|_S < \|u'(\tilde{c})\|_S. \tag{31}$$

Since the difference between the derivatives of marginal utility of consumption for the solutions that violate the transversality condition and the optimal solution is much larger than the difference between the derivatives of their corresponding capital paths, it is easier for the implicit bias of over-parameterized functions to detect the optimal solution. Therefore, in practice approximating the marginal utility yields faster convergence to the optimal solution.

In this argument we focus on cases where the initial value of capital is below the steady-state. For cases where the initial condition is above the steady-state the same argument can be made, see Figure 25 and the discussion in Appendix B.2.

3.4 Neoclassical growth model with multiplicity of equilibria: sequential form

The sequential version of the neoclassical growth model discussed and studied in Section 3 has a steady-state characterized by (k^*, c^*), while the solutions that violate the transversality condition approach a fixed point $(k, c) = (\tilde{k}_{\max}, 0)$. Given that for regular parameters $\tilde{k}_{\max}$ is extremely larger that k^*, the semi-norm of the capital path that converges to k^* is substantially smaller than those that violate the transversality condition. Therefore, it is easy for the implicit bias to distinguish the difference between these two solutions and pick the optimal solution path. However, a natural question that arises is: can the implicit bias of over-parameterized functions distinguish the optimal path in the presence of multiplicity of steady-states? Especially, when the steady-states are fairly close to each other.

In this section we address this question by studying the sequential neoclassical growth problem with a convex-concave production function of the form:

$$f(k) = a \max\{k^\alpha, b_1 k^\alpha - b_2\}, \tag{32}$$

where a, b_1, b_2 a are positive constants and $b_1 > 1$. [21] This production function is continuous, however, has a kink at $\left(\frac{b_2}{b_1 - 1}\right)^{\frac{1}{\alpha}}$. We embed this production function in the sequential setup of

[20] For the case of shadow prices, Assumption 1 can be relaxed. Because for all the semi-norms (and norms) we are aware of, an unbounded function has a bigger semi-norm than a bounded function.

[21] See Skiba (1978) for a comprehensive and detailed treatment of this problem in a continuous time setup.

neoclassical growth described in Section 3. Specifically, we solve the optimization illustrated in (28) with this convex-concave production function. In this experiment we focus on constant total factor productivity (i.e., $z_0 = 1$ and $g = 0$).[22]

This problem has two steady-states for capital:

$$k_1^* = \left(\frac{\beta^{-1} + \delta - 1}{a\alpha}\right)^{\frac{1}{\alpha-1}}$$
$$k_2^* = \left(\frac{\beta^{-1} + \delta - 1}{ab_1\alpha}\right)^{\frac{1}{\alpha-1}}.$$

with the corresponding steady-states for consumption c_1^* and c_2^*.

Figure 9 shows the results for $\beta = 0.9$, $\alpha = 0.33$, $\delta = 0.1$, $g = 0$, $z_0 = 1$, $a = 0.5$, $b_1 = 3$, $b_2 = 2.5$, and $k_0 = \{0.5, 1.0, 3.0, 4.0\}$.

In this experiment $\mathcal{X}_{\text{train}} = \{0, 1, 2, \ldots, 29\}$ and $\mathcal{X}_{\text{test}} = \{0, 1, 2, ..., 49\}$. We approximate the capital path using a deep neural network with four hidden layers, each with 128 nodes. Hidden layers use Tanh and the output layer uses Softplus as activation functions. The left panel shows the capital paths and the right panel shows the corresponding consumption paths. The dashed vertical lines separate the interpolation from the extrapolation region.

These results show that even in the presence of multiplicity of steady-states, the implicit bias picks up the right paths for capital and consumption in the vicinity of the steady-states. It is worth noting that when the initial capital is above k_2^* or below k_1^* the optimal solution has the lowest semi-norm and the implicit bias of deep neural networks can accurately represent the optimal solution.

Figure 10 shows the capital and consumption paths for a grid of initial conditions $k_0 \in [0.5, 1.75]$ and $k_0 \in [2.75, 4]$. The top panel shows the capital paths and the bottom panel shows the consumption paths. The dashed horizontal lines show the steady-states of capital k_1^* and k_2^* and the corresponding steady-states for consumption c_1^* and c_2^*. The dashed vertical lines separate the interpolation from the extrapolation region.

These results show that solutions are robust to variations in the initial condition for capital in the vicinity of the steady-states.

It is worth noting that the proposed method has two basins of attraction in the space of initial levels of capital. There exists a critical point, where above that point all the capital paths converge to the higher steady-state of capital (i.e., k_2^*) and below that point all the capital paths converge to the lower steady-state of capital (i.e., k_1^*). We do not expect our solution method to

[22] In this experiment we use Adam optimizer instead of LBFGS. The Adam optimizer is more stable than LBFGS. Due to Adam's longer process of optimization (more optimization steps), the solution is exposed to more implicit regularization, which is very desirable for this problem.

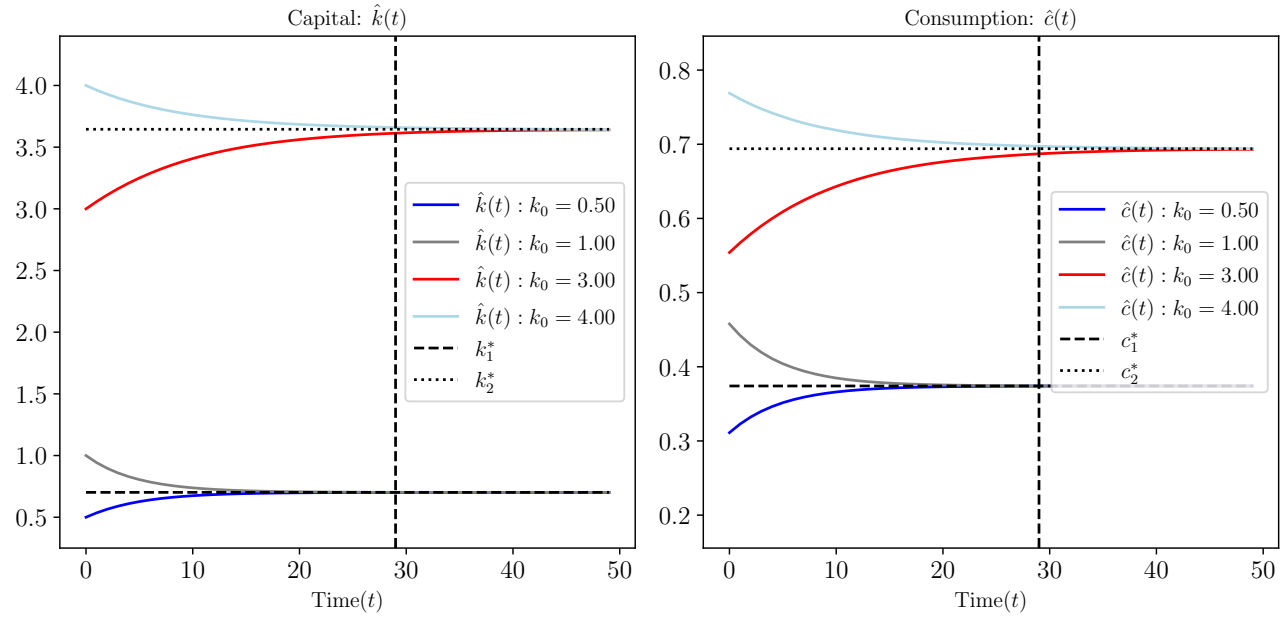

Figure 9: The approximate capital and consumption paths for the sequential neoclassical growth model with a convex-concave production function. The left panel shows the capital paths for four levels of initial capital $k_0 \in \{0.5, 1.0, 3.0, 4.0\}$ and the right panel shows the corresponding consumption paths. The dashed horizontal lines show the steady-states of capital k_1^* and k_2^* and the corresponding steady-states for consumption c_1^* and c_2^*. The dashed vertical line separates the interpolation from the extrapolation region.

provide a reliable solution in the vicinity of that critical point[23].

See the discussion in Appendix B.6 and Figure 28 for an analysis of the approximate solutions in the vicinity of the critical point.

3.5 Connection to turnpike theory

As discussed in the introduction, classic turnpike theorems provides conditions under which the solution of a finite-horizon dynamic problem is close to the infinite-horizon version of the problem. In particular, they emphasize how the dynamics local to a "destination" are not relevant if most of the time will be spent local to a stationary solution (i.e., the turnpike). In our paper, we are taking this further. Not only can we worrying about avoid local dynamics near the destination, but we can entirely focus on dynamics local to the origin—without even directly characterizing the steady-state.

Deep learning solutions and the turnpike In this paper we do not specify any terminal condition for the capital at the last point of the grid (i.e., t_N) and let the problem to be ill-posed

[23]See Benjamin Moll's lecture notes for a global treatment of this problem in a continuous time setup at https://benjaminmoll.com/wp-content/uploads/2020/06/skiba.pdf.

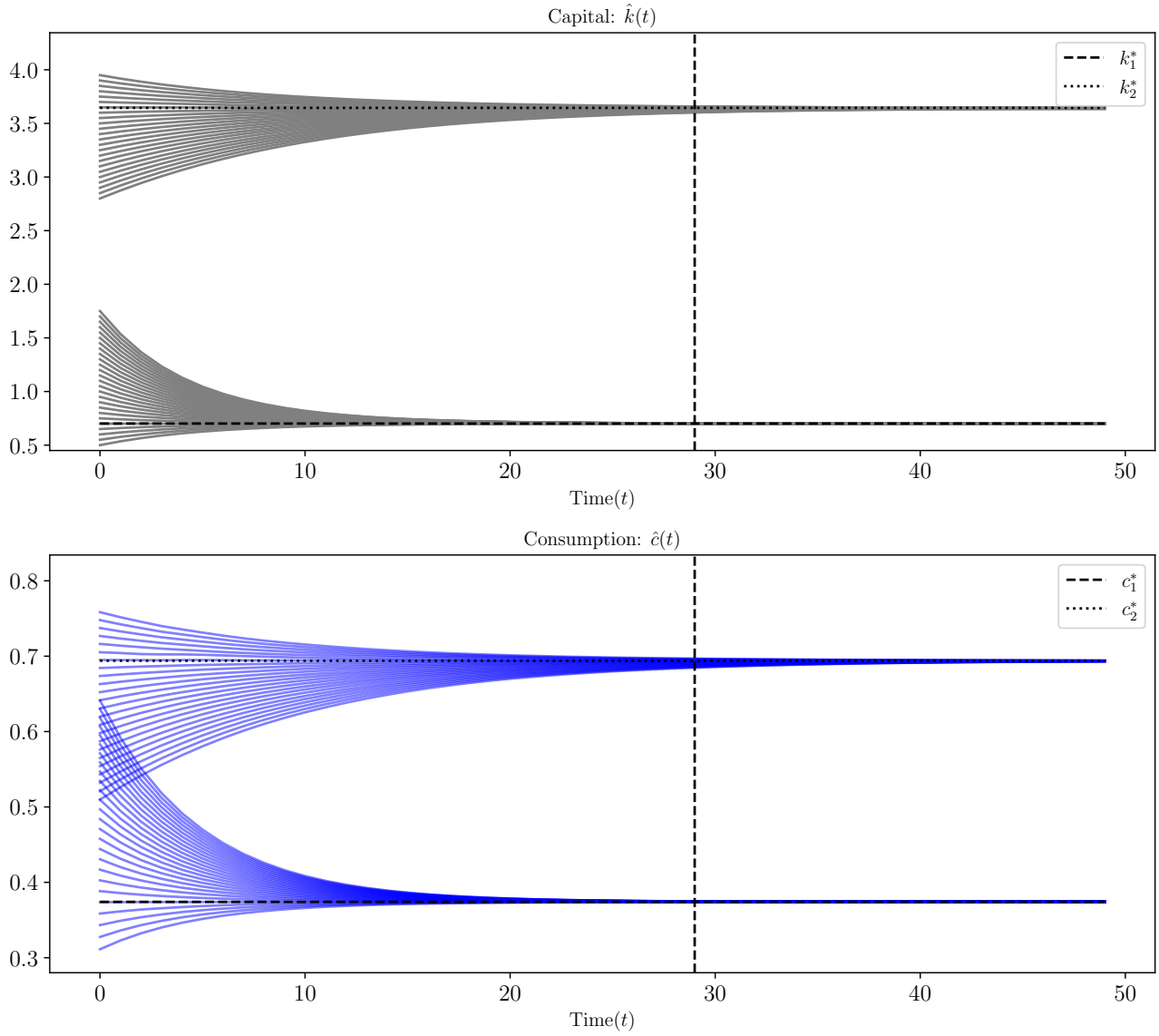

Figure 10: The approximate capital and consumption paths for the sequential neoclassical growth model with convex-concave production function. The grid for the initial condition of capital is from $[0.5, 1.75]$ and $[2.75, 4]$. The top panel shows the capital paths and the bottom panel shows the consumption paths. The dashed horizontal lines show the steady-states of capital k_1^* and k_2^* and the corresponding steady-states for consumption c_1^* and c_2^*. The dashed vertical line separates the interpolation from the extrapolation region.

and rely on the implicit bias of the deep neural networks to find the approximate path of the finite-horizon problem. As shown in Figure 5 the approximate capital path fully stays close to the path of infinite-horizon problem and does not deviate in the interpolation region. Moreover, in the extrapolation region it stays on the turnpike (at most 0.6% relative error for capital). As evident in Figure 6, these results are not sensitive to the time horizon used in $\mathcal{X}_{\text{train}}$. The approximate path for capital stays very close to the turnpike even when smaller time horizons are used (i.e.,

$t_N = 9$). As shown in Figure 7 and Figure 13, these results hold for non-stationary neoclassical growth model regardless of the functional specification of the growth. Therefore, deep learning solutions have a strong tendency toward the turnpikes (infinite-horizon paths) because of their implicit bias toward interpolating solutions with small derivatives.

4 The neoclassical growth model: recursive form

The sequential models outlined in the previous sections can be very useful to study deterministic setups and MIT shocks. However, they are not suitable for dealing with continuous shocks, such as shocks to total factor productivity. In this section we focus on the recursive version of the neoclassical growth model.

Setup: The Bellman equation for the neoclassical growth model can be written as:

$$v(k,z) = \max_{c,k'} \{u(c) + \beta v(k',z')\} \tag{33}$$

$$\text{s.t.} \quad k' = z^{1-\alpha} f(k) + (1-\delta)k - c \tag{34}$$

$$z' = (1+g)z \tag{35}$$

$$k' \geq 0, \tag{36}$$

where k is the capital, c is the consumption, and z is the total factor productivity. Capital depreciates with rate $\delta \in [0,1]$, the agent discounts the future with $\beta \in (0,1)$. We focus on the constant relative risk aversion utility functions $u(c) = \frac{c^{1-\sigma}}{1-\sigma}$, and production function of the form $f(k) = k^{\alpha}$.

The Euler equation for the recursive form of the neoclassical growth model can be written as:

$$u'(c) = \beta u(c') \big[z'^{1-\alpha} f'(k') + 1 - \delta \big], \tag{37}$$

where k' and c', z' are the capital, consumption and total factor productivity next period. Since this is a recursive model, k' and c are functions of capital k, and total factor productivity z, therefore, the domain of interest is $\mathcal{X} \equiv \mathbb{R}^2_+$. We are looking for functions $k'(k,z)$ and $c(k,z)$ such that they satisfy the Euler equation and the feasibility condition. Moreover, the optimality of the solution requires all the capital and consumption paths to satisfy the transversality condition for all initial conditions in $\mathcal{X}$:

$$0 = \lim_{T\to\infty} \beta^T u' \left(c^T(k_0,z_0) \right) k'^T(k_0,z_0) \quad \text{for all } (k_0,z_0) \in \mathcal{X}, \tag{38}$$

where $k'^T(k_0, z_0)$ and $c^T(k_0, z_0)$ are generated by iterating forward the optimal policy function for capital $k'(k, z)$, the consumption function $c(k, z)$, and the law of motion for total factor productivity up to the Tth period for a given k_0 and z_0.

4.1 Interpolating solution

As a generalization of the collocation approach one can pick a space of parametric functions $\mathcal{H}(\Theta)$, where $\Theta \equiv \{\theta_1, \ldots, \Theta_M\}$ represents the parameters. For instance, in a standard collocation method $\mathcal{H}(\Theta)$ can be the space of two-dimensional Chebyshev polynomials with M parameters.[24] The interpolating capital function $k(\cdot, \cdot; \theta) : \mathcal{X} \to R$ belongs to $\mathcal{H}(\Theta)$. In order to represent the domain X, one can pick a finite set of grid points $\mathcal{X}_{\text{train}} \subset X$. Since this is a two-dimensional recursive model:

$$\mathcal{X}_{\text{train}} \equiv \{k_1, \ldots, k_{N_k}\} \times \{z_1, \ldots, z_{N_z}\}.$$

This is a Cartesian product of sets of points in the capital and total factor productivity space. The Euler equation (i.e., equation 37), the feasibility condition (i.e, equation 34), the law of motion for the total factor productivity (i.e., equation 35), and the transversality condition (i.e., equation 38) form a system of equations. However, the transversality condition is a boundary condition at infinity. One can approximate the infinity by choosing a very large point in time, denoted by T. In the recursive case, T should be large enough such that the optimal capital path is very close to the steady-state at time T.

