ESSAYS ON ESTIMATING DYNAMIC DISCRETE CHOICE MODELS: METHODS AND APPLICATIONS

A Dissertation presented to
the Faculty of the Graduate School
at the University of Missouri

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

---

by

FANGDA WANG

Dr. Shawn Ni, Dissertation Supervisor

MAY 2022

The undersigned, appointed by the Dean of the Graduate School, have examined the dissertation entitled:

ESSAYS ON ESTIMATING DYNAMIC DISCRETE
CHOICE MODELS: METHODS AND APPLICATIONS

presented by Fangda Wang,
a candidate for the degree of Doctor of Philosophy and hereby certify that, in their opinion, it is worthy of acceptance.

Dr. Shawn Ni

Dr. Michael Podgursky

Dr. David Kaplan

Dr. Chong He

ACKNOWLEDGMENTS

First, I would like to thank my advisor, Dr. Shawn Ni for his support in research as well as in life. I am so lucky to receive academic training and start research life under his supervision. And it is not possible to accomplish this dissertation without his advice and encouragement along my PhD journey. I would also like to thank Dr. Michael Podgursky. I have always learned a lot from his insightful comments and suggestions. The experience of doing original and rigorous research with Dr. Ni and Dr. Podgursky is invaluable to me; those Eureka moments are wonderful.

Second, I would like to express my thanks to Dr. David Kaplan, who sits on my committee and is also the DDS. He is so kind and warm and willing to provide almost all kinds of help a PhD student may need. I would also like to thank Dr. Chong He, who provides very helpful suggestions for revising and extending my dissertation.

Third, I would like to express gratitude for the support from the Economics Department and EPARC. And I also appreciate the suggestions and supports by Dr. Xiqian Wang, Dr. Cheng Qian, Dr. Yong Bian, Qinhua Xie, and Qian Wu.

Finally, special thanks to my family and my girlfriend, for their endless caring and love, in such a volatile world.

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

ABSTRACT

Estimating dynamic discrete choice models (DDCM) is a common task in many disciplines, including various fields of economics. In a typical DDCM a forward-looking decision-maker chooses from a finite set of actions in each time period by maximizing the expected sum of current and discounted future values of an objective function. The parameters that determine the objective function are called structural parameters. For example, consider the problem of a senior teacher deciding the timing of retirement. The retirement decision is influenced by the current and future salaries, unobserved random factors, and pension rules. The teacher's objective function is the expected utility from the flow of salary or pension benefit. The structural parameters that shape the teacher's utility depict the teacher's preference and are independent of the environment (such as the pension rules.) Because the structural parameters are invariant to changes in environment, estimation of structural parameters is especially useful for simulating new policies that have not been employed in the past.

A DDCM is often presented as a dynamic programming (DP) problem. Solving the DP model involves solving the Bellman equation. Even with a limited set of state variables solving the Bellman equation can be time consuming. Conventional structural estimation requires repeatedly solving the Bellman equation. High computational cost limits applications of structural discrete choice models in policy analysis. The proposed research seeks improving computational efficiency for dynamic binary choice models (DBCM), through deep neural networks (DNN). The thesis consists of three chapters.

Chapter 1 proposes and implements a new method that uses DNN-aided learning to solve DP models during the process of estimation. We compare the new algorithm with several existing algorithms for estimating infinite horizon DBCM. The comparison of algorithm performance is made in the context of estimating three variants of

Rust's (1987) optimal engine replacement model, the benchmark model in the literature of structural estimation. We find that without sacrificing much accuracy, the new approach substantially cuts computational time of the conventional approach (in Rust (1987)), is comparable to the ones developed by Imai et al. (2009), Norets (2009) and Norets (2012), and has the potential to outperform others when the model is more complex than the Rust model. The reduction in computational costs also allows us investigate the shape of likelihood function more intensively, and we find that the benchmark Rust model with serially correlated error may be unidentified.

Chapter 2 focuses on structural estimation of DBCM in finite horizons. We consider teachers' optimal retirement problem. First, we show that the solutions to two similar models, DP and the option value model by Stock and Wise (1990) (SW), can both be presented by thresholds of preference errors. Second, we modify the three-step procedure of Norets (2012) to a simulated sample of teachers. We achieve reduction in computational time of the conventional nested algorithm by around 20-fold for DP and 5-fold for SW, without significant loss of accuracy. Lastly, the accuracy of the DNN aided algorithm is high enough to distinguish DP from SW as data generating model.

Chapter 3 applies the DNN-aided structural estimation to analyze the effect of Illinois teacher pension rules. We first estimate structural parameters using data on Illinois teacher retirement. The structural estimation accounts for the dependence of sample distribution on previous pension policies. Then, as an out-of-sample test, we use the estimated structural parameters to simulate teacher's response to a historical pension enhancement, the "22 upgrade". The estimated structural model produces good in- and out-of sample fit and is useful for policy simulations.

Chapter 1

Bayesian Estimation of Dynamic Discrete Choice Models with Deep Neural Networks

1.1 Introduction

A traditional procedure of structural estimation is as follows. Let the solution (i.e., an optimal policy function) of a structural model be $\mathbf{y} = f(\mathbf{X}, \boldsymbol{\theta}, \boldsymbol{\epsilon})$. Here $\mathbf{y}$ is the choice variable, $\mathbf{X}$ is a vector of (observable) state variable, $\boldsymbol{\theta}$ is a set of structural parameters, and $\boldsymbol{\epsilon}$ error term. In most cases the policy function is not available in analytical form. For a fixed $\boldsymbol{\theta}$, we *solve* for a numerical approximation of the policy function f, $\mathbf{y} = \hat{f}(\mathbf{X}, \boldsymbol{\theta}, \boldsymbol{\epsilon})$. For a given a solution method, the approximation error $||f - \hat{f}||$ is negatively related to the computation time of approximation.[1] For example, if one applies a grid search method, then with finer grids of $\mathbf{X}$ and $\boldsymbol{\epsilon}$, $||f - \hat{f}||$ decreases but computation time increases.

Denote the data of state- and choice variables by $\mathbf{D} = (\mathbf{X}, \mathbf{Y})$. From numerical

[1] Throughout the paper we use $||.||$ as a generic notation for distance between parameters or functions.

solution $\hat{f}$ and for given structural parameter $\boldsymbol{\theta}$ one can simulate data $\mathbf{D}$ and obtain a set of statistics $\hat{M}(\mathbf{D}, \boldsymbol{\theta})$. We *estimate* the structural parameter $\boldsymbol{\theta}$ by matching $\hat{M}(\mathbf{D}, \boldsymbol{\theta})$ with $\hat{M}(\mathbf{D}^*)$ based on observed data $\mathbf{D}^*$ for a given loss function: $\hat{\boldsymbol{\theta}} = argmin_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$ where $L(\boldsymbol{\theta}) = ||\hat{M}(\mathbf{D}, \boldsymbol{\theta}) - \hat{M}(\mathbf{D}^*)||$. In a setting for Bayesian inference we compute likelihood $L(\mathbf{D}^*, \boldsymbol{\theta})$ from numerical simulations of the structural model. Combining the approximated likelihood and a prior of $\boldsymbol{\theta}$ yields numerical draws of posterior. A posterior moment (e.g., posterior mean, posterior median, etc.) serves as the Bayesian estimator for $\boldsymbol{\theta}$. Regardless of how the estimator of $\boldsymbol{\theta}$ is obtained, one key component of the estimation step is the solution of the structural model, $\hat{f}$.

Typically an optimization algorithm (or in a Bayesian setting Markov Chain Monte Carlo algorithm) directs a search over parameter space. In the k-th iteration, the algorithm requires computing $L(\boldsymbol{\theta}^{(k)})$ (or in a Bayesian setting likelihood $L(\mathbf{D}^*, \boldsymbol{\theta}^{(k)})$) and to do so we need to solve for $\hat{f}(\mathbf{X}, \boldsymbol{\theta}^{(k)}, \boldsymbol{\epsilon})$. The estimation step ends with estimate $\hat{\boldsymbol{\theta}}$ when the improvement in the objective diminishes, and $\boldsymbol{\theta}^{(k)}$ converges to $\hat{\boldsymbol{\theta}}$. If the data $\mathbf{D}^*$ are generated by the true model $f(., \boldsymbol{\theta}, .)$ then the error of the estimator, $||\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}||$ stems only from the sampling error. However, in the presence of approximation error $||f - \hat{f}||$, even as data are generated from the true model f part of the estimation error $||\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}||$ is attributed to $||f - \hat{f}||$. If the solution error $||f - \hat{f}||$ is large then the search for estimator may fail to converge.

A technical challenge to structural estimation lies in computational cost. In the estimation step if the dimension of $\boldsymbol{\theta}$ is large, searching over the domain of parameter $\boldsymbol{\theta}$ may require computing $\hat{f}(., \boldsymbol{\theta}, .)$ many times. If the computation time for $\hat{f}$ is large then the estimation time may be prohibitive. This implies that in practice there is a trade off between accuracy in numerical solution and complicity (e.g., dimension of $\boldsymbol{\theta}$) of a feasible model. To ensure computation time is reasonable one may have to limit the number of parameters to be estimated, which makes the structural models overly restrictive for applications. The high computational cost is a major obstacle to proliferation of structural estimations in practice.

Table 1.1: Four types of algorithms

type	description	references
1	solve to estimate	Rust (1987), Rust (2000)
2	solve while estimate	Imai et al. (2009), Norets (2009), Ching et al. (2012)
3	solve, learn, then estimate	Norets (2012), Farrell et al. (2021), Chen et al. (2021)
4	learn to solve while estimate	this chapter

Estimating the structural parameters involve repetitively solving for $\hat{f}(\mathbf{X}, \boldsymbol{\theta}^{(k)}, \boldsymbol{\epsilon})$ in the kth iteration. For a dynamic structural model there are competing methods in solving for $\hat{f}$. Conventional solution methods (see, e.g., Rust (1987), Rust (2000) among others) solve for $\hat{f}(., \boldsymbol{\theta}^{(k)}, .)$ without benefiting from the information on $\hat{f}(., \boldsymbol{\theta}^{(j)}, .)$ (for $j < k$), the solutions obtained from the previous iterations. Given the values of $L(\boldsymbol{\theta}^{(j)})$ (for $j < k$), the minimization algorithm selects $\boldsymbol{\theta}^{(k)}$. But obtaining $\hat{f}(., \boldsymbol{\theta}^{(k)}, .)$ is just as costly as obtaining $\hat{f}(., \boldsymbol{\theta}^{(j)}, .)$. We call the type of these algorithms "solve to estimate."

An alternative type of algorithms (Imai et al. (2009), Norets (2009), Ching et al. (2012)) make use of the information on the solutions in the previous iterations. To avoid the cost in computing $\hat{f}(\mathbf{X}, \boldsymbol{\theta}^{(k)}, \boldsymbol{\epsilon})$ for all $\boldsymbol{\theta}^{(k)}$, in each iteration of parameters they update an approximation of $\hat{f}(\mathbf{X}, \boldsymbol{\theta}^{(k)}, \boldsymbol{\epsilon})$, $\hat{f^{(k)}}(., \boldsymbol{\theta}^{(k)}, .)$, only once, by averaging over previous values, $\hat{f^{(j)}}(., \boldsymbol{\theta}^{(j)}, .)$ (for $j < k$). Under some conditions, $\hat{f^{(k)}}(., \boldsymbol{\theta}^{(k)}, .)$ converges to $\hat{f}(., \boldsymbol{\theta}^{(k)}, .)$. We call this type of algorithms that iterate over both approximated solutions and parameters "solve while estimate."

The iterative processes in solving and estimating the structural models both generate large data. This suggests potential benefit from recent development in machine learning, especially deep learning. Machine learning can aid estimation in the following three-step procedure (Norets (2012), Farrell et al. (2021), Chen et al. (2021)). In the first step, the solution step, we obtain numerical $\hat{f}(., \boldsymbol{\theta}, .)$ for N combinations of state variable and parameters $(\mathbf{X}, \boldsymbol{\theta})$ and save the solution as a library. In the second step, the learning step, we fit an approximation structure with the built library (i.e., "learn" the solution). In this paper, we choose deep neural network (DNN) as the structure for its flexibility, and the process is also termed as "training a DNN". In the third step, the estimation step, we search for parameters $\boldsymbol{\theta}$ that minimizes the distance between simulated statistics and sample statistics. Rather than solving for $\hat{f}(\mathbf{X}, \boldsymbol{\theta}^{\#}, .)$ every time $\boldsymbol{\theta}$ is evaluated at a new value $\boldsymbol{\theta}^{\#}$, we approximate $\hat{f}(\mathbf{X}, \boldsymbol{\theta}^{\#}, .)$ by DNN. If the optimal policy can be accurately learned from a moderately sized library the total computational cost for estimation may be lower than the traditional approach for a given level of accuracy. We call this type of algorithms "solve, learn, then estimate."

The second algorithm ("solve while estimate") involves averaging over previous values and updating once, which significantly reduces computational burden, but might still be costly, if the state space is very large.[2] For the third algorithm ("solve, learn, then estimate"), once the library is built and DNN is trained, its performance is also fixed. And it is hard to tell how large the library should be and how accurate the DNN should be in advance, especially when library building is also time-consuming.

We propose a new algorithm, which combines the insights in algorithm 2 and algorithm 3. We set up a DNN for the solution, and maintain a dynamic library of inaccurate but continuously-improving solution library, along the estimation process. In each iteration, we randomize between two actions: 1) update and 2) not update. 1) If update, we first generate a guess of solution from the current DNN, update it only once, and save the new solution to the library, while deleting the oldest one. Then, we re-train the DNN with this dynamic library, and use the re-trained DNN as surrogate for the solution. 2) If not update, we simply use the current DNN as the surrogate. The probability of update is set to be negatively related with the

[2] Imai et al. (2009) provide a modification of their algorithm which only updates the solution for one vector of state variables in each iteration; but then it would take millions of iterations to obtain convergence in practice.

accuracy of the DNN in previous iterations. We call this type of algorithm "learn to solve while estimate" and examines whether it outperforms other algorithms when the state space is large.

The focus of this chapter is to summarize these types of algorithms for estimation of structural models and compare their performance in the context of estimation of a benchmark model–Rust's model of optimal engine replacement. There are competing methods for estimating the model (e.g., maximum likelihood, method of simulated moments, Bayesian estimation.) Here we present all algorithms under the framework of Bayesian estimation.

1.2 The Model and Bayesian Estimation

1.2.1 A General Dynamic Discrete Choice Model

Consider an infinite horizon Markov decision process with state variables (including those observable and unobservable to the researchers, x and ϵ, $s = (x, \epsilon) \in \mathcal{X} \times \mathcal{E}$,) control variables (action) in a finite set $y \in \mathcal{Y}(s)$, a instantaneous utility function (or the inverse cost function) $u(s, y)$, discount rate $\beta \in (0, 1)$, and transition law $\pi(s'|s, y)$. The economic agent wishes to maximize the expected discounted sum of utility:

$$\max_{y_0, y_1, \cdots} \mathbb{E} \sum_{t=0}^{\infty} \beta^t u(s_t, y_t)$$

subject to $y_t \in \mathcal{Y}(s_t)$, the feasible choice given state variable at s_t, and $s_{t+1} \sim \pi(s_{t+1}|s_t, y_t)$. Under regularity conditions (see, e.g., Rust (1996)), we can write the problem in a recursive form

$$V(s) = \max_{y \in \mathcal{Y}(s)} u(s, y) + \beta \mathbb{E}_{s' \sim \pi(s'|s,y)} V(s').$$

If we define the right hand side as a Bellman operator T that maps a function to another function, the above equation can be written compactly as

$$V = T(V).$$

Sometimes we also write the expected value function $\mathbb{E}_{s' \sim \pi(s'|s,y)} V(s') = \mathbb{E}V(s'|s,y)$ to highlight its dependence on s and y. In the bus engine replacement problem $s = (x, \epsilon)$ is the states, and $y \in \{0, 1\}$ is the action. Solving the value function $V(s)$ gives rise to the optimal policy $y(s)$. For solution methods of dynamic programming problems we only cover a few common value-function-based methods and omit others. [3]

Example 1 Rust model with normal AR(1) errors

The engine replacement model we solve and estimate is a modified version of Rust (1987). In this problem a researcher estimates the parameters of an optimal decision by a bus fleet manager from time series observations of the manager's decisions. At the beginning of each period, the bus fleet manager decides whether to replace the engine for each bus. Consider a bus with mileage x which also is observable to the researcher, and an idiosyncratic shock to maintenance ϵ which is only observable to the manager. If the manager replaces the engine, there is a replacement cost rc and the mileage is reset to zero next period. Otherwise, the running bus incurs cost θx, and the mileage will become $x' > x$. The discount rate, $\beta \in (0, 1)$ is fixed and known to the researcher. Written in recursive form, we have:

$$V(x, \epsilon) = \min_{y \in \{0,1\}} \{\theta x + \epsilon + \beta EV(x', \epsilon'|x, \epsilon, y = 0), rc + \beta EV(x', \epsilon'|x, \epsilon, y = 1)\}.$$

[3] For example, we do not consider conditional choice probability (CCP) based estimators (Hotz and Miller (1993), Hotz et al. (1994), Bajari et al. (2007)), Arcidiacono and Miller (2011) and mathematical programming with equilibrium constraints (MPEC, Su and Judd (2012)) in this paper.

For this binary choice problem, the first term on the right-hand-side is the value of keeping the engine, and the second term is the value of replacing the engine.

In the original paper, Rust (1987) assumes the maintenance cost shock ϵ follows an i.i.d. extreme valued distribution. This greatly reduces computational cost. Following subsequent studies (e.g., Norets (2012), Reich (2018)) we assume ϵ to be normal and AR(1). And to keep things simple, we assume the following transitional dynamics:

$$x', \epsilon' | (x, \epsilon, y) \sim x + 1, N(\rho\epsilon, 1), \ if \ y = 0;$$

$$\sim 0, N(0, \frac{1}{1-\rho^2}), \ if \ y = 1.$$

This problem is a stationary infinite horizon DP. If we define the right hand side of the minimization problem as the Bellman operator T, the problem can be solved by value function iteration: given the current guess of $V(x, \epsilon)$, we obtain its next value by apply the Bellman operator until convergence:

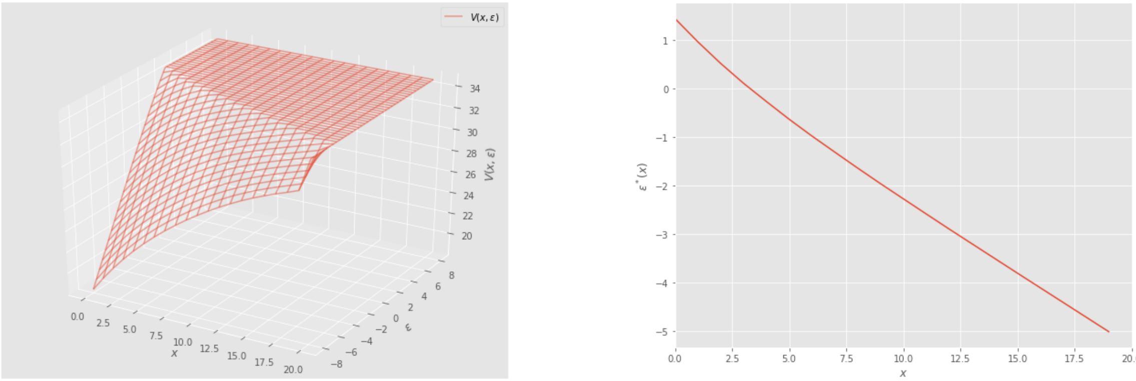
$$V \leftarrow TV$$

Also note that when $\rho \geq 0$, this problem also has a threshold strategy, that is, when the mileage is x, it is optimal to replace the engine if the shock to maintenance cost is too high:

$$\epsilon > \epsilon^*(x)$$

where $\epsilon^*(x)$ is the threshold that depends on x. We plot the value function and the threshold for one set of parameters in Figure 1.1. In the left panel, the value function for an given value of x increases as the error term ϵ increases, and then becomes constant (because upon replacing the engine the value is not dependent on the mileage and current maintenance cost.) In the right panel, the threshold itself is a decreasing function of mileage, due to linearity of the objective function in mileage and maintenance cost. The right panel shows that the threshold is almost linear in x, which contributes to difficulty in identifying the model, a topic we will pick up later in this chapter.

Figure 1.1: Value function and threshold for Rust model with AR(1) errors



Note: parameters are $(\theta, rc, \rho) = (0.3, 3.0, 0.5)$. Value function $V(x, \epsilon)$ is in the left panel, and threshold $\epsilon^*(x)$ is in the right panel.

Example 2 Rust model with IID errors

We also consider a special case where $\rho = 0$, and the model is simplified as:

$$\begin{aligned} V(x,\epsilon) &= \min_{y\in\{0,1\}} \{\theta x + \epsilon + \beta \mathbb{E}V(x+1,\epsilon'), rc + \beta\mathbb{E}V(0,\epsilon')\} \\ &= \min_{y\in\{0,1\}} \{\theta x + \epsilon + \beta \mathbb{E}V(x+1), rc + \beta\mathbb{E}V(0).\} \end{aligned}$$

Note that the expected value function no longer depends on the current error term when the errors are IID. We can write the above equation in terms of expected value functions:

$$\begin{aligned} \mathbb{E}V(x) &= \int V(x,\epsilon) dG(\epsilon) \\ &= \int \min_{a\in\{0,1\}} \{\theta x + \epsilon + \beta\mathbb{E}V(x+1), rc + \beta\mathbb{E}V(0)\} dG(\epsilon). \end{aligned}$$

And we still have to find the fixed point of the functional equation, with one fewer dimension:

$$\mathbb{E}V = T(\mathbb{E}V)$$

The threshold strategy trivially holds here. The manager would replace the engine if:

$$y = 1 \iff rc + \beta\mathbb{E}V(0) < \theta x + \epsilon + \beta\mathbb{E}V(x+1)$$

$$\iff \epsilon^*(x) \equiv rc + \beta\mathbb{E}V(0) - (\theta x + \beta\mathbb{E}V(x+1)) < \epsilon.$$

Example 3 Rust model with random effect (RE)

Our last example adds upon the IID model in example 2 with a term for time invariant heterogeneity (random effect $\alpha \sim N(\mu, \sigma^2)$, where we normalize $\mu = 0$ for identification) in the maintenance cost:

$$\begin{aligned} V(x, \alpha, \epsilon) &= \min_{y \in \{0,1\}} \{\theta x + \alpha + \epsilon + \beta\mathbb{E}V(x+1, \alpha, \epsilon'), rc + \beta\mathbb{E}V(0, \alpha, \epsilon')\} \\ &= \min_{y \in \{0,1\}} \{\theta x + \alpha + \epsilon + \beta\mathbb{E}V(x+1, \alpha), rc + \beta\mathbb{E}V(0, \alpha)\} \end{aligned}$$

The threshold strategy is that replacing the engine if:

$$y = 1 \iff rc + \beta\mathbb{E}V(0, \alpha) < \theta x + \alpha + \epsilon + \beta\mathbb{E}V(x+1, \alpha)$$

$$\iff \epsilon^*(x, \alpha) \equiv rc + \beta\mathbb{E}V(0, \alpha) - (\theta x + \beta\mathbb{E}V(x+1, \alpha)) < \epsilon.$$

Note that the expected value function depends on the mileage x as well as the time invariant heterogeneity α.

1.2.2 Bayesian Estimation

In the general model, assume that the utility function and the transition law are parametrized by a vector $\boldsymbol{\theta}$. We have a random sample of I individuals over T periods, with observable states and actions

$$Data = (\mathbf{X}, \mathbf{Y}) = \{x_{it}, y_{it}\}_{i=1,t=1}^{I,T}.$$

Given the prior distribution $p(\boldsymbol{\theta})$, the task is to derive the posterior distribution of the structural parameters $\boldsymbol{\theta}$ from these observations:

$$p(\boldsymbol{\theta}|\mathbf{X}, \mathbf{Y}) \propto p(\boldsymbol{\theta})L(\mathbf{X}, \mathbf{Y}|\boldsymbol{\theta}).$$

We compute the posterior numerically with Markov Chain Monte Carlo. More specifically, we run the Metropolis–Hastings (MH) loop where in iteration k, given the previous parameter values $\boldsymbol{\theta}^{k-1}$, the candidate draw $\boldsymbol{\theta}^*$ is obtained from a proposal distribution $q(\boldsymbol{\theta}^{k-1}, \boldsymbol{\theta}^*)$. We accept $\boldsymbol{\theta}^*$ as $\boldsymbol{\theta}^k$ with probability:

$$\min\{1, \frac{p(\boldsymbol{\theta}^*)L(\mathbf{X}, \mathbf{Y}|\boldsymbol{\theta}^*)q(\boldsymbol{\theta}^*, \boldsymbol{\theta}^{k-1})}{p(\boldsymbol{\theta}^{k-1})L(\mathbf{X}, \mathbf{Y}|\boldsymbol{\theta}^{k-1})q(\boldsymbol{\theta}^{k-1}, \boldsymbol{\theta}^*)}\}$$

and reject $\boldsymbol{\theta}^*$ and set $\boldsymbol{\theta}^k = \boldsymbol{\theta}^{k-1}$ otherwise.

