Applications of Deep Learning-Based Probabilistic Approach to "Combinatorial" Problems in Economics*

Ji Huang† Jinghai Yu‡
The Chinese University of Hong Kong

February 29, 2024

Abstract

Many "combinatorial" problems in economics arise from the static or discrete timing assumption that condenses a series of simple binary choices scattered randomly over time into a single instance. Leaning on this insight, we transform combinatorial choices into a sequence of binary choices in continuous time. The complexity of combinatorial choices turns into the dimensionality problem of dynamic optimization, which is overcome by applying a deep learning-based probabilistic approach. Two examples are provided for demonstration: 1) an exporting firm sporadically selects destinations among 100 potential interdependent markets; 2) a dynamic input-output network formation model involving 37 sectors.

JEL Classification: C63, G21, E44

Keywords: combinatorial choice, network formation, the curse of dimensionality, backward stochastic differential equation, deep learning

*Our CUHK colleague Vinci Chow provided tremendous technical support on computing hardware and software. Li Yanzhuo offered superb research assistance.

†Contact Details: 9/F Esther Lee Building, The Chinese University of Hong Kong, Shatin, Hong Kong, China. Email:jihuang@cuhk.edu.hk.

‡Email:jinghai.yu@link.cuhk.edu.hk

Introduction

"Combinatorial" problems are widely encountered in economics, particularly within the literature on international trade, industrial organization, and network formation. For example, an exporting firm needs to evaluate all possible combinations of destination markets to maximize its profit; a smartphone company needs to select the optimal combinations of its differentiated products; and there exists an inordinately large number of potential input-output networks between multiple sectors. The numerical resolution of these "combinatorial" problems is extremely challenging to derive, if not outright unattainable, using conventional methods.

In this paper, we aim to emphasize two points: one conceptual and the other methodological. Conceptually, we argue that numerous "combinatorial" problems in economic models are artificial or by-products of the timing assumption. If we construct a static or discrete-time model, various decisions dispersed over time in reality would be compressed to the onset of a period. By diffusing a "combinatorial" decision over time, more specifically over a continuous timeline, an agent faces a series of simple decisions (e.g., binary choices). Nonetheless, solving the dynamic optimization problem remains non-trivial due to the curse of dimensionality. This leads us to our second point, which suggests the application of a deep learning-based probabilistic approach to resolve dynamic optimization problems with high-dimensional state variables. We will expound further on our first point and subsequently illustrate the applications of our numerical method.

Let's consider a real-life scenario where an exporting firm decides which foreign markets to enter. It's hard to believe that the firm would make up its mind about the best combination of destination markets at the beginning of a year or its life cycle. The more plausible scenario is, as the company and its products grow, it considers branching out overseas and begins exploring a list of foreign markets. Over time, certain opportunities arise, and the firm actually enters some of these markets. Likewise, when college students start their freshman year, they do not decide who their friends will be for the rest of their four years. More likely, they meet others in various circumstances during college and form friendships over time. It's intuitive to see that many discrete choices, which are compressed into one instance in economic models, are actually spread out over time in real life.

Assuming that discrete choices are spread out over (continuous) time is not only closer to reality but also computationally more straightforward. Within a short time interval, an economic agent will have, at most, one binary choice to make, which is trivial to compute. This idea is inspired by Doraszelski and Judd (2012), who explores stochastic games with discrete states. Their primary point is that in the continuous-time setting, it is sufficient to consider only one player's state change within a short time interval. Computationally challenging cases, where states of multiple players jointly change, occur with negligible probabilities when the time interval is short enough. Similarly, in our case, it becomes extremely unlikely that an agent makes multiple discrete decisions simultaneously in the continuous-time setting.

When discrete choices are spread over time, the economic mechanism driving their interdependence remains intact. For instance, when a Chinese company begins selling electric vehicles to Chile, it understands that shipping costs will be more economical should it later enter other

South American markets. The interdependence (e.g., complementarity) between discrete choices spread out over time is captured by the value function of dynamic programming. If the present discrete choice contributes more significantly to potential future choices, it will increase the value function by a larger margin.

Solving dynamic optimization with a large state space may not necessarily be much easier than resolving combinatorial problems. The second contribution of this paper is to apply a deep learning-based probabilistic approach to the high-dimensional dynamic discrete choice problems outlined above (Huang, 2023b,a). In fact, the numerical approach we propose is applicable to both continuous states driven by diffusion shocks and discrete states driven by jump risks.

As an illustration, we consider a simple dynamic discrete choice problem. Suppose an agent's utility flow depends on uncontrolled continuous state x_t and two binary states y_t^1 and y_t^2, i.e., $u(x_t, y_t^1, y_t^2)$. And x_t follows

$$x_{t+\Delta} = x_t + \mu\left(x_t\right)\Delta + \sigma\left(x_t\right)\left(W_{t+\Delta} - W_t\right), \tag{1}$$

where Δ represents the length of time period, and $W_{t+\Delta} - W_t$ follows a normal distribution with a mean of zero and variance of Δ. If $y_t^1 = 1$, it changes to $y_t^1 = 0$ exogenously with a probability of $\lambda\Delta$ within $[t, t+\Delta]$; if $y_t^1 = 0$, the agent is granted an option to switch to $y_t^1 = 1$ with a probability of $\lambda\Delta$ within $[t, t+\Delta]$. The same type of jump risks apply to y_t^2. But y_t^1 and y_t^2 are two independent processes. Given the setting, the agent's value function satisfies

$$V\left(x_t, y_t^1, y_t^2\right) = u\left(x_t, y_t^1, y_t^2\right)\Delta + \max\left\{E\left[V\left(x_{t+\Delta}, y_{t+\Delta}^1, y_{t+\Delta}^2\right)\middle|\, x_t, y_t^1, y_t^2\right]\right\} \tag{2}$$

Since the jump risks driving y_t^1 and y_t^2 are independent over $[t, t+\Delta]$, we can disregard the joint movement of the two as it will occur with a probability of order Δ^2, becoming negligible when Δ is sufficiently small. Hence, we approximate the conditional expectation in equation (2) with

$$\begin{aligned} & 2e^{-\lambda\Delta} E\left[V\left(x_{t+\Delta}, y_{t+\Delta}^1, y_{t+\Delta}^2\right)\middle|\, x_t, y_{t+\Delta}^1 = y_t^1, y_{t+\Delta}^2 = y_t^2\right] \\ & \quad + (1 - e^{-\lambda\Delta})(1 - y_t^1)\max\left\{V\left(x_t, y_t^1 + 1, y_t^2\right), V\left(x_t, y_t^1, y_t^2\right)\right\} \\ & \quad + (1 - e^{-\lambda\Delta})(1 - y_t^2)\max\left\{V\left(x_t, y_t^1, y_t^2 + 1\right), V\left(x_t, y_t^1, y_t^2\right)\right\} \\ & \quad + (1 - e^{-\lambda\Delta}) y_t^1 V\left(x_t, y_t^1 - 1, y_t^2\right) + (1 - e^{-\lambda\Delta}) y_t^2 V\left(x_t, y_t^1, y_t^2 - 1\right) \end{aligned} \tag{3}$$

The first line above is the conditional expectation along the paths where no jump risks are realized, the second and third lines capture the expected long-run impacts of the agent's simple binary choices, and the last line represents the expected impacts of exogenous state changes. It is straightforward to observe that the complex combinatorial problems do not appear in our continuous-time setting because the probability that these combinatorial choices emerge goes to zero when the time interval we consider is short enough. Nevertheless, the interdependence between binary choices is still preserved by the value function, the present value of future utility flows.

Next, we consider the conditional expectation along the paths where there are no realizations

of jump risks. The probabilistic formulation of the recursive equation (2) is

$$V(x_{t+\Delta}, y_t^1, y_t^2) = V\left(x_t, y_t^1, y_t^2\right) - u\left(x_t, y_t^1, y_t^2\right)\Delta + z(x_t, y_t^1, y_t^2)(W_{t+\Delta} - W_t), \tag{4}$$

where $z(\cdot)$, is an unknown function of the current state (x_t, y_t^1, y_t^2) that we solve for along with $V(\cdot)$. In mathematics, equation (4) is referred to as a Backward Stochastic Differential Equations (BSDE), which effectively transforms an equation (2) at state (x_t, y_t^1, y_t^2) into infinitely many equations because it holds for any realization of $W_{t+\Delta} - W_t$ and that of jump risks.

If we follow the conventional analytic approach, we use equation (2) in a specific state to guide our search for the fixed point. This approach requires multiple evaluations of the value function to calculate the conditional expectation. However, with the probabilistic formulation, each realization of $W_{t+\Delta} - W_t$, as well as the corresponding evaluation of the value function, would independently discipline the search process via equation (4). This efficiency of each evaluation represents the advantage of the probabilistic approach.

To take advantage of modern Machine Learning technique, we approximate the valuation function $V(\cdot)$ and its volatility term $z(\cdot)$ with a feed-forward neural network, denoted as $\tilde{V}(\cdot;\Theta)$ and $\tilde{z}(\cdot;\Theta)$, respectively. Equation (2) and (4) suggest that the parameters Θ should solve the following optimization problem

$$\begin{aligned}
\min_{\Theta}: \quad & \frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}\left(\tilde{V}(\hat{x}^{i,j}, y^{1,i}, y^{2,i};\Theta) - \hat{V}(x^i, y^{1,i}, y^{2,i};\Theta)\right)^2 \\
& + \frac{1}{N}\sum_{i=1}^{N}\left(\max\left\{E\left[\tilde{V}\left(\hat{x}^{i,j}, \hat{y}^{1,i}, \hat{y}^{2,i};\Theta\right)\middle| x^i, y^{1,i}, y^{2,i}\right]\right\} + u(x^i)\Delta - \tilde{V}(x^i;\Theta)\right)^2 \\
\text{s.t.} \quad & \hat{V}(x^i, y^{1,i}, y^{2,i};\Theta) \equiv u(x^i, y^{1,i}, y^{2,i})\Delta - \tilde{V}(x^i, y^{1,i}, y^{2,i};\Theta) - \tilde{z}(x^i, y^{1,i}, y^{2,i};\Theta)w^{i,j} \\
& \hat{x}^{i,j} = x^i + \mu(x^i)\Delta + \sigma(x^i)w^{i,j} \\
& w^{i,j} \text{ is sampled independently from } N(0,\Delta) \\
& (x^i, y^{1,i}, y^{2,i}) \text{ are from a given set,}
\end{aligned} \tag{5}$$

where the conditional expectation operation follows the approximation (3) and $\hat{y}^{1,i}$ and $\hat{y}^{2,i}$ are random variables following state $(x^i, y^{1,i}, y^{2,i})$ and optimal binary choices. Note that this formulation can make use of parallel computing, as the evaluation of each sample path $(x^i, y^{1,i}, y^{2,i})$ is independent of others. More importantly, the task of coding is straightforward as we only need to generate sample paths, simulate the dynamics, and calculate the loss specified by the objective function (5) for a given set of parameter Θ. We fully outsource the search for the optimal Θ to industrial-level machine learning packages, such as TensorFlow and PyTorch.