For a given policy function for capital $k'(k, z; \theta) \in \mathcal{H}(\Theta)$ we define a consumption function $c(k, z; k'(\cdot; \theta))$ through the feasibility condition:

$$c\big(k, z; k'(\cdot; \theta)\big) \equiv z^{1-\alpha} f(k) + (1 - \delta)k - k'(k, z; \theta). \tag{39}$$

.

Given a space of parametric functions $\mathcal{H}(\Theta)$, a grid $\mathcal{X}_{\text{train}}$, and a very large point in time T, one can find the interpolating policy function for capital by solving the following optimization problem

$$\min_{\theta \in \Theta} \left[\frac{1}{|\mathcal{X}_{\text{train}}|} \sum_{(k,z) \in \mathcal{X}_{\text{train}}} \left(\frac{u'\big(c\big(k, z; k'(\cdot; \theta)\big)\big)}{u'\big(c\big(k'(k, z; \theta), (1+g)z; k'(\cdot; \theta)\big)\big)} - \beta\big[\big((1+g)z\big)^{1-\alpha} f'\big(k'(k, z; \theta)\big) + (1 - \delta)\big] \right)^2 + \underbrace{\frac{1}{|\mathcal{X}_{\text{train}}|} \sum_{(k_0,z_0) \in \mathcal{X}_{\text{train}}} \left(\beta^T u' \left(c^T\big(k_0, z_0; k'(\cdot; \theta)\big) \right) k'^T\big(k_0, z_0; \theta\big) \right)^2}_{\text{Is this necessary?}} \right]. \tag{40}$$

[24] Since this is a two-dimensional problem, one should adjust the Chebyshev polynomials and Chebyshev nodes.

The first term in the optimization problem is the Euler residuals and the second term is a finite approximation of the transversality condition. The non-negativity of $k'(k, z; \theta)$ can be built into $\mathcal{H}(\Theta)$.

Choice of $\mathcal{X}_{\text{train}}$: similar to the previous cases the choice of $\mathcal{X}_{\text{train}}$ is not very crucial. However, we provide results for different grids. We also evaluate the interpolating policy function for capital outside of $\mathcal{X}_{\text{train}}$ to study the generalization performance of the solutions.

In standard collocation methods, where the number of parameters is equal to the number of grid points, this problem can be solved exactly as a system of equations with a boundary condition at T. It is worth noting that the interpolating solution depend on T. Therefore, the minimization problem described in (40) defines a sequence of solutions indexed by T.

Is the transversality condition necessary? Without the transversality condition the recursive version of the neoclassical growth has more than one solution. Therefore, in the absence of the second term (finite approximation of the transversality condition) the minimization problem described in (40) has more than one solutions that minimize the objective function. As noted by Fernández-Villaverde et al. (2016), without explicitly imposing the transversality condition, it is necessary to verify that solutions satisfy the transversality condition. This verification process is done after solving the minimization problem by generating capital and consumption paths from some initial conditions.

In section Section 4.4 we establish how using over-parameterized functions ($M \gg N_k \times N_z$) and their optimization processes yield an interpolating solution that does not violate the transversality condition. Intuitively, the policy functions for capital that leads to violation of the transversality condition have bigger derivatives than the optimal policy function for capital. With this result, we do not need to be worried about the transversality condition. Therefore, using over-parameterized functions we can drop the transversality condition and solve:

$$\min_{\theta \in \Theta} \frac{1}{|\mathcal{X}_{\text{train}}|} \sum_{(k,z) \in \mathcal{X}_{\text{train}}} \left(\frac{u'\big(c\big(k, z; k'(\cdot; \theta)\big)\big)}{u'\big(c\big(k'(k, z; \theta), (1+g)z; k'(\cdot; \theta)\big)\big)} - \beta\big[\big((1+g)z\big)^{1-\alpha} f'\big(k'(k, z; \theta)\big) + (1-\delta)\big] \right)^2, \tag{41}$$

to obtain an interpolating solution that does not violate the transversality condition.

4.2 Results

Generating capital and consumption paths: in the recursive version of the neoclassical growth, the interpolating solution is a policy function for capital $k'(k,z;\theta^*)$, where:

$$\theta^* \equiv \underset{\theta\in\Theta}{\operatorname{argmin}} \frac{1}{|\mathcal{X}_{\text{train}}|} \sum_{(k,z)\in\mathcal{X}_{\text{train}}} \left(\frac{u'\big(c\big(k,z;k'(\cdot;\theta)\big)\big)}{u'\big(c\big(k'(k,z;\theta),(1+g)z;k'(\cdot;\theta)\big)\big)} - \beta\big[\big((1+g)z\big)^{1-\alpha} f'\big(k'(k,z;\theta)\big) + (1-\delta)\big]\right)^2.$$

Given $k'(k,z;\theta^*)$, we construct the consumption function $c\,(k,z;k'(\cdot;\theta^*))$ defined by (39). In order to generate a capital path up to period t, we apply the interpolating policy function for capital and the law of motion for total factor productivity t times from given initial conditions k_0 and z_0. Similarly, a consumption path is constructed by applying the consumption function t times.

This procedure provides a pair of approximate capital and consumption paths. In this section, we solve the minimization problem described in (41) for 100 times. Each time with a different random initialization (random seed) of the parameters of the deep neural network, then we generate a pair of capital and consumption paths. This procedure provides 100 pairs of approximate capital and consumption paths. We report the median, the 10th and 90th percentiles of these paths. We also report the median, the 10th, and the 90th percentiles of the relative errors between these approximate paths and the paths generated by the value function iteration method.

Figure 11 shows the result of the minimization problem described in (41) for $\beta = 0.9$, $\alpha = 0.33$, $\delta = 0.1$, $g = 0$, $k_0 = 0.4$, and $z_0 = 1$.

In this experiment we utilize a grid with 16 points between $k_1 = 0.8$ and $k_{N_k} = 2.5$. We use a deep neural network with four hidden layers, each with 128 nodes. Each hidden layer uses Tanh and the output layer uses Softplus as activation functions. The solid curve in the top-left panel, denoted by $\hat{k}(t)$, shows the median of the approximate capital paths. The dashed curve, denoted by $k(t)$, shows the capital path obtained by the value function iteration method. The solid curve is accompanied with the 10th and 90th percentiles of the approximate capital paths. However, they are so close to each other that they are not distinguishable in the plot. The gray rectangle shows the interpolation region, i.e., $[0.8, 2.5]$. The top-right panel shows the median of the relative errors between the approximate capital paths and the capital path obtained by the value function iteration method. The shaded region shows the 10th and 90th percentiles of the relative errors. The dashed part of the curve shows the median of relative errors in the extrapolation region (i.e., the parts of the approximate capital path outside of $[0.8, 2.5]$) and the solid part shows the median of relative errors in the interpolation region (i.e., the parts of the approximate capital path inside of $[0.8, 2.5]$). The solid curve, denoted by $\hat{c}(t)$, in the bottom-left panel shows the median of the

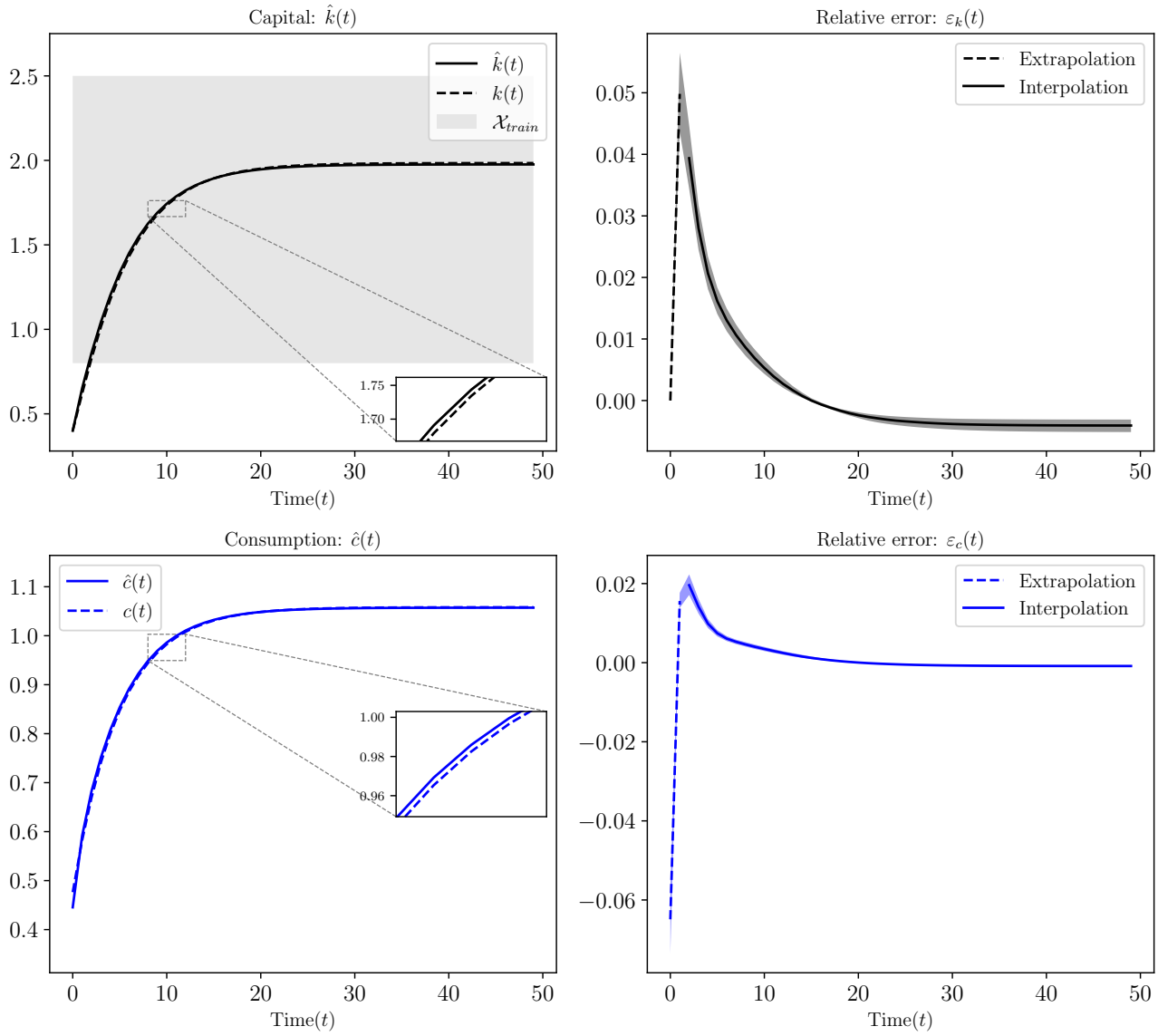

Figure 11: Comparison between the value function iteration and approximate solution using a deep neural network for the recursive neoclassical growth model. The left panels show the median of the approximate capital and consumption paths, denoted by $\hat{k}(t)$ and $\hat{c}(t)$. The dashed curves, denoted by $k(t)$ and $c(t)$, show the capital and consumption paths obtained by the value function iteration method. The right panels show the median of the relative errors between the approximate paths and the paths obtained by the value function iteration method. The shaded regions show the 10th and 90th percentiles. The dashed curves in the right panels show the relative errors for the parts of the approximate capital paths that lie outside of the interpolation regions. The gray rectangle in the top-right panel shows the interpolation region.

approximate consumption paths. The dashed curve shows the consumption path obtained by the value function iteration method. The bottom-right panel shows the median of the relative errors between the approximate consumption paths and the consumption path obtained by the value function iteration method. The shaded region shows the 10th and 90th percentiles of the

relative errors. The dashed part of the curve shows the median of relative errors for consumption where the corresponding levels of capital are in the extrapolation region and the solid part shows the median of relative errors for consumption where the corresponding levels of capital are in the interpolation region.

The results of this experiment are multi-fold. First, the implicit bias of deep neural networks yields a solution that automatically satisfies the transversality condition. Second, the solutions are accurate within $\mathcal{X}_{\text{train}}$. Third, given the initial level of capital $k_0 = 0.4$ lies outside of $\mathcal{X}_{\text{train}} = [0.8, 2.5]$, the generalization error are small (at most 5% for capital).

As stated in the original problem, the transversality condition should hold for all the initial level of capital, Figure 29 in Appendix C.1 shows the results for different initial conditions of capital both inside $\mathcal{X}_{\text{train}}$ and outside $\mathcal{X}_{\text{train}}$.

Far from the steady-state: in the last experiment, the steady-state for capital k^* lies inside of the interpolation region (i.e., $k^* \in [0.8, 2.5]$). In this experiment we investigate whether we can achieve accurate short-run dynamics by using a local grid around the initial condition for capital. More specifically, we use a grid $\mathcal{X}_{\text{train}} = \{k_1, \ldots, k_{N_k}\}$, where $k_{N_k} < k^*$.

Figure 12 shows the results for the recursive neoclassical growth using a local grid around the initial level of capital $k_0 = 0.9$. In this experiment we utilize a grid with 16 points between $k_1 = 0.8$ and $k_{N_k} = 1.5$. We use the same parameters and deep neural network as the previous experiment. The solid curve in the top-left panel, denoted by $\hat{k}(t)$, shows the median of the approximate capital paths. The dashed curve, denoted by $k(t)$, shows the capital path obtained by the value function iteration method. The shaded region shows the 10th and 90th percentiles of the approximate capital paths. The gray rectangle shows the interpolation region, i.e., $[0.8, 1.5]$. The top-right panel shows the median of the relative errors between the approximate capital paths and the capital path obtained by the value function iteration method. The shaded region shows the 10th and 90th percentiles of the relative errors. The dashed part of the curve shows the median of relative errors in the extrapolation region (i.e., the parts of the approximate capital path outside of $[0.8, 1.5]$) and the solid part shows the median of relative errors in the interpolation region (i.e., the parts of the approximate capital path inside of $[0.8, 1.5]$). The solid curve, denoted by $\hat{c}(t)$, in the bottom-left panel shows the median of the approximate consumption paths. The dashed curve shows the consumption path obtained by the value function iteration method. The shaded region shows the 10th and 90th percentiles of the approximate consumption paths. The bottom-right panel shows the median of the relative errors between the approximate consumption paths and the consumption path obtained by the value function iteration method. The shaded region shows the 10th and 90th percentiles of the relative errors. The dashed part of the curve shows the median of relative errors for consumption where the corresponding levels of capital are in the extrapolation region and the solid part shows the median of relative errors for consumption

where the corresponding levels of capital are in the interpolation region.