Example 1-3 (ctd)

For example 1-3, the data we have are the mileage and replacement decisions for each bus over the entire period. We assume all buses starts with zero mileage and treat a bus with replaced engine as a new bus. The task in example 1 is to estimate the value of deep parameters, θ, rc, ρ, from the data. The parameters are θ, rc in example 2, and θ, rc, σ in example 3.

The unconditional probability of observing bus i replaced in period τ is

$$p_\tau = \Pr(observing\ replace\ i\ in\ \tau) = \Pr(not\ replace\ i\ in\ 0, 1, ..., \tau - 1,\ replace\ in\ \tau).$$

This is a high dimensional integral in example 1 due to serial correlation in ϵ. (For efficient computation of this integral, see Appendix A.1) It can be factorized into the product of probabilities for the IID case in example 2. With time invariant heterogeneity in the manner of example 3, the product of probabilities needs to be integrated with respect to the distribution of α.

1.3 Algorithms

1.3.1 Solve to Estimate

This class of algorithm depicted in Figure 1.2 follows the nested fixed point theorem in Rust (1987).[4] There are two loops, an inner loop and an outer loop. The inner loop solves for the value function (or expected value function) by iterating the Bellman operator, for the current draw of parameters. The outer loop then use the solved model to calculate the likelihood for these parameters, and run the MH updating.

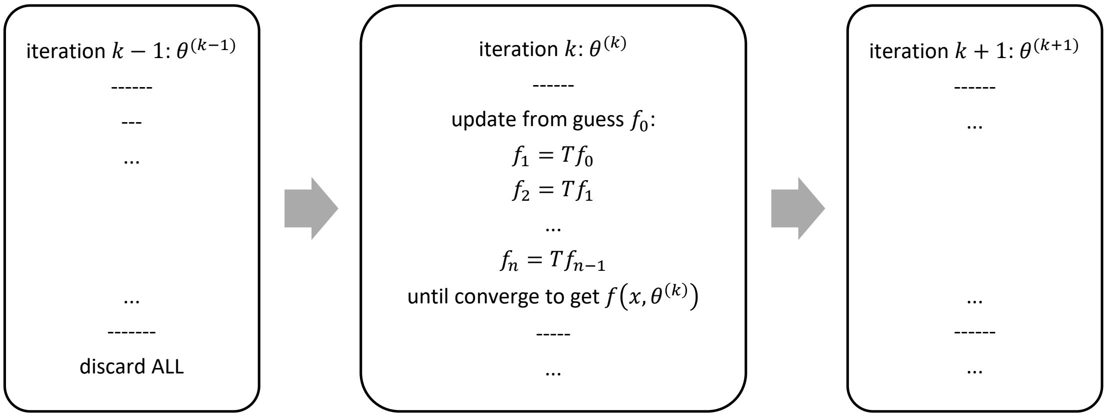
This procedure has two features: First, once the MH updating is done, the solved model is discarded. Second, in each iteration, the Bellman operator has to be applied many times until $||f_n - f_{n-1}||$ is within a pre-set tolerance level.

Example 2 (ctd)

The inner loop (with expectation computed via Monte Carlo integration) is:

[4] The original Rust paper makes several assumptions on the distribution of the error term (i.i.d. GEV) such that the expected value function can be written in analytic form. Here we present a more general formulation of the algorithm.

Figure 1.2: Solve to Estimate


iteration $k-1$: $\theta^{(k-1)}$

...
...
----
discard ALL
iteration k: $\theta^{(k)}$
------
update from guess f_0:
$f_1 = Tf_0$
$f_2 = Tf_1$
...
$f_n = Tf_{n-1}$
until converge to get $f(x, \theta^{(k)})$

...
iteration $k+1$: $\theta^{(k+1)}$

...
...

...

Note: T is the Bellman operator for value function iteration. f is an unknown function of state variable x and parameter θ. In iteration k, the parameter value is $\theta^{(k)}$. More general iterative solution methods such as Newton-Kontorovich iteration (Rust (2000)) is also applicable in this case.

i) given the value of $\{\theta, rc\}$, create grids for mileage $\{x_n\} = \{0, 1, ..., n\}$ and initialize the expected value to arbitrary values, say, $\{\mathbb{E}V(x_n)\} = \{0, 0, ..., 0\}$

ii) draw M values of ϵ_m from $N(0,1)$, and calculate

$$V(x_n, \epsilon_m) = \min_{y \in \{0,1\}} \{\theta x_n + \epsilon_m + \beta \mathbb{E}V(x_n + 1), rc + \beta \mathbb{E}V(0)\}$$

iii) update $\mathbb{E}V(x_n)$ for each x_n:

$$\mathbb{E}V(x_n) \longleftarrow \frac{1}{M} \sum_{m=1}^{M} V(x_n, \epsilon_m)$$

iv) repeat ii) and iii) until convergence.

The outer loop is:

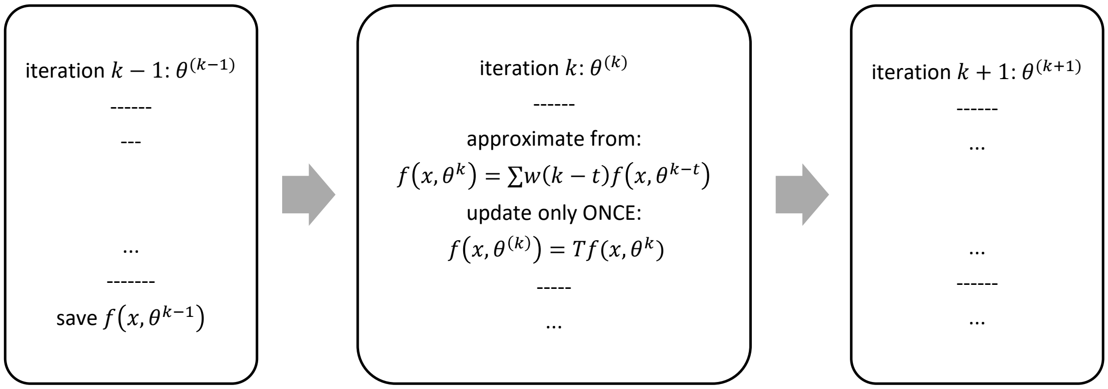
i) use the expected value function to calculate the threshold $\epsilon^*(x)$ for each x,

ii) calculate the likelihood function based on $\epsilon^*(x)$,

iii) run MH updating.

1.3.2 Solve while Estimate

Figure 1.3: Solve while Estimate


iteration $k-1$: $\theta^{(k-1)}$

...
----
save $f(x, \theta^{k-1})$
iteration k: $\theta^{(k)}$
------
approximate from:
$f(x, \theta^k) = \sum w(k-t) f(x, \theta^{k-t})$
update only ONCE:
$f(x, \theta^{(k)}) = Tf(x, \theta^k)$

...
iteration $k+1$: $\theta^{(k+1)}$

...
...

...

Note: T is the Bellman operator for value function iteration. f is the function of interest, which takes the state variable x and parameter θ as inputs. In iteration k, the (candidate) parameter value is $\theta^{(k)}$ (we drop the superscript * for convenience). $w(k-t)$ is the current weight for the saved function at iteration $k-t$, depending on the weighting kernel and the parameter values θ^{k-t}, θ^t.

The idea of algorithm 2 is to approximate the solution on the parameter space by some weighting kernel $\mathcal{K}$, according to Imai et al. (2009).[5] The most significant difference between algorithms 1 and 2 is the way we calculate the expected value function. In algorithm 1, everything is discarded at the end of each iteration and we repeatedly calculate the expectation from scratch. In algorithm 2, we store the value functions obtained in each iteration. Then, we calculate the expected value function as a weighted average of those saved value functions in previous iterations. In Figure 1.3, $f(x, \boldsymbol{\theta}^{k-t})$ denotes the value function obtained in iteration $k-t$ under parameter $\boldsymbol{\theta}^{k-t}$, and $w(k-t) = \dfrac{\mathcal{K}(\boldsymbol{\theta}^{k-t}, \boldsymbol{\theta}^t)}{\sum_{t=1}^{N(k)} \mathcal{K}(\boldsymbol{\theta}^{k-t}, \boldsymbol{\theta}^t)}$ the weight generated from some kernel measuring the distance between current parameter draw $\boldsymbol{\theta}^k$ and most recent draws $\boldsymbol{\theta}^{k-t}$ for $t = 1, 2, ..., N(k)$. Another difference is that, in algorithm 1, we apply the Bellman operator T until convergence to obtain the accurate solution. However,

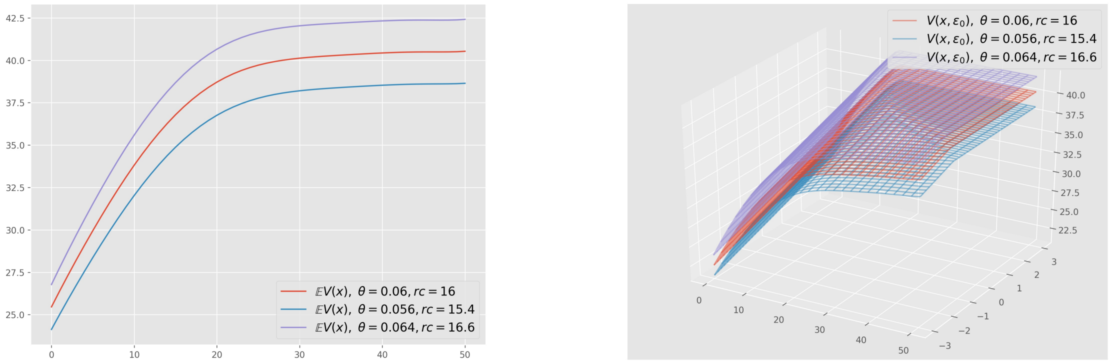
[5] Norets (2009) provide an alternative approach based on k-Nearest Neighbors (KNN).

in algorithm 2, we only apply it once in each loop. Under the conditions in Imai et al. (2009), the solution becomes more accurate as the chain runs and converges to the posterior distribution.

Example 2 (ctd)

The intuition of algorithm 2 can be visualized in Figure 1.4. We plot expected value function and value functions for the IID case with three sets of structural parameters: $\boldsymbol{\theta} = (\theta, rc) = (0.06, 16), (0.056, 15.4), (0.064, 16.6)$. Suppose that we already obtained the results for (0.06, 16) and (0.064, 16.6), and wish to solve for (0.056, 15.4). An good approximate would be some average of the previous two.

Figure 1.4: Expected value function and value function for Rust model with IID errors



Note: three sets of parameters are $\boldsymbol{\theta} = (\theta, rc) = (0.06, 16), (0.056, 15.4), (0.064, 16.6)$, from top (purple) to middle (red) and then bottom (blue). the left panel is for the expected value functions, and the right panel for the value functions.

The outer loop of the algorithm is unchanged, while the inner loop at iteration k is:

i) choose some weighting kernel $\mathcal{K}$ and length of recent memory $N(k)$

ii) update the expected value by

$$\hat{\mathbb{E}}V(x;\boldsymbol{\theta}^k) \longleftarrow \sum_{t=1}^{N(k)} \sum_{m=1}^{M} \hat{V}(x, \epsilon_m^{k-t}; \boldsymbol{\theta}^{k-t}) \frac{\mathcal{K}(\boldsymbol{\theta}^k, \boldsymbol{\theta}^{k-t})}{\sum_{t=1}^{N(k)} \mathcal{K}(\boldsymbol{\theta}^k, \boldsymbol{\theta}^{k-t})},$$

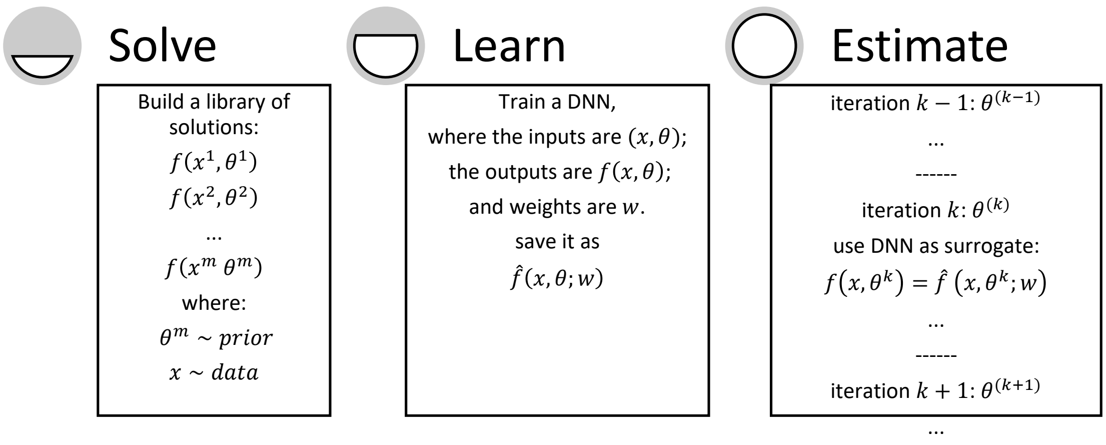
iii) draw M values of ϵ_m from $N(0,1)$ and calculate

$$\hat{V}(x, \epsilon_m^k; \boldsymbol{\theta}^k) \longleftarrow \min_{y \in \{0,1\}} \{\theta^* x + \epsilon_m + \beta \hat{\mathbb{E}} V(x+1; \boldsymbol{\theta}^k), rc^* + \beta \hat{\mathbb{E}} V(0; \boldsymbol{\theta}^k)\}.$$

Note that we no longer run the inner loop until convergence; instead, convergence is obtained along the process of parameter draws.

1.3.3 Solve, Learn, then Estimate

Figure 1.5: Solve, Learn, then Estimate



Solve

Build a library of solutions:
$f(x^1, \theta^1)$
$f(x^2, \theta^2)$
...
$f(x^m \theta^m)$
where:
$\theta^m \sim prior$
$x \sim data$

Learn

Train a DNN,
where the inputs are (x, θ);
the outputs are $f(x, \theta)$;
and weights are w.
save it as
$\hat{f}(x, \theta; w)$

Estimate

iteration $k-1$: $\theta^{(k-1)}$
...
------
iteration k: $\theta^{(k)}$
use DNN as surrogate:
$f(x, \theta^k) = \hat{f}\,(x, \theta^k; w)$
...
------
iteration $k+1$: $\theta^{(k+1)}$
...

Note: $f(x^m, \theta^m)$ is the m-th entry of the library of solutions, w stands for the parameter for the DNN here.

Instead of running nested loops in algorithm 1 and 2, we now follow a three-part process shown in Figure 1.5. The first step is building a library of solutions for different state variables $\mathbf{x}^m$ and parameters $\boldsymbol{\theta}^m$. The state variables are re-sampled from the data and the parameters are drawn from the prior distribution. Although we draw the state variables from the sample, we do not use the observed choice of action in the data but rather solve for the optimal policy under a random set of parameters. This is different from the method by Semenova (2018), who uses the observed policy to back out the optimal policy under the true parameters. Such use of data information is not cost-free; it may create bias in estimation. To correct the potential bias Chernozhukov et al. (2018) use a double machine learning technique. Moreover, since each draw of parameters are IID from the prior distribution, this process can be accelerated by parallel computation if we have access to multi-core computers.

The second step is learning the true f from the library of solutions. Multiple techniques are applicable for this purpose. Here we choose Deep Neural Networks (DNN) for its superior performance recently discovered by LeCun et al. (2015). And we can train DNN with modern optimization techniques to improve its accuracy.

The third step is estimating the structural parameters. Instead of solving the model, we use DPP as surrogate for the solution. That is, in each MCMC run we evaluate likelihood for a given set of parameters using approximated likelihood learned from DNN.

A brief introduction to DNN

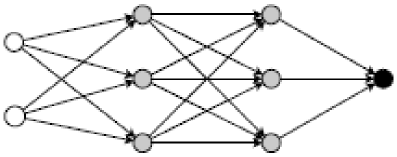
The neural networks for policy function consists $(\mathbf{z}, \mathbf{y})$ where $\mathbf{z} = (z_1, z_2, ..., z_m)'$ is an m-dimensional input, and $\mathbf{y} = (y_1, y_2, ..., y_l)'$ is a l-dimensional output vector of interest (it can be the expected value function, value function or threshold of the preference error that depicts the optimal policy). A feed-forward neural network with two hidden layers is depicted in Figure 1.6 and can be written as (see, e.g., Cho and

Sargent (1996)):

$$
\begin{aligned}
z_j^1 &= \sigma(b_j^1 + \sum_{i=1}^{m_0} w_{ji}^1 z_i^0) \\
z_j^2 &= \sigma(b_j^2 + \sum_{i=1}^{m_1} w_{ji}^2 z_i^1) \\
\hat{y}_j &= b_j^3 + \sum_{i=1}^{m_2} w_{ji}^2 z_i^2
\end{aligned}
$$

where we define $z_j^0 = z_j$ the initial input vector, $m_0 = m$ its dimension, and $\hat{y}_j$ the j-th entry of the DNN approximate. The $j \in \{1, ..., m_1\}$-th entry of the first hidden layer is z_j^1 and $j \in \{1, ..., m_2\}$-th entry of the second hidden layer is z_j^2. Moreover, $\sigma(\cdot)$ is the "activation function"[6], w_{ji}^k are weights of entry i of layer $k-1$ in determining entry j of layer k before activation. Neural networks with more layers simply take the output from previous process as the input of current process, and repeat the above weighting and activation procedure until reaching the final output.

Figure 1.6: A neural network with two hidden layers



Note: reproduced from Farrell et al. (2021), figure 2. Here, the input is two-dimensional, each hidden layer has three neurons, and the output is scalar, hence $m = m_0 = 2, m_1 = m_2 = 3, l = 1$.

Training neural networks takes several steps. First, we pick the number of hidden layers as well as number of perceptions in each layers (i.e., m_1, m_2 in our setting).

[6] Popular choices are Rectified Linear Unit (ReLu): $\sigma(x) = \max\{0, x\}$ and Sigmoid: $\sigma(x) = 1/(1 + \exp(-x))$.

Then we find the optimal weights that minimizes a loss function:

$$\begin{aligned}\|y-\hat{y}\| &= \|y-(b_j^3+\sum_{i=1}^{m_2} w_{ji}^2 z_i^2)\| \\ &= \|y-(b_j^3+\sum_{i=1}^{m_2} w_{ji}^2 \sigma(b_j^2+\sum_{i=1}^{m_1} w_{ji}^2 z_i^1))\| \\ &= \|y-(b_j^3+\sum_{i=1}^{m_2} w_{ji}^2 \sigma(b_j^2+\sum_{i=1}^{m_1} w_{ji}^2 \sigma(b_j^1+\sum_{i=1}^{m_0} w_{ji}^1 z_i^0)))\|\end{aligned}$$

The asymptotic performance of neural networks is guaranteed by the universal approximation theorem, which claims that under minor restrictions an objective function can be approximated arbitrarily well as long as m_1 and m_2 are large enough. Moreover, it only takes linearly many parameters to achieve this, while polynomial, spline, and trigonometric expansions take exponential number of parameters (according to Barron (1993).) Recently, Farrell et al. (2021), among others, also proves certain finite sample properties of DNN.

Example 1 (ctd)

Step 1 Solve

We build a library by solving the model for 3,000 sets of parameters from the prior distribution. We assume uniform priors: $\theta \sim Unif[0.1, 0.4]$, $rc \sim Unif[1.0, 4.0]$, $\rho \sim Unif[0.0, 0.9]$. Since this problem exhibits threshold strategy, we store the thresholds $\epsilon^*(x; \theta, rc, \rho)$ corresponding to mileage x and parameters θ, rc, ρ in the library, the structure of which is exemplified in Table 1.2.

Step 2 Learn

The DNN takes the one-dimensional state variable x and the three-dimensional structural parameters θ, rc, ρ as input, and the threshold $\epsilon^*(x; \theta, rc, \rho)$ as the output, as summarized in Table 1.3. Since output is continuously values, it is a standard regression problem. We would like the predicted thresholds be as close to the true

Table 1.2: Structure of the solution library

#	x	θ	rc	ρ	$\epsilon^*(x;\theta,rc,\rho)$
1	1	0.7	14.3	0.754	2.037
1	2	0.7	14.3	0.754	1.965
...	...	...	...	...	...
3000	19	0.6	12.6	0.279	-0.973
3000	20	0.6	12.6	0.279	-2.076

Note: a randomly build solution library for 3,000 sets of parameters and $x \in \{0, 1, ..., 20\}$, hence a total of 63,000 rows.

Table 1.3: Input and output of DNN

Variable Type	Variable Name	Notation
Input	state variables	x
Input	structural parameters	θ, rc, ρ
Output	critical values	$\epsilon^*(x;\theta,rc,\rho)$

Note: for Rust model with AR(1) errors.

ones as possible, measure by some loss functions such as mean squared error or mean absolute error.

We trained the DNN for 1,000 epochs, and the metric for the last epoch are reported in Table 1.4. We partition the sample into two sets: the training set and test set. Mean squared errors and mean absolute errors are both relatively small, suggesting a good fit.

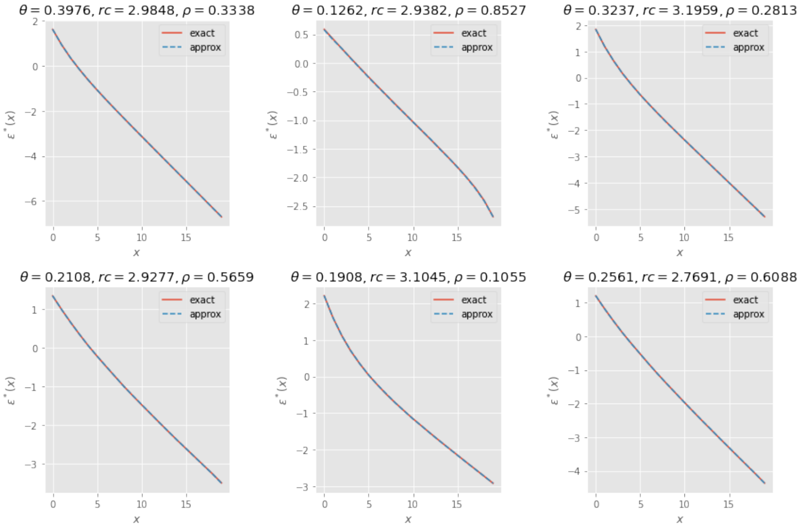
Table 1.4: Performance of DNN

	sample size	mean squared error	mean absolute error
train set	47250	8.7717×10^{-5}	0.0080
test set	15750	1.5017×10^{-4}	0.0109

Note: we randomly assign 75% (=47250/63000) rows of the library as the training set for DNN training, and the remaining 25% as the test set.

We also plot the true and predicted thresholds for six randomly chosen sets of parameters in Figure 1.7. The true and approximated solutions are visually indistinguishable.

Figure 1.7: Illustration of the fit of DNN



Note: exact thresholds (obtained by value function iteration) and approximate (obtained by DNN) for six randomly chosen sets of parameters.

Step 3 Estimate

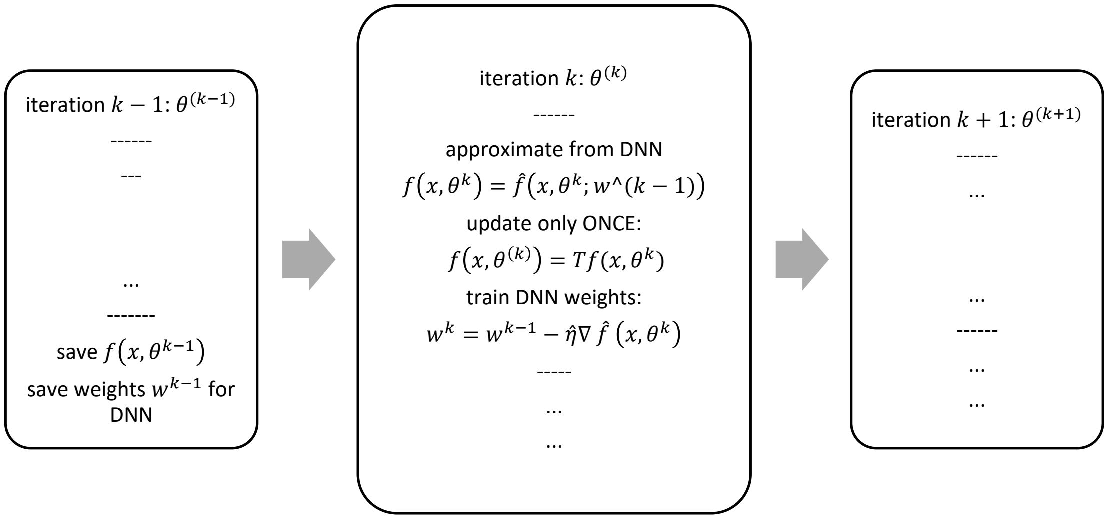
Given the satisfactory fit of the DNN, we use it as the surrogate for the true model for likelihood calculation and update by the MH algorithm.