As a demonstration, we solve a single firm's export destination selection problem and an input-output network formation problem. In Section 2, we construct a continuous-time version of the discrete-time firm export model by Alfaro-Urena, Castro-Vincenzi, Fanelli and Morales (2023). In this model, an exporting firm decides among 100 possible foreign markets for entry, with each destination featuring a state variable driving its demand. Therefore, there are 200 state variables in total. We demonstrate that the value function of the exporting firm preserves the cross-destination complementarities in its profit function.

Section 3 constructs a continuous-time network formation model based on the static production network setting by Kopytov, Mishra, Nimark and Taschereau-Dumouchel (2021). In this model, a social planner obtains opportunities stochastically over time to form an input-output link. There are 37 sectors in the economy, and each sector has a time-varying TFP driven by aggregate shocks. The dimensionality of the state variable is approximately 37×37 for the social planner's dynamic optimization problem. Our numerical exercise indicates that the socially optimal input-output linkages might not align with the ideal linkages from an individual sector's perspective.

Literature. Our paper contributes to the international trade and industrial organization literature, which often involves combinatorial discrete choice problems, typically in a static setting. For example, Jia (2008) study supermarkets' store location decisions and develop a global solution to the combinatorial problem when objective functions have the property of positive complementarities. Fan and Yang (2020) investigate the composition of differentiated product offerings in the U.S. smartphone market. Following Jia (2008), Antras, Fort and Tintelnot (2017) and Arkolakis, Eckert and Shi (2023) exploit positive and/or negative complementarities to solve sourcing or production location problems in trade literature. Other examples of combinatorial problems in the IO and trade literature include studies by Hendel (1999), Tintelnot (2017), Houde, Newberry and Seim (2023), and Oberfield, Rossi-Hansberg, Sarte and Trachter (2024).

Our paper is also related to the rapidly expanding body of literature on production networks (see reviews like Carvalho (2014); Carvalho and Tahbaz-Salehi (2019)). Our approach significantly contributes to the specific topic of input-output network formation, including studies by Oberfield (2018), Acemoglu and Azar (2020), Taschereau-Dumouchel (2020), and Dhyne, Kikkawa, Kong, Mogstad and Tintelnot (2023). All these network formation models are static. Our approach enables researchers in the field to explore dynamic network formation under idiosyncratic and aggregate shocks, and to characterize an input-output network's transition paths. The continuous-time setting is, in fact, more tractable for studying production networks. For example, Liu and Tsyvinski (2024) apply a continuous-time model to investigate the transmission of temporary shocks through a fixed input-output network.

Our approach to transforming static combinatorial problems into a sequence of binary choices aligns with strategic network formation models (Jackson, 2010). Similar to Currarini, Jackson and Pin (2009, 2010), agents in our setting randomly obtain opportunities over time to decide whether to form links. In a longer time, we observe the evolution of the overall network. In the econometric studies of network data, authors also follow this sequential move setting of network formation (for example, Mele (2017) and Christakis, Fowler, Imbens and Kalyanaraman (2020)).

The mathematical foundation of our numerical approach lies in nonlinear Backward Stochastic Differential Equations (BSDEs), beginning with the seminal work of Pardoux and Peng (1990). Following the advancement of Machine Learning in the past decade, applied mathematicians discover the numerical superiority of BSDEs when combined with deep learning, for solving high-dimensional Partial Differential Equations (Han, Jentzen and E, 2018). Inspired by

these works, Huang (2023b,a) introduce the deep learning-based probabilistic approach to the economics literature, recognizing that all forward-looking stochastic processes, like asset prices and continuation values, can be cast by BSDEs.

The organization of the paper is as follows. Section 1 illustrates the probabilistic formulation and the deep learning-based numerical method. Section 2 and 3 consider two examples: a single firm's dynamic discrete choice and a dynamic network formation model. In Section 4, we make a few remarks on wider applications of our numerical approach.

1 Deep Learning-Based Probabilistic Approach

In section, we present the detailed probabilistic and analytic formulation of a single agent's dynamic discrete choice problem and illustrate the deep learning-based numerical method based on the two formulations.

1.1 Basic Model

We consider the dynamic optimization of an agent whose flow utility $u(x_t, y_t)$ depends on continuous state variable $x_t \in R^N$ driven by Brownian motion

$$\mathrm{d}x_t = \mu(x_t, y_t)\,\mathrm{d}t + \sum_{m=1}^{M} \sigma^m(x_t, y_t)\,\mathrm{d}W_t^m \tag{6}$$

and discrete state variable $y_t = \left[y_t^1, y_t^2, \cdots, y_t^{\mathbf{J}}\right]$. Without loss of generality, we assume that all y_t^j are $0-1$ binary variables. The stochastic process x_t is assumed to be uncontrolled for simplicity. We refer readers who are interested in the probabilistic formulation of controlled state variables driven by diffusion process to Huang (2023b,a).

The discrete state variable is partially controlled by the agent in the following sense. First, if $y_t^j = 0$ the agent will be granted an opportunity of state switching with a probability $\lambda^j \mathrm{d}t$ over $[t, t+\mathrm{d}t]$. To switch $y_t^j = 0$ to $y_t^j = 1$ when she has the opportunity, the agent must pay a fixed utility cost s_j. The arrivals of these opportunities across all $y_t^j = 0$ are independent. Second, if $y_t^j = 1$ the state could switch to $y_t^j = 0$ with a probability $\lambda^j \mathrm{d}t$ over $[t, t+\mathrm{d}t]$.

In the corresponding static or discrete-time settings, the agent faces a standard combinatorial problem of selecting a combination of $\left[y_t^1, y_t^2, \cdots, y_t^{\mathbf{J}}\right]$ among $2^{\mathbf{J}}$ possibilities by exerting some efforts. In our continuous-time setting, the chance that the agent can alter two or more of her states jointly is negligible as that probability is of order $(\mathrm{d}t)^2$ or higher. Hence, the agent does not have a challenging combinatorial problem to solve at any point of the time. Nevertheless, when she has a binary choice to make, the agent still fully takes into account the long-run impact of her current decision, which in turn is captured by the value function. Now the difficulty of solving the dynamic optimization problem is to handle high-dimensional state variables (x_t, y_t). With the deep learning-based probabilistic approach, dimensionality is no longer a bottleneck. To give a sense of how powerful the approach is, the second example that we will show has 37×37 dimensions.

The value function of the agent is

$$V\left(x_t, y_t\right) = \max_{y_t} \int_0^{+\infty} e^{-\rho s} \left(u\left(x_s, y_s\right) - s_j \mathbf{1}\left\{y_{s,j} - y_{s-,j} > 0\right\}\right) \mathrm{d}s,$$

where ρ is the discount rate and y_{s-} is the left limit of the process $\{y_t\}$. $y_{s,j} - y_{s-,j} > 0$ implies that the agent is granted a chance to switch state j at time s and she decides to do so by exerting cost s_j. Note that x_t is a continuous state variable driven by diffusion processes, and y_t is a discrete state variable driven by jump processes. Next, we will combine the advantages of both probabilistic and analytic approaches to fully exploit the features of diffusion processes and jump processes.

1.2 Probabilistic Formulation

The probabilistic formulation defines the value function along all realized paths of exogenous shocks. For the current problem, given the agent's optimal decision the stochastic process of her continuation value $V_t = V\left(x_t, y_t\right)$ follows a backward stochastic differential equation (BSDE)

$$\begin{aligned}
\mathrm{d}V_t = & -\left(u\left(x_t, y_t\right) - \rho V_t\right) \mathrm{d}t + \sum_{m=1}^{M} \sigma_t^{V,m} \mathrm{d}W_t^m \\
& + \sum_{j=1}^{\mathbf{J}} \left(1 - y_t^j\right) \max\left\{V\left(x_t, y_t + \mathbf{1}^j\right) - V\left(x_t, y_t\right) - s_j, 0\right\} \mathrm{d}\Lambda_t^{0,j} \\
& + \sum_{j=1}^{\mathbf{J}} y_t^j \left(V\left(x_t, y_t - \mathbf{1}^j\right) - V\left(x_t, y_t\right)\right) \mathrm{d}\Lambda_t^{1,j} \\
V_{s+t} = & V\left(x_{s+t}, y_{s+t}\right) \qquad \text{for any } t \text{ and any initial date } s, \qquad (7)
\end{aligned}$$

where $\sigma_t^{V,m}, m = 1, \cdots, M$, are endogenous volatility terms of V_t, and $\mathrm{d}\Lambda_t^{0,j}$ and $\mathrm{d}\Lambda_t^{1,j}, j = 1, \cdots, \mathbf{J}$, capture the realizations of jump risks that affect the state transitions. Condition (7) states that the forward-looking stochastic process V_t and the backward-looking processes x_t and y_t must always satisfy the mapping $V\left(\cdot, \cdot\right)$. Regarding the deterministic terms, the agent's continuation value declines by the utility flow being realized $u\left(x_t, y_t\right) \mathrm{d}t$, and increases by the discounting effect $\rho V_t \mathrm{d}t$ that no longer applies from the perspective at time $t + \mathrm{d}t$. Note that the condition holds trivially along paths driven by jump risks due to the construction of the BSDE. Within this subsection, we will not consider paths with realizations of jump risks, i.e., we will only consider BSDE

$$\mathrm{d}V_t = -\left(u\left(x_t, y_t\right) - \rho V_t\right) \mathrm{d}t + \sum_{m=1}^{M} \sigma_t^{V,m} \mathrm{d}W_t^m \qquad (8)$$

$$V_{s+t} = V\left(x_{s+t}, y_{s+t}\right) \qquad \text{for any path } y_{s+t} = y_s, \text{ any } t \text{ and any initial date } s, \qquad (9)$$