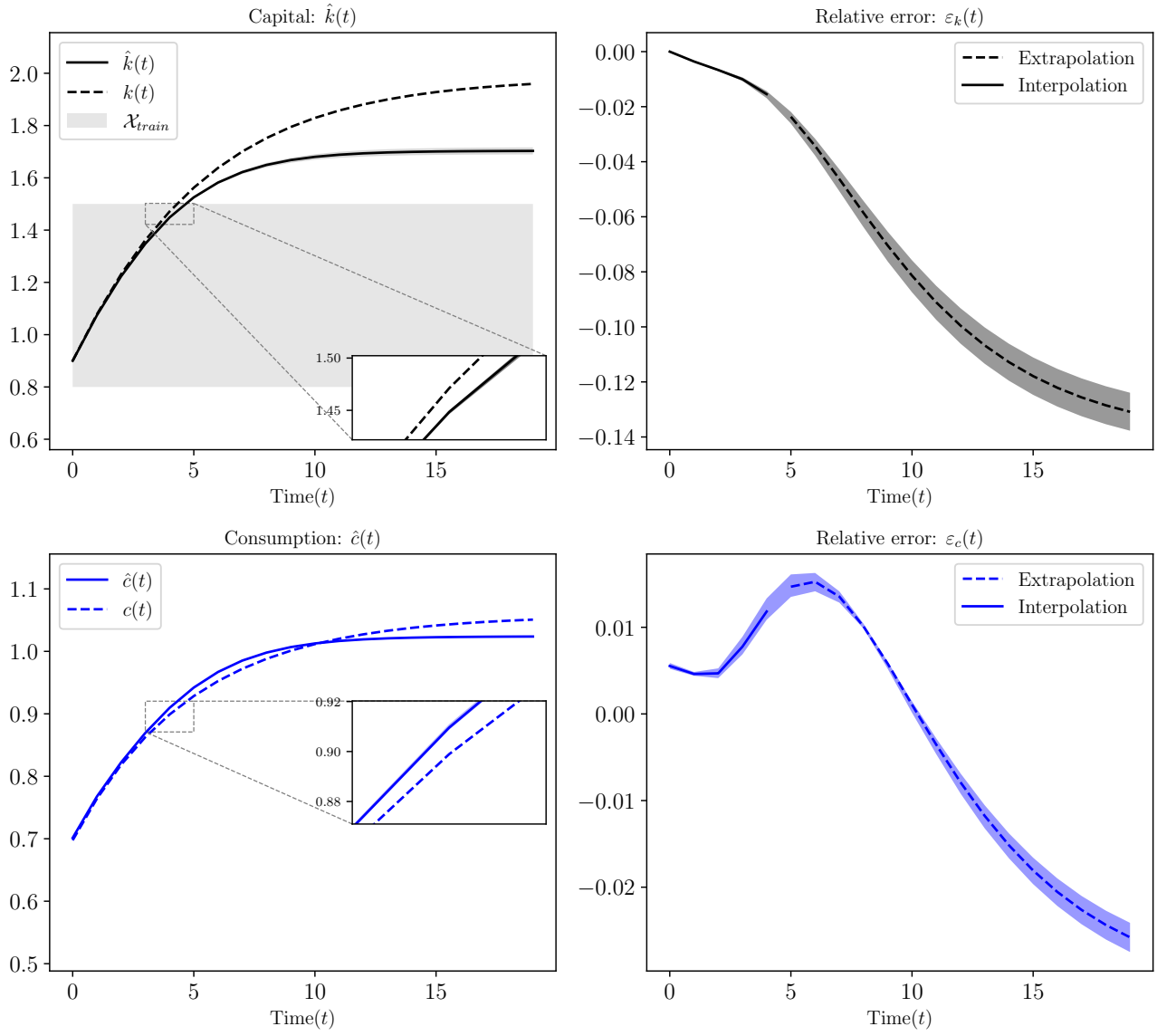

Figure 12: Comparison between the value function iteration and approximate solution using a deep neural network for the recursive neoclassical growth model using a local grid around the initial condition for capital $\mathcal{X}_{\text{train}} = \{0.8, \ldots, 1.5\}$. The left panels show the median of the approximate capital and consumption paths, denoted by $\hat{k}(t)$ and $\hat{c}(t)$. The dashed curves, denoted by $k(t)$ and $c(t)$, show the capital and consumption paths obtained by the value function iteration method. The right panels show the median of the relative errors between the approximate paths and the paths obtained by the value function iteration method. The shaded regions show the 10th and 90th percentiles. The dashed curves in the right panels show the relative errors for the parts of the approximate capital path that lie outside of the interpolation regions. The gray rectangle in the top-right panel shows the interpolation region.

These results show that one can achieve accurate short-run dynamics when the grid for capital is locally defined around the initial capital and is far from the steady-state. For the first few

periods the relative error for capital is less then 1%. Moreover, the relative errors stay bounded and do not grow excessively even after 20 periods.

Balanced growth path ($g > 0$): in this experiment we focus on the positive growth in total factor productivity (i.e. $g > 0$). We partially use an a priori economic intuition that the solution is homogeneous of degree one in z. We design the space of functions $\mathcal{H}(\Theta)$ such that it contains functions of the form

$$\hat{k}'(k, z; \theta) = zNN\left(\frac{k}{z}, z; \theta\right),$$

where $NN(\cdot, \cdot; \theta)$ is a deep neural network. We say partially, because the correctly specified functional form of a homogeneous of degree one is $zNN\left(\frac{k}{z}; \theta\right)$.

Figure 13 shows the results for recursive neoclassical growth for non-stationary total factor productivity (i.e., $g = 0.02$). We use 16 points in for $[0.8, 3.5]$ for capital, and 8 points in $[0.8, 1.8]$ for total factor productivity. The deep neural network $NN(\cdot, \cdot; \theta)$ is the same as the first experiment. The only difference is the linear term z.

The solid curve in the top-left panel, denoted by $\hat{k}(t)$, shows the median of the approximate capital paths. The dashed curve, denoted by $k(t)$, shows the capital path obtained by the value function iteration method. The shaded region shows the 10th and 90th percentiles of the approximate capital paths. The top-right panel shows the median of the relative errors between the approximate capital paths and the capital path obtained by the value function iteration method. The shaded region shows the 10th and 90th percentiles of the relative errors. The dashed part of the curve shows the median of relative errors in the extrapolation region (i.e., the parts of the approximate capital path outside of $[0.8, 3.5]$) and the solid part shows the median of relative errors in the interpolation region (i.e., the parts of the approximate capital path inside of $[0.8, 3.5]$). The solid curve, denoted by $\hat{c}(t)$, in the bottom-left panel shows the median of the approximate consumption paths. The dashed curve shows the consumption path obtained by the value function iteration method. The solid curve is accompanied with the 10th and 90th percentiles of the approximate consumption paths. However, they are so close to each other that they are not distinguishable in the plot. The bottom-right panel shows the median of the relative errors between the approximate consumption paths and the consumption path obtained by the value function iteration method. The shaded region shows the 10th and 90th percentiles of the relative errors. The dashed part of the curve shows the median of relative errors for consumption where the corresponding levels of capital are in the extrapolation region and the solid part shows the median of relative errors for consumption where the corresponding levels of capital are in the interpolation region.

The results of this experiment are multi-fold. First, the implicit bias of deep neural networks yields a solution that automatically satisfies the transversality condition. Second, the solutions are accurate within $\mathcal{X}_{\text{train}}$ (less than 1% relative error for capital in short- and medium-run).

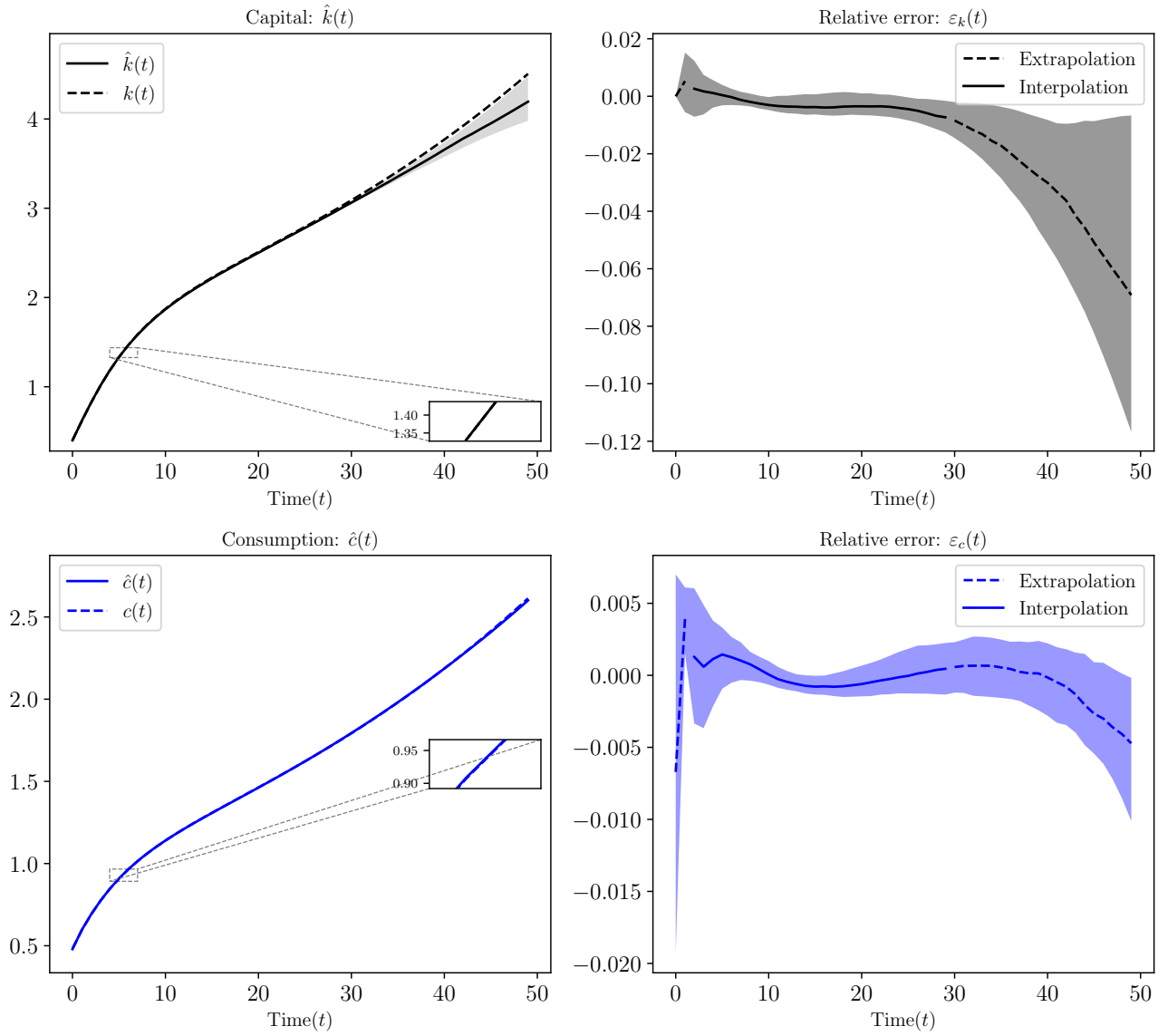

Figure 13: Comparison between the value function iteration and approximate solution using a deep neural network for the recursive neoclassical growth model with growth in total factor productivity (i.e., $g = 0.02$). We use $\mathcal{X}_{\text{train}} = [0.8, 3.5] \times [0.8, 1.8]$. The left panels show the median of the approximate capital and consumption paths, denoted by $\hat{k}(t)$ and $\hat{c}(t)$. The dashed curves, denoted by $k(t)$ and $c(t)$, show the capital and consumption paths obtained by the value function iteration method. The right panels show the median of the relative errors between the approximate paths and the paths obtained by the value function iteration method. The shaded regions show the 10th and 90th percentiles. The dashed curves in the right panels show the relative errors for the parts of the approximate capital path that lie outside of the interpolation regions.

Third, the solution generalizes well (less than 5% relative errors after 40 periods).

As stated in the original problem, the transversality condition should hold for all the initial conditions in X, Figure 30 in Appendix C.1 shows the results for different initial conditions of

capital both inside and outside of the interpolation region.

4.3 Neoclassical growth model with multiplicity of equilibria: recursive form

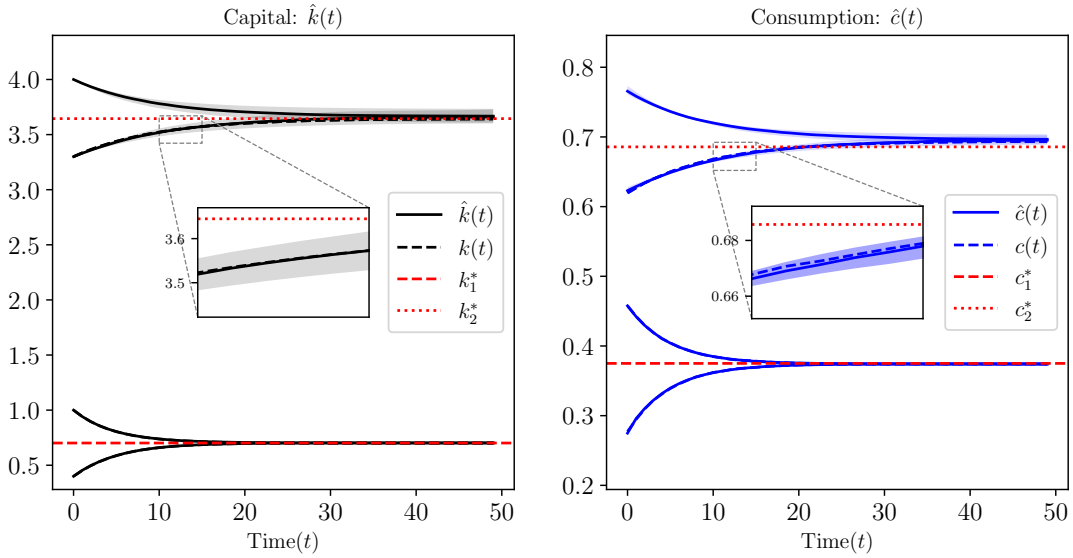

Figure 14

4.4 Minimum-norm interpretation and transversality condition

In this section we provide the connection between minimum-norm interpretation and the transversality condition for the case of recursive neoclassical growth. We focus on the case of zero total factor productivity growth (i.e., $g = 0$ and $z_0 = 1$).

As discussed in Section 3.3 the solutions that violate the transversality condition are associated with a sequence of consumption that approaches zero and a sequence of capital that approaches $\tilde{k}_{\max}$. Therefore, any policy function for capital $\tilde{k}'(k)$ that solves the Euler equation and feasibility condition, and violates the transversality condition, must have a fixed point at $\tilde{k}_{\max}$. Formally:

$$\tilde{k}'\left(\tilde{k}_{\max}\right) = \tilde{k}_{\max}.$$

The optimal policy function for capital $k'(k)$ has a fixed point at $k^* = \left(\frac{\beta^{-1}+\delta-1}{\alpha}\right)^{\frac{1}{\alpha-1}}$. Formally:

$$k'\left(k^*\right) = k^*.$$

Given the fact that $\tilde{k}_{\max} \gg k^*$, $\tilde{k}$ must intersect with 45 degree line way further to the right of k^*.

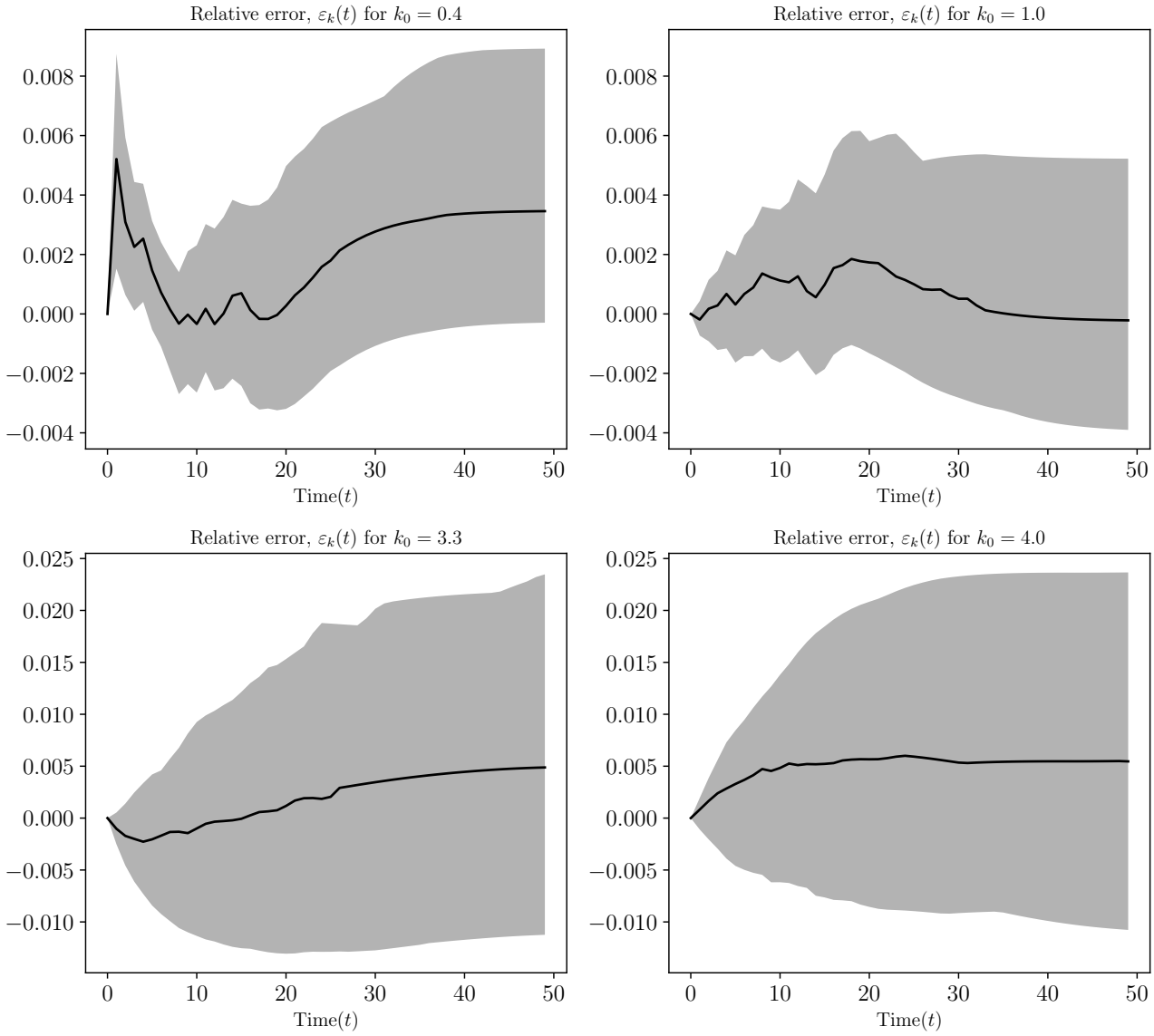

Figure 15

In Figure 17 the blue curve, denoted by $\tilde{k}'(k)$, shows a policy function for capital that violates the transversality condition and the black curve, denoted by $k'(k)$, shows the optimal policy function for capital. The dashed line shows the 45 degree line. $\tilde{k}'(k)$ intersects with the 45 degree line at $\tilde{k}_{\max} \approx 30$. [25]

The minimum-norm interpretation of the solutions of the minimization problem described in

[25] In this figure the solutions that violate the transversality condition (i.e., $\tilde{k}'(k)$) are generated by transforming the sequences of capitals that solve the sequential problem and violate the transversality condition.