1.3.4 Learn to Solve while Estimate

Algorithm 2 (“solve while estimate”) involves averaging over previous values and updating once, which significantly reduces computational burden, but might still be costly, if the state space is very large. For algorithm 3 (“solve, learn, then estimate”), once the library is built and DNN is trained, its performance is also fixed. In practice it is hard to decide how large the library should be and how accurate the DNN should be in advance, especially when library building is also time-consuming.

We propose a new algorithm (algorithm 4, “learn to solve while estimate”), combines the ideas in algorithm 2 and algorithm 3 (see Figure 1.8). We set up a DNN for the solution, and maintain a dynamic library of inaccurate but continuously-improving solution library, along the estimation process. In each iteration, we ran-

Figure 1.8: Learn to Solve while Estimate


iteration $k-1$: $\theta^{(k-1)}$

...
----
save $f(x,\theta^{k-1})$
save weights w^{k-1} for DNN
iteration k: $\theta^{(k)}$
------
approximate from DNN
$f(x,\theta^k) = \hat{f}(x,\theta^k; w^\wedge(k-1))$
update only ONCE:
$f(x,\theta^{(k)}) = Tf(x,\theta^k)$
train DNN weights:
$w^k = w^{k-1} - \hat{\eta}\nabla \hat{f}\,(x,\theta^k)$

...
...
iteration $k+1$: $\theta^{(k+1)}$

...
...

...
...

Note: w stands for the parameter for the DNN here. We only show the updating process in the figure.

domize between two actions: 1) update and 2) not update. 1) If update, we first generate current guess from the current DNN, update it only once, and save the new solution to the library, while deleting the oldest one. Then, we re-train the DNN with this dynamic library, and use the re-trained DNN as surrogate for the solution. 2) If not update, we simply use the current DNN as the surrogate. The probability of update is set to be negatively related with the accuracy of the DNN in previous iterations. One measure of the accuracy is the difference in the (log-) likelihood using the current and re-trained DNN.

Unlike algorithm 2, our new algorithm no longer needs to calculate the weighted average of previous solutions and also avoids the updating, when the DNN is accuracy enough. Unlike algorithm 3, our solution library is dynamic rather than static: it is inaccurate at the beginning since we only update once, but its accuracy improves as we run more iterations. By doing so, we avoid the high cost of building an accurate library beforehand, when state space is large. Moreover, in our new algorithm, the number of epochs for DNN training is also endogenously determined, rather than pre-determined as in algorithm 3.

Example 2 (ctd)

We maintain a dynamic library of solutions and a DNN. Since we need to update them along the MCMC runs, approximating the expected value function instead of the threshold is more convenient. So the output of DNN is $\mathbb{E}V(x;\theta,rc)$, and the inputs are x,θ,rc. In each iteration, we randomize between the following two actions:

1) update

i) generate current guess for $\mathbb{E}V(x;\theta,rc)$ from the current DNN with parameters w^{k-1}, apply Bellman operator only once:

$$\tilde{\mathbb{E}}V(x;\theta^k,rc^k) \leftarrow \hat{\mathbb{E}}V(x,\theta^k,rc^k;w^{k-1})$$

ii) save the current solution to the solution library, delete the oldest one

iii) re-train the DNN with the library, where the parameters are now w^k.

iv) use the re-trained DNN as the surrogate for the solution

$$\hat{\mathbb{E}}V(x;\theta^k,rc^k) \leftarrow \hat{\mathbb{E}}V(x,\theta^k,rc^k;w^k)$$

2) not update

i) use the current DNN as the surrogate directly:

$$\hat{\mathbb{E}}V(x;\theta^k,rc^k) \leftarrow \hat{\mathbb{E}}V(x,\theta^k,rc^k;w^{k-1})$$

The probability of updating (action 1) in next period is negatively related to the difference in the likelihood using two DNNs, one with w^{k-1} (hence $\tilde{\mathbb{E}}V(x;\theta^k,rc^k)$ as the surrogate) and another with w^k ($\hat{\mathbb{E}}V(x;\theta^k,rc^k)$ as the surrogate) for the most recent iteration when we update. When the difference is negligible, we would switch to not updating (action 2) with higher probability. In practice, we would like to build a small library and pre-train the DNN as the “warm-up” process, before we run the full-scale MCMC.

1.4 Results for Rust’s Engine Replacement Examples

1.4.1 Model with IID Errors

Table 1.5: Performance comparison, IID case

		# grid = 20			# grid = 100			# grid = 200		
	true	5%	median	95%	5%	median	95%	5%	median	95%
algo 1 solve to estimate										
θ	0.3000	0.2726	0.2995	0.3216	0.2677	0.2935	0.3159	0.2715	0.2917	0.3134
rc	3.0000	2.9562	2.9905	3.0331	2.9690	3.0039	3.0444	2.9659	3.0042	3.0431
time		87s			148s			268s		
algo 2 solve while estimate										
θ	0.3000	0.3214	0.3347	0.3482	0.2901	0.3166	0.3351	0.2954	0.3165	0.3399
rc	3.0000	2.9686	3.0081	3.0477	2.9941	3.0493	3.1317	2.9241	2.9824	3.0473
time		17s			17s			18s		
algo 3 solve, learn, then estimate										
θ	0.3000	0.2641	0.2805	0.2957	0.2747	0.2999	0.3193	0.2621	0.2865	0.3103
rc	3.0000	2.9916	3.0253	3.0580	3.0143	3.0496	3.0907	2.9999	3.0313	3.0712
time		solve 3s			5s			8s		
		learn 65s			65s			65s		
		estimate 50s			51s			50s		
algo 4 learn to solve while estimate										
θ	0.3000	0.2954	0.3102	0.3336	0.2954	0.3075	0.3432	0.2867	0.2992	0.3180
rc	3.0000	2.9536	2.9796	3.0047	2.9632	2.9846	3.0064	2.9381	2.9802	3.0141
time		warm-up 3s			4s			4s		
		MCMC 51s			50s			50s		

Note: performance comparison of algorithms, 10,000 MCMC run, 10,000 buses. The number of grids is for the error term ϵ.

We run the above four algorithms for the Rust model under three cases (IID, RE, AR(1)) respectively. As is standard in the literature, we assume that the discount rate $\beta = 0.99$ is known. For the mileage, we limit the possible values be $x \in [0, 1, ..., 20]$. For $x = 20$, the only available choice is to replace the engine. For all three cases, the variance of the innovation in cost is 1. The true parameters are $\theta = 0.3, rc = 3.0$ for all cases, $\sigma = 2.0$ for the RE case, and $\rho = 0.5$ for the AR(1) case. For full solution method, we set the maximum number of Bellman iteration to 10,000 and tolerance for value function error (measured by the sum of absolute error) as 0.00001. Our assumption of normally distributed errors rules out analytical solutions, and we rely on discrete grids for the numerical dynamic programming. The more number of grids for the errors, the more accurate our solution would be, at the cost of more computation. We consider three choices of the number of grids: 20, 100, and 200.

We simulate 10,000 buses until replacement for each choice of grid size, and run MCMC with chain length = 10,000 for the above four algorithms, resulting a total of 12 scenarios.[7] The first 1,000 iterations are discarded as burn-in runs, and we report the posterior median and the 5% and 95% posterior quantiles for each algorithm in Table 1.5. The computational time is reported in the bottom row of each panel.

We find that in most scenarios, the posterior 90% credible intervals are tight and contain the true parameters, while the posterior medians are close to the true values. For algorithm 1, the computational time increases from 87 seconds to 268 seconds as the grid size increases from 20 to 200. For algorithm 2, the computational time only increases from 17 seconds to 18 seconds. For algorithm 3, when grid size is 20, building the solution library takes 3 seconds , training the DNN takes 65 seconds, and

[7] Here we assume the mileage increases by 1 with certainty if not replaced, hence we only need to record the mileage at replacement for each bus, and the sample frequency of mileage at replacement would be a sufficient statistic here. In another word, the number of buses does not matter in the calculation of likelihood function. However, such shortcut does not work for stochastic mileage increment, say, a multinomial distribution over $\{0, 1, 2\}$. Under that case, we need to calculate the probability of observing the history for each bus, and likelihood function computation cost would be proportion to the number of sample.

run MCMC with DNN takes 50 seconds. Library building takes 8 seconds when grid size is 200, but the learning and estimating process takes same amount of time. For algorithm 4, warm up takes 3-4 seconds, and MCMC takes 50-51 seconds regardless of the grid size. To summarize, algorithm 2 performs well in IID case, where the state space is not large hence calculating the weighted average and updating the solution once is not costly. The two deep-learning aided algorithms (algorithm 3 and 4) spend more time with the additional DNN structure.

1.4.2 Model with RE

Now, we add random effect to the IID model. In this case, we need to solve the model for a set of possible values of the random effect α, deriving the unconditional replacement probability, and then (numerically) integrate them over the distribution of $\alpha \sim N(0, \sigma^2)$, where $\sigma = 2$. We reports the results for algorithms 1-3 in Table 1.6 in the same format as the last section, except that the number of grids not only stands for the grids for the error term ϵ, but also the random effect α.

First, we find that in most scenarios, similar to the IID case, the posterior credible intervals are tight and contain the true values, while the posterior medians are close to the true values. Second, the computational time for algorithm 1 increases from 2,648 seconds to 88,992 seconds when grid size increases from 20 to 200, reflecting the impact of curse of dimensionality. For algorithm 2, the computational time also increases from 171 seconds to 1,533 seconds. The most interesting case is algorithm 3. With 20 grids, it takes 3 seconds for solution, 83 seconds for learning, and 223 seconds for estimation, hence a total of 309 seconds, which is higher than algorithm 2. However, with 200 grids, it takes 12 seconds for solution, 83 second for learning, and 1340 seconds for estimation, so a total of 1,435 seconds, lower than 1,533 seconds for algorithm 2! The saving of time comes from DNN approximation: once we learn the threshold $\epsilon(x; \theta, rc, \alpha)$, we no longer need to solve the model many time for multiple

Table 1.6: Performance comparison, RE case

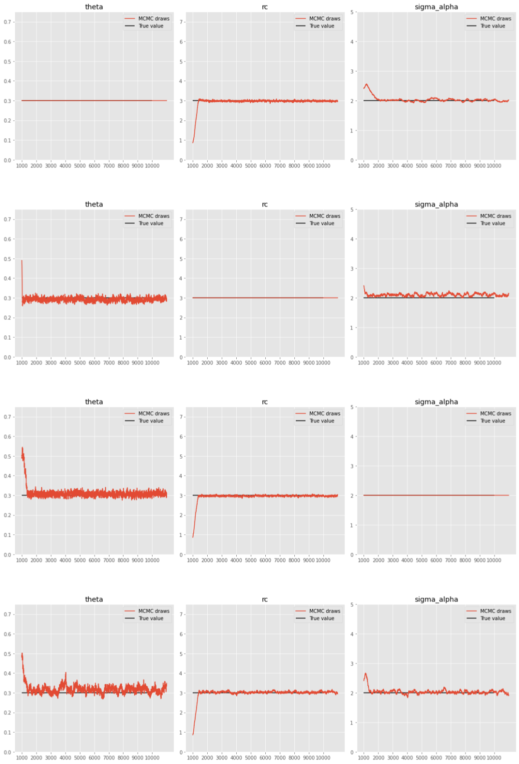
		# grid = 20			# grid = 100			# grid = 200		
	true	5%	median	95%	5%	median	95%	5%	median	95%
algo 1 solve to estimate										
θ	0.3000	0.3094	0.3460	0.3786	0.2394	0.2630	0.2879	0.2731	0.3020	0.3310
rc	3.0000	2.8644	2.9304	3.0174	3.0521	3.1352	3.2423	2.8990	2.9736	3.0674
σ	2.0000	1.8087	1.8896	1.9885	1.9952	2.0857	2.1846	1.8898	1.9695	2.0702
time		2648s			26441s			88992s		
algo 2 solve while estimate										
θ	0.3000	0.2726	0.3065	0.3373	0.2688	0.3055	0.3350	0.2612	0.2933	0.3222
rc	3.0000	2.9361	3.0072	3.1019	2.9161	2.9818	3.0751	2.9600	3.0306	3.1237
σ	2.0000	1.8886	1.9670	2.0575	1.8665	1.9456	2.0367	1.9536	2.0343	2.1222
time		171s			845s			1533s		
algo 3 solve, learn, then estimate										
θ	0.3000	0.2561	0.2742	0.2915	0.2688	0.3055	0.3350	0.2819	0.3159	0.3460
rc	3.0000	2.9501	3.0109	3.0888	2.9161	2.9818	3.0751	2.9309	3.0188	3.1242
σ	2.0000	2.0494	2.1249	2.2070	1.8665	1.9456	2.0367	1.9218	2.0265	2.1487
time		solve 3s			7s			12s		
		learn 83s			83s			83s		
		estimate 223s			693s			1340s		

Note: performance comparison of algorithms, 10,000 MCMC run, 10,000 buses. The number of grids is the same for 1) the error term ϵ and 2) the random effect α.

possible values of α in each iteration, but rather approximate them in a single call of DNN. And such saving of time would be more substantial if we run a longer MCMC chains and/or use more grids for the random effects.

A side issue for this extension of the Rust model is whether the model is still identified. In another word, would there be more than one set of parameters that generate the same data? We present the trace plots for four MCMC settings in Figure 1.9: in the first three panels, we fix one parameter at its true value, and estimate the other two; in the last panel, we estimate all three parameters at the same time. The trace plots show the chains converge to the posterior distribution that is centered around the true parameters, suggesting that including an additional parameter in this setting does not create difficulty in identification.

Figure 1.9: Results for RE with $\sigma = 2.0$



true	5%	post median	95%
0.3	0.3000	0.3000	0.3000
3.0	2.9322	2.9785	3.0243
2.0	1.9534	2.0105	2.0692

true	5%	post median	95%
0.3	0.2759	0.2916	0.3087
3.0	3.0000	3.0000	3.0000
2.0	2.0329	2.0993	2.1745

true	5%	post median	95%
0.3	0.2896	0.3067	0.3241
3.0	2.9187	2.9695	3.0197
2.0	2.0000	2.0000	2.0000

true	5%	post median	95%
0.3	0.2895	0.3159	0.3434
3.0	2.9577	3.0275	3.1093
2.0	1.9448	2.0198	2.1032

Note: 10,000 MCMC run, 10,000 buses. The number of grids is 11 for 1) the error term ϵ and 2) the random effect α.

1.4.3 Model with AR(1) Errors

The results for AR(1) errors are reported in Table 1.7, where we omit algorithm 1 due to its prohibitive computational cost.

First, we focus on the comparison of computational time. For algorithm 2, the computation time increases from 1,822 to 5,136 seconds when grid size increases from 20 to 200. For algorithm 3, the time for learning and estimation does not change a lot, but the time for library building increases from 1,109 to 8,613 seconds. For algorithm 4, the time for MCMC runs is around 2,100 seconds, while the time for warm-up increases from 122 second to 895 seconds. In summary, for Rust model with AR(1) errors, algorithm 4 achieves saving in computational time when the number of grids is large.

Table 1.7: Performance comparison, AR(1) case

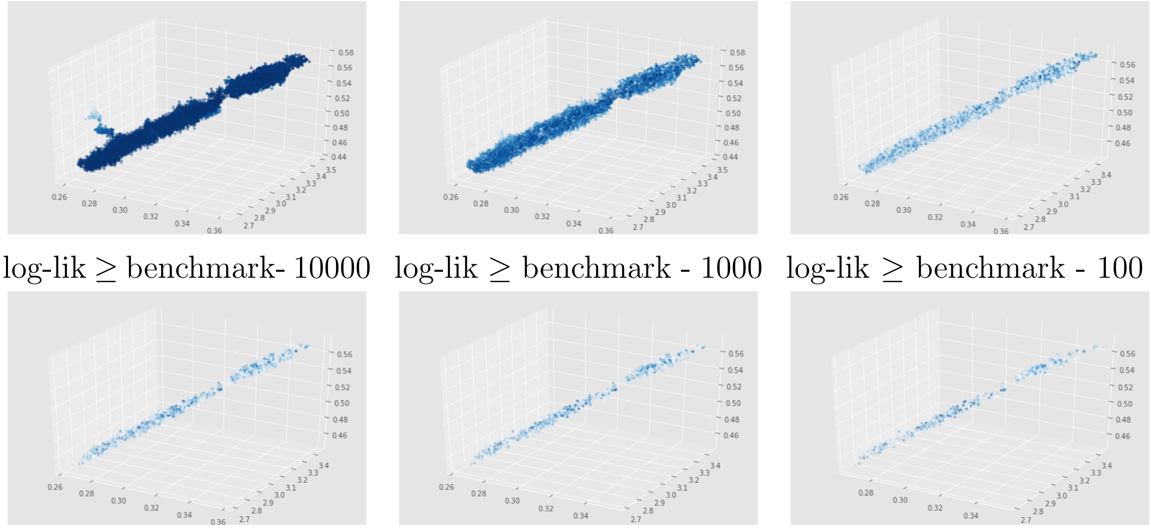
		# grid = 20			# grid = 100			# grid = 200		
	true	5%	median	95%	5%	median	95%	5%	median	95%
algo 2 solve while estimate										
θ	0.3000	0.2629	0.3095	0.3632	0.2525	0.2827	0.3184	0.2725	0.3217	0.3660
rc	3.0000	2.7308	3.1109	3.4819	2.6766	2.9075	3.1712	2.8492	3.2137	3.5457
ρ	0.5000	0.4395	0.5167	0.5707	0.4254	0.4679	0.5089	0.4615	0.5279	0.5693
time		1822s			4539s			5136s		
algo 3 solve, learn, then estimate										
θ	0.3000	0.2880	0.3090	0.3264	0.2811	0.3112	0.3384	0.2307	0.2634	0.2981
rc	3.0000	2.9587	3.2796	3.3889	2.8529	3.0525	3.2120	2.4854	2.7039	2.9683
ρ	0.5000	0.5065	0.5355	0.5679	0.4948	0.5213	0.5508	0.3955	0.4472	0.4987
time		solve 1109s			4206s			8613s		
		learn 83s			83s			83s		
		estimate 1965s			1950s			1950s		
algo 4 learn to solve while estimate										
θ	0.3000	0.3041	0.3300	0.3594	0.2286	0.2467	0.2664	0.2566	0.3060	0.3465
rc	3.0000	3.2908	3.5020	3.7275	2.4927	2.6159	2.7188	2.7714	3.3717	3.7423
ρ	0.5000	0.4234	0.4858	0.5261	0.3799	0.3983	0.4297	0.4253	0.4994	0.5347
time		warm-up 112s + 10s			443s + 10s			885s + 10s		
		MCMC 2107s			2025s			2133s		

Note: performance comparison of algorithms, 10,000 MCMC run, 10,000 buses. The number of grids is for the error term ϵ.

It is worth noting that the posterior credible interval is much wider under AR(1) for all algorithms; in some cases, it even does not contain the true values. This suggests difficulty in identification of model parameters. We now provide more evidence and explore possible causes for weak identification.

First, in Figure 1.10, we plot the sets of parameters with log-likelihood above certain values. We set the benchmark likelihood as the one for the sample under true parameters. As we decrease the value from 10,000 below the benchmark (top left) to 10 above the benchmark (lower right), the selected sets of parameters shrink. However, we find that these points lies along a line of the 3-dimensional parameter space. And such pattern persists even for points whose log-likelihood is above the benchmark.

Figure 1.10: MCMC draw of parameters whose log-likelihood is above certain values



log-lik $\geq$ benchmark- 10000 log-lik $\geq$ benchmark - 1000 log-lik $\geq$ benchmark - 100

log-lik $\geq$ benchmark - 10 log-lik $\geq$ benchmark + 0 log-lik $\geq$ benchmark + 10

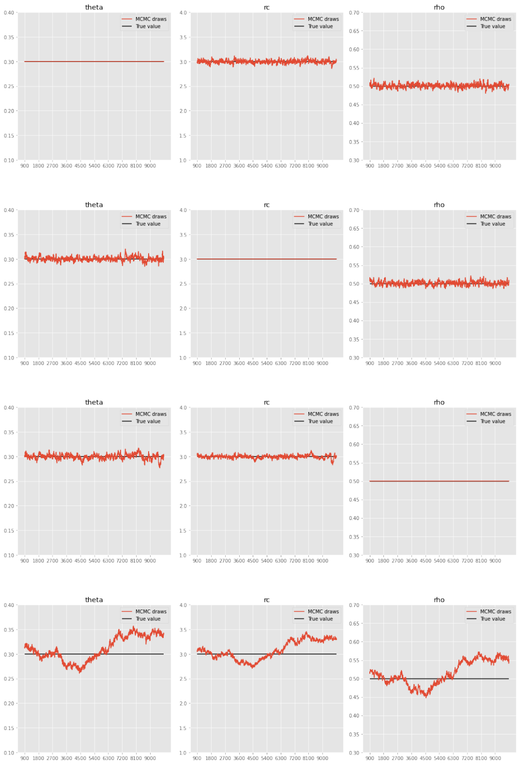
Note: benchmark = log-likelihood of the sample of 1,000,000 buses under the true parameters ($\ell(\theta = 0.3, rc = 3.0, \rho = 0.5) = -2022512$).

Second, we repeat the procedure that produced Figure 1.9 with data generated by the AR(1) model. In Figure 1.11, we first run three chains by fixing one of θ, rc, ρ for each chain, with 1,000,000 buses. These three chains all gives tight posterior credible interval around true values. However, if we allow all three parameters to be unrestricted, the chain no longer mixes for 10,000 runs. This pattern holds when the data generating ρ is 0.5, $0.7, 0.3$, 0.1, and even for $\rho = 0.0$. See Appendix Figure A.1-A.4 for more details.

There are some possible causes of weak identification: (i) Not enough variation in states: we assumes deterministic state dynamics, and the utility function is linear in parameters. (ii) Absence of other covariate: the only state variable is the mileage x. (iii) Accumulation approximation error from three sources: Gauss-Hermite quadrature and interpolation (in the solution process); deep neural network training (in the learning process); and GHK approximation in likelihood calculation (in the estimation process).

We conclude the section with a brief review of numerical findings by other studies that estimate the Rust model in the presence of serial correlated shocks. Norets (2009) integrates DP-MCMC method with simulated data. He finds a bi-modal distribution for ρ with actual data. Reich (2018) finds MLE tends to overestimate θ and but underestimates ρ, but the bias tends to diminish as the sample size increases. Blevins (2016) specifies the model differently and only reports few estimated ratio between costs. All of them assume stochastic process of mileage (e.g., multinomial and exponential distribution). We also experimented with stochastic process of mileage but still find evidence of weak identification.

Figure 1.11: Results for AR(1) with $\rho = 0.5$, 1,000,000 buses



true	5%	post median	95%
0.3	0.3000	0.3000	0.3000
3.0	2.9432	2.9964	3.0519
0.5	0.4912	0.5010	0.5099

true	5%	post median	95%
0.3	0.2939	0.3003	0.3083
3.0	3.0000	3.0000	3.0000
0.5	0.4921	0.5011	0.5098

true	5%	post median	95%
0.3	0.2911	0.2997	0.3080
3.0	2.9420	2.9975	3.0477
0.5	0.5000	0.5000	0.5000

true	5%	post median	95%
0.3	0.2724	0.3073	0.3462
3.0	2.7895	3.0464	3.3534
0.5	0.4628	0.5093	0.5608

1.5 A Car Replacement Example

In this section, we extend the above one-dimensional engine replacement problem to allow for multi-dimensional state variables. We label the new problem a "car replacement" problem. Specifically, let $\mathbf{x_t} = (x_{1t}, ..., x_{kt})^T$ be a $k-$ dimensional vector of state variables that may include engine mileage, condition of the car, etc.[8] A car owner decides whether to replace her old car with a new one by observing this state vector as well as idiosyncratic shock $\mathbf{\epsilon_t} = (\epsilon_{0t}, \epsilon_{1t})' \overset{i.i.d.}{\sim} G(\cdot)$ to the two actions $y_t \in \{0, 1\}$ (keeping the car or replacing it.) If she chooses to replace the car ($y_t = 1$) all state variables are reset to zero. If she keeps the car ($y_t = 0$) the increment for each state variable follows a known joint distribution.[9] Here, for simplicity we assume that the increments are identical and independent Bernoulli distribution with parameter p.[10] For example, when $k = 2$, if the owner keeps the car running and the current state is (x_{1t}, x_{2t}), then the state variables for the next period are:

$$(x_{1t+1}, x_{2t+1}) = \begin{cases} (x_{1t}, x_{2t}),\ \textit{with probability}\ (1-p)^2; \\ (x_{1t}+1, x_{2t}),\ \textit{with probability}\ p(1-p); \\ (x_{1t}, x_{2t}+1),\ \textit{with probability}\ p(1-p); \\ (x_{1t}+1, x_{2t}+1),\ \textit{with probability}\ p^2. \end{cases}$$

The value function is

$$V(\mathbf{x_t}, \mathbf{\epsilon_t}) = \min_{y_t \in \{0,1\}} \{\theta^T \mathbf{x_t} + \epsilon_{0t} + \beta \mathbb{E} V(\mathbf{x_{t+1}}, \mathbf{\epsilon_{t+1}} | y_t = 0),$$
$$rc + \epsilon_{1t} + \beta \mathbb{E} V(\mathbf{x_{t+1}}, \mathbf{\epsilon_{t+1}} | y_t = 1)\}$$

where we assumes a linear cost running cost with coefficient $\theta = (\theta_1, ..., \theta_k)^T$, and the replacement cost is constant rc. There exists a threshold strategy that we replace the

[8] This example is inspired by Imai et al. (2009) who give a single-firm entry-exit example in the fashion of Rust (1987), but with additional state variable to capture observable heterogeneity.