The core of the probabilistic formulation is that endogenous volatility terms of V_t must be such that Condition (7) or (9) always holds. The design of the probabilistic numerical scheme is based on this insight of BSDEs, which not only defines the fixed point $V\left(\cdot, \cdot\right)$ but also its

volatility terms. Given any initial date t and state (x_t, y_t), the fixed-point mapping $V(\cdot,\cdot)$ first yields $V_t = V(x_t, y_t)$. The volatility terms $\sigma_t^{V,m}, m = 1, \cdots, M$, are endogenous because they are such that for any realizations of $\left(W_{t+\Delta}^m - W_t^m, i = 1, \cdots, M\right)$ and sufficiently small positive constant Δ, the updated $V_{t+\Delta}$

$$V_{t+\Delta} = V_t - \left(u(x_t, y_t) - \rho V_t\right)\Delta + \sum_{m=1}^{M} \sigma_t^{V,m}\left(W_{t+\Delta}^m - W_t^m\right)$$

and updated $x_{t+\Delta}$

$$x_{t+\Delta} = x_t + \mu(x_t, y_t)\Delta + \sum_{m=1}^{M} \sigma^m(x_t, y_t)\left(W_{t+\Delta}^m - W_t^m\right)$$

satisfy the fixed-point mapping

$$V_{t+\Delta} = V(x_{t+\Delta}, y_t).$$

From the numerical perspective, the advantage of the probabilistic approach is that Condition (7) or (9) must hold along any simulated path. Therefore, we can take advantage of any single path and use Condition (7) or (9) to uncover the fixed-point mapping. However, for jump risks this advantage is no evident because we need to simulate a large number of paths to cover a significant proportion of paths with realized jump risks. Due to this concern, we revert back to the analytic formulation to efficiently captures the impacts of jump risks on the value function.

1.3 Analytic Formulation

The analytic formulation defines the value function as

$$\begin{aligned} V(x_t, y_t) &= u(x_t, y_t)\Delta + e^{-\rho\Delta} E\left[V(x_{t+\Delta}, y_{t+\Delta}) \middle| x_t, y_t\right] \\ &= u(x_t, y_t)\Delta - \rho\Delta V(x_t, y_t) + E\left[V(x_{t+\Delta}, y_{t+\Delta}) \middle| x_t, y_t\right]. \end{aligned} \tag{10}$$

The computation of the conditional expectation is composed of two components: diffusion and jump risks. For the diffusion risk,

$$E\left[V(x_{t+\Delta}, y_{t+\Delta}) \middle| x_t, y_{t+\Delta} = y_t\right] = \int_{w^1} \cdots \int_{w^M} V(x_{t+\Delta}, y_t) \prod_{m=1}^{M} f(w^m)\, \mathrm{d}w^M \cdots \mathrm{d}w^1, \text{ where}$$

$$x_{t+\Delta} = x_t + \mu(x_t, y_t)\Delta + \sum_{m=1}^{M} \sigma^m(x_t, y_t)\, w^m,$$

$$f(x) = \frac{1}{\sqrt{2\pi\Delta}} \exp\left(-\frac{x^2}{2\Delta}\right).$$

To evaluate the integration numerical, we will apply Gauss-Hermite quadrature. Note that while deriving the numerical integration, each node that we choose to evaluate $V(\cdot, y_t)$ is essentially a realization of Brownian shocks. Hence, we can make use of these evaluations twice to enhance the computation efficiency: one for the analytic approach and the other for the probabilistic

approach.

For the jump risk, we evalute $V\left(x_t, y_t + \mathbf{1}^j\right)$ if $y_t^j = 0$ and $V\left(x_t, y_t - \mathbf{1}^j\right)$ if $y_t^j = 1$. Then, equation (10) is approximated by

$$\begin{aligned}(1+\rho\Delta)\,V\left(x_t, y_t\right) = u\left(x_t, y_t\right)\Delta + \frac{1}{P_t} E\left[V\left(x_{t+\Delta}, y_{t+\Delta}\right) \middle| x_t, y_{t+\Delta} = y_t\right] \\ + \frac{1}{P_t}\sum_{j=1}^{\mathbf{J}}\left(1 - e^{-\lambda_j \Delta}\right) y_t^j V\left(x_t, y_t - \mathbf{1}^j\right) \\ + \frac{1}{P_t}\sum_{j=1}^{\mathbf{J}}\left(1 - e^{-\lambda_j \Delta}\right)\left(1 - y_t^j\right) \max\left\{V\left(x_t, y_t + \mathbf{1}^j\right) - s_j, V\left(x_t, y_t\right)\right\}, \text{ where}\end{aligned}$$

$$P_t \equiv 1 + \sum_{j=1}^{\mathbf{J}}\left(1 - e^{-\lambda_j \Delta}\right) y_t^j + \sum_{j=1}^{\mathbf{J}}\left(1 - e^{-\lambda_j \Delta}\right)\left(1 - y_t^j\right).$$

The feature of the analytic approach is to evaluate the value function $V\left(x_t, y_t\right)$ along multiple paths and incorporate their weighted sum into a single equation of conditional expectation. It has a comparative advantage over the probabilistic approach for jump risks, which requires a large number of simulated paths so as to capture the impacts of jump risks.

1.4 Deep Learning-Based Numerical Method

The first feature of our numerical method is to approximate the value function with deep neural network, which is denoted as $V(x, y; \Theta)$ and Θ stands for the set of parameters of the neural network. Audience who has no knowledge of neural network approximation can think of it as an alternative to Chebyshev polynomials. Chapter 6 of Goodfellow, Bengio and Courville (2016) is the standard reference for the basic neural network architecture, and we also recommend Chapter 5 of Zhang, Lipton, Li and Smola (2023).

Secondly, our scheme is simulation-based. Given a set of conjectured parameters Θ and the initial state (x_t, y_t) of an agent, we can find the initial continuation value $V_t = V(x_t, y_t; \Theta)$ and the volatility terms $\sigma_t^{V,m} = \sigma^{V,m}(x_t, y_t; \Theta), m = 1, \cdots, M$. Following the probabilistic approach, we simulate the backward-looking process x_t according to equation (6) and the forward-looking process V_t according to BSDE (8). To assess the accuracy of guessed parameter Θ, the terminal condition of BSDEs yields a loss

$$\text{Loss}_P = \|V_{t+\Delta} - V(x_{t+\Delta}, y_t; \Theta)\|^2.$$

To take into account the effects of jump risks, we resort to the analytic formulation and compute the loss

$$\text{Loss}_A = \|(1+\rho\Delta)V(x_t, y_t; \Theta) - u(x_t, y_t)\Delta - E\left[V\left(x_{t+\Delta}, y_{t+\Delta}; \Theta\right) \middle| x_t, y_t\right]\|^2.$$

The calculation of $E\left[V\left(x_{t+\Delta}, y_{t+\Delta}; \Theta\right) \middle| x_t, y_t\right]$ has been laid out in Section 1.3. For each sample path starting from (x_t, y_t), we compute two types of losses, and we can simulate as many paths as our computing hardware allows. It is important to note that the computation of a single

path is independent of other paths' computation. Hence, the construction of the losses can be fully paralleled.

Given the mapping from parameters Θ to the loss, i.e., $\text{Loss}_P + \text{Loss}_A$, the remaining task is to optimize Θ to minimize the loss, which can be completely outsourced to Machine Learning packages like TensorFlow or PyTorch in Python. The coding work of our numerical method is limited to the construction of two losses: Loss_P and Loss_A. The following two sections contain more detailed steps of constructing losses functions for specific models.

2 Exporting Dynamics

In this section, we transform a single firm's exporting dynamics problem in Alfaro-Urena, Castro-Vincenzi, Fanelli and Morales (2023) into the continuous-time setting. Because of the interdependence between different destinations, an exporting firm solves a combinatorial choice in each period while taking into account its dynamic effects in the discrete-time setting considered by Alfaro-Urena et al. (2023).

In our continuous-time setting, the chance that a firm decides whether to export to multiple destinations simultaneously is negligible. Nevertheless, the market interdependence or the complementarity is still kept as the long-run impacts of binary choices at any time are fully captured or encoded by the continuation value of the firm. Hence, we transform the combinatorial problem in Alfaro-Urena et al. (2023) into a sequential binary choice problem, whose static step is trivial to solve.

2.1 Basic Setting

Consider a firm could export to potential $\mathbf{J}$ destinations, whose exporting status is captured by a vector

$$y_t \equiv [y_{t,1}, y_{t,2}, \cdots, y_{t,\mathbf{J}}]^T,$$

where $y_{t,j}$ is a dummy variable indicating whether the firm exports to destination j at time t. If the firm exports to country j, its profit flow is

$$\pi\left(y_t, \nu_{jt}; j\right) = \zeta_j - \nu_{t,j} + \sum_{m \neq j} y_{t,m} c_{jm}$$

where ζ_j is time-invariant export revenue, $\nu_{t,j}$ is the export cost following

$$\mathrm{d}\nu_{t,j} = -\theta\left(\nu_{t,j} - \bar{\nu}_j\right)\mathrm{d}t + \sigma_j^{\nu,0}\mathrm{d}W_t^0 + \sigma_j^{\nu,1}\mathrm{d}W_t^1,$$

where c_{jm} captures the complementarities between export destinations. $\left\{W_t^0, W_t^1\right\}$ are two independent standard Brownian motions, which are common shocks that drive export costs across different destinations. The algorithm could easily accommodate settings with time-varying stochastic export revenues, destination-specific shocks, and multiple macro shocks.