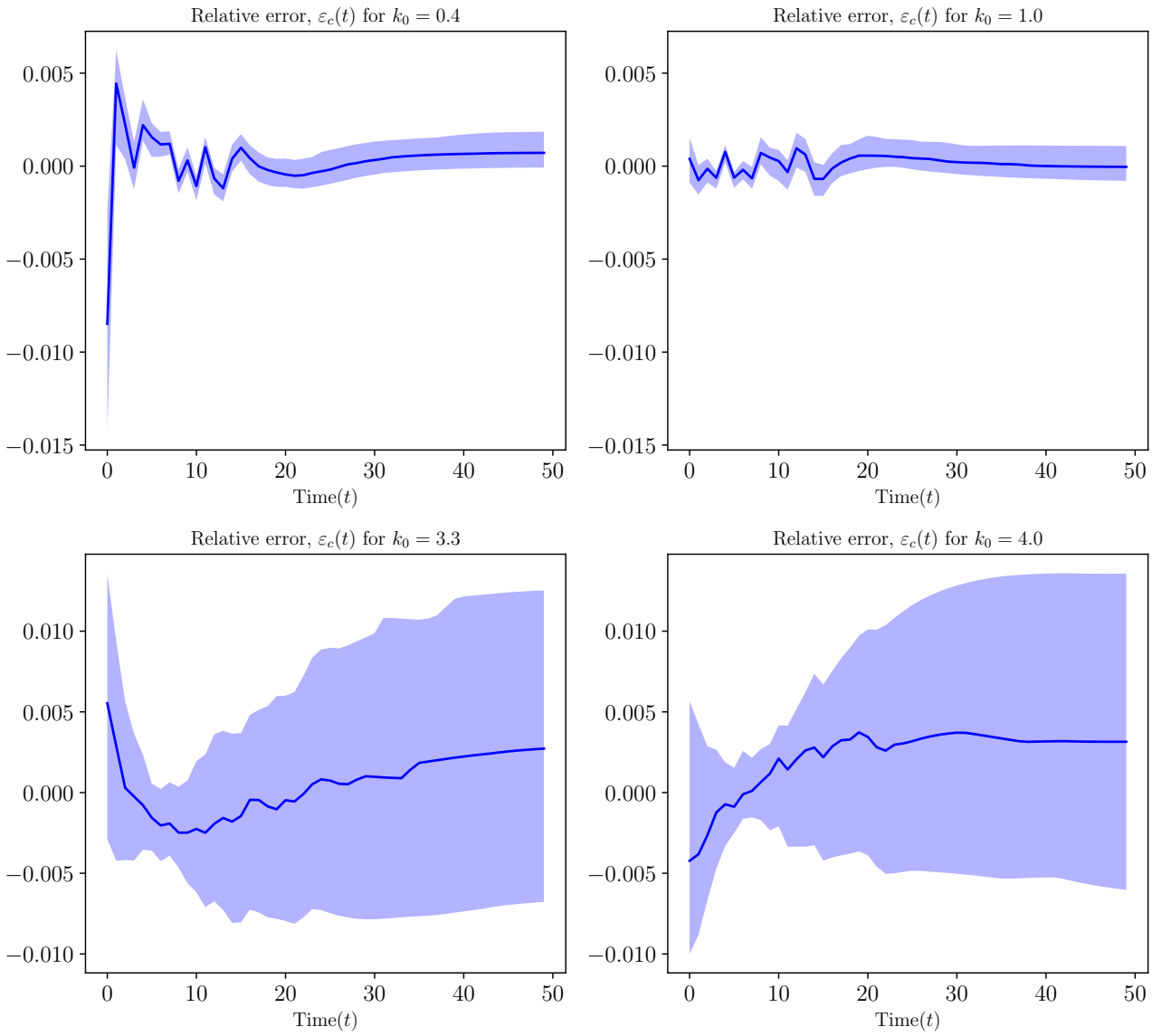

Figure 16

(40) for the limiting case of $T \to \infty$ can be written as

$$\min_{k'(\cdot;\theta)\in\mathcal{H}(\Theta)} \quad \|k'(\cdot;\theta)\|_S \tag{42}$$

$$\text{s.t.} u'\Big(c\big(k;k'(\cdot;\theta)\big)\Big) = \beta u'\Big(c\big(k'(k);k'(\cdot;\theta)\big)\Big)\Big[f'\big(k'(k;\theta)\big)+(1-\delta)\Big] \quad \text{for } k \in \mathcal{X}_{\text{train}} \tag{43}$$

$$k'(k;\theta) \geq 0 \quad \text{for } k \in \mathcal{X}_{\text{train}} \tag{44}$$

$$0 = \lim_{t\to\infty} \beta^T u'\left(c^T(k_0)\right) k'^T(k_0) \quad \text{for all } k_0 \in \mathcal{X}. \tag{45}$$

Since the total factor productivity is constant and is always equal to one, we replace $k'(k,z;\theta)$

by $k'(k;\theta)$ and $\mathcal{X}_{\text{train}}$ is defined as:

$$\mathcal{X}_{\text{train}} \equiv \{k_1, \ldots, k_{N_k}\}. \tag{46}$$

$\mathcal{H}(\Theta)$ is a space of over-parameterized functions and $\|\cdot\|_S$ satisfies Assumption 1.

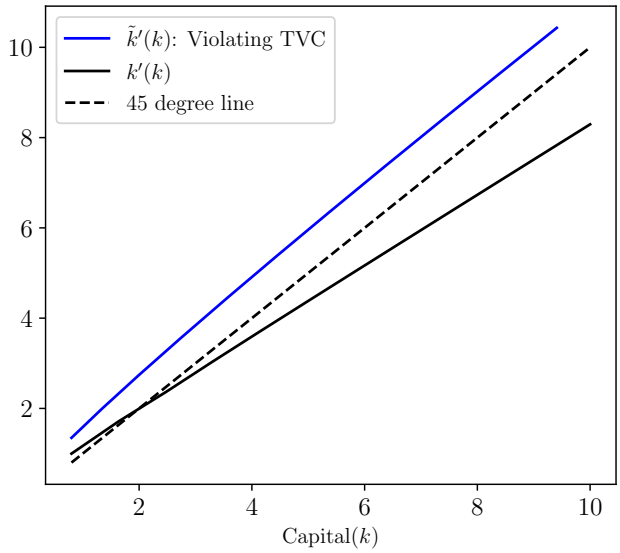

Figure 17: Comparison between the optimal solution, denoted by $k'(k)$, and a solution that violate the transversality condition, denoted by $\tilde{k}'(k)$, in the recursive neoclassical growth model. The solution that violates the transversality equation intersects with the 45 degree line at $\tilde{k}_{\max} \approx 30$.

As evident in Figure 17, $\tilde{k}'(k)$ has bigger derivatives than $k'(k)$. More specifically, for an arbitrary compact space of the form $[k_{\min}, k_{\max}]$:

$$\int_{k_{\min}}^{k_{\max}} \left|\frac{d\tilde{k}'}{dk}\right|^2 dk > \int_{k_{\min}}^{k_{\max}} \left|\frac{dk'}{dk}\right|^2 dk.$$

Therefore, by Assumption 1:

$$\|k'\|_S < \|\tilde{k}'\|_S.$$

Since the solutions that violate the transversality condition have bigger semi-norms, the over-parameterized interpolation eliminates those solutions. Therefore, the optimization problem described in (41) is enough to find the optimal policy function for capital $k'(k, z; \theta)$ without imposing any explicit regularity regarding the long-run behavior, such a finite approximation of the transversality condition.

5 Are Euler and value function residuals enough?

In dynamic models where there is multiplicity of solutions and some boundary conditions at infinity is required, finding the root of the functional operators such as Euler or value function residuals is not enough. Due to multiplicity, very low levels of residuals can be misleading and the interpolating solutions might represent non-optimal ones.

Minimizing the Euler residuals is not enough. In the recursive neoclassical growth model, we want to have a solution that minimizes the Euler residuals and satisfy the transversality condition. Equivalently, the optimal policy function for capital $k'(k)$ has to minimize the Euler residuals and have a fixed point at the steady-state k^*. As illustrated in Section 4.4, when we approximate the policy function for capital $k'(k)$ with a deep neural network, the implicit bias automatically picks the optimal solution. This happens because the policy functions for capital that violate the transversality condition have bigger derivatives.

Due to the trade off between the consumption function $c(k)$ and the policy function for capital $k'(k)$ through the feasibility condition, if $k'(k)$ has a bigger semi-norm then $c(k)$ has smaller semi-norm. Therefore, if one decides to approximate the consumption function instead of the policy function for capital they get solutions that violate the transversality condition. Equivalently, the corresponding policy function for capital $k'(k)$ has a fixed point at $\tilde{k}_{\max}$ instead of the steady-state k^*. We focus on the case of constant total factor productivity (i.e., $g = 0$ and $z_0 = 1$) and in the notation drop the total factory productivity as an input. The optimization we solve for approximating the consumption function for capital can be written as

$$\min_{\theta\in\Theta} \frac{1}{|\mathcal{X}_{\text{train}}|} \sum_{k\in\mathcal{X}_{\text{train}}} \left(\frac{u'\big(c(k;\theta)\big)}{u'\Big(c\big(k'(k;c(\cdot;\theta));\theta\big)\Big)} - \beta \left[f'\Big(k'\big(k;c(\cdot;\theta)\big)\Big) + (1-\delta) \right] \right)^2, \tag{47}$$

where for a given consumption function $c(k;\theta) \in \mathcal{H}(\Theta)$, the policy function for capital is defined as:

$$k'\big(k;c(\cdot;\theta)\big) \equiv f(k) + (1-\delta)k - c(k;\theta).$$

The policy function for capital is defined exactly and not approximated. The consumption function $c : \mathbb{R} \to \mathbb{R}_+$ is approximated by a deep neural network. Similar to the previous cases the grid for capital is defined as $\mathcal{X}_{\text{train}} \equiv \{k_1, \ldots, k_{N_k}\}$.

Figure 18 shows the comparison between approximating the policy function for capital $k'(k)$ versus approximating the consumption function $c(k)$ with a deep neural network. The economic parameters are the same as before. The solid curve in the top-left panel shows the median of the Euler residuals squared when $k'(k)$ is approximated with a deep neural network.[26] In this case,

[26] We solve the minimization problem described in (47) for 100 times, each time with a different random initial-

the Euler residual is defined as:

$$\varepsilon_E^k(k;\theta) \equiv \frac{u'\big(c\big(k;k'(\cdot;\theta)\big)\big)}{u'\Big(c\big(k'(k;\theta);k'(\cdot;\theta)\big)\Big)} - \beta\left[f'\big(k'(k;\theta)\big) + (1-\delta)\right],$$

where the superscript k denotes the Euler residuals when the policy function for capital is approximated with a deep neural network. The shaded region around the solid curve shows the 10th and the 90 percentiles of the Euler residuals squared. The bottom-left panel shows the median of the approximate policy functions for capital, when $k'(k)$ is approximated with a deep neural network. The figure also contains the 10th and the 90 percentiles of the approximate policy functions for capital. However, they are so close to each other that they are not distinguishable in the plot. In this case the approximate policy function for capital has a fixed point at $k^* \approx 2.0$. The solid curve in the top-right panel shows the median of the Euler residuals squared when $c(k)$ is approximated with a deep neural network. In this case, the Euler residual is defined as:

$$\varepsilon_E^c(k;\theta) \equiv \frac{u'\big(c(k;\theta)\big)}{u'\Big(c\big(k'(k;c(\cdot;\theta));\theta\big)\Big)} - \beta\left[f'\big(k'\big(k;c(\cdot;\theta)\big)\big) + (1-\delta)\right], \tag{48}$$

where the superscript c denotes the Euler residuals when the consumption function is approximated with a deep neural network. The shaded region shows the 10th and the 90 percentiles of the Euler residuals squared. The bottom-right panel shows the median of the approximate policy functions for capital, when $c(k)$ is approximated with a deep neural network. The shaded region shows the 10th and the 90 percentiles of the approximate policy functions for capital. In this case the approximate policy function for capital has a fixed point at $\tilde{k}_{\max} \approx 30$.

These results show that when the policy function for capital is approximated with a deep neural network, $k'(k)$ has a fixed point at $k^* \approx 2.0$ and it does not violate the transversality condition (the bottom-left panel). However, when the consumption function is approximated with a deep neural network, the policy function for capital has a fixed point at $\tilde{k}_{\max} \approx 30$ and it violates the transversality condition (the bottom-right panel). As shown in the bottom-right panel, this result is robust to changing different random initialization of the parameters of the deep neural network. Most importantly, the Euler residuals for the solutions that violate the transversality condition (the top-right panel) are systematically and substantially lower than the solution that does not violate the transversality condition. Therefore, having low Euler residuals is not sufficient for convergence to the optimal solution and can be very misleading.

From a theoretical perspective, a variation of the transversality condition is required to guarantee the value function is the unique solution of the Bellman formulation of the neoclassical growth model, see Le Van and Morhaim (2002) and Stokey et al. (1989).

ization of the parameters of the deep neural networks. We report the median, 10th, and 90th percentiles of these solutions.

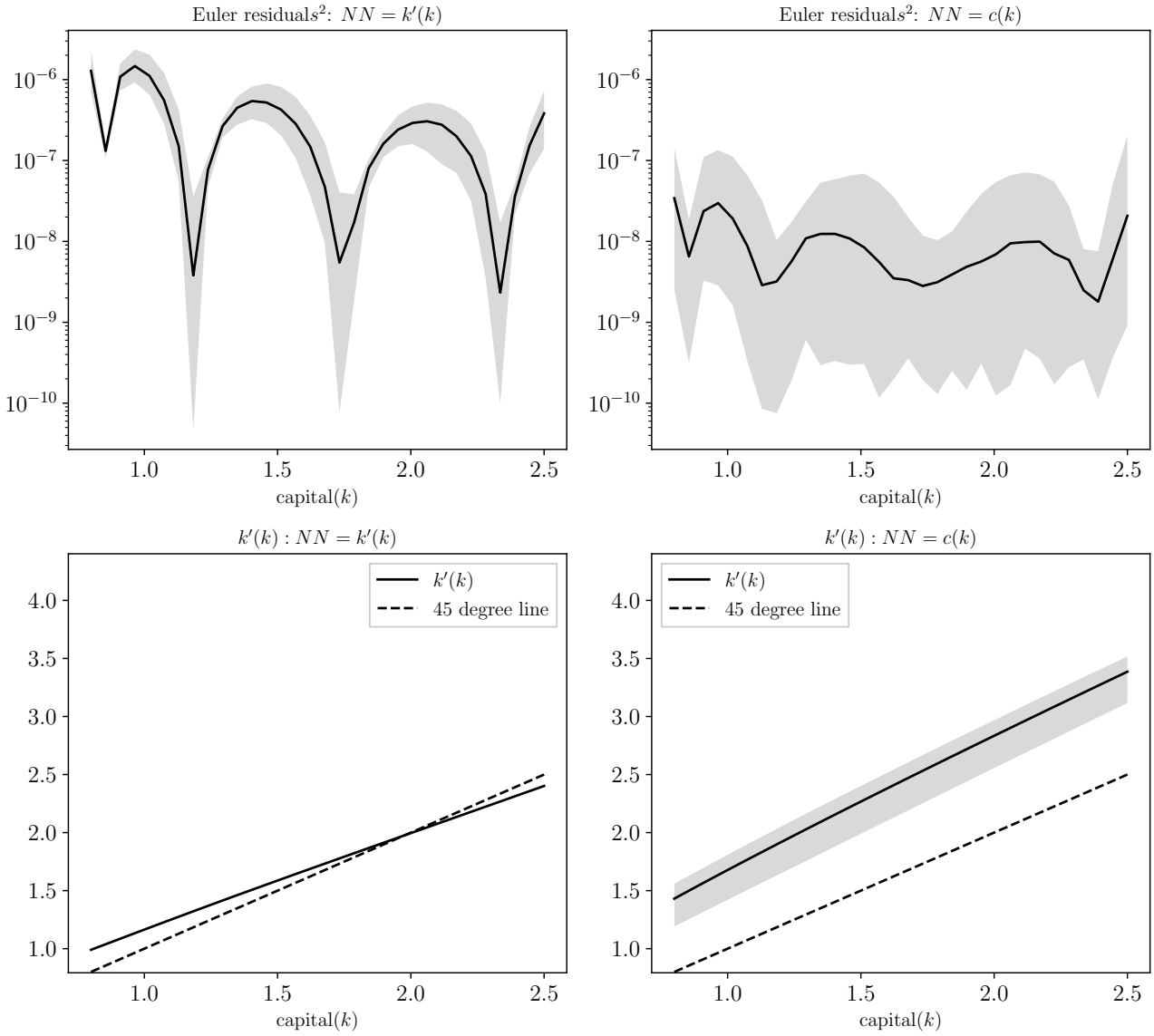

Figure 18: Comparison between approximating the policy function for capital $k'(k)$ versus the consumption function $c(k)$ with a deep neural network. The left panels show the median of Euler residuals and policy functions for capital when $k'(k;\theta)$ is approximated with a deep neural network. The right panels show the median of Euler residuals squared and policy functions for capital when $c(k)$ is approximated with a deep neural network. The solid curves show the medians and the shaded regions show the 10th and 90th percentiles over 100 different seeds.