[9] Note that even it is unknown, we can estimate it from the data, see Rust (1987) for details. Since this paper focuses on the estimation of parameters in the cost function only, we assume it is already known.

[10] We allow for stochastic evolution of state variables, since otherwise the possible sample path is unique and could be reparametrized to the one-dimensional case.

car if and only if

$$\epsilon_{0t} - \epsilon_{1t} < \epsilon^*(\mathbf{x_t})$$

Assume that the difference between error term follows a standard normal distribution $\epsilon_{0t} - \epsilon_{1t} \overset{i.i.d.}{\sim} N(0,1)$.[11] Finally, given the history of the state variables and replacement decision of a car, we can write out the likelihood function as the product involving standard normal CDF. Taking logarithm and summing up for all cars, we arrive at the sample likelihood, and Bayesian estimation is then feasible.

As the dimension of state vector, k, increases, solving the model can be very costly. Numerically, a common practice is to use finite grids for these state variables. Say we use M equally spaced grids between $[0, \bar{x}]$ for each state variable, then we need to perform value function iteration over M^k points until convergence.[12] Even one iteration is time-consuming when M^k is large for algorithm 2. One possible time-saving approach is to limit the number of grids, but it could take longer for the Markov Chain to converge, since the underlying non-parametric estimator needs a large volume of data to perform well. For algorithm 3, since solving the model is now very difficult, an accurate library of solutions may take too long to build, and we cannot determine ex ante how large the library should be. However, algorithm 4 seems promising since we do not need to build an accurate library ex ante. Moreover, using DNN as the approximator typically requires fewer data than other non-parametric approaches, if the high-dimensional function exhibits some low-dimensional patterns.

We fix $k = 2$ and consider three choices of M: 21 (hence each state variable takes values in 0, 1, ..., 20), 41 and 61. The true values for two cost coefficients are $\theta_1 = 0.1, \theta_2 = 0.2$, while the replacement costs are $rc = 6.0$ for $M = 21$, 12.0 for 41,

[11] We follow the tradition of two error terms for two actions to avoid unnecessary divergence, though it can be shown they give raise to the same model with one error term. See appendix for some numerical experiments.

[12] Another approach is to use randomization, function approximation, and some combinations of the two, see e.g., Rust (1997), Judd et al. (2014). Our algorithm 4 exploits the effectiveness of DNN in approximation and does not involve much tuning. Combining it with the estimation task is also natural in Bayesian settings.

and 18.0 for 61 to make sure those extended mileage grids are indeed possible and can be observed from the data.[13] The parameter for the increment of state variable, p is set to be 1/2 and known.

The estimation results are summarized in the following tables. They show that: 1) All algorithms yield accurate posterior distribution centered around true parameter values. 2) With a small number of grids, $M = 21$, the two DNN-aided algorithms take extra time for library building and DNN training. 3) With a medium number of grids, $M = 41$, algorithm 2 takes shorter time since doing one value function iteration is not that costly. 4) With a large scale of grids, $M = 61$, algorithm 3 takes too long to build the library, and for algorithm 2 even one iteration is costly, while algorithm 4 takes the least time to finish.[14] Our analysis suggests that if we further increase M, the advantage of algorithm 4 would be more obvious.

1.6 Concluding Remarks

This chapter proposes and implements a new method ("learn to solve while estimate") that solves the model and learn the pattern of the solution along the estimation process. We contrast it with three competing algorithms ("solve to estimate", "solve while estimate", and "solve, learn, then estimate") in an infinite horizon setup and use three variants of Rust's engine replacement problem as the benchmark. We find that 1) "solve to estimate" is vulnerable to the curse of dimensionality; 2) "solve while estimate" is computationally efficient for problem with small state space; 3) "solve, learn, then estimate" is suited for small scale models that are time-consuming to estimate; 4) "learn to solve while estimate" outperforms others when the state

[13] The priors are: $\theta_1 \sim U[0.05, 0.15], \theta_2 \sim U[0.15, 0.30]$. For $M = 21$, $rc \sim U[3, 9]$; $M = 41$, $rc \sim U[6, 18]$; $M = 61$, $rc \sim U[9, 27]$. We use 21 grids points for the Gauss-Hermite quadrature of the error term ϵ. Simple Monte Carlo requires thousands of grids to achieve similar accuracy of value function iteration.

[14] As in previous experiments, for large scales, algorithm 1 takes too long and we omit it here.

Table 1.8: Performance comparison, $M = 21$

	true	post mean	post std	5%	post median	95%
algo 1 solve to estimate						
theta1	0.1000	0.0999	0.0022	0.0963	0.0999	0.1034
theta2	0.2000	0.1950	0.0031	0.1898	0.1950	0.2000
rc	6.0000	5.8918	0.0680	5.7741	5.8960	5.9991
time						14020s
algo 2 solve while estimate						
theta1	0.1000	0.0998	0.0021	0.0964	0.0998	0.1032
theta2	0.2000	0.1950	0.0030	0.1901	0.1949	0.2000
rc	6.0000	5.8891	0.0639	5.7851	5.8902	5.9957
time						13670s
algo 3 solve, learn, then estimate						
theta1	0.1000	0.0980	0.0025	0.0936	0.0982	0.1018
theta2	0.2000	0.1916	0.0027	0.1872	0.1916	0.1960
rc	6.0000	5.8978	0.0682	5.7745	5.9027	6.0056
time						16873s
algo 4 learn to solve while estimate						
theta1	0.1000	0.0962	0.0021	0.0930	0.0959	0.0994
theta2	0.2000	0.1925	0.0025	0.1885	0.1922	0.1976
rc	6.0000	5.8603	0.0719	5.7287	5.8704	5.9460
time						16959s

Note: performance comparison of algorithms, 10,000 MCMC run, 10,000 buses. The number of grids is for each state variable.

space is large and accurate solution is costly to obtain.

Algorithms that reduce computational cost also allow us to investigate the likelihood function more closely, and we find that the benchmark Rust model with serially correlated error may be unidentified. The issue of identification is left for future investigation. Numerical simulation of this chapter suggests that algorithm 4 may be more useful for estimating structural models that are more costly to solve than the engine replacement problem. Further explorations will be left for future research.

Table 1.9: Performance comparison, $M = 41$

	true	post mean	post std	5%	post median	95%
algo 1 solve to estimate						
theta1	0.1000	0.1005	0.0016	0.0977	0.1004	0.1031
theta2	0.2000	0.1970	0.0025	0.1928	0.1970	0.2010
rc	12.0000	11.9274	0.1094	11.7332	11.9282	12.1132
time						47313s
algo 2 solve while estimate						
theta1	0.1000	0.0996	0.0016	0.0971	0.0996	0.1023
theta2	0.2000	0.1985	0.0024	0.1946	0.1985	0.2023
rc	12.0000	11.9405	0.1094	11.7690	11.9357	12.1276
time						22862s
algo 3 solve, learn, then estimate						
theta1	0.1000	0.0971	0.0027	0.0942	0.0968	0.1009
theta2	0.2000	0.1940	0.0056	0.1892	0.1932	0.1990
rc	12.0000	11.7685	0.2811	11.5177	11.7242	12.1242
time						29581s
algo 4 learn to solve while estimate						
theta1	0.1000	0.1019	0.0015	0.0993	0.1024	0.1040
theta2	0.2000	0.1949	0.0015	0.1924	0.1950	0.1972
rc	12.0000	11.8862	0.0945	11.7048	11.9052	12.0070
time						29799s

Table 1.10: Performance comparison, $M = 61$

	true	post mean	post std	5%	post median	95%
algo 2 solve while estimate						
theta1	0.1000	0.1027	0.0015	0.1003	0.1027	0.1051
theta2	0.2000	0.1997	0.0022	0.1963	0.1997	0.2034
rc	18.0000	18.0974	0.1509	17.8735	18.0868	18.3577
time						33694s
algo 3 solve, learn, then estimate						
theta1	0.1000	0.0953	0.0080	0.0883	0.0907	0.1105
theta2	0.2000	0.2021	0.0124	0.1897	0.1959	0.2244
rc	18.0000	18.3123	1.1760	17.1967	17.6920	20.5436
time						41691s
algo 4 learn to solve while estimate						
theta1	0.1000	0.0996	0.0014	0.0977	0.0994	0.1016
theta2	0.2000	0.2019	0.0023	0.1987	0.2013	0.2051
rc	18.0000	17.9791	0.1262	17.7803	17.9843	18.2247
time						31682s

Chapter 2

Comparison of Two Optimal Retirement Models: Stock-Wise versus Dynamic Programming

2.1 Introduction

Following the notation in Chapter 1, we consider a policy function in a structural model $\mathbf{y} = f(\mathbf{X}, \boldsymbol{\theta}, \boldsymbol{\epsilon})$ and its numerical approximation $\mathbf{y} = \hat{f}(\mathbf{X}, \boldsymbol{\theta}, \boldsymbol{\epsilon})$, with choice variable $\mathbf{y}$, state variable $\mathbf{X}$, structural parameters $\boldsymbol{\theta}$, and error term $\boldsymbol{\epsilon}$. In some cases we seek to identify from a finite sample of data two competing but similar models $\mathbf{y} = f_1(\mathbf{X}, \boldsymbol{\theta}_1, \boldsymbol{\epsilon}_1)$ and $\mathbf{y} = f_2(\mathbf{X}, \boldsymbol{\theta}_2, \boldsymbol{\epsilon}_2)$. For this exercise large approximation errors $||f_j - \hat{f}_j||$ $(j = 1, 2)$ may lead to biased conclusions.[1]

Can one estimate a complicated structural model based on accurate solutions? In this paper we argue that for a class of binary choice problem (an optimal retirement problem with serially correlated errors in particular) the answer is yes.

In the retirement problem, an economic agent (say a senior school teacher) consid-

[1] The difference of optimal policy in the models is affected by the error $||\hat{f}_1 - \hat{f}_2||$. $||f_1 - f_2|| = ||(f_1 - \hat{f}_1) - (f_2 - \hat{f}_2) + (\hat{f}_1 - \hat{f}_2)|| < ||f_1 - \hat{f}_1|| + ||f_2 - \hat{f}_2|| + ||\hat{f}_1 - \hat{f}_2||$.

ers the optimal timing for an irreversible retirement from continuing teaching. Factors that influence the decision include an experience-dependent salary schedule, a pension formula that relates retirement benefit to the age and experience at the time of retirement, a mortality table that determines life expectancy, a utility function that sets preference of salary income relative to pension benefit, and an observed serially correlated errors that measures the teacher's preference to teaching.

The solution to this class of problem is characterized by a threshold of the current preference error that depends on age and experience (the state variable $\mathbf{X}$) and preference parameter $\boldsymbol{\theta}$. If the current preference error is below the threshold, then it is optimal to retire, otherwise it is optimal to stay teaching. This means the solution f of the structural model takes the form of the threshold of errors. The thresholds exhibit a pattern conducive to machine learning.

We present a three-step approach aided by deep learning neural network for the problem (algorithm 3 "solve, learn, then estimate" in Chapter 1.) In the first step, the solution step, we obtain numerical $\hat{f}(.,\boldsymbol{\theta},.)$ for N combinations of state variable and parameters $(\mathbf{X},\boldsymbol{\theta})$ and save the solution as a library. In the second step, the learning step, we approximate $\hat{f}(\mathbf{X},\boldsymbol{\theta}^{\#},.)$ by learning from the pre-built library. The tool for the machine learning is DNN. If the optimal policy can be accurately learned with a moderately sized library the total computational cost for estimation may be lower than the traditional approach for a given level of accuracy. Finally, in the third step, the estimation step, we search for parameters $\boldsymbol{\theta}$ that minimizes simulated statistics and sample statistics. Rather than solving for $\hat{f}(\mathbf{X},\boldsymbol{\theta}^{\#},.)$ every time $\boldsymbol{\theta}$ is evaluated at a new value $\boldsymbol{\theta}^{\#}$, we use the learned DNN as surrogates.

Numerical analysis of this approach is based on two structural models of teacher retirement under the same set of factors (pension rules, salary schedule, mortality rate, utility function, serially correlated preference errors). One is the forward-solving optional value model by Stock and Wise (1990) (SW) model, another is dynamic programming (DP) model. The difference between the two models is subtle. The SW model does not factor in the option value of future options of continuing teaching. On the other hand, it has a lower computational cost than dynamic programming because it does not involving solving for the value function.

The paper contributes to the tool box for policy analysis of pension incentives on retirement. First, given a pension rule we can save the approximated numerical solution to a structural model in a library. To estimate parameters from a different data set, other researchers can retrieve solutions from the library, thus skip the costliest step in structural estimation, making estimation of structural models for retirement easier and more common. We simulate data from a SW model and a DP model in turn, then estimate the data generating model aided by DNN. We find the DNN aided estimation is computationally feasible and can recover the data-generating model as well as data generating parameters. For a sample of 21,412 teachers the DNN-aided approach estimates the SW model in 400 seconds for the SW model and the DP model in 3 hours. Under the traditional approach, with the same estimation error it took 2,000 seconds to estimate the SW model and 36 hours to estimate the DP model.

Second, we find that it is feasible to estimate structural models for teacher retirement in reasonable time with high accuracy in approximation of the structural model. The high accuracy renders it possible to identify similar structural models from data. When the data are generated from a SW model the estimated model easily rejects the DP model, and vice versa.

The paper is organized as follows. Section 2 presents the SW model of optimal retirement timing and a DP version of the problem. Solutions to both problems have the feature that when the current value of serially correlated preference errors is below a threshold it is optimal to retire. Given the state variable the threshold that solves the SW model is above that of the DP model (i.e., if it is optimal to retire in the DP model then it is optimal to so in the SW models.) Section 3 outlines the three-step strategy: the solution step, learning step and estimation step. Section 4 presents numerical simulations with the focus on accuracy and computation time.

2.1.1 Related literature

A number of recent studies have developed on the intersection between machine learning (in particular deep learning) and structural estimation, see the summary by Iskhakov et al. (2020) and Igami (2020). Farrell et al. (2021) provide theoretical justification of using DNN as the approximation structure and Farrell et al. (2020) discuss possible applications of DNN. Semenova (2018) applies machine learning to estimation of DDCM under the Conditional Choice Probability (CCP) framework. The most relevant work for this study is Norets (2012). We adopt his strategy but make two modifications. We approximate the threshold instead of the value function; and we train the DNN with a loss function that weights thresholds by their importance.

Another strand literature related to the present study is solutions of optimal retirement problem, see Stock and Wise (1990), Ni and Podgursky (2016) for the SW model, and Gustman and Steinmeier (1986), Rust and Phelan (1997), Berkovec and Stern (1991) for DP model. Lumsdaine et al. (1992) and Belloni (2008) compare the property of SW and DP models. However, these comparisons are made with different model setups: In the SW model the error term is assumed to be AR(1) normal, but in the DP model, the error term is assumed independently and identically distributed (iid) generalized extreme value distribution (GEV). One reason for assuming iid errors in the DP model is the high computational cost associated solving DP models with AR(1) errors. Here we compare the SW and DP models assuming *exactly* the same structural parameters and distribution of error terms. The DNN aided algorithm makes it feasible to estimate a DP model with AR(1) errors.[2]

[2] Even with DNN, we still need to solve the DP model under some parameter values. For that pur-

2.2 Teachers' Optimal Retirement Problem with Normal Serially Correlated Errors

2.2.1 A general model

We work under a finite-horizon setting. We assume the maximum lifespan is T. The value function depends on age and experience. Note that the retirement problem presents a few special features that may be used to save computation cost. One is that the choice set is state dependent: a retiree can only stay retired. The state variables are also restricted in some fashion: age and experience can only go up by one per year. The teacher currently in teaching force decides whether to retire at the end of the year, after enjoying the utility of current period salary. We assume that there is an AR(1) error in preference to teaching and no preference shock after retirement. The retirement benefit for retirees depends on age and experience.

The retirement eligible range of age and experience combination is $(a, e) \geq (a_0, e_0)$, where the boundary (a_0, e_0) is defined by pension rules. We denote the values by the age and experience in the initial period, (a, e). For instance, $y_{(a,e)}(t)$ is the salary of a teacher in year t with (a, e) in the initial year 0 (hence with age $a+t$ and experience $e+t$ in year t.)

With a fixed salary schedule there are two sources of uncertainty: the uncertainty of survival of mortality and uncertainty in preference shocks.

The range for age $a \leq A$ and experience $e \leq A - 22$. We denote survival probability from age a to age $a+k$ by $G(a, a+k)$. For a teacher alive in year t we denote the probability of survival to period $s > t$ as $\pi(s|t) = G(a+t, a+s) = G(a+t, a+t+1)..G(a+s-1, a+s)$.

The current pension wealth (the discounted pension wealth at the time of separation) $W_{(a,e)}(t)$ has the following properties: (1) The eligibility depends on age and

pose, we use the quadrature-interpolation method following Stinebrickner (2000) and Stinebrickner (2001).

experience. (2) For eligible retirees, the annual benefit $B_{(a,e)}(s)$ in year s is a function of experience. The year t value of pension wealth is $W_{(a,e)}(t) = \sum_{s=t}^{T} G(a + s, a + s + 1)\beta^{s-t}(B_{(a,e)}(s,t))^{\gamma}$, where $B_{(a,e)}(s,t)$ is the year-t value of annual retirement benefit received in year s by the teacher with initial (a, e) retired in year t. The year-t value of pension wealth for a teacher with initial (a, e) retired in year $s > t$ is $\beta^{s-t}\mathbb{E}_t W_{(a,e)}(s) = \beta^{s-t}\pi(s|t)W_{(a,e)}(s)$. For any age $W_{(a,e)}(t)$ is increasing in experience e. With age over a retirement eligible threshold $W_{(a,e)}(t)$ is decreasing in age because for the fixed e, a larger a means fewer years to collect benefits.

The utility function for period t is

$$[(\kappa_t y_{(a,e)}(t))^{\gamma} + \nu_t],$$

where $\kappa_t = \kappa(\frac{60}{a+t})^{\kappa_1}$ is an age-dependent parameter of leisure, with $0 < \kappa \leq 1$ during working years and captures the disutility of working. The unobserved innovations in preferences are AR(1):

$$\nu_t = \rho\nu_{t-1} + \epsilon_t. \tag{2.1}$$

We assume ϵ_t is iid $N(0, \sigma^2)$. The preference error ν captures the preference for teaching. A larger ν favors continuing teaching relative to retirement. Teachers with the same initial (a, e) choose different time to retire in response to realizations unobserved preference errors. We assume $0 \leq \rho < 1$ for two reasons. One is that the nature of the unobserved heterogeneity in such as health or preference to teaching likely has a positive serial correlation for each teacher. Another reason is that empirical estimates of ρ that fit teacher samples are invariably in $(0, 1)$.

The teacher's expected utility in period t is a function of expected retirement in year r (with $r = t, \cdots, T$ and T depends on the upper bound on age). In period t, the expected utility of retiring in period r is the discounted sum of pre- and post-retirement expected utility

$$V_t(r) = \underbrace{\mathbb{E}_t\{\sum_{s=t}^{r-1} \beta^{s-t}[(\kappa_s y_{(a,e)}(s))^\gamma + \nu_s]\}}_{\text{expected utility before retirement}} + \underbrace{\mathbb{E}_t \beta^{r-t} W_{(a,e)}(r)}_{\text{expected utility after retirement}}.$$

The expectation for the first term pertains to the unobserved innovations in preferences and survival probability, and the expectation of the second term only pertains to the survival probability. By definition $V_t(t) = W_{(a,e)}(t)$.

2.2.2 The dynamic programming solution

The value function of current teacher with preference error ν_t is $V_{(a,e)}(t, \nu_t)$, is

$$V_{(a,e)}(t, \nu_t) = max\{U_{(a,e)}(t, \nu_t) + \nu_t, \quad W_{(a,e)}(t)\} \tag{2.2}$$

where $U_{(a,e)}(t, \nu_t)$ is the expected value function of continuing teaching:

$$U_{(a,e)}(t, \nu_t) = [\kappa_t y_{(a,e)}(t)]^\gamma + \beta G(a+t, a+t+1)\mathbb{E}_\epsilon V_{(a,e)}(t+1, \nu_{t+1}). \tag{2.3}$$

The teacher chooses to retire when $V_{(a,e)}(t, \nu_t) = W_{(a,e)}(t)$; i.e., $U_{(a,e)}(t, \nu_t) + \nu_t \leq W_{(a,e)}(t)$. The following proposition says the expected value of continuing teaching is increasing in the preference error ν.

Proposition 1: $U_{(a,e)}(t, \nu)$ is increasing in ν.

Proof. If $U_{(a,e)}(t+1, \nu_{t+1})$ is increasing in ν_{t+1} then $V_{(a,e)}(t+1, \nu_{t+1})$ is increasing in ν_{t+1} and $\mathbb{E}_\epsilon V_{(a,e)}(t, \rho\nu_t + \epsilon_{t+1})$ is increasing in ν_t. Hence from (C.1) $U_{(a,e)}(t, \nu_t)$ is increasing in ν_t. The proposition is proved by backward induction.

Denote a threshold ν_t^* that satisfies

$$U_{(a,e)}(t, \nu_t^*) + \nu_t^* = W_{(a,e)}(t). \tag{2.4}$$

Proposition 2: There is a unique $\nu_{(a,e)}^*(t)$. This follows from Proposition 1.

Proposition 2 suggests a threshold strategy for optimal retirement timing: A teacher with initial age-experience (a, e) chooses to stay (retire) in year t if $\nu_t > (\leq$ $)\ \nu_{(a,e)}^*(t)$. We later also denote the threshold $\nu_{(a,e)}^*(t)$ as $f_{(a,e)}^*(t)$ to indicate that it is an optimal policy.

2.2.3 The Stock-Wise solution

In the SW "option value" model the expected gain from retirement at age r over retirement in the current period is

$$\begin{aligned} G_{(a,e)}(r,t) &= \mathbb{E}_t V_t(r) - \mathbb{E}_t V_t(t) \\ &= \underbrace{\sum_{s=t}^{r-1} \pi(s|t)\beta^{s-t}(k_s y_{(a,e)}(s))^\gamma + \sum_{s=r}^{T} \pi(s|t)\beta^{s-t}(B_{(a,e)}(s,r))^\gamma - \sum_{s=t}^{T} \pi(s|t)\beta^{s-t}(B_{(a,e)}(s,t))^\gamma}_{g_{(a,e)}(r,t)} \\ &+ \underbrace{\sum_{s=t}^{r-1} \pi(s|t)\beta^{s-t}\mathbb{E}_t \nu_s}_{K_{(a,e)}(r,t)\nu_t} . \end{aligned} \tag{2.5}$$

We denote the sum of the first three terms, a function of current salary and experience, by $g_t(r)$. The last term equals $\sum_{s=t}^{r-1} \pi(s|t)(\beta\rho)^s \nu_t$. We denote it as $K_t(r) = \sum_{s=t}^{r-1} \pi(s|t)(\beta\rho)^s$ (which depends on unknown parameters) times the error term given in (3.4.2). The SW retirement decision can thus be formulated as choosing $r = t, \cdots, T$ that maximizes

$$G_t(r) = g_t(r) + K_t(r)\nu_t.$$

Let

$$r_t^\dagger = argmax\ g_t(r)/K_t(r),$$

The teacher retires if

$$G_t(r) \leq 0\ \forall\ r > t; \quad i.e.\ \frac{g_t(r^\dagger)}{K_t(r^\dagger)} \leq -\nu_t. \tag{2.6}$$

Hence the probability that teacher retires in period t is $Prob(\frac{g_t(r^\dagger)}{K_t(r^\dagger)} \leq -\nu_t)$.