If a firm does not have an exporting channel to destination j by time t, over a time interval $[t, t + \mathrm{d}t]$, it is granted a chance with probability $\lambda^0 \mathrm{d}t$ of establishing such a channel by paying

a one-time fixed cost s_j. Once such a channel exists, it will last until a Poisson shock arrives and the intensity of the shock is λ^1. It is assumed that the chance of establishing an exporting channel or the vanishing of a channel is independent across different destinations. Therefore, we can overlook the scenario that the firm decides multiple exporting destinations simultanenously within a short time interval $[t, t + \mathrm{d}t]$ because its probability is the order of $(\mathrm{d}t)^N, N \geq 2$. The combinatory choice is the main challenge faced by Alfaro-Urena et al. (2023).

The dynamic optimization problem of a firm is to decide whether to establish an exporting channel whenever it has a chance to maximize

$$V(y_0, \nu_0) = \max_{y_t} \int_0^{+\infty} e^{-\rho t} \left(\sum_{j=1}^{\mathbf{J}} y_{t,j} \pi(y_t, \nu_{t,j}; j) - s_j \mathbf{1}\{y_{t,j} - y_{t-,j} > 0\} \right) \mathrm{d}t,$$

where the state variables of the firm are y_t and $\nu_t \equiv [\nu_{t,1}, \nu_{t,2}, \cdots, \nu_{t,\mathbf{J}}]$. Note that given the value function $V(y_t, \nu_t)$, the firm's policy function is straightforward. It will establish an exporting channel to destination j if the firm is granted the chance and

$$V\left(y_{t-} + \mathbf{1}^j, \nu_t\right) \geq V(y_{t-}, \nu_t) + s_j,$$

where $\mathbf{1}^j$ is a $\mathbf{J}$-dimensional vector whose j'th element is one and other elements are zeroes. Although the continuous-time setting avoids the complicated combinatorial decisions faced by discrete-time settings, the model still preserves the interdependence between destinations as we will show later that the increase in the value function will be higher if forming an exporting channel to a new destination contributes more to other destinations' profits.

The current setting can accommodate **active searching** by assuming a certain cost function of the search effort that increases λ^0, the chance of establishing an exporting channel. The marginal benefit of searching is

$$\max\left\{V\left(y_{t-} + \mathbf{1}^j, \nu_t\right) - V(y_{t-}, \nu_t) - s_j, 0\right\}.$$

Note that allowing for active searching does not increase the dimensionality of the firm's dynamic optimization problem.

2.2 Probabilistic and Analytic Formulations

As the above discussion indicates, the value function plays a critical role for characterizing a firm's optimal dynamic choices. In this section, we will present probabilistic and analytic formulations that define the value function, and both play a critical role in numerical schemes that solve for $V(y_t, \nu_t)$.

The BSDE that gives rise to $V(y,\nu)$ is

$$\begin{aligned}
\mathrm{d}V_t = &-\left(\sum_{j=1}^{\mathbf{J}} y_{t,j}\pi\left(y_t,\nu_{t,j};j\right)-\rho V_t\right)\mathrm{d}t+\sigma_t^{V,0}\mathrm{d}W_t^0+\sigma_t^{V,1}\mathrm{d}W_t^1\\
&+\sum_{j=1}^{\mathbf{J}}\left(1-y_{t-,j}\right)\max\left\{V\left(y_{t-}+\mathbf{1}^j,\nu_t\right)-V\left(y_{t-},\nu_t\right)-s_j,0\right\}\mathrm{d}\Lambda_t^{0,j}\\
&+\sum_{j=1}^{\mathbf{J}} y_{t-,j}\left(V\left(y_{t-}-\mathbf{1}^j,\nu_t\right)-V\left(y_{t-},\nu_t\right)\right)\mathrm{d}\Lambda_t^{1,j}
\end{aligned}$$

where $\mathrm{d}\Lambda_t^{0,j}$ indicates the shock that a firm is granted the opportunity to start exporting to destination j and $\mathrm{d}\Lambda_t^{1,j}$ is the shock that the firm's exporting channel to destination j vanishes. Over time $[t,t+\mathrm{d}t]$, the continuation declines by utility flow that has been realized $\sum_{j=1}^{\mathbf{J}} y_{jt}\pi\left(y_t,\nu_{jt};j\right)\mathrm{d}t$ and increases by the discounting effect $\rho V_t\mathrm{d}t$ that only applies to V_t rather than $V_{t+\mathrm{d}t}$. $\sigma_t^{V,0}$ and $\sigma_t^{V,1}$ capture the impacts of the two Brownian shocks on the continuation value. The coefficient of $\mathrm{d}\Lambda_t^{0,j}$ is the increase in the firm's continuation value if it has the opportunity to initialize an exporting channel to destination j; The coefficient of $\mathrm{d}\Lambda_t^{0,j}$ is the change to the continuation value if the existing channel to destination j disappears.

The analytic formulation is well-known in the economics profession. Given Δ is a small positive number,

$$\begin{aligned}
V\left(y_t,\nu_t\right) &\simeq \sum_{j=1}^{\mathbf{J}} y_{t,j}\pi_t\left(y_t,\nu_{jt};j\right)\Delta+e^{-\rho\Delta}E_t\left[V\left(y_{t+\Delta},\nu_{t+\Delta}\right)\right]\\
&\simeq \sum_{j=1}^{\mathbf{J}} y_{t,j}\pi_t\left(y_t,\nu_{jt};j\right)\Delta+e^{-\rho\Delta}E_t\left[V\left(y_{t+\Delta},\nu_{t+\Delta}\right)\right]-E_t\left[V\left(y_{t+\Delta},\nu_{t+\Delta}\right)\right]+E_t\left[V\left(y_{t+\Delta},\nu_{t+\Delta}\right)\right]\\
&\simeq \sum_{j=1}^{\mathbf{J}} y_{t,j}\pi_t\left(y_t,\nu_{jt};j\right)\Delta-\rho V\left(y_t,\nu_t\right)\Delta+E_t\left[V\left(y_{t+\Delta},\nu_{t+\Delta}\right).\right]
\end{aligned}$$

After a few steps of derivation, the calculation of $V\left(y_t,\nu_t\right)$ boils down to the term $E_t\left[V\left(y_{t+\Delta},\nu_{t+\Delta}\right)\right]$, which can be approximated by

$$\begin{aligned}
E_t\left[V\left(y_{t+\Delta},\nu_{t+\Delta}\right)\right] \simeq &\frac{1}{P_t}E_t\left[V\left(y_{t+\Delta},\nu_{t+\Delta}\right)\mid y_{t+\Delta}=y_t\right]+\frac{1-e^{-\lambda_1\Delta}}{P_t}\sum_{j=1}^{\mathbf{J}} y_{t,j}V\left(y_t-\mathbf{1}^j,\nu_t\right)\\
&+\frac{1-e^{-\lambda_0\Delta}}{P_t}\sum_{j=1}^{\mathbf{J}}\left(1-y_{t,j}\right)\max\left\{V\left(y_t+\mathbf{1}^j,\nu_t\right)-s_j,V\left(y_t,\nu_t\right)\right\} \qquad (11)
\end{aligned}$$

where

$$P_t\equiv 1+\left(1-e^{-\lambda_1\Delta}\right)\sum_{j=1}^{\mathbf{J}} y_{t,j}+\left(1-e^{-\lambda_0\Delta}\right)\sum_{j=1}^{\mathbf{J}}\left(1-y_{t,j}\right)$$

In this approximation, we drop terms of order higher than Δ. The term $E_t\left[V\left(y_{t+\Delta},\nu_{t+\Delta}\right)\mid y_{t+\Delta}=y_t\right]$

is further approximated by

$$\frac{1}{4}V\left(y_t, \nu_t - \rho\left(\nu_t - \bar{\nu}\right)\Delta + \sigma^{\nu,0}\sqrt{\Delta} + \sigma^{\nu,1}\sqrt{\Delta}\right) + \frac{1}{4}V\left(y_t, \nu_t - \rho\left(\nu_t - \bar{\nu}\right)\Delta + \sigma^{\nu,0}\sqrt{\Delta} - \sigma^{\nu,1}\sqrt{\Delta}\right)$$
$$+ \frac{1}{4}V\left(y_t, \nu_t - \rho\left(\nu_t - \bar{\nu}\right)\Delta - \sigma^{\nu,0}\sqrt{\Delta} + \sigma^{\nu,1}\sqrt{\Delta}\right) + \frac{1}{4}V\left(y_t, \nu_t - \rho\left(\nu_t - \bar{\nu}\right)\Delta - \sigma^{\nu,0}\sqrt{\Delta} - \sigma^{\nu,1}\sqrt{\Delta}\right),$$

which takes four pairs of $\left(\mathrm{d}W_t^0, \mathrm{d}W_t^1\right)$'s realizations.[1] Note that we will make use of the four realizations and construct the loss of probabilistic formulation along each realization.

2.3 Numerical Schemes

We approximate the value function $V\left(y, \nu\right)$ and its volatility terms $\sigma^{V,0}\left(y, \nu\right)$ and $\sigma^{V,1}\left(y, \nu\right)$ with feedforward neural network, which consists of three shared layers followed by one subnet for $V(\cdot)$ and one for $\sigma^{V,0}(\cdot)$ and $\sigma^{V,0}(\cdot)$. Each subnet contains two independent layers. Each layer has 256 nodes.[2]

Our numerical scheme is simulation-based. The majority of coding is to construct the loss function based on simulated sample paths. The deep learning package like TensorFlow or PyTorch in Python will assume the job of optimizing the parameters of the neural network and minimizing the loss function.

We choose an arbitrary terminal date T and discretized the interval $[0, T]$ into multiple subinterval of length Δ. At initial date $t = 0$, we randomly generate N firms with initial states $(y_0^n, \nu_0^n, n = 1, \cdots, N)$, where the sample index n will be dropped hereafter as the numerical operations are identical across different firms except that different sample paths experience different realization of shocks. For each sample path, two independent Brownian processes $\left\{W_t^0, W_t^1\right\}$ are generated, i.e.,

$$W_{t+\Delta}^i - W_t^i \sim \mathbf{N}\left(0, \Delta\right), i = 0, 1.$$

There is no realization of Poisson shocks along each path since the analytic formulation has incorporated their impacts on the continuation value function. Hence, we can drop the time subscript of y_t.