Can explicit regularization solve the problem? As shown in Figure 18 approximating the consumption function $c(k)$ instead of policy function for capital leads to solutions that violate the transversality condition. One natural question that arises is whether explicit regularization of the parameters of the deep neural network can fix this problem. One of the most common methods of explicit regularization is L_2 regularization. This regularization is achieved by penalizing the

L_2 norm of the parameters of the neural networks (i.e., $\sum_{\theta_i \in \Theta} \theta_i^2$).[27] Therefore, the minimization takes the following form

$$\min_{\theta \in \Theta} \left[\frac{1}{|\mathcal{X}_{\text{train}}|} \sum_{k \in \mathcal{X}_{\text{train}}} \left(\frac{u'\big(c(k;\theta)\big)}{u'\Big(c\big(k'(k;c(\cdot;\theta));\theta\big)\Big)} - \beta \left[f'\big(k'\big(k;c(\cdot;\theta)\big)\big) + (1-\delta) \right] \right)^2 + \lambda \sum_{\theta_i \in \Theta} \theta_i^2 \right],$$

where λ is the penalization coefficient. We solve this minimization, with and without the explicit regularization, for 100 times. Each time with a different random initialization of the parameters of the deep neural network. We report the median, the 10th, and 90th percentiles of the results.

Figure 19 shows the results for approximating the consumption function with and without regularization. The left panels show the results without regularization. The right panels show the results with L_2 regularization with penalization coefficient $\lambda = 10^{-7}$. The solid curve in the top panels shows the median of the Euler residuals squared defined in equation (48), and the shaded regions show the 10th and 90th percentiles of the Euler residuals squared. The solid curves in the bottom panels shows the median of the policy functions for capital, i.e., $k'\left(k;c(\cdot;\theta)\right)$ and the shaded regions show the 10th and 90th percentiles of the approximate policy functions for capital.[28]

These results show that explicit regularization cannot lead to solutions that do not violate the transversality condition. However, this regularization reduces the variation of the policy functions for capital.

Minimizing the Bellman residuals is not enough. Similarly, for the sequential models such as linear asset pricing low residual errors can be misleading. As noted before the solutions can be written as:

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

From the theoretical perspective, given that the dividends are positive, prices should be non-negative for every time period. Therefore, negative values of ζ are not allowed and as showed in Proposition 1 the price based on the fundamentals has the lowest semi-norm for all non-negative values of ζ. However, when $\mathcal{X}_{\text{train}} = \{t_1, \ldots, t_N\}$ only contains small values, negative values of ζ can leak into the interpolating solutions. It is important to note that they are still interpolating solutions and solve the optimization problem accurately. This is because when t_N is small the solutions with negative ζ have smaller semi-norms.

We solve the minimization problem described in (12) for 100 times. Each time with a different random initialization of the parameters of the deep neural network. We report the median, the 10th, and the 90th percentiles of the results.

Figure 20 shows the results for the sequential asset pricing model for $\mathcal{X}_{\text{train}} = \{0, 1, \ldots, 14\}$.

[27] This is also called weight decay, and it is very easy to implement in PyTorch and TensorFlow.

[28] Here we use Adam optimizer. This kind of regularization is not feasible in L-BFGS.

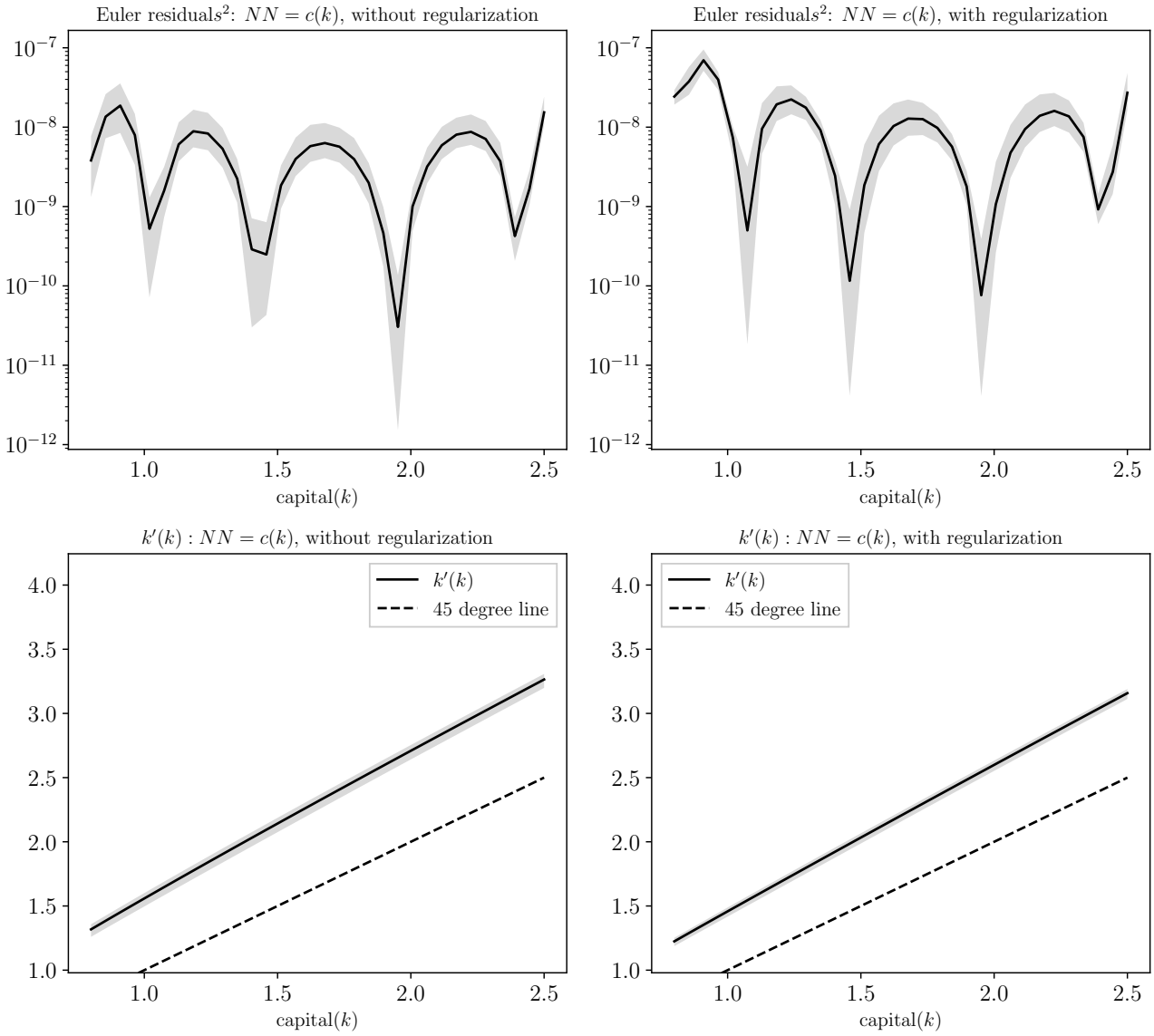

Figure 19: The effect of the L_2 explicit regularization on the solutions, when the consumption function $c(k)$ is approximated with a deep neural network. The left panels show the results without regularization. The right panels show the results with L_2 regularization with penalization coefficient $\lambda = 10^{-9}$. The solid curves in the top panels show the median of the Euler residuals squared. The solid curves in the bottom panels show the median of the approximate policy functions for capital, i.e., $k'\left(k; c(\cdot\theta)\right)$. The shaded regions show the 10th and 90th percentiles.

The parameters and the deep neural network is the same as before. The dashed curve in the left panel shows the price based on the fundamentals and the solid curve shows the median of the approximate prices. The shaded region shows the 10th and 90th percentiles of the approximate prices. The solid curve in the right panel shows the median of the relative errors. The shaded region shows the 10th and 90th percentiles of the relative errors.

These results show the leakage of negative value of ζ into the interpolating solutions. Since ζ

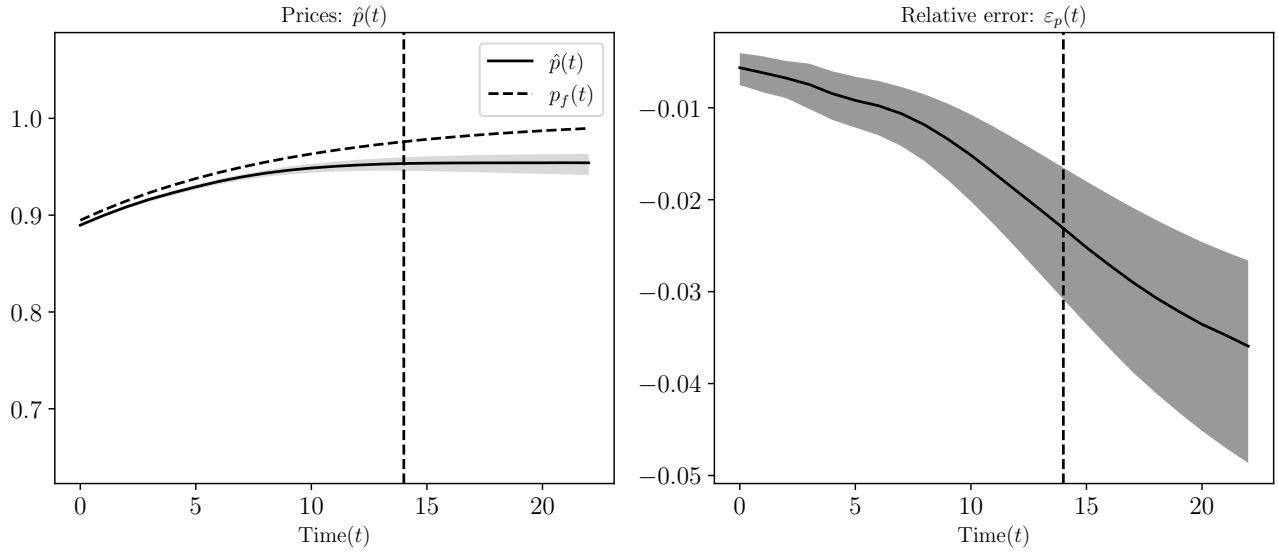

Figure 20: Comparison between the accurate and approximate solution using a deep neural network for the sequential linear asset pricing model for small t_N. The dashed vertical lines separate the interpolation from the extrapolation region. In the left panel the solid curve shows the median of th approximate prices. The dashed curve shows the price based on the fundamentals. The solid curve in the right panel shows the median of relative errors between the approximate price and the price based on the fundamentals. The shaded regions show the 10th and the 90th percentiles of the relative errors.

is negative and non-negativity of $p(\cdot;\theta)$ is built into $\mathcal{H}(\Theta)$, the approximate solutions are always bounded between 0 and $p_f(t)$.

Figure 21 is the result of the same experiment with $\mathcal{X}_{\text{train}} = \{0, 1, \ldots, 9\}$. Since the approximate prices solve the interpolation problem accurately, then each of them can be written as:

$$\hat{p}(t) = p_f(t) + \hat{\zeta}\beta^{-t}.$$

By setting $t = 0$ we can find the corresponding $\hat{\zeta}$ as:

$$\hat{\zeta} = \hat{p}(0) - p_f(0).$$

The solid blue line shows the median of $\hat{p}(t) - \hat{\zeta}\beta^{-t}$. If each solution solves the interpolation problem accurately, then $\hat{p}(t) - \hat{\zeta}\beta^{-t}$ must be very close to $p_f(t)$. The blue shaded region shows the 10th and 90th percentiles for $\hat{p}(t) - \hat{\zeta}\beta^{-t}$.

These results confirm that when t_N is small (in this example $t_{\text{N}} = 9$) the approximate solution accurately solves the interpolation problem. This is evident from observing that $p_f(t) \approx \hat{p}(t) - \hat{\zeta}\beta^{-t}$ very accurately in $\mathcal{X}_{\text{train}}$. The 10th and the 90th percentiles for $\hat{p}(t) - \hat{\zeta}\beta^{-t}$ are very close to each other. This happens because the variations in the approximate prices are caused by very small variations in $\hat{\zeta}$. Moreover these results confirm that the leakage of negative values of ζ

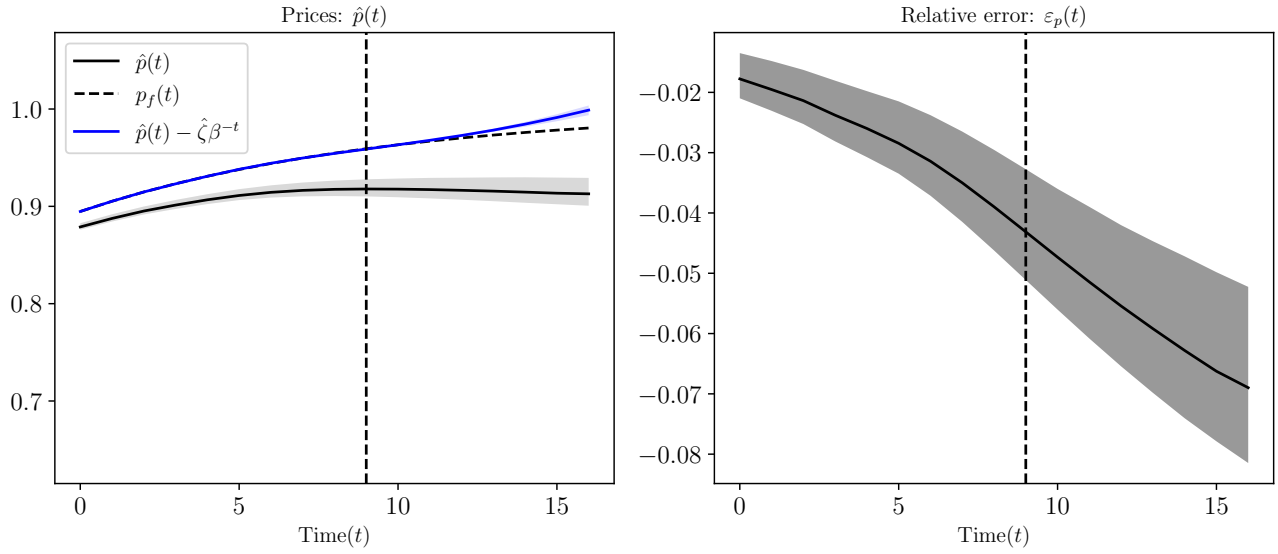

Figure 21: Analyzing the approximate solutions of the sequential linear asset pricing model for small t_{N}. In this experiment $\mathcal{X}_{\text{train}} = \{0, 1, \ldots, 9\}$. The dashed vertical lines separate the interpolation from the extrapolation region. In the left panel the black solid curve, denoted by $\hat{p}(t)$, shows the median of the approximate prices. The dashed curve, denoted by $p_f(t)$, shows the price based on the fundamentals. The blue solid curve shows the median of $\hat{p}(t) - \hat{\zeta}\beta^{-t}$. The right panel shows the difference between the approximate prices and the price based on the fundamentals. The shaded regions show the 10th and 90th percentiles.

gets more severe for smaller values of t_N. However, the approximate solutions are still bounded between 0 and $p_f(t)$. Therefore, the bias is always downward.

6 Implicit bias of deep neural networks

For a given space of over-parameterized functions $\mathcal{H}(\Theta)$, a residual $\ell(\cdot, \cdot)$, and a grid $\mathcal{X}_{\text{train}}$ the interpolation problem can be written as

$$\min_{\Psi(\cdot;\theta)\in\mathcal{H}(\Theta)} \frac{1}{|\mathcal{X}_{\text{train}}|} \sum_{x\in\mathcal{X}_{\text{train}}} \ell(\Psi(\cdot;\theta), x)^2. \tag{49}$$

Since in over-parameterized interpolation problems, such as interpolation with deep neural networks, the number of parameters is larger than the number of grid points there are many interpolating solutions that can achieve $\frac{1}{|\mathcal{X}_{\text{train}}|} \sum_{x\in\mathcal{X}_{\text{train}}} \ell(\Psi(\cdot;\theta), x)^2 = 0$.