Denote

$$f_{(a,e)}^\dagger(t) = \nu_{(a,e)}^\dagger(t) = -\frac{g_t(r^\dagger)}{K_t(r^\dagger)} \tag{2.7}$$

then the teacher stays if $\nu_t > f_t^\dagger$.

2.2.4 Comparing the retirement decisions in DP and SW

Under the same parameter and preference shocks the retirement decision under DP in year t may differ from that under SW because the latter does not factor in the value of options in years after t. To compare optimal decisions in DP and SW models, note that in (2.2) with initial age a and experience e,

$$\begin{aligned}
&U_{(a,e)}(t,\nu_t) \\
&= [(k_t y_{(a,e)}(t))^\gamma + \nu_t] + \beta G(a+t, a+t+1)\mathbb{E}_t max\{U_{(a,e)}(t+1,\nu_{t+1}) + \nu_{t+1}, W_{(a,e)}(t+1)\} \\
&\geq [(k_t y_{(a,e)}(t))^\gamma + \nu_t] + \beta G(a+t, a+t+1)\mathbb{E}_t\{U_{(a,e)}(t+1,\nu_{t+1}) + \nu_{t+1}\} \\
&= [(k_t y_{(a,e)}(t))^\gamma + \nu_t] + \beta G(a+t, a+t+1)\mathbb{E}_t\{[k_{t+1} y_{(a,e)}(t+1)]^\gamma + \nu_{t+1}\} \\
&\quad + \beta G(a+t+1, a+t+2)\mathbb{E}_{t+1} max\{U_{(a,e)}(t+2,\nu_{t+2}) + \nu_{t+2}, W_{(a,e)}(t+2)\} \\
&\geq \mathbb{E}_t\{\sum_{s=t}^{t+1} \beta^{s-t}[(k_s y_{(a,e)}(s))^\gamma + \nu_s] + \sum_{s=t+2}^{T} \beta^{s-t}[(B_{(a,e)}(s,t+2))^\gamma]\}.
\end{aligned} \tag{2.8}$$

The last step uses the fact that the year-t expected value of pension wealth for retiring in $t+2$ is $\beta^2\mathbb{E}_t\mathbb{E}_{t+1}W_{(a,e)}(t+2) = \pi(t+2|t)\sum_{s=t+2}^{T} G(a+s, a+s+1)\beta^{s-t}[(B_{(a,e)}(s, t+2))^{\gamma}]$.

Repeated application of the same argument implies that

$$U_{(a,e)}(t, \nu_t) \geq \mathbb{E}_t\{\sum_{s=t}^{r-1} \beta^{s-t}[(k_s y_{(a,e)}(s))^{\gamma} + \nu_s] + \sum_{s=r}^{T} \beta^{s-t}[(B_{(a,e)}(s, r))^{\gamma}]\}, \qquad (2.9)$$

for any $r > t$. The result (2.8) can be viewed as a special case of $r = t + 2$. Hence if $U_{(a,e)}(t, \nu_t) + \nu_t \leq W_{(a,e)}(t)$ then $V_t(r) \leq V_t(t)$. Hence we have the following proposition.

Proposition 3. If a teacher chooses to retire under DP, then given the same parameters and preference shocks she chooses to retire under SW. The reverse is not true. If it is optimal for a teacher to retire under the SW model it may not be optimal for her to retire under the DP model.

This proposition says if $\nu_t > \nu^{\dagger}_{(a,e)}(t)$ then $\nu_t > \nu^{*}_{(a,e)}(t)$. Hence $\nu^{\dagger}_{(a,e)}(t) \geq \nu^{*}_{(a,e)}(t)$ for all t and (a, e). The numerical solutions $\nu^{\dagger}_{(a,e)}(t)$ is less costly to compute then $\nu^{*}_{(a,e)}(t)$.

2.2.5 Solution technique

Solving DP

Given the value of structural parameters, we need to solve the model for each (a, e, t) and all ν_t. Since ν_t follows an AR(1) process, its support is the entire $\mathbb{R}$. A common practice is approximating ν_t by finite grids through discretization or projection.

Following Stinebrickner (2001), we use backward induction on discrete grids with the help of Gauss-Hermite quadrature to calculate the expected value function. We utilize monotonicity of value function and the fact optimal policy takes the form of threshold of preference errors to reduce computational burden.

Step 1 discretization

Discretize the state space for the error term ν_t into equally spaced grids with lower and upper bound ν^L, ν^U such that for each period,

$$\rho\nu_t^L + \sqrt{2}\sigma m^1 > \nu_{t+1}^L,\ \rho\nu_t^U + \sqrt{2}\sigma m^p < \nu_{t+1}^U$$

where m^p is the p-th root for the Hermite polynomial $H_p(m)$ (there are p roots for the p-th order polynomial, namely, $m^1, m^2, ..., m^p$). [3] The weights corresponding to these roots are $w^1, w^2, ..., w^p$.

Step 2 backward induction

Step 2.1: $t = T$. This is the last period, comparing the two alternatives yields the optimal choice.

Step 2.2: Repeat for each $t < T$. First, note that the values for the grids in period $t+1$ are already given, for any grid point ν_t we can approximate the value for the j-th point of Gaussian quadrature, $\rho\nu_t + \sqrt{2}\sigma m^j$ by interpolation. Say the closest points to that point is ν_{t+1}^k and ν_{t+1}^{k+1}, and the associated value at these two points are $V(\nu_{t+1}^k)$ and $V(\nu_{t+1}^{k+1})$. By mean-value theorem, there exists a $\xi \in [0,1]$ s.t.:

$$V(\rho\nu_t + \sqrt{2}\sigma m^j) = \xi V(\nu_{t+1}^k) + (1-\xi)V(\nu_{t+1}^{k+1})$$

where in practice we can choose $\xi = \frac{\rho\nu_t + \sqrt{2}\sigma m^j - \nu_{t+1}^k}{\nu_{t+1}^{k+1} - \nu_{t+1}^k}$. Then, by Gaussian-Hermit quadrature, we approximate conditional expectation of the value function by

$$\mathbb{E}V(\nu_{t+1}|\nu_t) \approx \frac{1}{\sqrt{\pi}}\sum_{j=1}^{p} w^j V(\rho\nu_t + \sqrt{2}\sigma m^j).$$

[3] Because these roots are symmetric we only need to know the positive ones.

There are several advantages of Gauss-Hermite quadrature over equal spaced grids. The first is its low computation cost: before we find the critical values, with naive grids we need to compute weighted summation over n possible future scenarios. However, with Gauss-Hermite quadrature, we only need $3p$ times of evaluation (p for finding the nodes, $2p$ for calculate the value.) Second, we no longer need to calculate and save the transition matrix for the discretized markov process of ν, which could be prohibitively large (if grid size is 1000, the transition matrix is 1000 by 1000, hence we need evaluation around 500,000 integrals.)

Solving SW

Unlike solving the DP model, we can calculate the SW threshold directly with forward solution. For each $t \leq T$, we calculate $g_t(r)$ and $K_t(r)$ for all possible $r = t, ..., T$. Then we find the optimal $r_t^\dagger$ that maximizes the ratio $g_t(r)/K_t(r)$, and record the ratio as the SW threshold for time t.

2.3 Structural Estimation of Retirement Problems

As noted earlier, structural estimation generally involves solving a structural model for a given set of parameters and searching for parameters that maximizes the fit of the structural model to the data.

The traditional approach is based on repeatedly solving the structural model numerically as the optimization algorithm searches over the parameter space. The solution and estimation steps iterate until the gain from the search diminishes.

The deep-learning-aided approach involves additional steps in between of solution and estimation: instead of solving the structural model for each set of parameters dictated by the maximization algorithm of the estimation step, here we solve the model for a pre-chosen set of parameters to build a library that links each set of parameters to a solution, then we train the neural network through deep learning to obtain the optimal weights of activation. The trained neural network maps any set of state variables and parameters to a solution of structural model. Lastly, with this mapping, we estimate the parameters without the need to solve for the structural model while searching through the parameter space. In each step there are multiple options of implementation which could affect the computation time and accuracy. Appendix Table B.1 summarizes some of these options. We describe our choice for the numerical experiment in section 4.

2.3.1 Solution Step: Library Building

Assume that we are estimating a SW model. First, we build a library of solutions by selecting a collection of deep parameters $\boldsymbol{\theta}$ and state variables (a, e), solving for thresholds $f^{\dagger}_{a,e}(t; \boldsymbol{\theta})$.

We randomly draw a given number of age and exp cells based on their observed frequency in the data, pair each sets of deep parameters with one age-exp cell, and solve the model by backward induction to obtain thresholds $f^{\dagger}_{a,e}(t; \boldsymbol{\theta})$ for $t = 0, ..., T$.

2.3.2 Learning Step: Training DNN

We specify a multi-layer neural network for approximating the critical values as a function of state variable and parameters summarized in Table 2.1. The input is a 9-dimensional vector (3 for state variables, and 6 for structural parameters), and the output is a scalar.

2.3.3 Estimation Step: MLE

We estimate the model parameter $\boldsymbol{\theta}$ using a sample observations of teachers' decision with observed (a, e) in the initial year. We compute the likelihood of the

Table 2.1: Input and output of the DNN for threshold

Variable Type	Variable Name	Notation
Input	state variables	$s = (a, e, t)$
Input	structural parameters	$\boldsymbol{\theta} = (\beta, \gamma, \kappa, \kappa_1, \rho, \sigma)$
Output	critical values	$f^{\dagger}_{a,e}(t; \boldsymbol{\theta})$

sample from the probability of each observed choice. The parameter is estimated via maximizing the simulated likelihood.

Given the critical values sequence (subscript a, e omitted)

$$\mathbf{f}^{\dagger} = (f^{\dagger}(0), f^{\dagger}(1), ..., f^{\dagger}(T)).$$

and the AR(1) process

$$\nu_t = \rho \nu_{t-1} + \epsilon_t \text{ where } \epsilon_t \sim N(0, \sigma^2).$$

we calculate the retirement probability at year $t \in \{0, 1, ..., T\}$ by:

$$p_t = \Pr(\textit{retire at year } t) = \Pr(\nu_0 > f^{\dagger}(0), ..., \nu_{t-1} > f^{\dagger}(t-1), \nu_t \leq f^{\dagger}(t)).$$

Then denote the entire sequence of probabilities of retirement as

$$\mathbf{p}^{\dagger} = (p^{\dagger}_0, p^{\dagger}_1, ..., p^{\dagger}_T).$$

Note that this is the unconditional probability which can be log summed with the retirement counts for each year to obtain the sample likelihood. And the estimator is obtained by maximizing the sample (log-) likelihood. Since we need calculate the unconditional retirement probabilities for all possible combination of (a, e), an alternative to the GHK algorithm is the Discretization filter (Tauchen (1986), Adda et al. (2003), Farmer (2021)) which discretizes the AR(1) process into finite state

Markov chains. We apply the equal-probability discretization in Adda et al. (2003), with the details presented in Appendix B.3.

The algorithm for estimating the DP model is similar to the one above, except with $f_t^\dagger$ replaced by f_t^* for the same initial period (a, e). Also note that both DP and SW admit this critical value representation, hence we can use the same computation and approximation framework for both models.

2.4 Numerical Experiments

In this section we conduct numerical analysis on the teacher retirement problem. We simulate retirement data from a DP or SW model based on a given set of parameters. We then estimate parameters by MLE using traditional and deep-learning-aided approaches. Finally, we estimate a DP model where the true data are generated from a SW model, and vice versa, and study whether we can distinguish them from the true data-generating model.

2.4.1 Institutional Background and Specification of Data Generating Model

In a typical public school system teacher compensation are in two forms: salary and retirement benefit. We generate data based on a simplified version of the Illinois TRS teacher pension system. For active teachers, salaries are paid solely based on the experience/years of service one accumulated in the system. For data generation we assume senior teachers have master's degrees. The salary schedule is derived from regressing the log wage on a cubic function of experience. With the Illinois data, we obtain the following equation:

$$Y(e) = w_0 exp\{b_1 e + b_2 e^2 + b_3 e^3\}$$

where e is the experience, $Y(e)$ is the salary.[4] We assume teachers have full-time employment and accumulate one credit for each year of service. Each year a teacher can either keep working or claim retirement. Retirement is irreversible: once a teacher claims retirement, she can no longer return to teaching.

For retired teachers, pension benefits are determined by years of service one already accumulated in the system at the point of retirement. Table 2.2 summarizes age-and-experience dependent eligibility requirements for claiming retirement benefit in Illinois.

Table 2.2: Retirement Eligibility of Illinois TRS Teachers

Year of Service	Age	Description
5	62	Normal Retirement
10	60	Normal Retirement
20	55	At Reduced Rate
35	55	Normal Retirement

Note: the reduced rate is calculated as follows: Calculate the years before reaching age 60 and service experience of 35 years. The smaller one of the two times 6% equals the discounted annuity.

Note that experience is fixed after claiming retirement, hence the only post retirement the only time-varying variable is age. We can write the pension benefit as

$$B(a_t, e_t) = Q(a_t, e_t)b(e_t)$$

where $Q(a_t, e_t)$ is an indicator for retirement eligibility which depends on age and experience at the time of claim. The benefit formula is

$$b(e_t) = rep\_factor * e_t * FAS.$$

Here, the replacement factor rep_factor is the percentage of final salary for one year of

[4] The estimated coefficients are: $w_0 = 35,000$, $b_1 = 0.0402516942$, $b_2 = -0.0009063110$, and $b_3 = 0.0000061209$. And we assume salary grows at constant compound rate of 2.1% after $exp \geq 40$.

service. The replacement factor may depend on total years of service, e_t, as in Illinois before 1998. Here, we follow the Illinois rules post 1998 and assume the replacement factor is a constant, 2.2%. FAS is the final average salary, which by the Illinois rules is the average of the highest salaries over four consecutive years in last ten years.[5]

Data Generating Process

Based on the Illinois TRS pension rules we simulate 21,412 female teachers with initial age between 47-54, initial experience between 5-40, track their retirement decisions for seven years. To avoid unrealistic age and exp profile and to mimic patterns of observed data, we use the age and experience distribution of Illinois TRS members in 2006. We then calculate the threshold for each age-exp cell with parameter values at $(\beta, \gamma, \kappa, \kappa_1, \rho, \sigma) = (0.96, 0.70, 0.60, 1.30, 0.70, 0.31 \times 10^4)$. Finally, for each teacher, we generate an AR(1) process with $(\rho, \sigma) = (0.70, 0.31 \times 10^4)$ and record the retirement decision.

2.4.2 Solution to Two Models under the Same Set of Structural Parameters

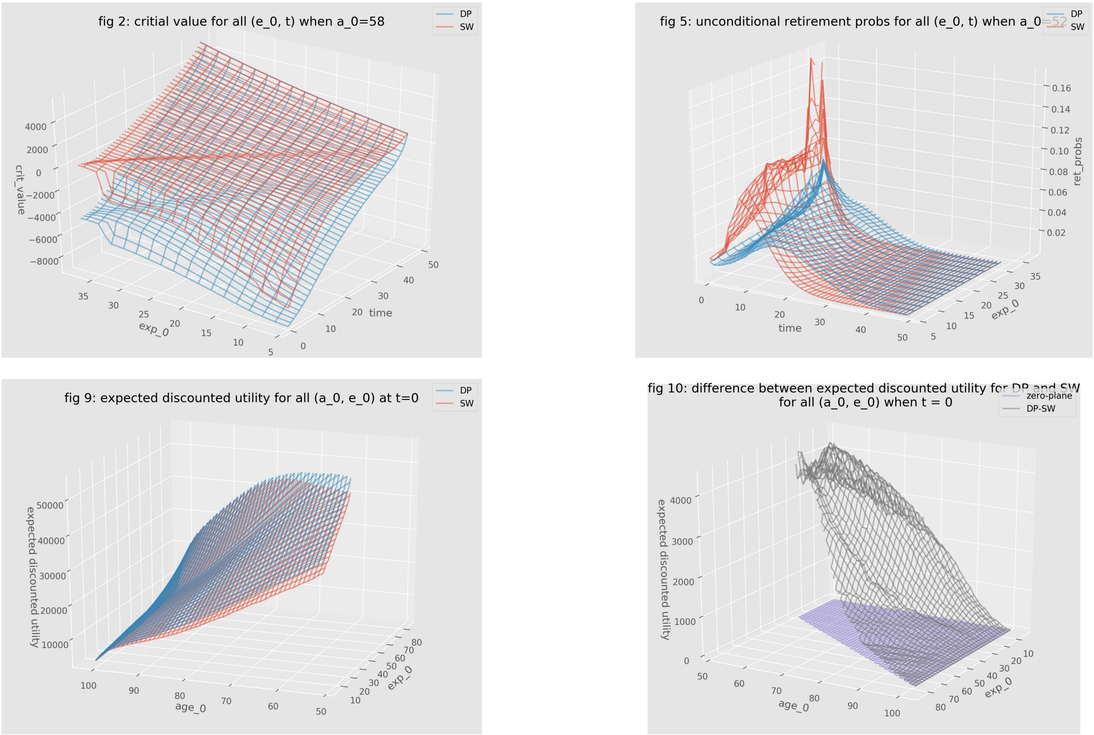
Panel (a) of Figure 2.1 plots the vector of thresholds for teachers with same initial age ($a = 58$ and different initial experience) under DP (blue lines) and SW (red lines), with the same set of parameters. The DP thresholds are always below the SW thresholds, and the gap decreases in teacher's age, as predicted by Propositions 1-3. Panel (b) plots the unconditional retirement probability of these teachers over time. It shows that based on the SW model teachers tend to retire earlier than in the DP model, and their retirement years are more concentrated.

Panel (c) plots the expected discounted lifetime utility for a typical teacher with

[5] Illinois pension rules also allow the option of actuarial calculation, which provide retirees the annuity based on the actuarial equivalent amount of how much they and their employers contributed to the system. We ignore this uncommon option in data generation.

different initial age and experience, based on DP and SW models. To compute the expected utilities we compare generated AR(1) errors against the thresholds under each model and record the predicted retirement decisions. Then we calculate the discounted sum of utilities and take the average. Due to its "sub-optimality", expected utility of the SW model is below that in the DP model. Panel (d) plots this difference in the expected utilities. The difference tends to be larger for younger teachers. For a teacher aged 60 with 30 years of experience, the difference is around 3,000. With $\gamma = 0.7$, the difference in welfare can be compensated by a one-time payment of about \$ 90,000 in 2010 dollars.

Figure 2.1: Comparison of SW and DP under same parameters



Note: Both models are solved under $(\beta, \gamma, \kappa, \kappa_1, \rho, \sigma) = (0.96, 0.70, 0.60, 1.30, 0.70, 0.31 \times 10^4)$. Panel a (upper left) shows the threshold as a function of initial experience (e_0) and time period t for a cohort of teachers with initial age $a_0 = 58$. Panel b (upper right) shows the unconditional retirement probability. Panel c (lower left) shows the expected value function, and panel d (lower right) shows the difference in expected value functions. In panel a-c, we use red color for SW and blue for DP.

2.4.3 Implementation of the Three-Step Procedure

Solution Step

Following the procedure in chapter 1, we first generate 3000 random sample of structural parameters from the following uniform distributions:

$$\beta \sim Unif(0.93, 0.98), \gamma \sim Unif(0.10, 1.00), \kappa \sim Unif(0.10, 1.00),$$
$$\kappa_1 \sim Unif(0.50, 2.00), \rho \sim Unif(0.10, 0.90), \sigma \sim Unif(2000, 4000).$$

Then, we randomly draw 3000 age and experience cells based on their observed frequency in the data. Next, we assign each sets of deep parameters with one age-exp cell, and solve the model by backward induction to obtain thresholds $f^{\dagger}_{a,e}(t; \boldsymbol{\theta})$ for $t = 0, ..., T$. Finally, we only retain the relevant thresholds (i.e., keep those with $t < T^*$) and store them in a table.

Learning Step

After obtaining the library of solutions (here, the thresholds), we specify a DNN for which the inputs and outputs are summarized in Table 2.1. We then train the DNN for 1,000 epochs. To improve the efficiency of training, we keep the data with thresholds in [-15,000, 15,000]. This is because for a Gaussian random variable, the probability of observing a realization over 5σ is around 10^{-6}, and for our numerical example, $\sigma = 3100$. In addition, instead of using the mean square error loss, we let the loss function be Gaussian weighted squared errors

$$L(y, \hat{y}) = (y - \hat{y})^2 \frac{\phi(y/\sigma)}{\phi(0)} = (y - \hat{y})^2 exp(-\frac{y^2}{2\sigma^2}).$$

This loss function weighs more heavily thresholds closer to zero. More details in training design and training results are reported in Appendix B.

Estimation step

We conduct maximum likelihood estimation with the Nelder-Mead algorithm. For each age-experience cell, following Adda et al. (2003) and Farmer (2021) we calculate the unconditional retirement probability for each year in the sample period through equal-probability discretization of the AR(1) process. The details are given in appendix B. Since the likelihood is approximated by discrete Markov chain, we experiment with the limit of tolerance of approximation errors and set it at 2.0.

2.4.4 Results with Correctly Specified Model

We first estimate the data-generating model–we first generate the sample of teacher retirement by the DP (SW) model and a set of structural parameters, and then estimate the structural parameters with the DP (SW) model.

Table 2.3: Computational time comparison for DP

method	step	task	time	details
conventional	solve	solve for all cells	131500s	532.5s per loop
	estimate	calculate likelihood	74s	0.3s per loop
		total	around 36h	
3-step	solve	build the library	7500s	3000 cells
	learn	train the DNN	201s	1000 epochs
	estimate	calculate likelihood	89s	0.3s per loop
		total	around 2h	

Note: to make the computation time comparable, we assume both methods need 247 iterations for the MLE to converge. We have 213 initial age-exp cells, and it takes 2.5 seconds to solve the DP for one cell, on average.

In Table 2.3, we compare the computation time of the conventional “solve to estimate” approach and the 3-step “solve, learn, then estimate” approach for the DP model. We use the GHK algorithm for a simulated estimator for likelihood. Unlike running a Bayesian MCMC where the length of the chain is pre-set, the number of iterations in MLE is random.[6] To make a fair comparison of estimation time we remove the difference due to different MLE iterations in both cases. We use the actual number of iterations for the 3-step procedure and rescale the computation time for the conventional approach. Both methods use the same solver of the DP problem, that solves for one age-experience cell in 2.5 seconds on average. The conventional method needs to solve for ALL cells in each iteration, and because the sample contains 213 cells of age-experience combinations, it takes the conventional method 532 seconds on solution, and the likelihood calculation takes negligible time of 0.3 second. In total, conventional method takes 36 hours to finish 247 iterations.

The three-step method only solves the model for 3,000 cells under 3,000 different parameter values, which takes 7,500 seconds if they are solved sequentially. And the time can be further decreased if we do parallel computation. The training of DNN takes 201 seconds. And with the trained DNN, estimation only takes 89 seconds for 247 iterations, with the majority cost for likelihood calculation. The cost for obtaining a solution of the DP model from trained DNN is negligible.

Table 2.4 presents the mean and standard deviation of the estimates of structural parameters for 100 repetitions. In each repetition, we first simulate a random sample of teachers with the same age-exp distribution, then estimate sturctural parameters of the DP model by the three-step procedure. Note that in each repetition, the library we build in the first step and the neural network we trained in the second step may differ. The mean for each parameter is close to the true value, and the standard deviation is relatively small. This suggests the three-step procedure recovers the true parameters accurately.

Tables 2.5-2.6 are the counterparts of Tables 2.3-2.4 for SW. The main findings are similar: the three-step procedure is faster than the conventional method, and it can accurately recover true parameters. Because it is much faster to solve for SW than

[6] For estimation, commonly used algorithms include gradient descent, Newton's method and stochastic gradient descent.

Table 2.4: Accuracy for 100 repetitions under DP

	β	γ	κ	κ_1	ρ	$\sigma \times 10^4$
TRUE	0.9600	0.7000	0.6000	1.3000	0.7000	0.3100
mean	0.9598	0.6918	0.6075	1.3181	0.7105	0.3117
std.dev	0.0087	0.0180	0.0210	0.0409	0.0250	0.0126

for DP, the computational advantage of the three-step method for the SW model is smaller than it computational advantage for the DP model. For the SW model the total computation time of conventional method is around 5 times of that of the three-step method. For the DP model the total computation time of conventional method is around 20 times of that of the three-step method.