The numerical operation is the same for each interval $[t, t + \Delta]$. Taking the state $[y, \nu_t]$ entering the interval and the continuation value V_t, the construction of the loss function is as follow

[1] The choice of the four realizations is guided by Gauss-Hermite quadrature.

[2] We refer readers to Chapter 6 of Goodfellow et al. (2016) and Chapter 5 of Zhang et al. (2023) for terminologies of deep learning.

1. Calculate the firm's profit π_t

$$\pi_t = y^T(\zeta - \nu_t) + y^T C y, \text{ where}$$
$$C = \begin{bmatrix} 0 & c_{0,1} & \cdots & c_{0,J-2} & c_{0,J-1} \\ c_{1,0} & 0 & \cdots & c_{1,J-2} & c_{1,J-1} \\ c_{2,0} & c_{2,1} & \cdots & c_{2,J-2} & c_{2,J-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ c_{J-1,0} & c_{J-1,1} & \cdots & c_{J-1,J-2} & 0 \end{bmatrix}$$

2. Calculate $E_t[V(y_{t+\Delta}, \nu_{t+\Delta})]$. To calculate $E_t[V(y, \nu_{t+\Delta}) \mid y_{t+\Delta} = y]$, first update $\nu_{t,j}, j = 1, \cdots, J$

$$\begin{aligned}
\nu^{uu}_{t+\Delta,j} &= \nu_t - \theta(\nu_t - \bar{\nu}_j)\Delta + \sigma_j^{\nu,0}\sqrt{\Delta} + \sigma_j^{\nu,1}\sqrt{\Delta} \\
\nu^{ud}_{t+\Delta,j} &= \nu_t - \theta(\nu_t - \bar{\nu}_j)\Delta + \sigma_j^{\nu,0}\sqrt{\Delta} - \sigma_j^{\nu,1}\sqrt{\Delta} \\
\nu^{du}_{t+\Delta,j} &= \nu_t - \theta(\nu_t - \bar{\nu}_j)\Delta - \sigma_j^{\nu,0}\sqrt{\Delta} + \sigma_j^{\nu,1}\sqrt{\Delta} \\
\nu^{dd}_{t+\Delta,j} &= \nu_t - \theta(\nu_t - \bar{\nu}_j)\Delta - \sigma_j^{\nu,0}\sqrt{\Delta} - \sigma_j^{\nu,1}\sqrt{\Delta}.
\end{aligned}$$

Then, to plug in the value function approximated by the neural network $V(\cdot,\cdot)$, we have

$$E_t[V(y, \nu_{t+\Delta}) \mid y_{t+\Delta} = y] = \frac{V\left(y, \nu^{uu}_{t+\Delta,j}\right) + V\left(y, \nu^{ud}_{t+\Delta,j}\right) + V\left(y, \nu^{du}_{t+\Delta,j}\right) + V\left(y, \nu^{dd}_{t+\Delta,j}\right)}{4}$$

Evaluate $V\left(y + \mathbf{1}^j, \nu_t\right)$ if $y_j = 0$; evaluate $V\left(y - \mathbf{1}^j, \nu_t\right)$ if $y_j = 1$. Then, we calculate $E_t[V(y_{t+\Delta}, \nu_{t+\Delta})]$ according to (11) and the loss of the analytic formulation is

$$\text{Loss}_A \Leftarrow \text{Loss}_A + \frac{\Delta}{T}\left((1 + \rho\Delta)V_t - \pi_t\Delta - E_t[V(y_{t+\Delta}, \nu_{t+\Delta})]\right)^2$$

3. Given the realized Brownian shocks $W^i_{t+\Delta} - W^i_t, i = 0, 1$, update forward SDE of ν_t and BSDE of V_t

$$\begin{aligned}
\nu_{t+\Delta,j} &= \nu_{t,j} - \theta(\nu_{t,j} - \bar{\nu}_j)\Delta + \sigma_j^{\nu,0}\left(W^0_{t+\Delta} - W^0_t\right) + \sigma_j^{\nu,1}\left(W^1_{t+\Delta} - W^1_t\right), j = 1, \cdots, J \\
V_{t+\Delta} &= V_t - (\pi_t - \rho V_t)\Delta + \sigma_t^{V,0}\left(W^0_{t+\Delta} - W^0_t\right) + \sigma_t^{V,1}\left(W^1_{t+\Delta} - W^1_t\right),
\end{aligned}$$

where $\sigma^{V,0}_{t,j}, \sigma^{V,1}_{t,j}$ are given by the network with the state input (y, ν_t). To take advantage of four realizations used for calculating Loss_A, compute

$$\begin{aligned}
V^{uu}_{t+\Delta} &= V_t - (\pi_t - \rho V_t)\Delta + \sigma_t^{V,0}\sqrt{\Delta} + \sigma_t^{V,1}\sqrt{\Delta} \\
V^{ud}_{t+\Delta} &= V_t - (\pi_t - \rho V_t)\Delta + \sigma_t^{V,0}\sqrt{\Delta} - \sigma_t^{V,1}\sqrt{\Delta} \\
V^{du}_{t+\Delta} &= V_t - (\pi_t - \rho V_t)\Delta - \sigma_t^{V,0}\sqrt{\Delta} + \sigma_t^{V,1}\sqrt{\Delta} \\
V^{dd}_{t+\Delta} &= V_t - (\pi_t - \rho V_t)\Delta - \sigma_t^{V,0}\sqrt{\Delta} - \sigma_t^{V,1}\sqrt{\Delta}.
\end{aligned}$$

Table 1: Loss of Untrained Sample

	analytic loss	*probabilistic loss*
average	1.89×10^{-7}	1.06×10^{-7}
st.d.	3.37×10^{-7}	2.07×10^{-7}
$85th$ *percentile*	3.28×10^{-7}	1.83×10^{-7}
$90th$ *percentile*	4.46×10^{-7}	2.56×10^{-7}
$95th$ *percentile*	7.06×10^{-7}	4.08×10^{-7}

The loss of the probabilistic formulation is

$$\begin{aligned}\text{Loss}_P \Leftarrow \text{Loss}_P &+ \frac{\Delta}{T}\left(V_{t+\Delta} - V\left(y, v_{t+\Delta}\right)\right)^2 \\ &+ \frac{\Delta}{T}\left(V^{uu}_{t+\Delta} - V\left(y, v^{uu}_{t+\Delta}\right)\right)^2 + \frac{\Delta}{T}\left(V^{ud}_{t+\Delta} - V\left(y, v^{ud}_{t+\Delta}\right)\right)^2 \\ &+ \frac{\Delta}{T}\left(V^{du}_{t+\Delta} - V\left(y, v^{du}_{t+\Delta}\right)\right)^2 + \frac{\Delta}{T}\left(V^{dd}_{t+\Delta} - V\left(y, v^{dd}_{t+\Delta}\right)\right)^2\end{aligned}$$

The calculation of the next sub-interval will take $(y, \nu_{t+\Delta})$ as the entering state along with the continuation value $V_{t+\Delta}$. The computation repeats until the terminal date T. The deep learning package will be used to minimize the total loss

$$\text{Loss}_A + \text{Loss}_P.$$

2.4 Numerical Exercises

For numerical exercises, we try to recover parameter values used or estimated in Alfaro-Urena et al. (2023). If not, we use the closest alternative parameter values. The difficulty of combinatorial problems is the enormous number of possible combinations to consider. Our numerical example contains 100 destinations, which leads to 2^{100} combinations, a number larger than what a standard 64-bit system can represent.

Accuracy. The critical aspect of our numerical scheme is whether we can derive the value function up to a certain degree of accuracy with only a limited number of sample paths that we simulate, which is $400,000$ for current exercise. To assess the performance of our numerical scheme, we generate a separate set of $20,000$ sample paths that are not used for training or solving for the value function, and plug in these untrained samples to the loss function above. Table 1 presents the losses normalized by the value function, i.e.,

$$\frac{1}{V^2}\left(\text{Loss}_A + \text{Loss}_P\right).$$

Considering that the size of our neural network is relatively small, the "out of sample" performance of our numerical scheme is reasonably good.

Complementarity. The key feature of the model by Alfaro-Urena et al. (2023) is the complementarity between exporting destinations. We next examine the contribution of adding a particular destination to the continuation value that cannot be explained by the destination's

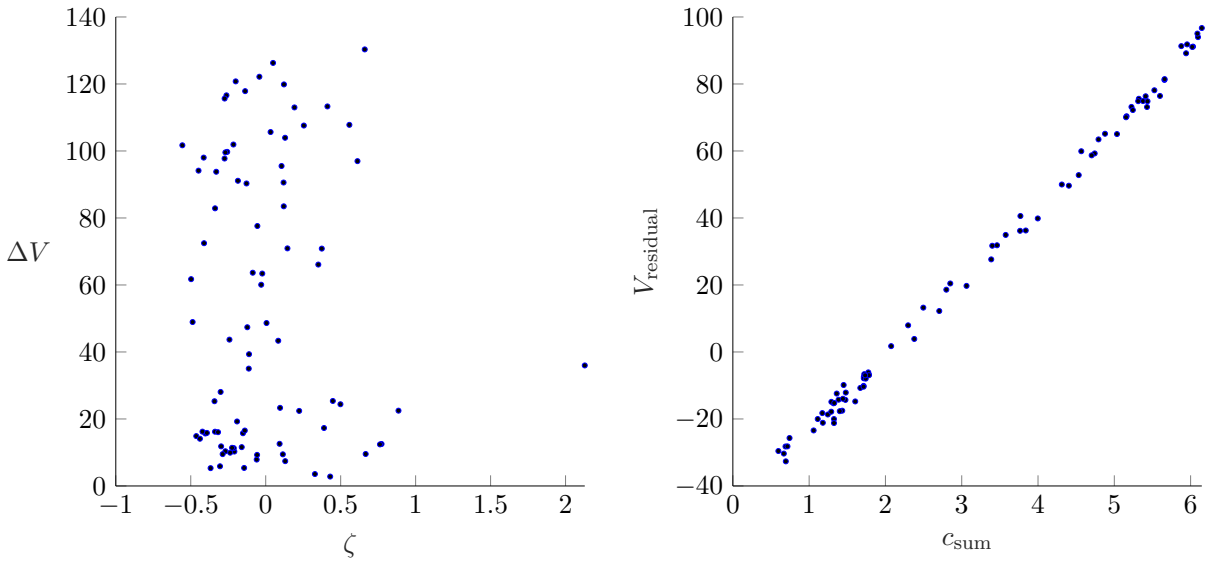

Figure 1: Complementarity

revenue itself. For a given state (y_t, ν_t), we can calculate the increase in continuation value by adding destination j, i.e.,