Discussion regarding the validity of Assumption 1. Over-parameterized function approximation, specially deep learning, has shown an incredible generalization power, see Zhang et al. (2021). Given that they typically have many more parameters than data points, they are capable

of achieving exact interpolation. One might expect this exact interpolation leads to over-fitting and hence poor generalization power. However, in many empirical settings it has been shown that is not the case, to name a few see Belkin et al. (2019), Neyshabur et al. (2014).

One approach to avoid over-fitting in over-parameterized problems is to use explicit regularization. The two most common methods are L_1 and L_2 regularization. L_1 regularization corresponds to penalizing the sum of the absolute values of the parameters in the approximating function (similar to Lasso regression). L_2 regularization corresponds to penalizing the sum of the squared of parameters (similar to ridge regression). Surprisingly, it has been shown that, even without any explicit regularization, they can still achieve good generalization.

These two observations have led researches to believe that the optimization methods used in over-parameterized interpolation posses an intrinsic implicit bias toward a specific class of solutions, for example see Neyshabur et al. (2017), Arora et al. (2019), and Smith et al. (2021). Understanding the nature of this implicit bias is still an active field of research in computer science, optimization theory, and statistics. However, it is mostly believed that the implicit bias is caused by the first order optimization methods such as gradient descent and stochastic gradient descent, which are widely used in over-parameterized interpolation problems. It has been established that the first order optimization methods favor flat minima in the space of parameters. In this context the flatness/sharpness of the minima is characterized by the trace of the Hessian matrix of the objective function. For instance, in regression problems, Damian et al. (2021) show that the stochastic gradient descent minimizes

$$\|\Psi(\cdot;\theta)\|_S \equiv -\frac{1}{\eta}\mathrm{tr}\log\big(I - \frac{\eta}{2}\frac{1}{|\mathcal{X}_{\text{train}}|}\sum_{x\in\mathcal{X}_{\text{train}}}\nabla^2_\theta \ell(\Psi(\cdot;\theta),x)^2\big),$$

where I is the $M\times M$ identity matrix, η is the size of the optimization step (learning rate), and $\nabla^2_\theta\ell(\Psi(\cdot;\theta),x)^2$ is the Hessian of the objective function. In the interpolation regime, where zero of the objective function (i.e., the objective function defined in 49) is reached, there is a tight connection between the trace of the Hessian and the gradient of the interpolating function with respect to its parameters. Blanc et al. (2020) show that the stochastic gradient descent minimizes

$$\|\Psi(\cdot;\theta)\|_S \equiv \frac{1}{|\mathcal{X}_{\text{train}}|}\sum_{x\in\mathcal{X}_{\text{train}}}\|\nabla_\theta\Psi(x;\theta)\|_2^2,$$

which is the L_2 norm of the the gradient of the interpolating function with respect to the parameters.

Convergence to a flat minima in the space of parameters does not provide enough information about the properties of the function with respect to its inputs such as smoothness and small derivatives. Due to the multiplicative structure of deep neural networks, Ma and Ying (2021) observe that there is a tight connection between the flatness of a minima and the norm of the

gradient of the function with respect to its inputs. More specifically, in an interpolation regime, stochastic gradient descent as an optimization algorithm implicitly minimizes the Sobolev semi-norm of the approximating function. In other words deep neural networks along with their optimizers have a tendency to find approximating functions that have small derivatives (in our case not explosive).

Here we provide the mathematical definition of Sobolev $1-2$ semi-norm for a univariate function.

Definition 1 (Sobolev 1-2 seminorm)**.** *Let Ψ be a univariate function from a compact space in $\mathcal{X} \subset \mathbb{R}$ to $\mathbb{R}$, Sobolev 1-2 semi-norm is defied as:*

$$\|\Psi\|_{1,2,\mathcal{X}} \equiv \left(\int_{\mathcal{X}} \left| \frac{d\Psi}{dx} \right|^2 dx \right)^{\frac{1}{2}}.$$

Therefore, among two differentiable interpolating functions Ψ_1 and Ψ_2 such that

$$\left(\int_{\mathcal{X}} \left| \frac{d\Psi_1}{dx} \right|^2 dx \right)^{\frac{1}{2}} > \left(\int_{\mathcal{X}} \left| \frac{d\Psi_2}{dx} \right|^2 dx \right)^{\frac{1}{2}},$$

there is an implicit bias toward Ψ_2.

It is worth mentioning that Maennel et al. (2018) study the case of a one hidden layer neural network with ReLU activation function ($\text{ReLU}(x) = \max\{0, x\}$) with low-dimensional data points. They show, when the initial parameters are set to have small values, the gradient descent converges to a data dependent (as opposed to network size dependence) linear interpolation which is an indication of an implicit bias toward low Sobolev semi-norms.

Remark: all the theoretical and empirical results in the literature regarding the implicit bias of over-parameterized interpolation focus on regression and classification problems. The interpolation problems we solve in theoretical equilibrium models have a different nature and the objective functions (such as Bellman or Euler residuals) are slightly different from regression problems. However, as verified in the result sections of this paper, we strongly believe the same results can be proved for the sort of problems we solve.

7 Conclusion

Dynamic models of forward-looking agents in economics require conditions on long-run behavior (e.g, transversality and no-bubble condition). These conditions rule out explosive solutions. In this paper, we show that how deep learning approximations automatically satisfy these conditions without explicitly imposing them through calculating the steady-state even in models with steady-state multiplicity. What makes deep learning approximations special is its inherent

implicit bias toward functions with small derivatives that match the class of solutions we are interested in dynamic models. These results suggest that deep learning may let us calculate accurate transition dynamics with high-dimensional state spaces without directly solving for the boundary conditions at infinity.

References

Arora, S., N. Cohen, W. Hu, and Y. Luo (2019): "Implicit regularization in deep matrix factorization," *Advances in Neural Information Processing Systems*, 32.

Athey, S. and G. W. Imbens (2019): "Machine Learning Methods That Economists Should Know About," *Annual Review of Economics*, 11, 685–725.

Auerbach, A. J., L. J. Kotlikoff, et al. (1987): *Dynamic fiscal policy*, Cambridge University Press.

Azinovic, M., L. Gaegauf, and S. Scheidegger (2022): "DEEP EQUILIBRIUM NETS," *International Economic Review*.

Belkin, M. (2021): "Fit without fear: remarkable mathematical phenomena of deep learning through the prism of interpolation," *Acta Numerica*, 30, 203–248.

Belkin, M., D. Hsu, S. Ma, and S. Mandal (2019): "Reconciling modern machine-learning practice and the classical bias–variance trade-off," *Proceedings of the National Academy of Sciences of the United States of America*, 116, 15849–15854.

Blanc, G., N. Gupta, G. Valiant, and P. Valiant (2020): "Implicit regularization for deep neural networks driven by an Ornstein-Uhlenbeck like process," *Proceedings of Machine Learning Research vol*, 125, 1–31.

Brunnermeier, M. K. (2017): *Bubbles*, London: Palgrave Macmillan UK, 1–8.

Calvano, E., G. Calzolari, V. Denicolo, and S. Pastorello (2020): "Artificial intelligence, algorithmic pricing, and collusion," *American Economic Review*, 110, 3267–97.

Childers, D., J. Fernández-Villaverde, J. Perla, C. Rackauckas, and P. Wu (2022): "Differentiable State Space Models and Hamiltonian Monte Carlo Estimation," *NBER Working Paper*.

Damian, A., T. Ma, and J. D. Lee (2021): "Label noise sgd provably prefers flat global minimizers," *Advances in Neural Information Processing Systems*, 34, 27449–27461.

Duarte, V. (2018): "Machine learning for continuous-time finance," Tech. rep., Working paper.

Duffy, J. and P. D. McNelis (2001): "Approximating and simulating the stochastic growth model: Parameterized expectations, neural networks, and the genetic algorithm," *Journal of Economic Dynamics and Control*, 25, 1273–1303.

Ebrahimi Kahou, M., J. Fernández-Villaverde, J. Perla, and A. Sood (2021): "Exploiting Symmetry in High-Dimensional Dynamic Programming," Working Paper 28981, National Bureau of Economic Research.

Ekeland, I. and J. A. Scheinkman (1986): "Transversality conditions for some infinite horizon discrete time optimization problems," *Mathematics of operations research*, 11, 216–229.

Fernández-Villaverde, J., J. F. Rubio-Ramírez, and F. Schorfheide (2016): "Solution and estimation methods for DSGE models," in *Handbook of macroeconomics*, Elsevier, vol. 2, 527–724.

Fernández-Villaverde, J., S. Hurtado, and G. Nuño (2019): "Financial Frictions and the Wealth Distribution," Working Paper 26302, National Bureau of Economic Research.

Gentzkow, M., B. Kelly, and M. Taddy (2019): "Text as Data," *Journal of Economic Literature*, 57, 535–74.

Judd, K. L. (2002): "The parametric path method: an alternative to Fair–Taylor and L–B–J for solving perfect foresight models," *Journal of Economic Dynamics and Control*, 26, 1557–1583.

Kamihigashi, T. (2005): "Necessity of the transversality condition for stochastic models with bounded or CRRA utility," *Journal of Economic Dynamics and Control*, 29, 1313–1329.

Krusell, P. and A. A. Smith, Jr (1998): "Income and wealth heterogeneity in the macroeconomy," *Journal of political Economy*, 106, 867–896.

Le Van, C. and L. Morhaim (2002): "Optimal growth models with bounded or unbounded returns: a unifying approach," *Journal of economic theory*, 105, 158–187.

Liang, T. and A. Rakhlin (2020): "Just interpolate: Kernel "Ridgeless" regression can generalize," *The Annals of Statistics*, 48.

Ma, C. and L. Ying (2021): "The Sobolev regularization effect of stochastic gradient descent," *arXiv preprint arXiv:2105.13462*.

Maennel, H., O. Bousquet, and S. Gelly (2018): "Gradient descent quantizes relu network features," *arXiv preprint arXiv:1803.08367*.

Maliar, L., S. Maliar, J. B. Taylor, and I. Tsener (2020): "A tractable framework for analyzing a class of nonstationary Markov models," *Quantitative Economics*, 11, 1289–1323.

Maliar, L., S. Maliar, and P. Winant (2021): "Deep learning for solving dynamic economic models." *Journal of Monetary Economics*, 122, 76–101.

Marimón, R. (1989): "Stochastic turnpike property and stationary equilibrium," *Journal of Economic Theory*, 47, 282–306.

McKenzie, L. W. (1976): "Turnpike theory," *Econometrica*, 841–865.

Neyshabur, B., R. Tomioka, R. Salakhutdinov, and N. Srebro (2017): "Geometry of optimization and implicit regularization in deep learning," *arXiv preprint arXiv:1705.03071*.

Neyshabur, B., R. Tomioka, and N. Srebro (2014): "In search of the real inductive bias: On the role of implicit regularization in deep learning," *arXiv preprint arXiv:1412.6614*.

Shen, Z., R. Zhang, M. Dell, B. C. G. Lee, J. Carlson, and W. Li (2021): "LayoutParser: A unified toolkit for deep learning based document image analysis," in *International Conference on Document Analysis and Recognition*, Springer, 131–146.

Skiba, A. K. (1978): "Optimal growth with a convex-concave production function," *Econometrica*, 527–539.

Smith, S. L., B. Dherin, D. G. Barrett, and S. De (2021): "On the origin of implicit regularization in stochastic gradient descent," *arXiv preprint arXiv:2101.12176*.

Stokey, N. L., R. Lucas Jr, and E. C. Prescott (1989): *Recursive methods in economic dynamics*, Harvard University Press.

Zhang, C., S. Bengio, M. Hardt, B. Recht, and O. Vinyals (2021): "Understanding deep learning (still) requires rethinking generalization," *Communications of the ACM*, 64, 107–115.

Appendix A Sequential linear asset pricing

A.1 Learning the growth rate

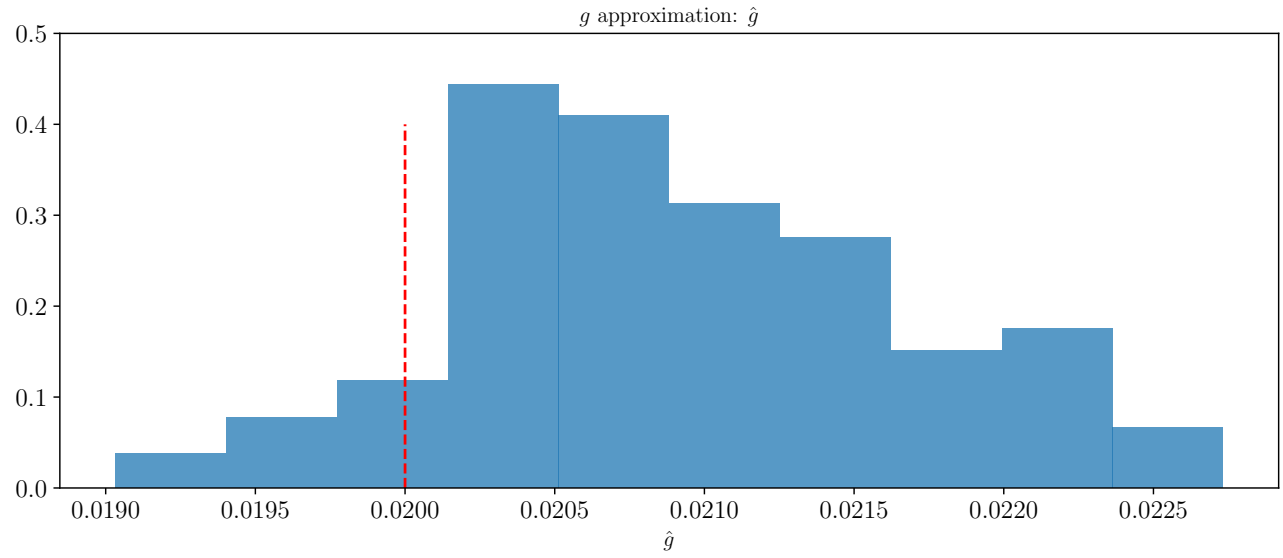

Figure 22: The distribution of the approximated growth rate of the dividends in the sequential linear asset pricing model. The dashed vertical line shows the true value of g.

Figure 22 shows the distribution for the approximated growth rate for the sequential linear asset pricing model. The approximated growth rate is defined as

$$\hat{g} \equiv e^{\phi} - 1.$$

This figure shows the approximate growth rate for 100 seeds, the red vertical line shows the true growth rate, i.e., $g = 0.02$. The approximation is slightly biased to the right. However, given the optimization problem is extremely non-convex, the accuracy of approximation is impressive. This results shows that the algorithm can separate the growth path from the stationary solution.

A.2 Misspecification of growth in prices

In this experiment we use a linear function to approximate the solution of the sequential linear asset pricing model with growing dividends:

$$\hat{p}(t;\theta) = tNN(t;\theta_1) + \phi,$$

where $\theta \equiv \{\phi, \theta_1\}$, $NN(\cdot;\theta_1)$ is a deep neural network, and ϕ is a parameter that needs to be found in the optimization process.

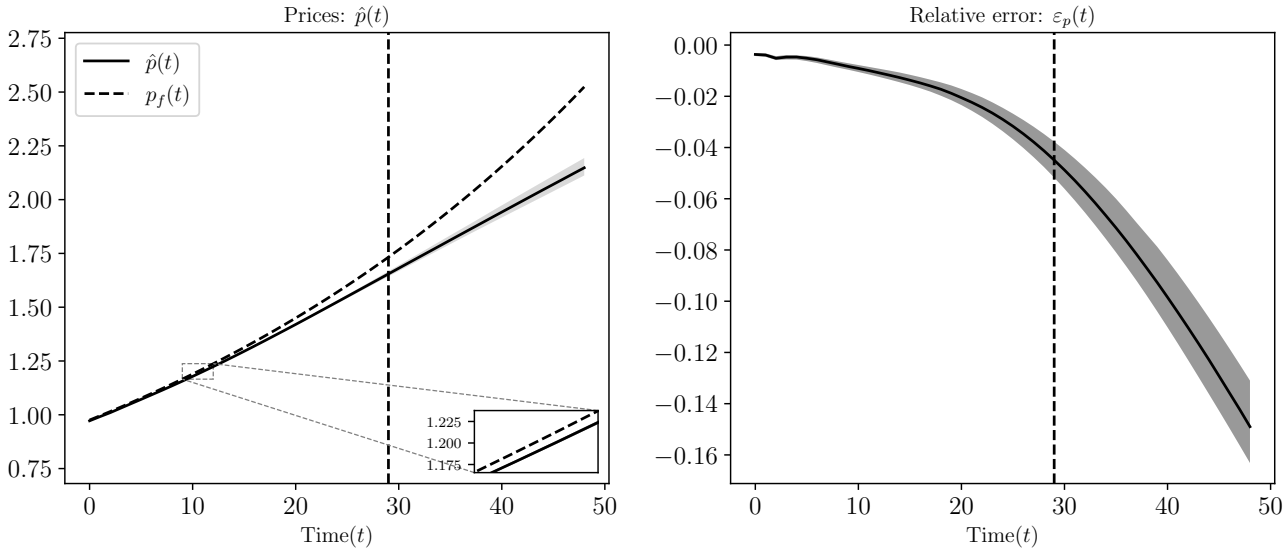

Figure 23: Comparison between the price based on the fundamentals and the price approximated by a deep neural network for the sequential linear asset pricing model with growing dividends in the presence of functional misspecification of the growth. The solid curve in the left panel shows the median of the approximate price paths, the dashed curve shows the price based on the fundamentals. The solid curve in the right panel shows the median of the relative errors. The shaded regions show the 10th and 90th percentiles. The dashed vertical lines separate the interpolation from the extrapolation region.