Table 2.5: Computational time comparison for SW

method	step	task	time	details
conventional	solve	solve for all cells	1893s	7.7s per loop
	estimate	calculate likelihood	74s	0.3s per loop
		total	around 2000s	
3-step	solve	build the library	108s	3000 cells
	learn	train the DNN	201s	1000 epochs
	estimate	calculate likelihood	89s	0.3s per loop
		total	around 400s	

Note: to make the computation time comparable, we assume both methods need 247 iterations for the MLE to converge. We have 213 initial age-exp cells, and it takes 0.036 seconds to solve the SW for one cell, on average.

Table 2.6: Accuracy for 100 repetitions under SW

	β	γ	κ	κ_1	ρ	$\sigma \times 10^4$
TRUE	0.9600	0.7000	0.6000	1.3000	0.7000	0.3100
mean	0.9622	0.6958	0.6138	1.3065	0.6999	0.3075
std.dev	0.0103	0.0072	0.0372	0.0818	0.0204	0.0195

Finally, Table 2.7 reports the MLE estimates for one sample (using the true parameters as the initial guess.) Columns (2) and (4) reports the estimated structural parameters of the data generating models (SW model for Column (2) and the DP model for Column (4)). In both cases, MLE recovers the true parameters with small standard errors.

Table 2.7: MLE results for one sample

	(1)	(2)	(3)	(4)	(5)
model	true	sim SW, est SW	sim SW, est DP	sim DP, est DP	sim DP, est SW
β	0.9600	0.9598	0.9306	0.9617	0.9507
		(0.0006)	(0.0010)	(0.0003)	(0.0004)
γ	0.7000	0.7001	0.6911	0.6982	0.7665
		(0.0009)	(0.0006)	(0.0008)	(0.0001)
κ	0.6000	0.6041	0.6004	0.6059	0.5769
		(0.0016)	(0.0027)	(0.0002)	(0.0012)
κ_1	1.3000	1.3384	1.3139	1.3107	1.2694
		(0.0040)	(0.0035)	(0.0059)	(0.0009)
ρ	0.7000	0.7006	0.7287	0.7013	0.7657
		(0.0014)	(0.0014)	(0.0012)	(0.0003)
$\sigma \times 10^4$	0.3100	0.3125	0.3179	0.3113	0.2953
		(0.0007)	(0.0025)	(0.0008)	(0.0021)
log-lik		-32847.9420	-33925.7708	-15716.2422	-16332.2856

Note: "sim SW, est DP" means the sample is simulated under the SW solution with the true parameters, and then estimated under DP. "sim DP, est SW" means the sample is simulated under the DP solution with the true parameters, but then estimated under SW.

2.4.5 Model Identification

We now examine whether the deep-learning-aided estimation enables identification of similar structural models: can we tell whether the retirement data are generated from DP or SW?

In column (3) of Table 2.7, the sample is simulated under the SW model with the true parameters, but we estimate the parameters through a DP model via the three-step method. As is expected, with model mis-specification the estimates for structural parameters differs from the true values. The sample log-likelihood of the

Table 2.8: 100 repetitions for mis-specified models

Case 1: sim SW, est DP						
	β	γ	κ	κ_1	ρ	$\sigma \times 10^4$
TRUE	0.9600	0.7000	0.6000	1.3000	0.7000	0.3100
mean	0.9331	0.6410	0.4801	1.4685	0.5940	0.3510
std.dev	0.0061	0.0326	0.1418	0.2720	0.1179	0.0450
Case 2: sim DP, est SW						
	β	γ	κ	κ_1	ρ	$\sigma \times 10^4$
TRUE	0.9600	0.7000	0.6000	1.3000	0.7000	0.3100
mean	0.9532	0.7516	0.6262	1.2239	0.7726	0.2877
std.dev	0.0181	0.0135	0.0678	0.1555	0.0374	0.0382

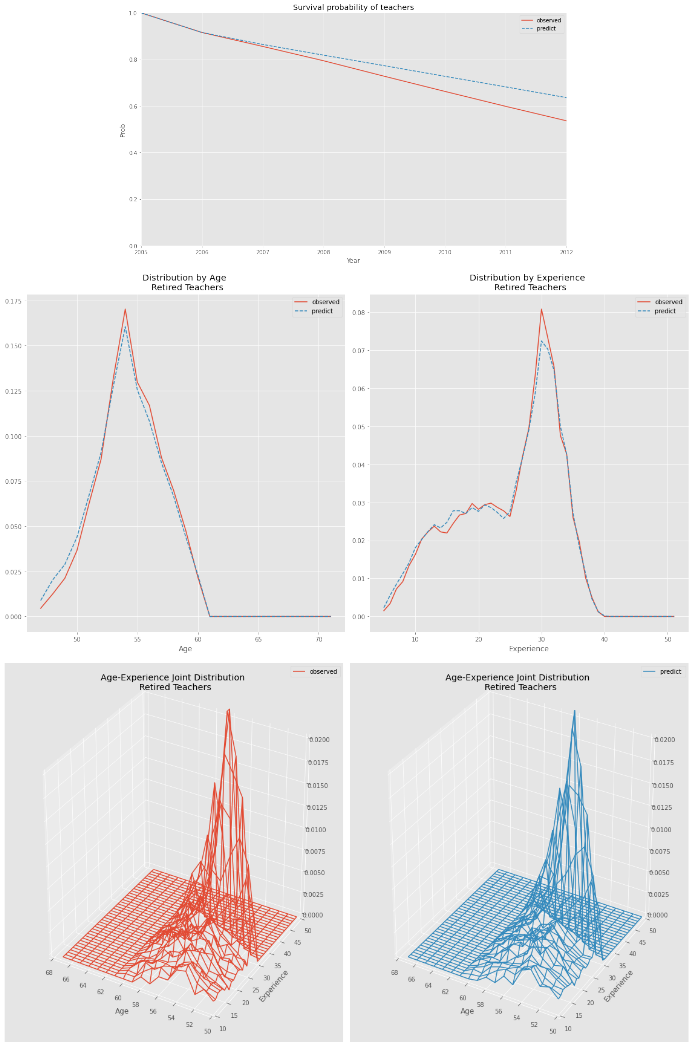
Note: “sim SW, est DP” means the sample is simulated under the SW solution with the true parameters, and then estimated under DP. “sim DP, est SW” means the sample is simulated under the DP solution with the true parameters, but then estimated under SW.

mis-specified is significantly lower than that of the true model. Similar results are found in column (5), where the sample is simulated under SW but we estimate it by DP. We then run 100 repetitions for the above two mis-specified models, and report the mean and standard deviation of parameter estimates in Table 2.8. Not surprisingly, the parameter estimates from mis-specified models differ from their true values.

As for structural estimation, we are more interested in whether the mis-specified model (with the corresponding “mis-estimated” parameters) makes similar predictions of teacher’s retirement behavior as the correctly specified model. For the sample simulated by SW and estimated by DP, we further generate a sample with DP under the estimated values of parameters, and plot some summary statistic in Figure 2.2.

The upper panel of Figure 2.2 is the survival rate over time, where the red solid line is for the generated data, and the blue dashed line is for the predicted data. The middle panel is the marginal distribution of age and experience of retired teachers. The lower panel is the joint distribution of age and experience of retired teachers. The

Figure 2.2: Simulated by SW, Estimated with DP



Note: the upper panel plots the survival rate. The middle panels plot the marginal distribution of age (left) and experience (right) of retired teachers (at the time of retirement). We use red solid line for the simulated data and blue dashed lines for the predicted data with mis-specified model and the corresponding parameter estimates. The lower panels plot the joint distribution of age and experience of retired teachers (at the time of retirement), where the red lines in the left is for the simulated data and the blue lines in the right for the predicted data.

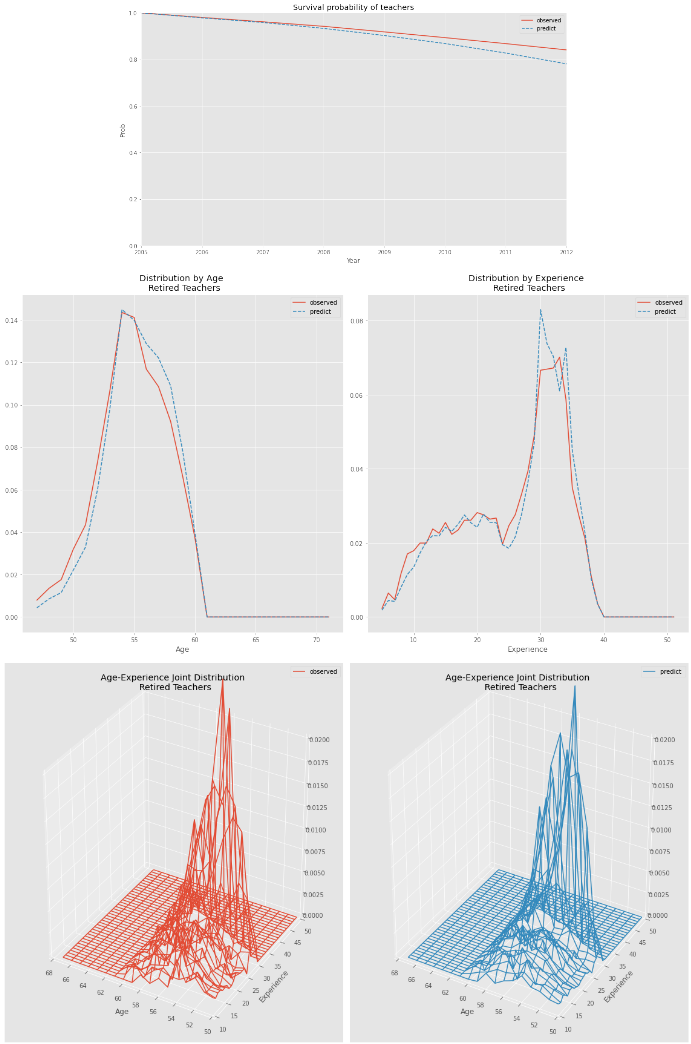
left one is for the generated data, and the right one is for the predicted pattern. The predicted survival rate is above the true one–the estimated DP model under-predict retirement generated by the SW model. The estimated model mimics the spikes in the marginal distribution of age and experience of retired teachers, but the spike is not as large as the true model. The joint distribution of age and experience of the mis-specified model exhibit a qualitatively similar pattern as the generated data.

Figure 2.3 presents the results for data generated by the DP model, and estimated by the SW model. First, comparing the true models for SW and DP in Figure 2.2 and 2.3 shows that the survival rates are higher under DP than SW, given the structural parameters. Moreover, the marginal distribution of age and experience under DP is not as concentrated as SW. Next, we focus on Figure 2.3. The survival rate of the true model generated by the DP model is higher than the estimated model with the SW model. The marginal distribution of age and experience is sharper under the estimated model, and the joint distribution under the estimated model is more concentrated.

2.5 Concluding Remarks

This chapter focuses on structural estimation of DBCM in finite horizons. We consider teachers' optimal retirement problem. First, we show that two popular models, DP and SW, both admit the threshold strategy under our general framework, and the threshold for DP is always higher than SW under the same set of parameters, so teachers retire earlier under SW. Second, we modify the three-step procedure in Norets (2009) to a simulated sample of teachers. We achieve reduction in computational time around 20-fold for DP and 5-fold for SW with respect to the conventional nested algorithm, without significant accuracy loss. Lastly, we show that our method is able to identify the subtle difference between DP and SW from the fit of the survival rate and distribution of age and experience of retirement teachers.

Figure 2.3: Simulated by DP, Estimated with SW



Note: the upper panel plots the survival rate. The middle panels plot the marginal distribution of age (left) and experience (right) of retired teachers (at the time of retirement). We use red solid line for the simulated data and blue dashed lines for the predicted data with mis-specified model and the corresponding parameter estimates. The lower panels plot the joint distribution of age and experience of retired teachers (at the time of retirement), where the red lines in the left is for the simulated data and the blue lines in the right for the predicted data.

Chapter 3

Behavior Responses to Voluntary Pension Upgrades: Structural Estimation for Illinois Public School Teachers

3.1 A Framework for Evaluating Effect of Pension Rules

During the 1990s, many states enhanced pension benefits for public K-12 teachers, sometimes multiple times over several years. In the state of Illinois, defined benefit (DB) pension costs for K-12 public school teachers have been rising in the past decades and created severe fiscal pressure.

> *" In the current school year, 36% of the money the state allocates to education will be diverted to pension payments. This represents a 200% increase in spending on teacher pensions since 2000, compared with only a 20% increase on classroom spending during that period. Despite massive funding increases and diminishing social services, the five state-run retirement sys-*

> *tems – which serve teachers, state workers, public university employees, judges and members of the Illinois General Assembly – are only about 40% funded, a shortfall of $137 billion by the state's accounting."* [1]

Given the extent of the impending pension crisis major policy interventions may be necessary.[2] Assessing the labor supply and fiscal effect of major pension reforms requires predicting outcomes of policies that generally have not been implemented in the past, thus precluding the use of conventional policy evaluation tools.

Analysis of future policies should be based on economic models that can fit historical data of teacher retirement under different pension rules. Before entrusting an economic model for predicting teacher behavioral responses to any proposed reforms, one should verify whether the model can explain the teacher retirement behavior under the current pension rules as well as the change in retirement behavior induced by the pension enhancements in the 1990s.

Evaluating the effects of the pension enhancements on teacher retirements is challenging. Under DB pension rules, financial incentives in retirement decision are functions of teachers' age and experience. Our previous empirical research suggests unobserved heterogeneity in preference among teachers plays an important role in teachers' retirement decision and the aggregate retirement rate. As multiple enhancements occurred sequentially, earlier enhancements affected the population exposed to later enhancements. Evolution of distribution of the preference of teachers who remain teaching given age and experience depends on the historical policies. Ignoring this dynamics of sample selection results in biased estimates of policy effects.

[1] From `https://www.illinoispolicy.org/illinois-pension-costs-debt-are-growing-far-faster-than-state-predicted/` and `https://www.illinoispolicy.org/reports/pensions-vs-schools/`.

[2] Unfunded pension liabilities have generated calls for reforms of DB plans (see Costrell and Podgursky (2009), Brown (2013), Costrell and McGee (2010), Fitzpatrick and Lovenheim (2014), Friedberg and Turner (2010), Knapp et al. (2016), Ni and Podgursky (2016), Novy-Marx and Rauh (2011), Malanga and McGee (2018), Doherty (2012), Backes et al. (2016), Kim et al. (2021), among others).

In addition, the full impact of a policy's effect on retirement may take a long time to materialize. This means the short-term effect may differ from the long-term effects: in the short term the effects of a policy change depends on current distribution and for the long term the full effects of a policy change may take long time to materialize because the retirement decision occurs in the future.

With these challenges in mind, we propose a structural model approach to assessing the effect of any given policy rule on teachers' retirement decision. While the discussion focuses on pension enhancements and teacher retirement, the framework applies to evaluation of policy changes in other contexts. The structural model approach is based on a dynamic programming model of retirement in which teachers make retirement decisions by maximizing an objective function given the pension rules and with time-varying unobserved heterogeneity. We first estimate a set of "structural parameters" that quantify the nature of teachers' preferences (such as teachers' willingness in delaying receiving income, and preference towards risk). The structural parameters are independent of pension rules. We then use estimated structural parameters to predict teachers' decisions when facing a different pension rules.

In application of the model for the teachers in the sample, we not only account for the dynamic dependence of unobserved heterogeneity as teachers make retirement decisions over time, but also the initial unobserved heterogeneity, based on the and and experience of the teachers in the initial sample year. The structural approach is desirable for policy evaluation because the estimated structural parameters are applicable beyond the data sample. Another advantage of the structural approach lies in its ability predicting new policies that differ from those implemented in the past. Because experimenting on pension policies are costly and time consuming, guidance from economic models are particularly useful. The ultimate goal of the empirical research on pension effect on retirement is guiding policies in pension reform. A structural model is suitable for that purpose.

A structural model is based on restrictive assumptions and omits empirically important factors. These limitations likely render misfit of data. However, structural models produce often produce good out-of-sample fit, suggesting they capture the key factors in the retirement decisions.

Two types of structural models, option value models and dynamic programming models, have been used to model optimal retirement decision (see chapter 2 for more details). In option value models of Stock and Wise (1990) and Ni and Podgursky (2016), time-varying unobserved heterogeneity is modeled as AR(1) preference errors. The option value model used in these studies can be more easily solved in a forward-looking fashion. However, an option value model only values the option of retirement of the current but ignores the value of of having such an option in future years, which makes it unsuitable to model optimal sequential decisions by teachers in some settings. In contrast, dynamic programming models account for the value of future options but need to be solved through backward induction, which is computationally more burdensome. Some studies use dynamic programming models but make simplifying assumption on the unobserved heterogeneity. For example, Knapp et al. (2019) construct a structural model of retirement with independent and identical generalized extreme value distribution (i.i.d. GEV) of per-period error term with a random effect of time-invariant heterogeneity. Empirical evidence suggests serial correlation is essential in modeling policy-dependent heterogeneity.

Estimation of the dynamic programming model with serially correlated preference errors is computationally costly. A key innovation of this dissertation (chapter 1) is proposing a machine learning algorithm for reducing the computation time of estimating structural models. We estimate a dynamic programming model with AR(1) preference errors using a modified three-step procedure and deep neural networks (i.e., algorithm 3 in chapter 1, which is tested with simulated retirement data in chapter 2).

For empirical application, we estimate the structural parameters in a dynamic programming model using administrative data of 2005-2012 from Illinois public school teacher's retirement system. In the dynamic programming model, teachers' retirement decision is driven by pension incentives, structural parameters defined utility function, and serially correlated unobserved preference errors. With the structural parameters we can calculate the distribution of preference errors for given a combination of age and experience, and for in-sample test, the probability of retirement of the 2005-2012 sample. Then as an out-of-sample test, we use the estimated model to evaluate the effect of the 2.2 upgrade option implemented in 1998 that offers the option to an earlier sample of Illinois teachers to upgrade pre-1998 service credits to a more generous formula with 2.2% flat rate of pension benefit for each year of service. As an example of history-dependence of sample distribution, in estimating the effect of the "2.2 upgrad" we take into account of another enhancement, the Early Retirement Incentives (ERI) during 1993-1995 that allowed teachers with at least five years of service to purchase additional five years of age and service, conditional on them retiring immediately. Note that teachers' decisions on both ERI and "2.2 upgrade" are based on the unobserved heterogeneity, and teachers who faced the "2.2 upgrade" decisions were a selected sample that declined the offer of ERI.

For policy analysis, we focus on estimating the effect of 2.2 upgrade. We find that around 87% of teachers in the 1998 sample took the upgrade. By matching teachers with the same age and experience profile to an earlier cohort, we find that takers retired around 1.3 years earlier on average, with a substantial variation across different age-exp cells.

3.2 Illinois TRS Rules and Data Sample

3.2.1 Illinois Pension Rules

Before discussing the pension upgrade in 1998, we first summarize the pension rules of public school teachers in Illinois. Illinois public school teachers are enrolled in TRS (except those in Chicago, who are enrolled in Chicago Teachers' Pension Fund, CTPF). During the time of employment, teachers contribute a proportion of their salary to the defined benefit (DB) pension system, matched by the employer. Eligibility for pension is based on combinations of age-experience (accumulated service credit) requirement, see Table 3.1.[3]

Table 3.1: Retirement Eligibility

Year of Service	Age	Description
5	62	Normal Retirement
10	60	Normal Retirement
20	55	At reduced rate: 6% for each year under 60; or under ERO*
35	55**	Normal Retirement

Note: * ERO stands for Early Retirement Options.
** If the retirement annuity is at least 74.6 percent of the final average salary and the teacher will reach age 55 between July 1 and Dec. 31, TRS considers him/her to have attained age 55 on the preceding June 1. Moreover, if a teacher meets some criteria of the state of Illinois, he/she can also apply for rule of 85.
Source: TRS (2018a), TRS (2018b).

A vested member (i.e., accumulated five years of service in the system) who do not satisfy the above requirements may separate first and wait until qualified. For those not vested members, refund option is also available.

[3] The TRS teachers are classified into several tiers of pension plans. Before September 2018, there were two tiers: Tier 1 (2) are those first contributed to TRS before (after) Jan. 1, 2011 or (and) have (no) pre-existing creditable service with a reciprocal pension system prior to Jan. 1, 2011. Tier 1 and Tier 2 members are covered by defined benefit plans of different parameters. In September 2018, a new group, Tier 3, was created. Tier 3 is a hybrid of defined benefit and defined contribution plans.

Pension Benefit

Upon claiming retirement, TRS will calculate the pension benefit according to two approaches: experience-based formula and actuarial calculation, and use the one gives the higher amount. The experience-based formula is

$$B = r * exp * FAS$$

where r is the replacement factor which equals 2.2% after 1998, exp is the credible service year accrued by the member. According to Chapter 5 of TRS (2018a), credible service years includes regular service, sabbatical leave, sick leave, optional service and reciprocal service, among others. [4] FAS is the final average salary, which is the average of four consecutive annual salaries among the last ten years.

3.2.2 Sample for Structural Estimation

This paper relies on the data from TRS. It has two types of data: i) personal data such as name, age/age at claim, date hired, contribution accumulation, total service credit, claim date, claim type, etc. ii) Payroll and service credit data, which record the employment status, employer name, salary, earnings, service credit, and days paid for every TRS member for each fiscal year. Sufficient statistics including individuals' From the data we can compute age, experience, gender, and retirement decision.

The 2005-06 cohort is used for estimating deep parameters for structural models. We construct the sample of teachers who satisfy the following criterion: 1) Work full time in 2005-2006 school year. That is, service credit equals 1 in that year. 2) Have 5 or more years of experience at the end of 2005-2006. i.e., total service credit ≥ 5 at

[4] TRS has a series of rule of classify which service years are credible; moreover, members could also purchase optional service year if qualified; and upon retirement, unused sick leave credit (max 2 years) also counts towards total credible service years. For more details, check the TRS member guide at `https://www.trsil.org/members/retired/guide`.

that time. 3) Age between 47 and 54 at the end of 2005-2006. 4) Work full time from 2005-2006 until retirement. These four restrictions balances maximize observations and make it possible for cell-based estimation.

Table 3.2 shows the retirement pattern over time of 27,299 teachers, with average age of 51.01 and average experience of 20.81 at the end of school year 2005-06. 516 teachers retired in July 2006, and the number jumped to 1,340 in 2007, back to 949 in 2008, then rose steadily from 1,377 in 2009 to 2,535 in 2012. There were 17,146 teachers remaining active in school year 2012-13. Among all teachers in 2005, 22% or 5,887 were male, and male teachers are more likely to retire, which results in a fewer male share (19%) among active teachers in 2012.[5]

Table 3.2: Basic retirement facts

	# Teachers	Age	Exp	# Male Teachers	Male Share
all_teachers_in_2005-2006	27,299	51.01	20.81	5887	0.22
retire_in_July06	516	52.75	24.96	149	0.29
retire_in_July07	1,340	54.39	29.25	373	0.28
retire_in_July08	949	54.83	29.08	254	0.27
retire_in_July09	1,377	55.69	30.36	378	0.27
retire_in_July10	1,512	56.30	30.72	386	0.26
retire_in_July11	1,924	56.88	30.72	537	0.28
retire_in_July12	2,535	57.75	29.76	557	0.22
not_retired_in_2012-2013	17,146	57.21	24.71	3253	0.19

Note: The sample includes teachers who earn full credit (=1) at 2005-2006 school year, with total service credit (TSC, not including unused sick leave credit) at the end of 05-06 school year greater than or equal to 5, with age between 47 and 54.

3.3 A Dynamic Programming Model of Retirement

As in chapter 2, we set up a dynamic programming model with structural parameters as $(\beta, \gamma, \kappa, \kappa_1, \rho, \sigma)$. For the threshold strategy and the three-step algorithm, see

[5] We define the time of retirement as the date a teacher separate from her work, which is not necessary the time when she claims pension benefits. Teachers retire under various retirement plans.

the discussion there for more details.

3.3.1 Modeling the endogenous distribution of preference errors

Denote the optimal policy $f_t^{\dagger}$ by the threshold preference error $f^{\dagger}(a,e,t) = \nu^*_{(a,e)}(t)$, i.e., the decision rule is that a teacher with preference error (for teaching) ν_t retires in t if $f^{\dagger}(a,e,t) \leq -\nu_t$, and continue to teach otherwise. The threshold of the preference error in period t, f_t, depends on the set up of the decision model, the model parameter vector b (that determines the teacher's preference), teacher's observables related to retirement decision (such as age and experience) in period t, $\mathbf{s}_t$, and the pension rules R. To make the dependence implicit we write the threshold condition as a model of latent variable

$$y_t^* = f(\mathbf{s}_t, R_t, R_{t-1}...) + \nu_t, \qquad (1)$$

where $d_t = 1$ if $y_t^* \leq 0$; $d_t = 0$ if $y_t^* > 0$. By definition, $y_t^* \leq 0$ iff $\nu_t \leq -f^+(\mathbf{s}_t, R_t, R_{t-1}...)$. The key challenge is to identify the threshold function $f^+(\mathbf{s}_t, R_t, R_{t-1}...)$.