$$\Delta V(y_t^{-i}, \nu_t) = V(1, y_t^{-i}, \nu_t) - V(0, y_t^{-i}, \nu_t),$$

where y_t^{-i} denotes a vector comprised of all elements of y_t except the i'th one. We simulate $50,000$ sample paths for 1000 years and obtain the "long-run" distribution of (y_t, ν_t), with which we calculate the average of $\Delta V(y_t^{-i}, \nu_t)$ denoted as ΔV^i. The left panel of Figure 1 is the scatter plot $(\Delta V^i, \zeta_i)$ of all destinations that have positive ΔV^i, which indicates that the revenue of a destination cannot explain much of its contribution to the overall value function of the firm. In light of this observation, we run a linear regression of ΔV^i against ζ_i and obtain the residual term denoted as V^i_{residual}. To capture the complementary effect of adding a destination, we calculate the contribution of destination i to other destinations' profits as

$$c^i_{\text{sum}} = \sum_{j=1, j\neq i}^{\mathbf{J}} c_{ji}.$$

The right panel of Figure 1 clearly shows that when a destination contributes more to other markets' profits adding this destination would increase the firm's continuation value by a larger margin.

3 Dynamic Network Formation

We extend the static production network formation model of Kopytov, Mishra, Nimark and Taschereau-Dumouchel (2021) into a continuous-time model. The key distinction of our continuous-time setting is that we can ignore the scenario that multiple input-output links are formed simultaneously within a sufficiently short time interval. Therefore, the massive combinatorial problem is transformed into a sequence of binary choice sub-problems. Nevertheless, the state

space is beyond the capability of conventional numerical methods. The dimension of the state variable is 37×37 given 37 sectors in Kopytov et al. (2021) and each sector's productivity.

3.1 Basic Setting

The economy consists of N sectors, each producing a differentiated product, and a representative consumer, who has the utility function over N types of products

$$\frac{1}{1-\gamma}\exp\left((1-\gamma)\log\left(\prod_{i=1}^{N}\left(\frac{C_i}{\beta_i}\right)^{\beta_i}\right)\right).$$

The consumer supplies a unit of labor inelastically and the labor wage serves as the numeraire. Let β denote $[\beta_1, \beta_2, \cdots, \beta_N]$.

Each sector has a representative firm, which hires labor and takes outputs of other sectors as intermediate inputs. Firms in all sectors behave competitively with zero profits in each period. Firm i or sector i is endowed with a full-pledged production function

$$F\left(\varepsilon_t^i, \alpha^{i1}, \cdots, \alpha^{iN}, X^{i1}, \cdots, X^{iN}\right) = e^{\varepsilon_t^i}\left(L^i\right)^{1-\sum_{j=1}^{N}\alpha^{ij}}\prod_{j=1}^{N}\left(X^{ij}\right)^{\alpha^{ij}},$$

if it is fully connected with all other sectors, where ε_t^i is the TFP, $\left[\alpha^{i1}, \alpha^{i2}, \cdots, \alpha^{iN}\right]$ denoted as α^i is the vector of intermediate input shares, and $\left[X^{i1}, X^{i2}, \cdots, X^{iN}\right]$ denoted as X^i is the vector of inputs. ε_t^i follows

$$\mathrm{d}\varepsilon_t^i = -\phi\left(\varepsilon_t^i - \bar{\varepsilon}^i\right)\mathrm{d}t + \sum_{m=1}^{M}\sigma_{im}\mathrm{d}B_t^m,$$

where $\left(B_t^1, \cdots, B_t^M\right)$ are independent Brownian motions.

Over time, an existing link between sectors may vanish, and new links could be formed. Let $Y_t^i = \left[Y_t^{i1}, Y_t^{i2}, \cdots, Y_t^{iN}\right]$, a vector of dummy variables, captures the linkage status for sector i. Given Y_t^i, the production function is

$$\begin{aligned} F\left(\varepsilon_t^i, \alpha^i, X^i, Y_t^i\right) &= e^{\varepsilon_t^i - a^i\left(Y_t^i\right)}\left(L^i\right)^{1-\sum_{j=1}^{N}\alpha^{ij}Y_t^{ij}}\prod_{j=1}^{N}\left(X^{ij}\right)^{\alpha^{ij}Y_t^{ij}}, \text{ where} \\ a^i\left(Y_t^i\right) &= \min\left\{1, \left(\alpha^i \odot Y_t^i - \alpha^i\right)^T H\left(\alpha^i \odot Y_t^i - \alpha^i\right)\right\}, \\ H &= W^T K W, \end{aligned} \tag{12}$$

$\odot$ denotes the element-by-element product, W is a $(N+1) \times N$ matrix, and K is a $(N+1) \times$

$(N+1)$ positive definite diagonal matrix

$$W = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}, K = \begin{bmatrix} \kappa^{i0} & 0 & 0 & \cdots & 0 \\ 0 & \kappa^{i1} & 0 & \cdots & 0 \\ 0 & 0 & \kappa^{i2} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \kappa^{iN} \end{bmatrix}.$$

The function $a^i\left(Y_t^i\right)$ captures the negative impact of the technology's deviation from the full-pledged benchmark. Given an input-output network $Y_t = \left[Y_t^{ij}\right]$, the productivity vector $\varepsilon_t = \left[\varepsilon_t^1, \cdots, \varepsilon_t^N\right]^N$, productivity deviation $a\left(Y_t\right) = \left[a^1\left(Y_t^1\right), \cdots, a^N\left(Y_t^N\right)\right]$, the competitive equilibrium yields the utility for the consumer

$$\begin{aligned} u\left(Y_t, \varepsilon_t\right) &= \frac{\exp\left(\left(1-\gamma\right) f_t\right)}{1-\gamma}, \text{ where} \\ f_t &= \beta \mathscr{L}\left(Y_t\right)\left(\varepsilon_t - a\left(Y_t\right)\right) \\ \mathscr{L}\left(Y_t\right) &= \left(I - A \odot Y_t\right)^{-1}. \end{aligned}$$

$\mathscr{L}\left(Y_t\right)$ is the Leontief inverse, and f_t can be interpreted as the log of GDP at time t.

3.2 Social Planner

We assume that a social planner makes the network formation decisions. To simplify the computation, assume that an N-state Markov chain z_t governs which sector's technology is subject to change over $[t, t+\mathrm{d}t]$. When $z_t = i$, each existing link that sector j supplies its input to sector i (i.e., $Y_{t-}^{ij} = 1$) may vanish with a probability $\lambda \mathrm{d}t$ independently. Also when $z_t = i$, the social planner gains the opportunity to establish a new link that sector j supplies to sector i with a probability $\lambda \mathrm{d}t$. This opportunity appears independently across all sector with $Y_{t-}^{ij} = 0$.

The only choice that the social planner makes is whether to establish a sector's input link when she has the opportunity to do so. The objective function for the social planner is

$$V\left(Y_t, z_t, \varepsilon_t\right) = \max_{Y_t} \int_t^{+\infty} e^{-\rho s} u\left(Y_s, \varepsilon_s\right) \mathrm{d}s$$

with state variables Y_t, z_t, and ε_t. The key to characterize the network formation of the economy, i.e., the social planner's policy function, is to solve for the value function. Next, we will illustrate

The BSDE that the continuation value V_t follows is

$$\mathrm{d}V_t = -\left(u\left(Y_t, \varepsilon_t\right) - \rho V_t\right) \mathrm{d}t + \sum_{m=1}^{M} \sigma_t^{V,m} \mathrm{d}B_t^m$$

along the path where no jump risks are realized, where $\sigma_t^{V,m}, m = 1, \cdots, M$ reflect the impacts

of exogenous shocks to productivity. The analytic formulation is

$$\begin{aligned} V\left(Y_t, z_t, \varepsilon_t\right) &= u\left(Y_t, \varepsilon_t\right)\Delta + e^{-\rho\Delta} E_t\left[V\left(Y_{t+\Delta}, z_{t+\Delta}, \varepsilon_{t+\Delta}\right)\right] \\ &\simeq u\left(Y_t, \varepsilon_t\right)\Delta - \rho V\left(Y_t, z_t, \varepsilon_t\right)\Delta + E_t\left[V\left(Y_{t+\Delta}, z_{t+\Delta}, \varepsilon_{t+\Delta}\right)\right]. \end{aligned}$$

The key of the analytic formulation is to approximate the term $E_t\left[V\left(Y_{t+\Delta}, z_{t+\Delta}, \varepsilon_{t+\Delta}\right)\right]$. When Δ is small enough, suppose $z_t = j$ we have

$$\begin{aligned} E_t\left[V\left(Y_{t+\Delta}, z_{t+\Delta}, \varepsilon_{t+\Delta}\right)\right] = &\frac{1}{P_t} E_t\left[V\left(Y_t, z_{t+\Delta}, \varepsilon_{t+\Delta}\right) \middle| Y_{t+\Delta} = Y_t, z_{t+\Delta} = j\right] \\ &+ \frac{1 - e^{\lambda\Delta}}{P_t} \sum_{i=1, i\neq j}^{N} V\left(Y_t, i, \varepsilon_t\right) + \frac{1 - e^{\lambda\Delta}}{P_t} \sum_{i=1, i\neq j}^{N} Y_t^{ji} V\left(Y_t - \mathbf{1}^{ji}, j, \varepsilon_t\right) \\ &+ \frac{1 - e^{\lambda\Delta}}{P_t} \sum_{i=1, i\neq j}^{N} \left(1 - Y_t^{ji}\right) \max\left\{V\left(Y_t + \mathbf{1}^{ji}, j, \varepsilon_t\right), V\left(Y_t, j, \varepsilon_t\right)\right\}, \end{aligned} \tag{13}$$

where

$$P_t = 1 + \left(1 - e^{\lambda\Delta}\right) 2\left(N - 1\right)$$

The evaluation of $E_t\left[V\left(Y_t, z_{t+\Delta}, \varepsilon_{t+\Delta}\right) \middle| Y_{t+\Delta} = Y_t, z_{t+\Delta} = j\right]$ follows Gauss-Hermite quadrature as in the previous dynamic firm exporting example.