Figure 23 shows the results for sequential linear asset pricing model with growing dividends and a misspecification of the functional form of the growth. We use the same neural network we utilized in the correctly specified form. The only difference is the linear term. In the left panel, the solid curve, denoted by $\hat{p}(t)$, shows the median of the approximate price paths, the dashed curve, denoted by $p_f(t)$ shows the solution based on the fundamentals. The shaded region shows the 10th and 90th percentiles of the approximate price paths. The right panel shows the median of the relative errors. The shaded region shows the 10th and 90th percentiles of the relative errors. The dashed vertical lines separate the interpolation from the extrapolation region.

These results show that even in the presence of misspecification the long-run errors do not impair the accuracy of short- and medium-run dynamics (at most 2% relative error after 10 periods). However, the misspecification can reduce the generalization power of the solution.

Appendix B Sequential neoclassical growth

B.1 Zero as a fixed point for capital

As discussed before $k = 0$ is a repulsive fixed point for the neoclassical growth model. In this experiment we investigate whether the approximate solutions utilizing deep neural networks pick

a capital path that converges to the repulsive fixed point due to numerical errors.

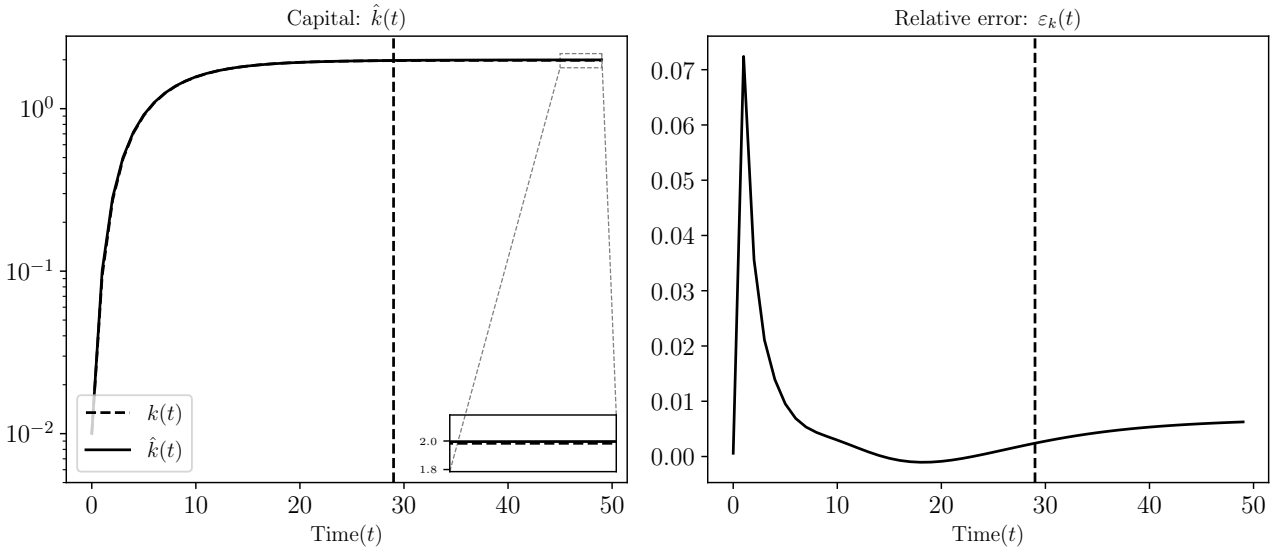

Figure 24: Comparison between the value function iteration and approximate solution using a deep neural network for the sequential neoclassical growth model for a small level of initial capital $k_0 = 0.01$. The solid curve, in the left panel, shows the approximate capital path and the dashed curve shows the solution obtained by the value function iteration method. The right panel shows the relative errors for the capital paths. The dashed vertical lines separate the interpolation from the extrapolation region.

Figure 24 shows the results for the sequential neoclassical growth model for a small level of initial capital. In this experiment we use the same parameters and deep neural network as the sequential neoclassical growth model with no total factor productivity growth, except the initial condition for capital. We use $k_0 = 10^{-2}$. In the left panel the solid curve shows the approximate capital path and the dashed curve shows the solution obtained by the value function iteration method. The right panel shows the relative errors between the approximate capital path and the solution obtained by the value function iteration method. The dashed vertical lines separate the interpolation from the extrapolation region.

This result shows that even for a small levels of initial capital the solutions do not converge to $k = 0$. Therefore, the solutions can detect that k_0 is a repulsive fixed point. Moreover, the short- and medium-run solutions are accurate and are not impaired by the long-run errors (at most 0.8%).

B.2 Solutions violating the transversality condition: initial capital above the steady-state

The blue curves in Figure 25 show a set of paths for capital, consumption and marginal utility of consumption (shadow prices), denoted by $\tilde{k}(t), \tilde{c}(t)$, and $u'(\tilde{c}(t))$, that satisfy the Euler equation

and feasibility condition, and violate the transversality condition.[29] The black curves, denoted by $k(t)$, $c(t)$, and $u'(c(t))$, show the optimal paths for capital, consumption and marginal utility of consumption. These paths satisfy the Euler equation, feasibility condition and the transversality condition. The steady-states for capital and consumption in the optimal solution are denoted by k^* and c^*.

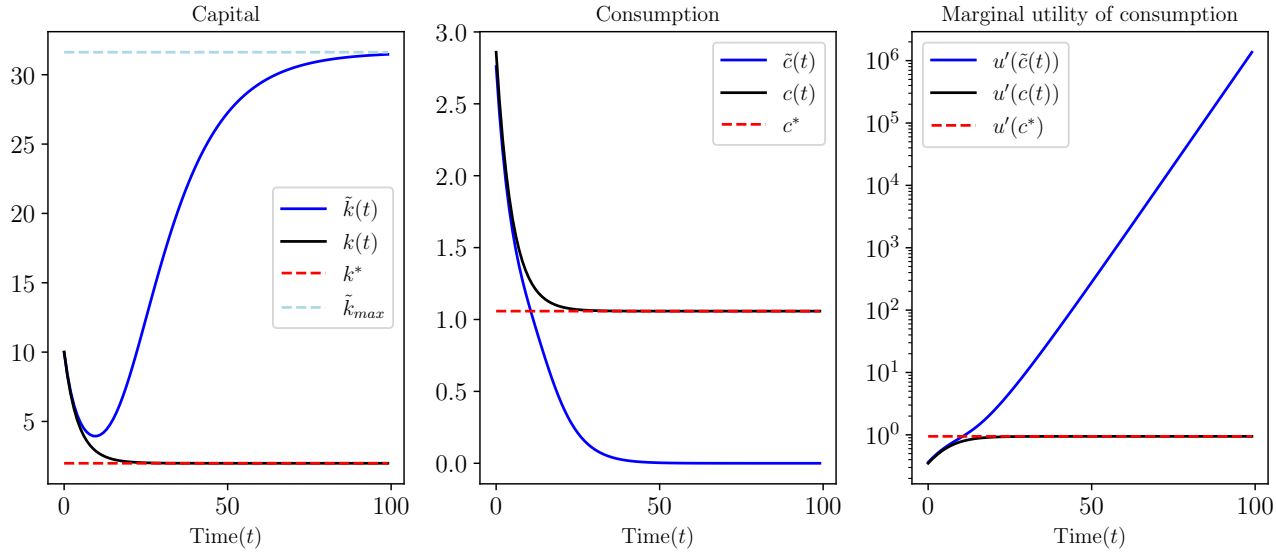

Figure 25: Comparison between the optimal solution and the solutions that violate the transversality condition for $k_0 > k^*$. The blue curves denoted by $\tilde{k}(t)$, $\tilde{c}(t)$, and $u'(\tilde{c}(t))$ show a set of capital, consumption, and marginal utility of consumption paths that violate the transversality condition. The black curves denoted by $k(t)$, $c(t)$, and $u'(c(t))$ show the capital, consumption, and marginal utility of consumption paths for the optimal solution. The steady-states for capital an consumption are denoted by k^* and c^*.

As evident in Figure 25 the capital path that violate the transversality condition has higher derivatives over its domain. More formally, let $\tilde{k}(t)$ be a capital path violating the transversality condition and $k(t)$ be the optimal solution, then in a compact space of the form $[0, T]$:

$$\int_0^T \left|\frac{d\tilde{k}}{dt}\right|^2 dt > \int_, ^T \left|\frac{dk}{dt}\right|^2 dt.$$

Therefore, by Assumption 1:

$$\|k\|_S < \|\tilde{k}\|_S.$$

[29] The parameters we used are $k_0 = 10$, $\beta = 0.9$, $\alpha = 0.33$, and $\delta = 0.1$.

B.3 An alternative way of approximating the optimal solution

Another approach to solve the sequential neoclassical growth model is to simultaneously approximate consumption and capital functions. In this case we pick a space of over-parameterized functions $\mathcal{H}(\Theta)$, and a grid $\mathcal{X}_{\text{train}} = \{t_1, \ldots, t_N\}$ and find the the capital and consumption function $[k(\cdot;\theta), c(\cdot;\theta)] \in \mathcal{H}(\Theta)$ by solving the following optimization problem:

$$\min_{\theta\in\Theta} \left[\frac{1}{|\mathcal{X}_{\text{train}}|} \sum_{t\in\mathcal{X}_{\text{train}}} \left(\beta\big[z(t+1)^{1-\alpha} f'(k(t+1;\theta)) + 1 - \delta\big] - \frac{u'(c(t;\theta))}{u'(c(t+1;\theta))} \right)^2 + \right.$$
$$\left. \frac{1}{|\mathcal{X}_{\text{train}}|} \sum_{t\in\mathcal{X}_{\text{train}}} \big(z(t)^{1-\alpha} f(k(t;\theta)) + (1-\delta)k(t;\theta) - c(t;\theta) - k(t+1;\theta)\big)^2 + (k(0;\theta) - k_0)^2 \right],$$

where $z(t)$ is evaluated by the law of motion for z:

$$z(t) = (1+g)^t z_0 \quad \text{for } t \in \mathcal{X}_{\text{train}}.$$

Non-negativity of capital and consumption can be built into $\mathcal{H}(\Theta)$. Here we omit the results because they are identical to the results illustrated in Section 3.2.

B.4 Being far from the steady-state

Figure 26 shows the results for the sequential neoclassical growth model with $\mathcal{X}_{\text{train}} = \{0, 1, \ldots, 4\}$. The dashed vertical lines separate the interpolation from the extrapolation region. The solid curve in the top-left panel shows the median of the approximate capital paths, the dashed curve shows the capital path obtained by the value function iteration method, and the shaded regions shows the 10th and 90th percentiles of the approximate capital paths. The solid curve in the top-right panel shows the median of the relative errors between the approximate capital paths and the capital path obtained by the value function iteration method, the shaded region shows the 10th and 90th percentiles of the relative errors. The solid curve in the bottom-left panel shows the median of the approximate consumption paths, the dashed curve shows the consumption path obtained by the value function iteration method, and the shaded region shows the 10th and 90th percentiles of the approximate consumption paths. The solid curve in the bottom-right panel shows the median of the relative errors between the approximate consumption paths and the consumption path obtained by the value function iteration method, the shaded region shows the 10th and 90th percentiles of the relative errors.

Theses results show that the approximate solutions are robust to using short time horizons in $\mathcal{X}_{\text{train}}$. The long-run errors do not impair the accuracy of short-run dynamics (less than 1% relative errors in capital in the first three periods).

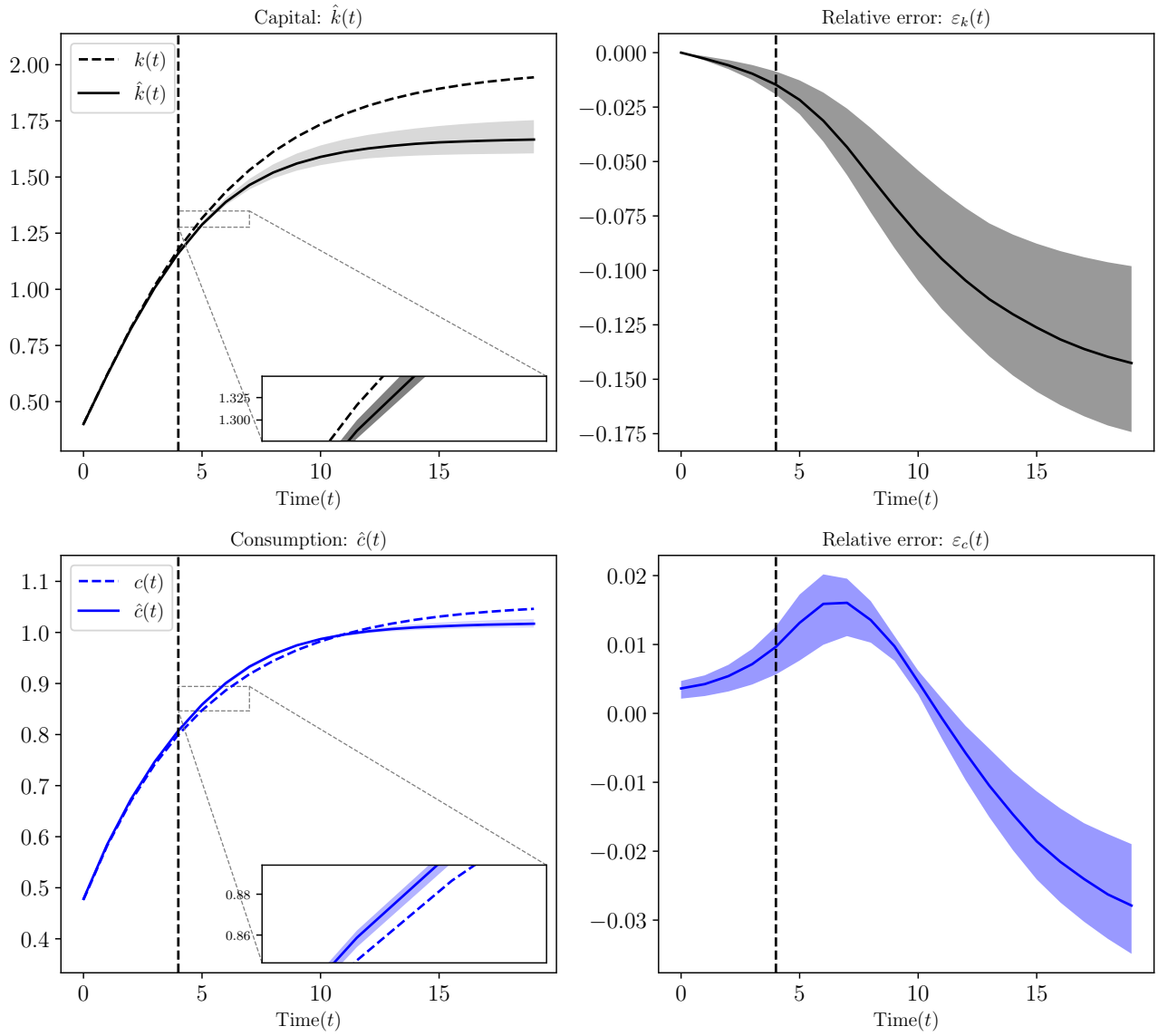

Figure 26: Comparison between the value function iteration and approximate solution using a deep neural network for the sequential neoclassical growth model with a short time horizon (i.e., $\mathcal{X}_{\text{train}} = \{0, 1, \ldots, 4\}$). The solid curves in the left panels show the median of the approximate capital and consumption paths. The dashed curves show the capital and consumption paths obtained by the value function iteration method. The solid curves in the right panels show the median of the relative errors between the approximate solutions and solutions obtained by the value function iteration. The shaded regions show the 10th and 90th percentiles. The dashed vertical lines separate the interpolation from the extrapolation region.

B.5 Misspecification of growth in capital and consumption paths

In this experiment we use a different functional form to approximate the solution of the sequential neoclassical growth model with growing total factor productivity:

$$\hat{k}(t;\theta) = tNN(t;\theta_1) + \phi,$$

where $\theta \equiv \{\phi, \theta_1\}$, $NN(\cdot;\theta_1)$ is a deep neural network, and ϕ is a parameter that needs to be found in the optimization process.