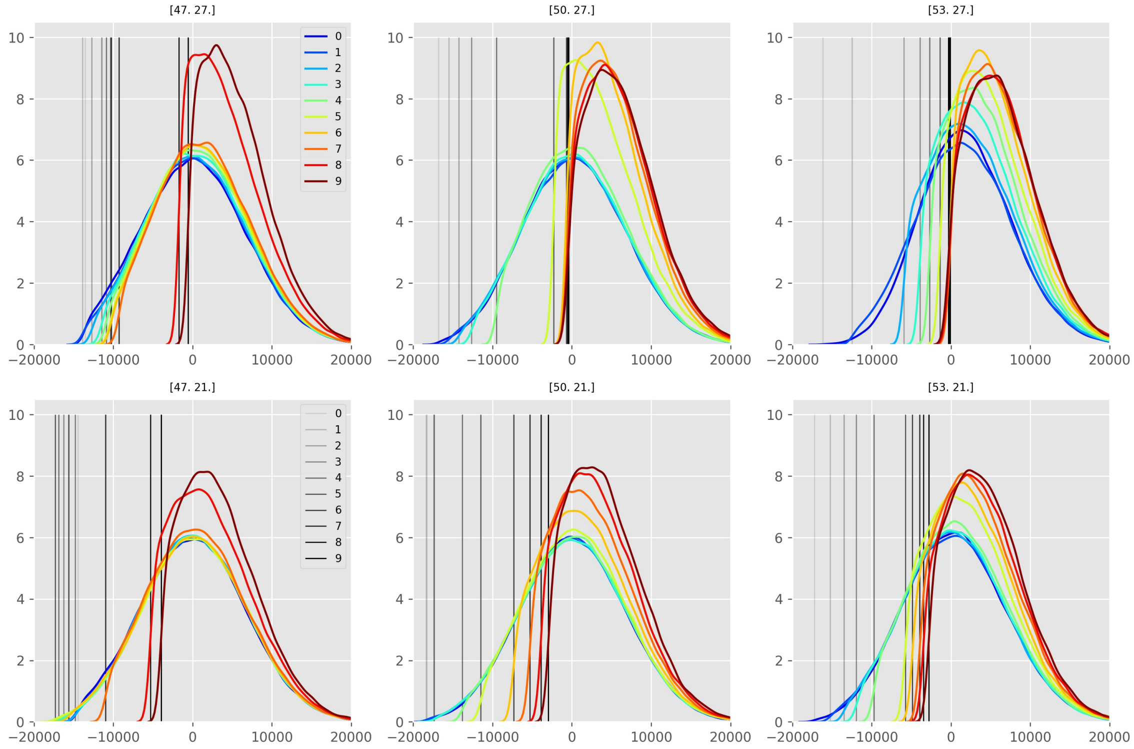
Teacher i's history is given by $\mathbf{d}_i = (0,..,0,1)$ (with $n_i - 1$ 0's before the number 1) we assign a sequence of latent variables in (1) with $y_t^* > 0$ for $t = 1, 2, .., n_i - 1$ and $y_t^* \leq 0$ for $t = n_i$. The structural model depends on the assumptions on teacher preference and the environment in which teachers make retirement decisions. These assumptions imply restrictions in (1).

In making decision in period t, the teacher faces preference error ν_t drawn from distribution conditional on the history up to t: $F(\nu_t| - \nu_j < f_j^+ \; for \; all \; j < t)$. The history of the teacher includes the historical pension rules and realized preference shocks. With a given structural model we can track the thresholds.

An empirical indication of a shift in the threshold f^+ is that retirement behavior shifts for a given $\mathbf{s}$. If over time the retirement decisions of different cohorts of teachers with the same age-experience combination (say $age = 47, exp = 27$, see the top left panel of Figure 3.1) change then the there is an underline shift in thresholds f. The shift in the thresholds implies the distribution of ν_t differs by cohorts.

The key to implementing this approach is the structural model that produces the thresholds f^+ for given the state variables and policy history.

Figure 3.1: Evolution of preference shock across time



Note: 1) we plot the distribution of the preference shock in each period for the remaining teachers immediately after they decide whether to retire. The top left panel is for teachers with initial age of 47 and exp of 27, the top middle panel for 50 and 27,etc.
2) The distribution is generated by simulating 100,000 paths for each age-exp combination, and trace them for 10 years. The vertical lines corresponds to the thresholds, and only those whose preference shock to the right of threshold remain.
3) We use the DP model with estimated structural parameters for the 2005-12 cohorts in Table 3.3 to generate these thresholds.

3.3.2 MLE of Structural Parameters using the 2005-12 Sample

We apply the three-step procedure to the current problem. Because we do not observe the date of birth, we specify the likelihood function as a finite mixture of two possible cases: i) age is correctly measured; ii) true age is 1 year older. We run MLE corrected for measurement error: denote the unconditional probability of a teacher with initial age and exp (a, e) retires at period t as $p(a, e, t)$. Assume that age is correctly measured with probability λ, then we would observe a teacher with (a, e) retiring at t with probability $\lambda p(a, e, t) + (1 - \lambda)p(a + 1, e, t)$. We set $\lambda = 0.5$ and the MLE results are reported in Table 3.3.

Table 3.3: MLE estimates of structural parameters for Illinois data 2005-2012

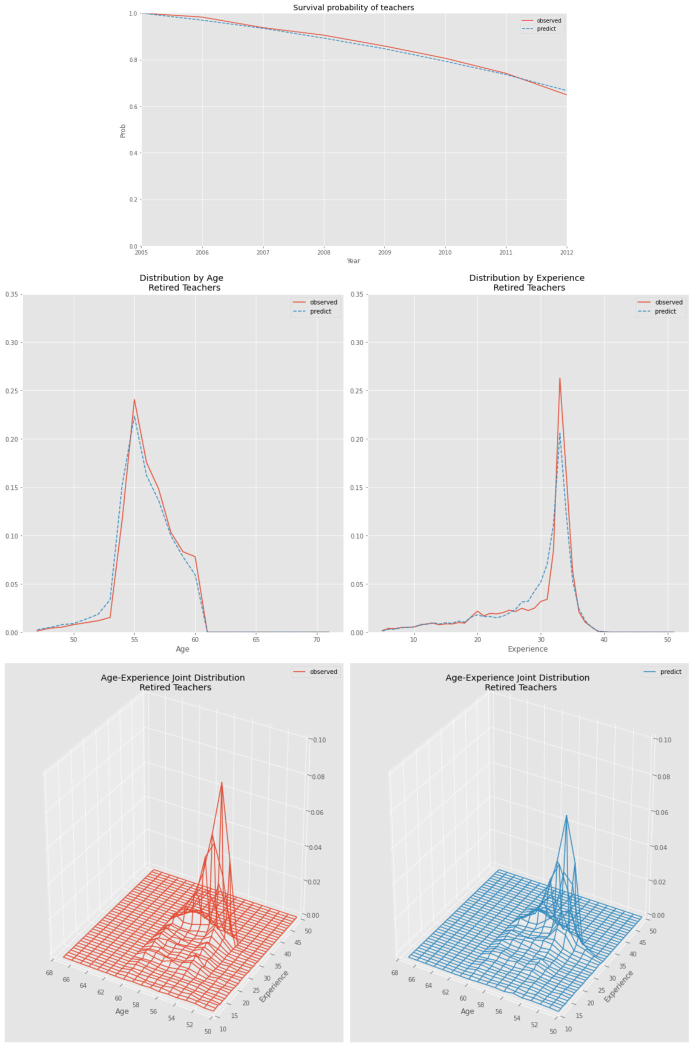
Parameters	β	γ	κ	κ_1	ρ	$\sigma \times 10^4$
	0.9304	0.7907	0.3874	1.5438	0.6981	0.3488
	(0.0002)	(0.0003)	(0.0013)	(0.0005)	(0.0026)	(0.0004)
log-lik	-22959.1596					

Note: The estimation is based on female teachers only in the 2005-2012 sample. Standard errors are in parentheses, which are obtained by inverting the numerical Hessian matrix of the log-likelihood. Corrected for: i) measurement error in age; ii) left-censoring, and iii) with a fixed effect for year 2008.

3.3.3 In-sample Goodness of Fit

We report the predicted survival rate, age-exp distribution for retired teachers (both marginal and joint distributions) in Figure 3.2. The dynamic programming model provides reasonable fit to the observed data. It tracks the observed survival curve very well. The model under-predicts the spike at 33-34 in the marginal distribution of experience of retired teachers. As a result, the spike of the joint distribution at (55, 33-43) is also under-predicted. Overall the estimated model provides a good fit in-sample.

Figure 3.2: DP for Illinois data



Note: the upper panel plots the survival rate. The middle panels plot the marginal distribution of age (left) and experience (right) of retired teachers (at the time of retirement). We use red solid line for the observed data and blue dashed lines for the predicted data with estimated parameters. The lower panels plot the joint distribution of age and experience of retired teachers (at the time of retirement), where the red lines in the left is for the observed data and the blue lines in the right for the predicted data.

3.4 Policy Analysis of the "2.2 upgrade" as an Out-of-Sample Test

Teachers can raise the replacement factor for their service years prior to July 1, 1998 to 2.2% by paying a cost. We call it the "2.2 upgrade". The 2.2 upgrade in

1998 only concerned Tier 1 members. Pensions of Tier 1 teachers experienced several waves of pension enhancements. The 2.2 upgrade was one of them.

After 1998, teachers were given the option to upgrade their replacement rate for service years prior 1998 to 2.2 percent by paying a price equal to:

$$P_{it} = \min\big(\frac{Exp_{1998}}{100}, \frac{20}{100}\big) \times Salary_{it}$$

where Exp_{1998} denotes the service years a teacher has earned before 1998, and $Salary_{it}$ is the highest salary rate during the four school years before the teacher applies to make the upgrade contribution. Typically, this is the salary of the teacher at the time of payment. This means that the price of upgrade is approximately 1% for each year of service before 1998, with the total capped at 20% of annual salary.

It is important to note that with or without the upgrade, the experience-based retirement annuity is capped at 75 percent of final average salary. Under the old formula without the upgrade, teachers hit this cap at 38 years of service. Under the new formula with the upgrade they hit it at 34 years. Thus the gain in pension wealth from the upgrade declines after 34 years and hits zero at 38 years. This means that teachers who planned on longer careers and retirement at a later age receive smaller or zero benefits from the upgrade.

3.4.1 Data of the Response to "2.2 Upgrade"

The sample for out-of-sample test is a cohort of 19,126 active teachers with 22-28 years of experience at the end of 1997-98 school year. This is the same cohort we built in Ni et al. (2022). The sample was initially studied by Fitzpatrick (2015), but Ni et al. (2022) had the service credit more accurately measured.

Among teachers with 22-28 years experience in 1998, 87% purchased the 2.2 upgrade by 2019. We label them "takers". The rest of the group, "non-takers", did not purchase by 2019. Teachers with at least 22 years experience in 1998 and still working in 2014 are considered non-takers. We find the take-up decision on the upgrade strongly correlated with net realized pension benefit.

Figure 3.3 shows the observed age and experience distribution for takers and non-takers in 1998 and at retirement claim date. All of them suggest a strong separating pattern, where takers retire earlier and enjoy positive net benefits while non-takers retire later and have negative net benefit of upgrade.

The upgrade status and retirement timing for teachers are correlated with initial experience in 1998. Senior teachers take the upgrade more, and almost all teachers above 26 years of experience retire by 2012. Among those with 22 years of experience in 1998, 90% retired by 2012. Those still working in 2012 are non-takers (since if they work full time, their service credit is at least 36 years, not including sick leave service. Hence the 2.2 upgrade would be worthless for them.)

3.4.2 A Dynamic Programming Model with "22 upgrade" as a Recurring Option

In the beginning of each period, a teacher makes two decisions in sequence immediately before observing the preference shock: a) whether to purchase the upgrade, if she has not purchased yet; and b) whether to retire, given the current upgrade status.

We first set up the Bellman equations for retirement. This decision depends on whether the teacher already purchased the "22 upgrade". Consider a teacher with initial age-exp (a, e) who has already purchased the upgrade by period t, the only decision is to retirement or not. Let $W(a, e, t, 1)$ be the expected discounted lifetime utility of pension for the upgraded benefit, and $V(a, e, t, 1, \nu_t)$ the value function:

$$V(a, e, t, 1, \nu_t) = \max\{u(a, e, t) + \nu_t + \beta EV(a, e, t+1, 1, \nu_{t+1}), W(a, e, t, 1)\}.$$

The first term on the right-hand-side is the value of teaching for at least one more year with the upgraded pension, the second term is the value of retiring now. By backward induction, there exists a threshold $\nu^*(a,e,t,1)$ such that the teacher retires at period t (conditional on not retired yet) if and only if $\nu_t \leq \nu^*(a,e,t,1)$. We can then solve the model using this threshold property and obtain the value function $V(a,e,t,1,\nu_t)$.

The utility function $u(a,e,t)$ for period t is the same one in the previous section, $(\kappa_t y_{(a,e)}(t))^\gamma$, with same specification of age-dependent parameter of leisure $\kappa_t = \kappa(\frac{60}{a+t})^{\kappa_1}$ $(0 < \kappa \leq 1)$ during working years. The unobserved innovations in preferences are again AR(1):

$$\nu_t = \rho\nu_{t-1} + \epsilon_t.$$

As in the earlier context, the dependence on teacher is not reflected in the labeling of variables.

Next, consider the case that the teacher who has just made the non-purchasing decision in period t, and the only decision is whether to retire. Let $U(a,e,t,\nu_t)$ be the value function of teacher in period t who have not yet made the upgrade decision. Let $W(a,e,t,0)$ be the expected discounted lifetime utility of pension for the standard benefit with the upgrade, and $V(a,e,t,0,\nu_t)$ the value function of retirement. Then

$$V(a,e,t,0,\nu_t) = \max\{u(a,e,t) + \nu_t + \beta EU(a,e,t+1,\nu_{t+1}), W(a,e,t,0)\}.$$

The first term on the right-hand-side is the value of teaching for at least one more year without the pension upgrade, the second term is the value of retiring now. The optimal policy admits a threshold strategy in this case: the teacher retires if and only if $\nu_t \leq \nu^*(a,e,t,0)$.

Finally, we set up the Bellman equations for purchasing the "22 upgrade". Consider a teacher with initial age-exp (a,e) who has not purchased the upgrade by period

t. Let $disutil(a, e, t, \delta)$ be the dis-utility of the one-time payment δ of the upgrade, which is subtracted from the take-home income from last period. [6] We have:

$$U(a, e, t, \nu_t) = \max\{V(a, e, t, 1, \nu_t) - disutil(a, e, t, \delta), V(a, e, t, 0, \nu_t)\}.$$

The first term on the right-hand-side is the value of upgrading this year, the second term is the value of not upgrading now. Then one upgrades if and only if

$$V(a, e, t, 1, \nu_t) - disutil(a, e, t, \delta) \geq V(a, e, t, 0, \nu_t),$$

Since we already solved $V(a, e, t, 0, \nu_t)$ and $V(a, e, t, 1, \nu_t)$, and the disutility of payment can also be easily calculated, we can pin down the optimal actions given the value of ν_t.

3.4.3 Numerical Procedure

We solve the above model for each (age, experience) cell (a, e) by 1) calculate $W(a, e, t, 1)$ and then $V(a, e, t, 1, \nu_t)$ through backward induction. 2) Calculate $W(a, e, t, 0)$. 3) Given the value of $V(a, e, t, 1, \nu_t)$, solve for $V(a, e, t, 0, \nu_t)$ and $U(a, e, t, \nu_t)$ simultaneously. Note that we need to integrate ν_{t+1} out in $EU(a, e, t + 1, \nu_{t+1})$ via Gaussian-Hermit quadrature. We make several simplifying assumptions

[6] Consider the utility in period t with and without payment. The utility function without payment is the same as before: $u^0(a, e, t) \equiv u(a, e, t) = (k(a, t)((1 - c)y(e, t) - 0))^\gamma$, and we assume the utility with payment δ is $u^\delta(a, e, t) = (k(a, t)((1 - c)y(e, t) - \delta))^\gamma$, so the disutility is:

$$u^0(a, e, t) - u^\delta(a, e, t) = (k(a, t)(1 - c)y(e, t))^\gamma - (k(a, t)((1 - c)y(e, t) - \delta))^\gamma.$$

Finally, since we assume the payment is subtracted from last-period's income (this is to ensure that the payment is always subtracted from wage income; otherwise, the disutility depends on whether the teacher retires or not), we replace t with $t - 1$ to get:

$$disutil(a, e, t, \delta) = u^0(a, e, t - 1) - u^\delta(a, e, t - 1).$$

for now, which will be relaxed later.[7]

3.4.4 Account for the effect of ERI on the sample used for analysis of the "2.2 upgrade"

Now, consider teachers at the end of school year 1992-93 who were offered the ERI option between 1993-1995. The cost and benefit of ERI are summarized in the following table.

Table 3.4: Two Early Retirement Polices

Policy	Effective	Eligibility	Benefit	Price (Cost)
ERO	1979-2016	$age \geq 55,\ exp \geq 20$	undiscounted benefit	one-time payment (see note i)
ERI	1992-1994	$age \geq 50,\ exp \geq 5$	$age + 5,\ exp + 5$	one-time payment (see note ii) must retire immediately

Note: i) Only concerned Tier-1 members; ii) Fitzpatrick and Lovenheim (2014) noted "The fee for employees was 4 percent of the highest annual salary for the past five years for each of the five additional years of age and service purchased; the fee for employers was 12 percent of the employee's highest annual salary from the last five years for each additional year purchased."

Teachers in 1993 faced three choices: 1) keep working, 2) retire under ERI, 3) retire without ERI. We assume the 1998 2.2 upgrade was not foreseen when these teachers made the decision on ERI in 1993.

Then we solve the models in the following steps. i) First, we use backward induction under old formula to obtain the value function and threshold for year 1994, 1995, ..., assuming no further pension enhancement. ii) Second, we use the value function

[7] These assumptions are:

1) No ERO and actuarial calculation.

2) Payment is one-time, done at the beginning of each period, and partly by the employees. Although the required employee payment for the upgrade is 20% of previous year salary for all teachers, since they are senior teachers (exp = 22-28 in 1998), in the data we find that some fractions are paid by their employers. In this exercise, based on the sample average we assume that employers pay 50% of the upgrade fee.

3) For now we ignore the automatic increase of the replacement factor after 1998.

for year 1994 as the continuation value for not retiring in 1993 with or without ERI. iii) Finally, we solve for the optimal decision rule for ERI. For simplicity, for ERI we assume the teachers purchase the minimum of 5 years and the difference between the current service credit and the one that attains the 75% cap, i.e., 38 years. We model the payment for ERI in the same way as 2.2 upgrade: subtract it from the take-home income of the last period.[8] Then, the effect of ERI is fully captured by the increase in the threshold for the year it is offered. And we use the new thresholds f^+ to simulate the distribution of the preference shock ν_t from the year ERI is offered, and only keep those sequences for teachers still active in 1997-1998. In this way, we account for the left truncation induced by ERI.

3.4.5 Out-of-sample Fit

The same structural parameters in Table 3.3 $(\beta, \gamma, \kappa, \kappa_1, \rho, \sigma)$ estimated under the current rules and the 2005-12 sample are used to simulate the response of the 1997-98 cohort to the option of the "2.2 upgrade" and ERI. Following Fitzpatrick (2015) and Ni et al. (2022), we focus on a sample of senior teachers with experience between 22 and 28 in 1998, totaled at 19,126.[9] We reproduce the observed age-experience distribution for takers and non-takers in Figure 3.3 from Ni et al. (2022).[10] We plot the age (left) and experience (right) distribution for takers and non-takers in 1998 (upper) and at claim date (lower). Then, in Table3.5, we report the fraction of teachers taking the upgrade and the retirement rate across time for subgroups of teachers with different experience in 1998. Finally, in Figure 3.4 and Table 3.6, we repeat the above analysis for the simulated data with our model in this section and

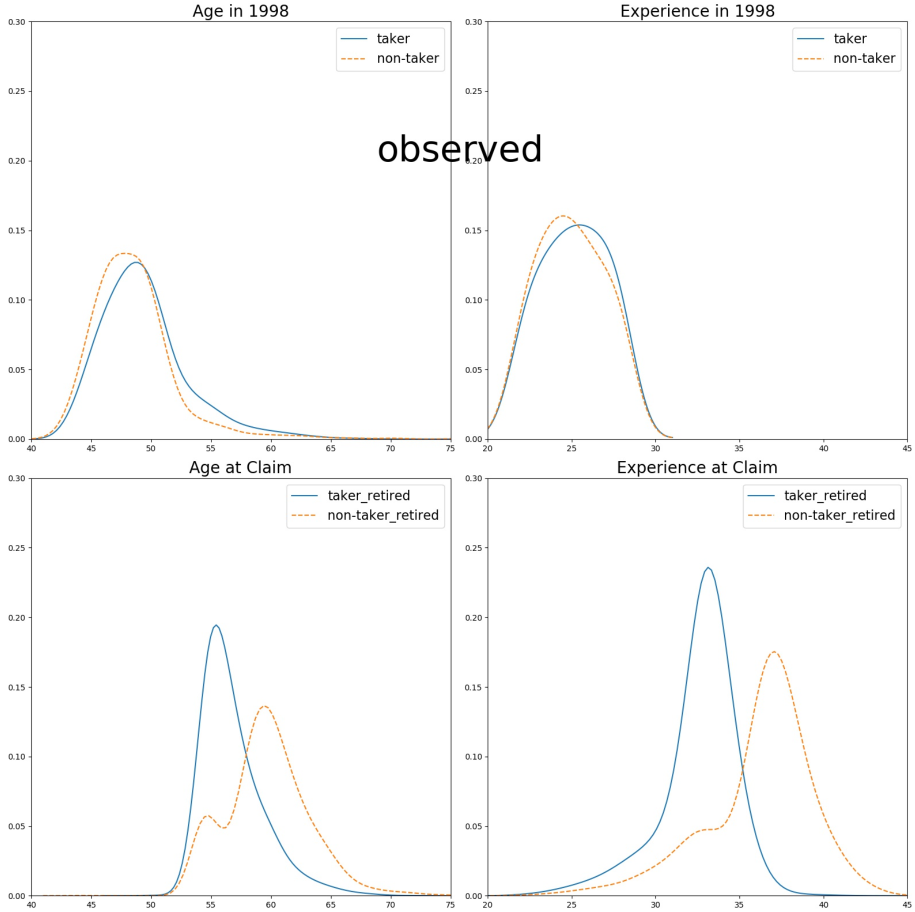
[8] Since there are two waves of ERI, to avoid complexity of introducing the timing choice of ERI, we assume (based on the data) that 64% of the population is offered ERI only in the end of school year 1992-1993, and the remaining 36% in 1993-1994.

[9] This is the sample size in Ni et al. (2022). Fitzpatrick (2015) uses a different sample; for discrepancies between these two, see Ni et al. (2022).

[10] The scale for the x-axis in the upper panels is changed to be the same as the lower panels.

the parameter estimates of Table 3.3.[11]

Figure 3.3: Observed age-experience distribution for takers and non-takers



[11] That is, we use the same sample of teachers, but generate our own retirement and upgrade decisions.