Extensions. The network formation setting above is quite stylized. Our numerical scheme can accommodate various modeling choices. For example, we could allow the social planner to exert costly efforts to maintain existing input-output links or enhance the probability of forming a new link. Such extension does not increase the dimensionality of the problem. Moreover, we could extend the model such that each sector has a delegate who makes the input-output link formation decisions. Although this extension would not increase the dimension of the model, it requires solving more dynamic optimization problems, because of which solving the model requires relatively more computing power. When the network formation is decentralized, we could also consider bargaining that occurs during the network building process given the characterization of continuation values in hand.

3.3 Numerical Schemes

As in the previous example, we approximate the value function $V\left(Y, z, \varepsilon\right)$ and its volatility terms $\sigma_t^{V,1}$ and $\sigma_t^{V,2}$ with feedforward neural network. However, as the dimensionality of the current problem is much higher than the previous one: 37×38 versus 200, although the network has the same number of layers, the number of nodes for each shared layer is 512 instead of 256 in the previous example.

Given an arbitrary terminal date T, interval $[0, T]$ is divided into small ones of length Δ. At initial date $t = 0$, we randomly generate K economies with initial states $\left(Y_0, z_0, \varepsilon_0\right)$. The sample index is omitted for brevity. For each economy, two independent Brownian shocks are generated $\left\{W_t^1, W_t^2\right\}$. As in the previous example, no Poisson shocks are realized for each economy since the analytic formulation has incorporated their impacts on $V\left(Y_t, z_t, \varepsilon_t\right)$. Hence,

the time subscripts of Y_t and z_t are also omitted. As it is not a trivial computational task, we calculate save the Leontief inverse

$$\mathscr{L}(Y) = I + \sum_{i=1}^{10} (A \odot Y)^i$$

and also

$$a^i\left(Y_t^i\right) = \min\left\{1, \left(\alpha^i \odot Y_t^i - \alpha^i\right)^T H \left(\alpha^i \odot Y_t^i - \alpha^i\right)\right\}, i = 1, \cdots, N,$$

for each Y before the simulation starts.

The numerical steps are identical for each period $[t, t+\Delta]$. Taking the entering state $[Y, z, \varepsilon_t]$ and the continuation value V_t, the construction of the loss function follows

1. First calculate the utility

$$u(Y, \varepsilon_t) = \frac{\exp\left((1-\gamma) f_t\right)}{1-\gamma}, \text{ where}$$
$$f_t = \beta \mathscr{L}(Y)\left(\varepsilon_t - a(Y)\right)$$

2. Calculate $E_t\left[V\left(Y_{t+\Delta}, z_{t+\Delta}, \varepsilon_{t+\Delta}\right)\right]$. The first term involved is

$$E_t\left[V\left(Y_{t+\Delta}, z_{t+\Delta}, \varepsilon_{t+\Delta}\right) | Y_{t+\Delta} = Y, z_{t+\Delta} = z\right],$$

which requres the updates of $\varepsilon_{t+\Delta}$. For each j,

$$\begin{aligned}
\varepsilon_{t+\Delta,j}^{uu,j} &= \varepsilon_t^j - \phi\left(\varepsilon_t^j - \bar{\varepsilon}^j\right)\Delta + \sigma_{j,1}\sqrt{\Delta} + \sigma_{j,2}\sqrt{\Delta} \\
\varepsilon_{t+\Delta,j}^{ud} &= \varepsilon_t - \phi\left(\varepsilon_t^j - \bar{\varepsilon}^j\right)\Delta + \sigma_{j,1}\sqrt{\Delta} - \sigma_{j,2}\sqrt{\Delta} \\
\varepsilon_{t+\Delta,j}^{du} &= \varepsilon_t - \phi\left(\varepsilon_t^j - \bar{\varepsilon}^j\right)\Delta - \sigma_{j,1}\sqrt{\Delta} + \sigma_{j,2}\sqrt{\Delta} \\
\varepsilon_{t+\Delta,j}^{dd} &= \varepsilon_t - \phi\left(\varepsilon_t^j - \bar{\varepsilon}^j\right)\Delta - \sigma_{j,1}\sqrt{\Delta} - \sigma_{j,2}\sqrt{\Delta}.
\end{aligned}$$

To plug in the value function approximated by the neural network $V(Y, z, \varepsilon)$, we have

$$\begin{aligned}
&E_t\left[V\left(Y_{t+\Delta}, z_{t+\Delta}, \varepsilon_{t+\Delta}\right) | Y_{t+\Delta} = Y, z_{t+\Delta} = z\right] \\
&= \frac{V\left(Y, z, \varepsilon_{t+\Delta,j}^{uu}\right) + V\left(Y, z, \varepsilon_{t+\Delta,j}^{ud}\right) + V\left(Y, z, \varepsilon_{t+\Delta,j}^{du}\right) + V\left(Y, z, \varepsilon_{t+\Delta,j}^{dd}\right)}{4}
\end{aligned}$$

Evaluate $V\left(Y + \mathbf{1}^{zi}, z, \varepsilon_t\right)$ if $Y^{zi} = 0$; evaluate $V\left(Y - \mathbf{1}^{zi}, z, \nu_t\right)$ if $Y^{zi} = 1$. Then, we calculate $E_t\left[V\left(Y_{t+\Delta}, z_{t+\Delta}, \varepsilon_{t+\Delta}\right)\right]$ according to (13) and the loss of the analytic formulation is

$$\text{Loss}_A \Leftarrow \text{Loss}_A + \frac{\Delta}{T}\left(\left(1 + \rho\Delta\right) V_t - u(Y, \varepsilon_t)\Delta - E_t\left[V\left(Y_{t+\Delta}, z_{t+\Delta}, \varepsilon_{t+\Delta}\right)\right]\right)^2$$

3. Use Brownian shocks $W^i_{t+\Delta} - W^i_t, i = 1, 2$, to update forward SDE of ε_t and BSDE of V_t

$$\varepsilon^j_{t+\Delta} = \varepsilon^j_t - \phi\left(\varepsilon^j_t - \bar{\varepsilon}^j\right)\Delta + \sigma_{j,1}\left(W^1_{t+\Delta} - W^1_t\right) + \sigma_{j,2}\left(W^2_{t+\Delta} - W^2_t\right), j = 1, \cdots, J$$
$$V_{t+\Delta} = V_t - \left(u\left(Y, \varepsilon_t\right) - \rho V_t\right)\Delta + \sigma^{V,1}_t\left(W^1_{t+\Delta} - W^1_t\right) + \sigma^{V,2}_t\left(W^2_{t+\Delta} - W^2_t\right),$$

where $\sigma^{V,0}_{t,j}, \sigma^{V,1}_{t,j}$ are given by the network with the state input (Y, z, ε_t). To fully utilize the four realizations of $\left(\mathrm{d}W^1_t, \mathrm{d}W^2_t\right)$ used for calculating Loss_A, compute

$$\begin{aligned}
V^{uu}_{t+\Delta} &= V_t - \left(u\left(Y, \varepsilon_t\right) - \rho V_t\right)\Delta + \sigma^{V,1}_t\sqrt{\Delta} + \sigma^{V,2}_t\sqrt{\Delta} \\
V^{ud}_{t+\Delta} &= V_t - \left(u\left(Y, \varepsilon_t\right) - \rho V_t\right)\Delta + \sigma^{V,1}_t\sqrt{\Delta} - \sigma^{V,2}_t\sqrt{\Delta} \\
V^{du}_{t+\Delta} &= V_t - \left(u\left(Y, \varepsilon_t\right) - \rho V_t\right)\Delta - \sigma^{V,1}_t\sqrt{\Delta} + \sigma^{V,2}_t\sqrt{\Delta} \\
V^{dd}_{t+\Delta} &= V_t - \left(u\left(Y, \varepsilon_t\right) - \rho V_t\right)\Delta - \sigma^{V,1}_t\sqrt{\Delta} - \sigma^{V,2}_t\sqrt{\Delta}.
\end{aligned}$$

The loss of the probabilistic formulation is

$$\begin{aligned}
\mathrm{Loss}_P \Leftarrow \mathrm{Loss}_P &+ \frac{\Delta}{T}\left(V_{t+\Delta} - V\left(Y, z, \varepsilon_{t+\Delta}\right)\right)^2 \\
&+ \frac{\Delta}{T}\left(V^{uu}_{t+\Delta} - V\left(Y, z, \varepsilon^{uu}_{t+\Delta}\right)\right)^2 + \frac{\Delta}{T}\left(V^{ud}_{t+\Delta} - V\left(Y, z, \varepsilon^{ud}_{t+\Delta}\right)\right)^2 \\
&+ \frac{\Delta}{T}\left(V^{du}_{t+\Delta} - V\left(Y, z, \varepsilon^{du}_{t+\Delta}\right)\right)^2 + \frac{\Delta}{T}\left(V^{dd}_{t+\Delta} - V\left(Y, z, \varepsilon^{dd}_{t+\Delta}\right)\right)^2.
\end{aligned}$$

The calculation of the following period will take $(Y, z, \varepsilon_{t+\Delta})$ as the entering state along with the continuation value $V_{t+\Delta}$. The computation repeats until the terminal date T. The deep learning package will be used to minimize the total loss

$$\mathrm{Loss}_A + \mathrm{Loss}_P.$$

3.4 Numerical Exercises

Regarding parameter values of numerical exercises, we try to uncover values used or estimated in Kopytov et al. (2021). Otherwise, we use our most educated conjectures according to the literature. The dimensionality of the current problem is around 6 times larger than the previous problem (37×38 versus 200). Therefore, we choose a larger neural network, although its depth is still the same as the previous example (i.e., the same number of layers).