Figure 27 shows the results for the sequential neoclassical growth model with non-stationary total factor productivity in the presence of functional misspecification of the growth. We use the same deep neural network as before for $NN(\cdot;\theta_1)$. The only difference is the linear term.

The solid curve in the top-left panel shows the median of the approximate capital paths and the dashed curve shows the capital paths obtained by the value function iteration method. The shaded region shows the 10th and 90th percentiles of the approximate capital paths. The solid curve in the top-right panel shows the median of the relative errors between the approximate capital paths and the capital path obtained by the value function iteration method. The shaded region shows the 10th and 90th percentiles of the relative errors. The solid curve in the bottom-left panel shows the median of the approximate consumption paths, the dashed curve shows the consumption path obtained by the value function iteration method, and the shaded region shows the 10th and 90th percentiles of the approximate consumption paths. The solid curve in the bottom-right panel shows the median of the relative errors between the approximate consumption paths and the consumption path obtained by the value function iteration method. The shaded region shows the 10th and 90th percentiles of the relative errors.

These results show that the long-run errors do not impair the accuracy of the short- and medium-run dynamics even in the presence of functional misspecification of growth in total factor productivity. Moreover, the approximate solutions generalize well in the extrapolation region (less than 4% relative errors after 50 periods).

B.6 Analysis of the approximate solution in the vicinity of the critical point

Figure 28 shows the capital and consumption paths for a grid of initial conditions $k_0 \in [0.5, 4]$ for the sequential neoclassical growth model with the convex-concave production function. The top panel shows the capital paths and the bottom panel shows the consumption paths. The dashed horizontal lines show the steady-states of capital k_1^* and k_2^* and the corresponding steady-states for consumption c_1^* and c_2^*. The red trajectories show the approximate solutions near the critical point. The non-monotonicity of the red consumption paths (or equivalently the inflection

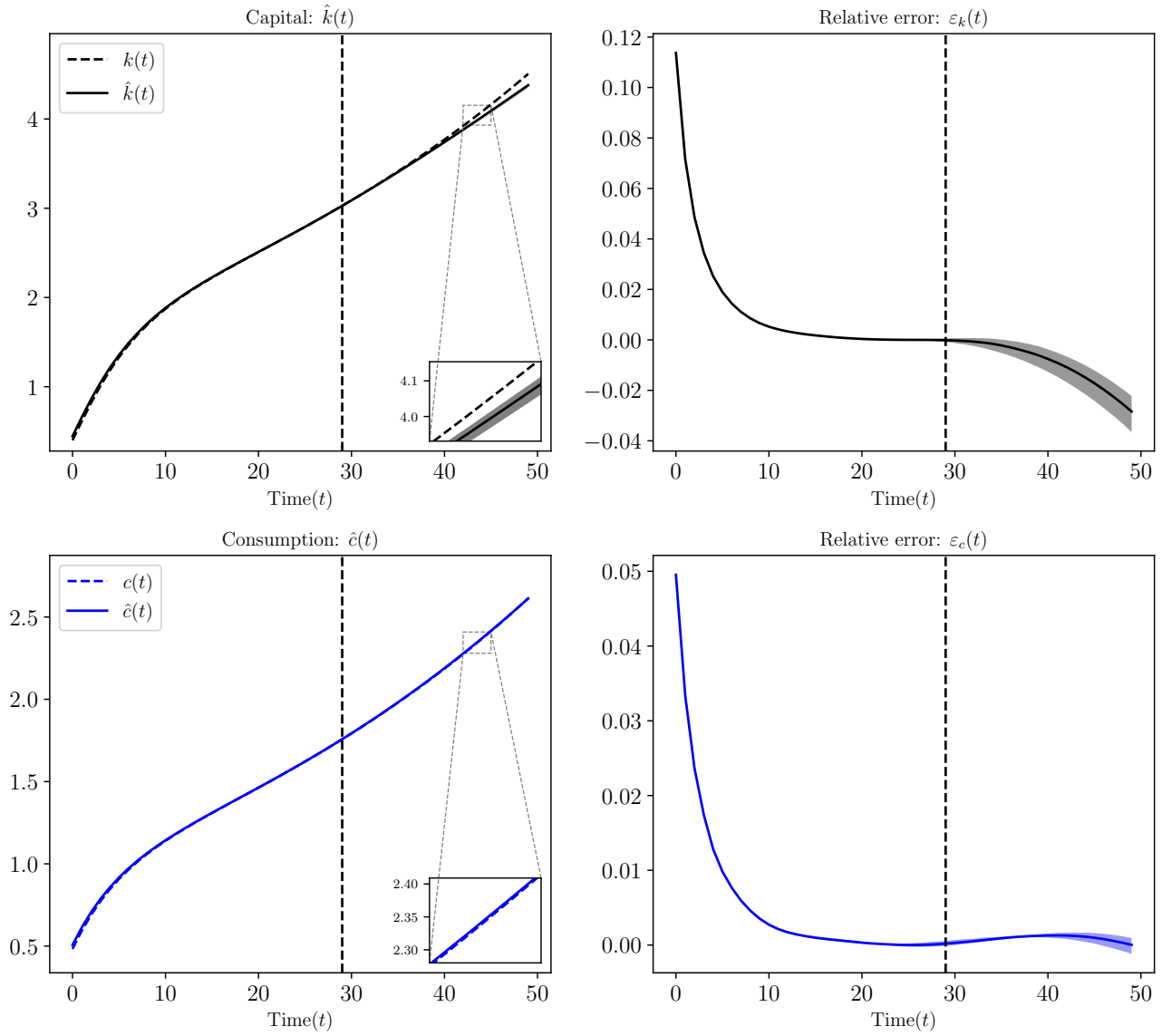

Figure 27: Comparison between the value function iteration and approximate solution using a deep neural network for the sequential neoclassical growth model with growth in total factor productivity in the presence of functional misspecification of the growth. The solid curves in the left panels show the median of the approximate capital and consumption paths. The dashed curves show the capital and consumption paths obtained by the value function iteration method. The solid curves in the right panels show the median of the relative errors between the approximate solutions and solutions obtained by the value function iteration method. The shaded regions show the 10th and 90th percentiles. The dashed vertical lines separate the interpolation from the extrapolation region.

points in red capital paths) indicates that these approximate solutions are not correct. However, the algorithm detects there is a critical point in the space of initial levels of capital around $k \approx 2.5$. Due to the discontinuity in the derivative of the production function it is cumbersome to confirm whether algorithm finds the right critical point and it is beyond the scope of this paper.

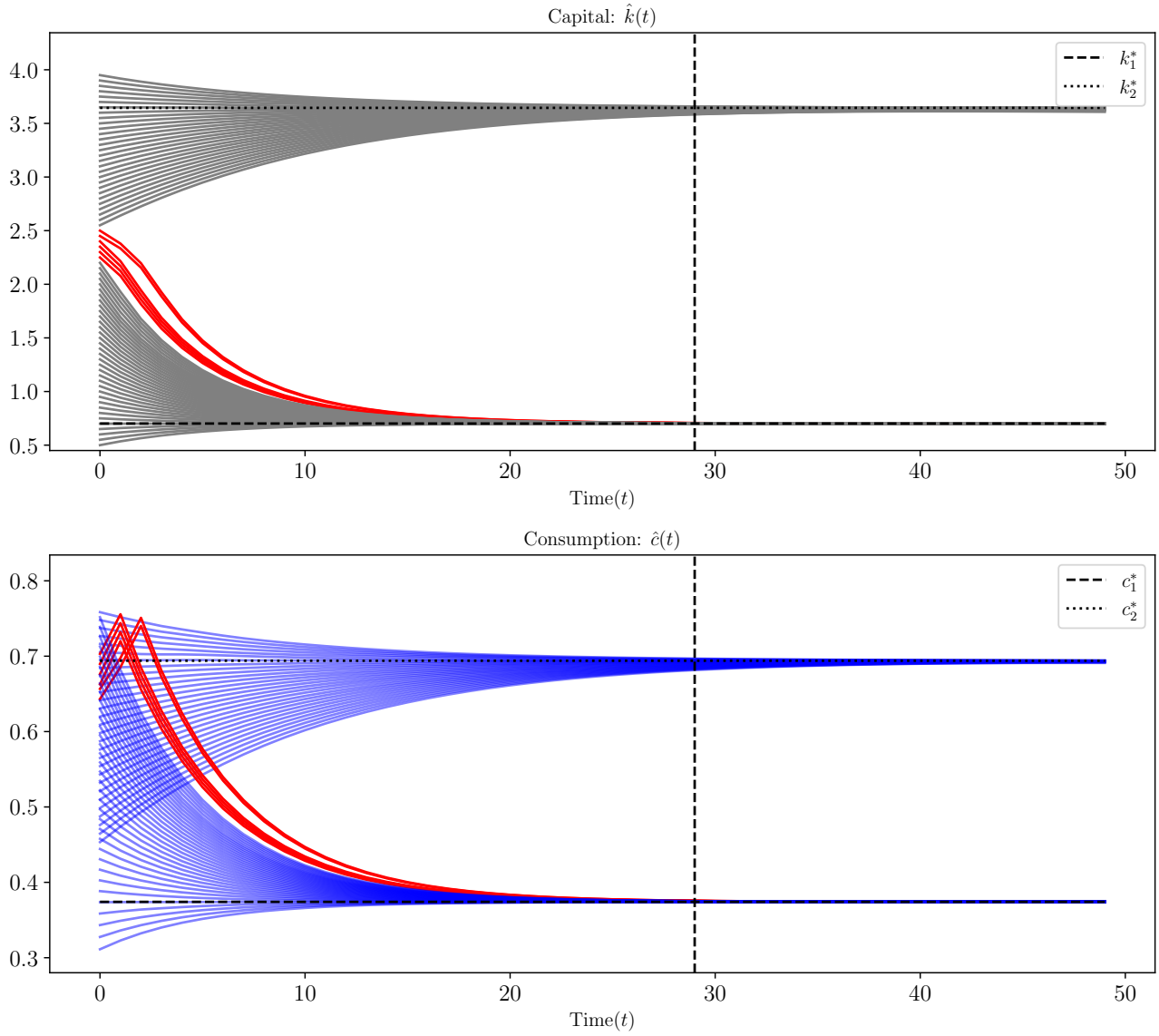

Figure 28: The approximate capital and consumption paths for the sequential neoclassical growth model with convex-concave production function. The grid for the initial condition of capital is from $[0.5, 4]$. The top panel shows the capital paths and the bottom panel shows the consumption paths. The red trajectories show the capital and consumption in the vicinity of the bifurcation point. The dashed horizontal lines show the steady-states of capital k_1^* and k_2^* and their corresponding steady-states for consumption c_1^* and c_2^*. The dashed vertical line separates the interpolation from the extrapolation region.

Appendix C Recursive neoclassical growth

C.1 Robustness of the solution to the initial condition of capital

As stated in the recursive formulation of the neoclassical growth problem, the optimality of the solution requires that all the capital consumption paths generated by the policy function to

satisfy the transversality condition.

Figure 29 shows the approximate capital and consumption paths for a range of initial conditions of capital in this case of $g = 0$ and $z_0 = 1$. The solid curves in the left panel, denoted by $\hat{k}(t)$, show the approximate capital paths from three initial conditions for capital outside of the interpolation region, $k_0 \in \{0.5, 3.25, 4.0\}$. The solid curves in the right panel show the corresponding consumption paths. The dashed curves, denoted by $k(t)$ and $c(t)$, show the capital and consumption paths obtained by the value function iteration method. The gray rectangle shows the interpolation region.

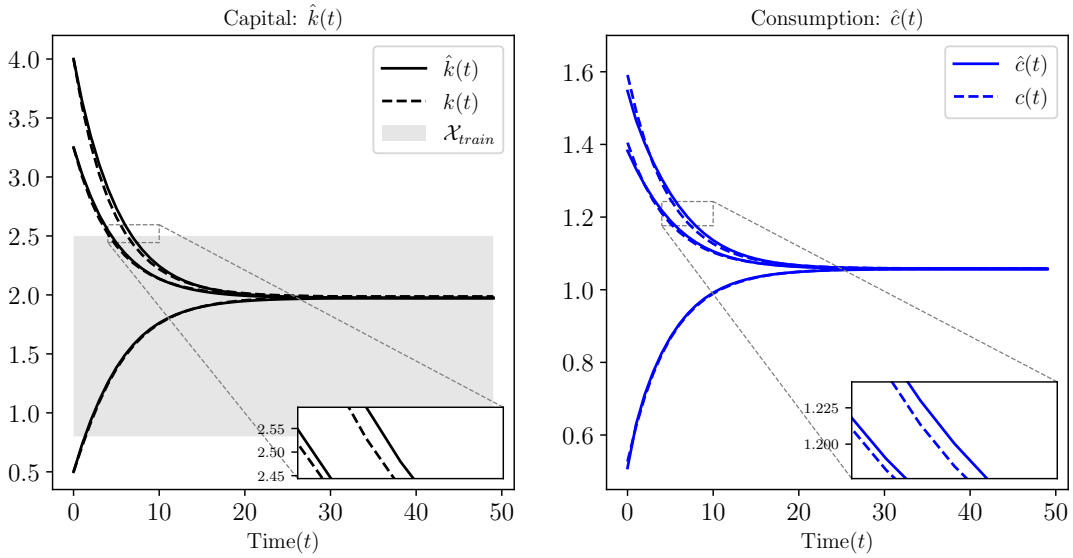

Figure 29: Comparison between the value function iteration and approximate solution using a deep neural network for the recursive neoclassical growth model for three different initial levels of capital. The left panel shows the approximate capital paths from three different initial conditions for capital outside of $\mathcal{X}_{\text{train}}$, i.e., $k_0 \in \{0.5, 3.25, 4.0\}$. The right panel shows the corresponding consumption paths. The dashed curves, denoted by $k(t)$ and $c(t)$, show the capital and consumption paths obtained by the value function iteration method.

These results show that the policy function for capital satisfies the transversality condition in the interpolation and the extrapolation regions.

Figure 30 shows the result of the recursive neoclassical growth problem for a range of initial conditions of capital in the presence of non-stationary total factor productivity (i.e., $g = 0.02$).

The solid curves in the left panel, denoted by $\hat{k}(t)$, show the approximate capital paths from three initial conditions for capital $k_0 \in \{0.5, 3.25, 4.0\}$. The solid curves in the right panel show the corresponding consumption paths. The dashed curves, denoted by $k(t)$ and $c(t)$, show the capital and consumption paths obtained by the value function iteration method.

These results show that even in the presence of non-stationary, the approximate policy function for capital satisfies the transversality condition inside and outside of the interpolation region.

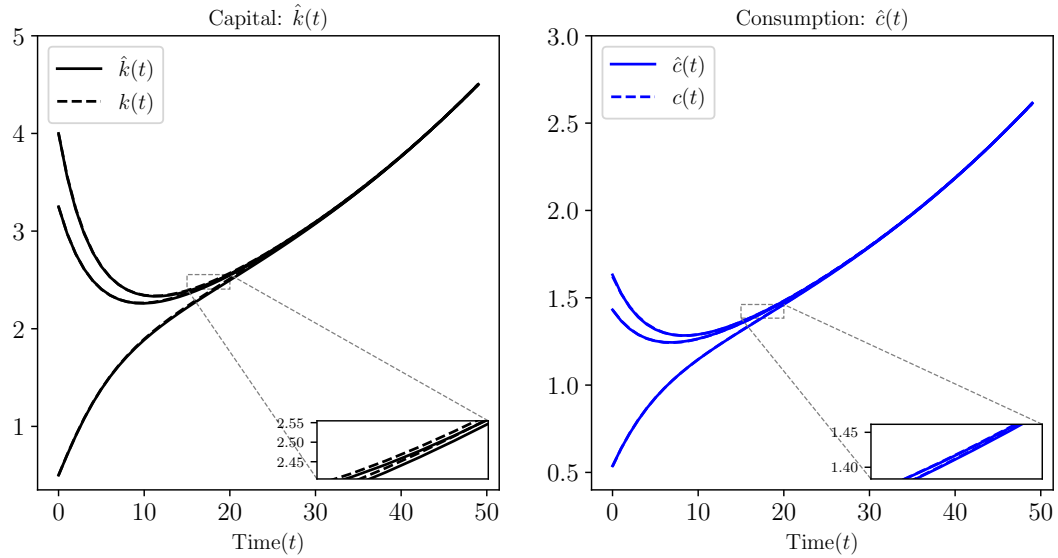

Figure 30: Comparison between the value function iteration and approximate solution using a deep neural network for the recursive neoclassical growth model with growth in total factor productivity (i.e., $g = 0.02$) for three different initial levels of capital. The left panel shows the approximate capital paths from three different initial conditions for capital, i.e., $k_0 \in \{0.5, 3.25, 4.0\}$. The right panel shows the corresponding consumption paths. The dashed curves, denoted by $k(t)$ and $c(t)$, show the capital and consumption paths obtained by the value function iteration method.