Table 2: Loss of Untrained Sample

	analytic loss	*probabilistic loss*
average	7.40×10^{-6}	9.44×10^{-8}
st.d.	2.08×10^{-6}	1.35×10^{-7}
85th percentile	9.60×10^{-6}	1.91×10^{-7}
90th percentile	1.02×10^{-5}	2.45×10^{-7}
95th percentile	1.12×10^{-5}	3.53×10^{-7}

Accuracy. The size of simulated sample paths used for training the value function is $220,000$,

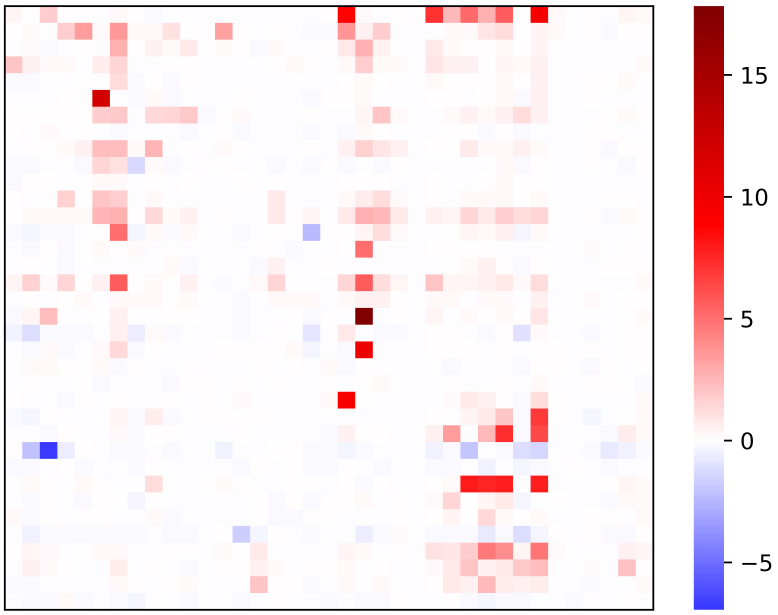

Figure 2: Marginal Value of a Input Link, ΔV
The heat map presents the average increase in the social planner's value function if an input-output is established. The magnitude is of the percentage increase in the value function. The red color indicates the increase in the continuation value, and the blue color indicates the decrease.

which is smaller than the previous example. This is because the current example is a general equilibrium model and the calculation of each sample path occupies much more memory that that of the previous partial equilibrium model. We use $10,000$ untrained sample paths to test the accuracy and we normalize the loss with the scale of the value function. Table 2 displays summary statistics of the losses based on untrained sample. It appears that our numerical method approximates the value function to a reasonably good degree.

Network Effects. Does the social planner find it optimal to form an input-output link when it is optimal for an output producer's perspective? Our simple model may shed light on this question. A sector's production technology, equation (12) implies that any deviation from the benchmark technology would lower the sector's TFP. Therefore, it is always optimal from the sector's perspective to form a link if the relevant inputs are in its full-pledged technology. Our numerical results suggest that the social planner may have different thoughts. We define ΔV^{ij} as

$$E\big[V(\mathbf{1}^{ij} + Y_t^{-ij}, z_t, \varepsilon) - V(Y_t^{-ij}, z_t, \varepsilon)\big],$$

where Y_t^{-ij} denotes an input-output network matrix with ij'th element being zero, and the expectation is taken over a simulated sample. The heat map of Figure 2 shows that it is not socially optimal to form certain input-output links although such links benefit sectors that expand their intermediate inputs.

4 Final Remarks

The key insight of our paper is that a large class of "combinatorial" problems in economics are essentially approximations of sequential moves made by economic agents in real life. Given this simple idea, we transform typical combinatorial problems into sequential binary choice problems in the continuous-time setting. Although such transformation leads to rather simple calculation in static steps, the dimensionality of the state space is enormous for conventional numerical methods. To overcome the curse of dimensionality, we apply the recent deep learning-based probabilistic approach.

There are several features of our numerical approach that previous examples do not fully display. First, the deep learning-based probabilistic approach provides the global solution of models with aggregate shocks. Hence, scholars using our method can easily explore the transition dynamics of a network given an idiosyncratic or aggregate shock. Second, the probabilistic framework is general enough to incorporate almost all economic modeling ingredients. We refer readers to Huang (2023b,a) for applications in international finance, asset pricing, and heterogeneous-agent macroeconomics. We believe the deep learning-based probabilistic approach opens a brand new avenue for scholars in international trade, IO, social network, and input-output linkages to draw tighter connections with international finance, asset pricing, and macroeconomics.

Although we only show how to solve models with a given set of parameters, our numerical method also helps quantitative works substantially. Since the dimensionality is no longer a primary concern, we can treat all parameters, which we are interested in estimating, as time-invariant state variables and solve the "large" model in both state and parameter space at once. Then, we can search for the best parameter estimations within the parameter space without re-solving the model every time trying new parameter estimates.

References

Acemoglu, Daron and Pablo D Azar (2020) "Endogenous production networks," *Econometrica*, Vol. 88, pp. 33–82.

Alfaro-Urena, Alonso, Juanma Castro-Vincenzi, Sebastián Fanelli, and Eduardo Morales (2023) "Firm export dynamics in interdependent markets,"Technical report, National Bureau of Economic Research.

Antras, Pol, Teresa C Fort, and Felix Tintelnot (2017) "The margins of global sourcing: Theory and evidence from us firms," *American Economic Review*, Vol. 107, pp. 2514–2564.

Arkolakis, Costas, Fabian Eckert, and Rowan Shi (2023) "Combinatorial Discrete Choice: A Quantitative Model of Multinational Location Decisions,"Technical report, National Bureau of Economic Research.

Carvalho, Vasco M (2014) "From micro to macro via production networks," *Journal of Economic Perspectives*, Vol. 28, pp. 23–48.

Carvalho, Vasco M and Alireza Tahbaz-Salehi (2019) "Production networks: A primer," *Annual Review of Economics*, Vol. 11, pp. 635–663.

Christakis, Nicholas, James Fowler, Guido W Imbens, and Karthik Kalyanaraman (2020) "An empirical model for strategic network formation," in *The Econometric Analysis of Network Data*: Elsevier, pp. 123–148.

Currarini, Sergio, Matthew O Jackson, and Paolo Pin (2009) "An economic model of friendship: Homophily, minorities, and segregation," *Econometrica*, Vol. 77, pp. 1003–1045.

——— (2010) "Identifying the roles of choice and chance in network formation: Racial biases in high school friendships," *Proceedings of the National Academy of Sciences*, Vol. 107, pp. 4857–4861.

Dhyne, Emmanuel, Ayumu Ken Kikkawa, Xianglong Kong, Magne Mogstad, and Felix Tintelnot (2023) "Endogenous production networks with fixed costs," *Journal of International Economics*, Vol. 145, p. 103841.

Doraszelski, Ulrich and Kenneth L Judd (2012) "Avoiding the curse of dimensionality in dynamic stochastic games," *Quantitative Economics*, Vol. 3, pp. 53–93.

Fan, Ying and Chenyu Yang (2020) "Competition, product proliferation, and welfare: A study of the US smartphone market," *American Economic Journal: Microeconomics*, Vol. 12, pp. 99–134.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016) *Deep learning*: MIT press.

Han, Jiequn, Arnulf Jentzen, and Weinan E (2018) "Solving high-dimensional partial differential equations using deep learning," *Proceedings of the National Academy of Sciences*, Vol. 115, pp. 8505–8510.

Hendel, Igal (1999) "Estimating multiple-discrete choice models: An application to computerization returns," *The Review of Economic Studies*, Vol. 66, pp. 423–446.

Houde, Jean-François, Peter Newberry, and Katja Seim (2023) "Nexus Tax Laws and Economies of Density in E-Commerce: A Study of Amazon's Fulfillment Center Network," *Econometrica*, Vol. 91, pp. 147–190.

Huang, Ji (2023a) "Breaking the Curse of Dimensionality in Heterogeneous-Agent Models: A Deep Learning-Based Probabilistic Approach," *Available at SSRN 4649043*.

——— (2023b) "A Probabilistic Solution to High-Dimensional Continuous-Time Macro and Finance Models."

Jackson, Matthew O (2010) *Social and economic networks*: Princeton university press Princeton.

Jia, Panle (2008) "What happens when Wal-Mart comes to town: An empirical analysis of the discount retailing industry," *Econometrica*, Vol. 76, pp. 1263–1316.

Kopytov, Alexandr, Bineet Mishra, Kristoffer Nimark, and Mathieu Taschereau-Dumouchel (2021) "Endogenous production networks under supply chain uncertainty," *Available at SSRN 3936969*.

Liu, Ernest and Aleh Tsyvinski (2024) "A Dynamic Model of Input-Output Networks," *Review of Economic Studies*, p. rdae012.

Mele, Angelo (2017) "A structural model of dense network formation," *Econometrica*, Vol. 85, pp. 825–850.

Oberfield, Ezra (2018) "A theory of input–output architecture," *Econometrica*, Vol. 86, pp. 559–589.

Oberfield, Ezra, Esteban Rossi-Hansberg, Pierre-Daniel Sarte, and Nicholas Trachter (2024) "Plants in space," *Journal of Political Economy*, Vol. 132, pp. 000–000.

Pardoux, Etienne and Shige Peng (1990) "Adapted solution of a backward stochastic differential equation," *Systems & Control Letters*, Vol. 14, pp. 55–61.

Taschereau-Dumouchel, Mathieu (2020) "Cascades and fluctuations in an economy with an endogenous production network," *Available at SSRN 3115854*.

Tintelnot, Felix (2017) "Global production with export platforms," *The Quarterly Journal of Economics*, Vol. 132, pp. 157–209.

Zhang, Aston, Zachary C Lipton, Mu Li, and Alexander J Smola (2023) *Dive into deep learning*: Cambridge University Press.