Figure 3.4: Simulated age-experience distribution for takers and non-takers

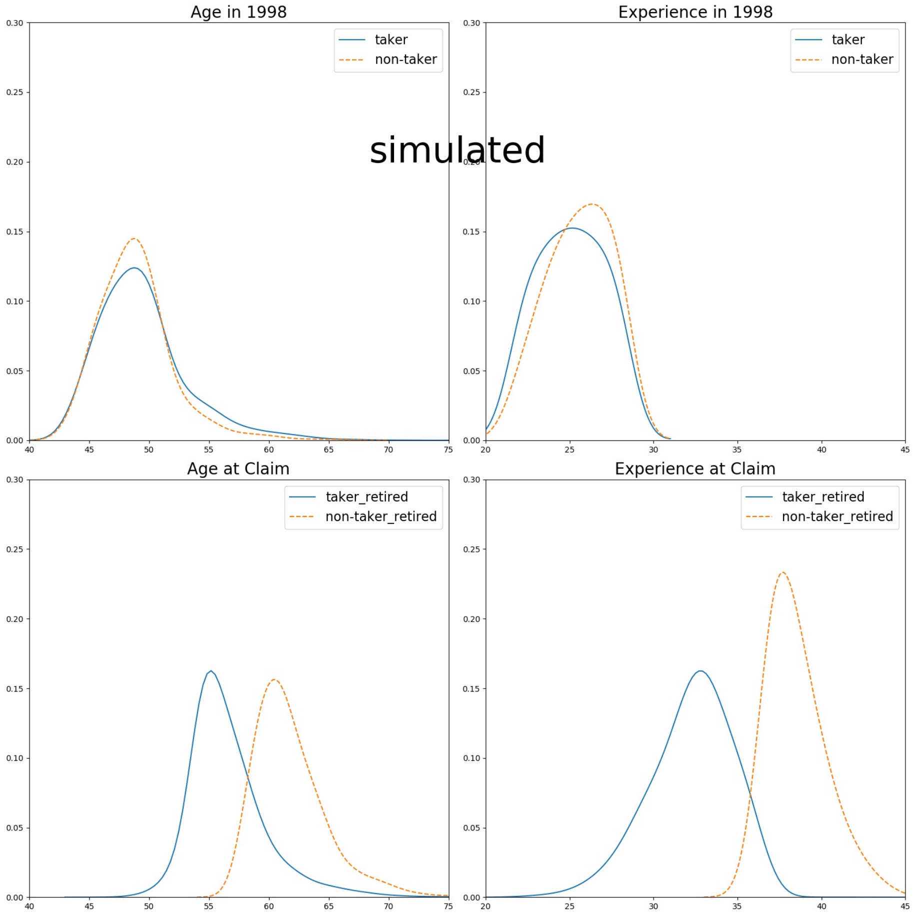


Table 3.5: Upgrade decision and retirement timing, observed data

exp at 98	upgrade	1998	2000	2002	2004	2006	2008	2010	2012	obs_1
22	0.83	0.01	0.03	0.06	0.12	0.21	0.32	0.75	0.90	2385
23	0.84	0.01	0.03	0.06	0.13	0.21	0.56	0.85	0.94	2430
24	0.85	0.01	0.03	0.07	0.14	0.33	0.78	0.90	0.96	2897
25	0.87	0.01	0.04	0.08	0.16	0.63	0.87	0.94	0.98	3006
26	0.89	0.01	0.05	0.10	0.25	0.83	0.92	0.96	0.99	2972
27	0.90	0.01	0.05	0.13	0.63	0.89	0.95	0.97	0.99	2599
28	0.89	0.01	0.06	0.17	0.80	0.91	0.96	0.97	0.99	2837
whole sample	0.87	0.01	0.04	0.10	0.32	0.59	0.78	0.91	0.97	19126

Table 3.6: Upgrade decision and retirement timing, simulated data

exp at 98	upgrade	1998	2000	2002	2004	2006	2008	2010	2012	obs_1
22	0.80	0.00	0.01	0.03	0.10	0.24	0.44	0.67	0.80	2385
23	0.81	0.00	0.01	0.05	0.13	0.29	0.56	0.75	0.86	2430
24	0.79	0.00	0.02	0.07	0.19	0.40	0.65	0.79	0.88	2897
25	0.80	0.01	0.03	0.11	0.25	0.55	0.74	0.85	0.91	3006
26	0.79	0.01	0.05	0.16	0.38	0.64	0.79	0.88	0.94	2972
27	0.79	0.02	0.08	0.23	0.54	0.73	0.85	0.91	0.94	2599
28	0.78	0.02	0.13	0.35	0.63	0.78	0.88	0.93	0.96	2837
whole sample	0.79	0.01	0.05	0.15	0.32	0.53	0.71	0.83	0.90	19126

Note: Payment for the “2.2 upgrade” is made one-time and half of it paid by employer, with possible refund. ERO is not modeled.

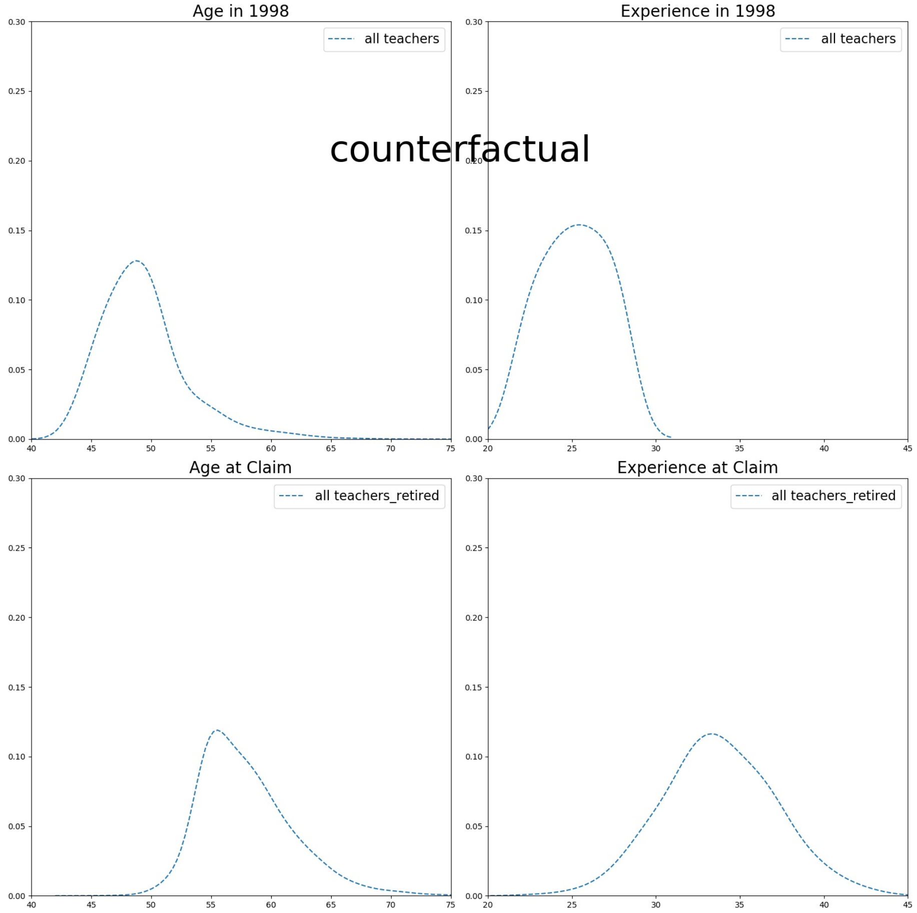
Figure 3.3 shows that takers and non-takers are similar in terms of their age-experience profile in 1998. However, they separate at the time of retirement. Takers tend to retire around age 55 with 33-34 years of experience, while the distribution of age for non-takers at retirement is bi-modal: one at 55 and another at 60. Moreover, non-takers likely retire with 37-38 years of experience, though the distribution of experience of non-takers is less concentrated than takers. Table 3.5 shows that the average upgrade rate is 87%, and it is increasing in experience in 1998. In 1998 only 1% of the sample retire, while in 2012, 97% are retired.

The simulated data (Figure 3.4 and Table 3.6) also exhibit the similarity of takers and non-takers in 1998 and then separation at retirement. Figure 3.4 and Table 3.6 match the pattern of Figure 3.3 and Table 3.5. The out-of-sample match is not perfect though. Since we do not include ERO and actuarial calculation in our model, the age and experience distribution of non-takers at retirement is uni-modal: non-takers retire around age 60 with experience 37-38. The average upgrade rate is 79%, significantly lower than the true rate of 87%, possibly due to the assumption that we do not allow refund and payment in installment. And the upgrade rate is decreasing in experience in 1998, in contrast to the true pattern. Finally, the retirement rate is also lower in our simulated data: it is only 90% in 2012, 7% lower than the observed value. This may be due to the low upgrade rate, since takers typically retire earlier than non-takers.

3.4.6 The effect of 2.2 upgrade: structural estimation

Finally, we construct the counterfactual without 2.2 upgrade for the 1997-98 cohort. The age and experience distribution is shown in Figure 3.5. We find that without 2.2 upgrade, the distribution of age and experience will be flatter and exhibits a single peak. Based on this counterfactual, we found the introduction of "2.2 upgrade" moved retirement year forward by roughly 1.3 years on average.

Figure 3.5: Counterfactual age-experience distribution in the absence of the “2.2 upgrade” option



3.5 Concluding Remarks

We formulate a dynamic programming model of retirement with time-varying unobserved heterogeneity. We estimate the model with data on a later cohort using a modified three-step procedure and deep neural networks (DNN). Our structure model fits well both in-sample and out-of-sample. The estimated model can be used for simulating counterfactual scenarios based on historical teacher population and analyzing hypothetical rules of future pension reforms based on the current teacher population.

Appendix A

Additional Results for Chapter 1

A.1 GHK for Unconditional Replacement Probabilities

Denote the normal $N(m, \sigma^2)$ truncated at (a, b) as $TN_{(a,b)}(m, \sigma^2)$. The CDF of standard normal is $\Phi(.)$.

The following algorithm computes the probability of replacing in period x, $p(x)$, for a new engine in period 0:

$p(0) = 1 - \Phi(\epsilon^*(0)\sqrt{1-\rho^2}) = \Phi(-\epsilon^*(0)\sqrt{1-\rho^2})$.

1. Starting in period 0, obtain K ϵ_0s that satisfy $\epsilon_0^{(k)} < \epsilon^*(0)$ by drawing from the right truncated $TN_{(-\infty,\epsilon^*(0))}(0, \frac{1}{1-\rho^2})$, $k = 1, .., K$.

2. For $0 < t < x$, given $\epsilon_{t-1}^{(k)}$ draw $\mu_t^{(k)}$ from $TN_{(-\infty,\epsilon^*(t)-\rho\epsilon_{t-1}^{(k)})}(0, 1)$, hence $\epsilon_t^{(k)} = \rho\epsilon_{t-1}^{(k)} + \mu_t^{(k)} < \epsilon^*(t)$.

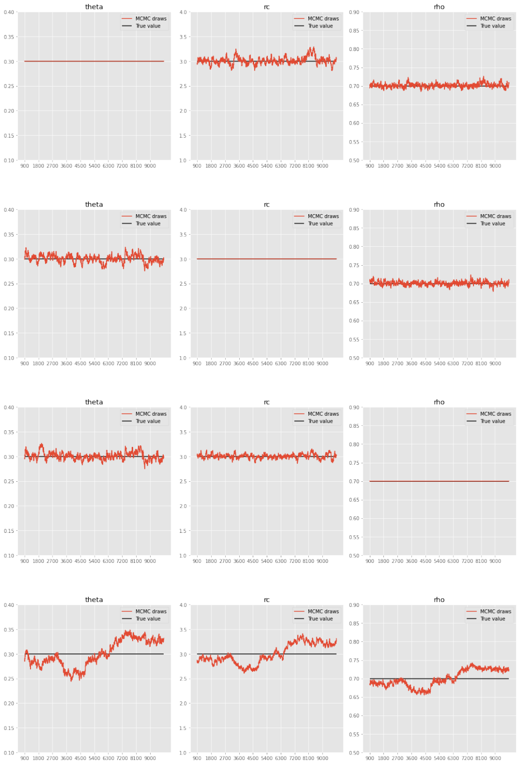
Note $\epsilon_t^{(k)}$ is a biased draw of the error. But the following estimate of the probability of replacing engine in period $x > 0$ is unbiased:

$$p(x) \approx \frac{1}{K}\sum_{k=1}^{K}[\prod_{t=1}^{x-1}\Phi(\epsilon^*(t)-\rho\epsilon_{t-1}^{(k)})]\Phi(-\epsilon^*(t)+\rho\epsilon_{x-1}^{(k)}). \tag{A.1}$$

$K = 200$ would generate accurate approximation.

A.2 Results with 1,000,000 Buses

Figure A.1: Results for AR(1) with $\rho = 0.7$, 1,000,000 buses



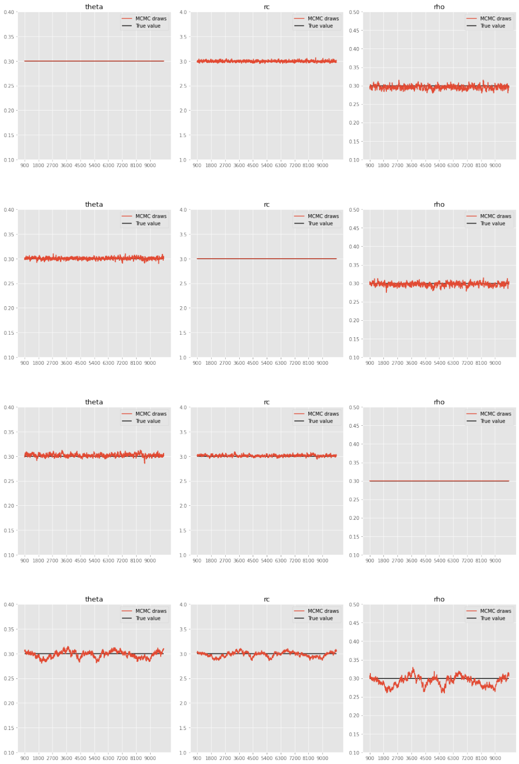
true	5%	post median	95%
0.3	0.3000	0.3000	0.3000
3.0	2.9034	3.0086	3.1509
0.7	0.6921	0.7012	0.7116

true	5%	post median	95%
0.3	0.2854	0.3007	0.3125
3.0	3.0000	3.0000	3.0000
0.7	0.6920	0.7004	0.7098

true	5%	post median	95%
0.3	0.2891	0.3005	0.3157
3.0	2.9292	3.0053	3.0861
0.7	0.7000	0.7000	0.7000

true	5%	post median	95%
0.3	0.2586	0.2954	0.3391
3.0	2.6888	2.9461	3.3065
0.7	0.6638	0.6949	0.7321

Figure A.2: Results for AR(1) with $\rho = 0.3$, 1,000,000 buses



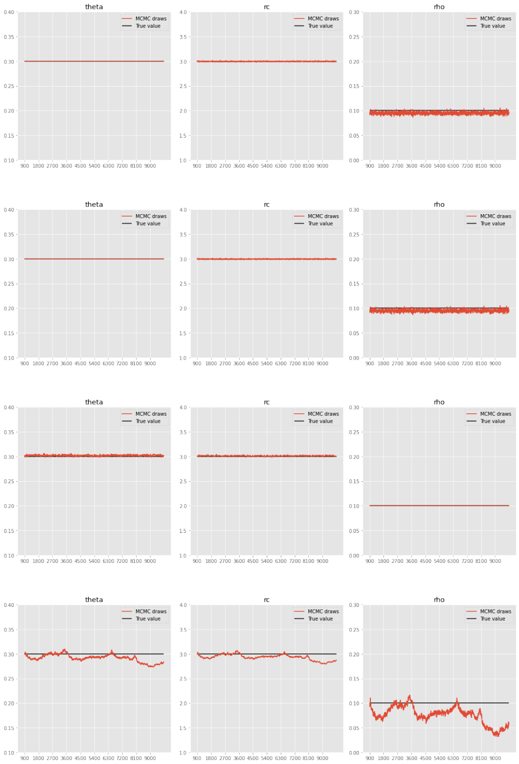
true	5%	post median	95%
0.3	0.3000	0.3000	0.3000
3.0	2.9712	2.9944	3.0216
0.3	0.2874	0.2957	0.3041

true	5%	post median	95%
0.3	0.2968	0.3006	0.3044
3.0	3.0000	3.0000	3.0000
0.3	0.2887	0.2974	0.3047

true	5%	post median	95%
0.3	0.2976	0.3021	0.3074
3.0	2.9811	3.0126	3.0467
0.3	0.3000	0.3000	0.3000

true	5%	post median	95%
0.3	0.2863	0.2990	0.3079
3.0	2.8902	2.9861	3.0585
0.3	0.2693	0.2931	0.3123

Figure A.3: Results for AR(1) with $\rho = 0.1$, 1,000,000 buses



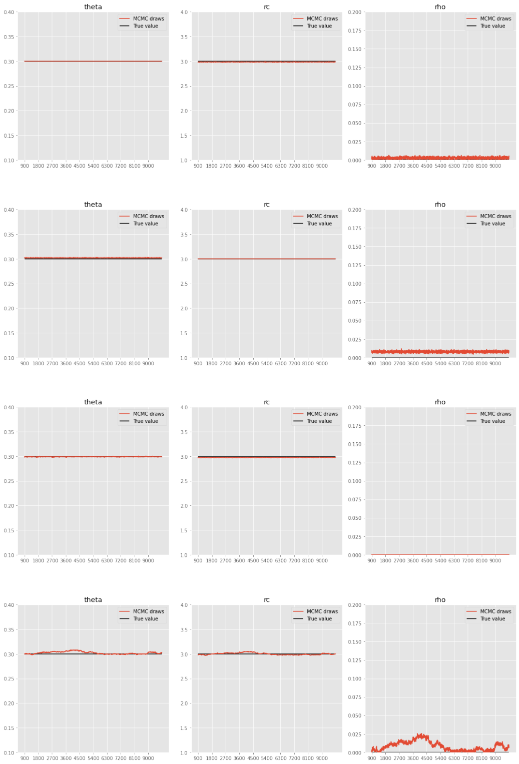
true	5%	post median	95%
0.3	0.3000	0.3000	0.3000
3.0	2.9881	2.9953	3.0037
0.1	0.0912	0.0948	0.0991

true	5%	post median	95%
0.3	0.2993	0.3005	0.3015
3.0	3.0000	3.0000	3.0000
0.1	0.0923	0.0962	0.0998

true	5%	post median	95%
0.3	0.2999	0.3020	0.3040
3.0	2.9962	3.0114	3.0257
0.1	0.1000	0.1000	0.1000

true	5%	post median	95%
0.3	0.2767	0.2924	0.3029
3.0	2.8238	2.9393	3.0166
0.1	0.0422	0.0782	0.1022

Figure A.4: Results for AR(1) with $\rho = 0.0$, 1,000,000 buses



true	5%	post median	95%
0.3	0.3000	0.3000	0.3000
3.0	2.9803	2.9830	2.9861
0.0	0.0009	0.0026	0.0044

true	5%	post median	95%
0.3	0.3017	0.3021	0.3025
3.0	3.0000	3.0000	3.0000
0.0	0.0061	0.0077	0.0095

true	5%	post median	95%
0.3	0.2984	0.2990	0.2997
3.0	2.9702	2.9751	2.9802
0.0	0.0000	0.0000	0.0000

true	5%	post median	95%
0.3	0.2992	0.3017	0.3073
3.0	2.9763	2.9978	3.0430
0.0	0.0006	0.0069	0.0208

A.3 Thresholds for car replacement problem

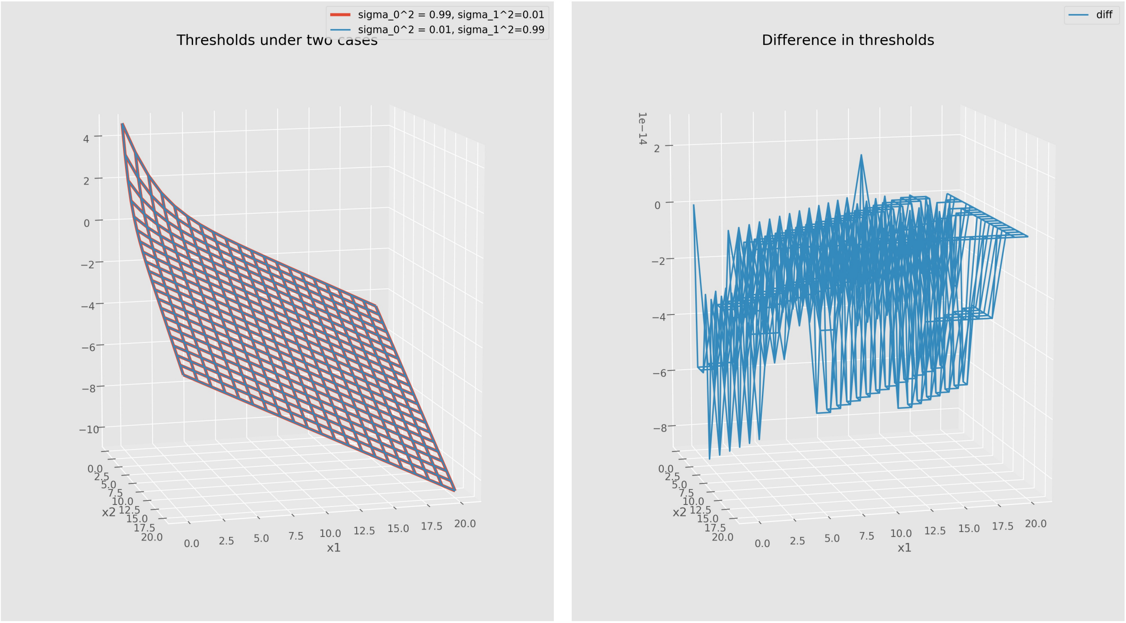
Recall that the threshold strategy states that we replace the car if and only if:

$$\epsilon_{0t} - \epsilon_{1t} < \epsilon^*(\mathbf{x_t})$$

And we only assume that the difference between error term follows a standard normal distribution $\epsilon_{0t} - \epsilon_{1t} \overset{i.i.d.}{\sim} N(0,1)$. However, in the calculation of the expected valuation, we need the joint distribution of both ϵ_{0t} and ϵ_{1t}. Here, we consider two extreme cases: i) $\sigma_{0t} \overset{i.i.d.}{\sim} N(0, \sigma_0^2 = 0.01), \sigma_{0t} \overset{i.i.d.}{\sim} N(0, \sigma_1^2 = 0.99)$ and ii) $\sigma_{0t} \overset{i.i.d.}{\sim} N(0, \sigma_0^2 = 0.99), \sigma_{0t} \overset{i.i.d.}{\sim} N(0, \sigma_1^2 = 0.01)$. Both cases gives us $\epsilon_{0t} - \epsilon_{1t} \overset{i.i.d.}{\sim} N(0,1)$.

With two-dimensional state vector, we plot the corresponding thresholds under these two settings in the left panel. Red lines are for case i) and blue lines for case ii). They overlap perfectly. The right panel shows their difference, and the range is between $-8 \times 10^{-14} \sim 2 \times 10^{-14}$, so they are indistinguishable at least at magnitude of 10^{-12}. Hence the choice of the joint distribution does not matter, we only require the difference between error terms to follow i.i.d. standard normal distribution. This also justifies the simplification in the engine replacement problem in example 1-3.

Figure A.5: Thresholds for car replacement problem under two settings for error term distribution



Note: the left panel shows the thresholds for the car replacement problem in section 1.7 for two-dimensional state vector under these two settings: i) red lines for $\sigma_{0t} \overset{i.i.d.}{\sim} N(0, \sigma_0^2 = 0.01), \sigma_{0t} \overset{i.i.d.}{\sim} N(0, \sigma_1^2 = 0.99)$; ii) blue lines for $\sigma_{0t} \overset{i.i.d.}{\sim} N(0, \sigma_0^2 = 0.99), \sigma_{0t} \overset{i.i.d.}{\sim} N(0, \sigma_1^2 = 0.01)$. The right panel shows their differences, with the scale in the y-axis is $\times 10^{-14}$.

A.4 Specification of DNN

Table A.1: structure of DNN for AR(1)

layer	shape	no. param	note
input	4		
hidden 1	32	160	32*4(for w) + 32*1 (for b)
hidden 2	32	1056	32*32 + 32*1
output	1	33	32*1 + 1*1
	total	1249	

Table A.2: hyperparameters of DNN for AR(1)

hyperparams	choice
activation function	relu
learning rate	0.01
optimizer	adam
loss function	customized
no. epochs	1000
batch size	424
package	tensorflow 2.1
platform	google colab

Appendix B

Additional Results for Chapter 2

B.1 Possible Choices for Implementation

We divide the numerical procedures into four categories: 1) DP solution, 2) library building, 3) DNN training, and 4) estimation. All of them are relevant to the deep learning-aided approaches (i.e., algorithm 3 and 4 in chapter 1), while only 1) and 4) are relevant to the traditional method (i.e., algorithm 1 "solve to estimate"). For each procedure, there are several options with corresponding hyper-parameters. For example, for estimation, we need to decide whether to use maximum likelihood estimation (MLE), Bayesian estimation, or generalized (simulated) method of moment (GMM, SMM). And if we choose MLE, we then need to determine which method to calculate the likelihood, initial guess of parameters, maximum number of iteration, and tolerance for convergence.

Table B.1: Options for traditional and deep-learning-aided approaches to structural estimation

	traditional	deep-learning-aided	options	hyperparameters	note (common choices)
DP solution	yes	yes	naïve grid	number of girds	equal probability discretization
			GH quadrature	number of GH roots	
				number of grids	equal space discretization
			monte carlo	number of MC draws	
library building	no	yes	same as solution	see above	
			library shape	size of the library	
				library DGP	for params and states
DNN training	no	yes	DNN structure	neural network	or other ML techniques
				activation function	relu or sigmoid
				loss function	MSE or some weighted average
				optimization algorithm	standard newton or adam
				number of iteration	
estimation	yes	yes	MLE	likelihood calculation	GHK
					discretization filter
					others
				error tolerance	
			Bayesian	prior distribution	
				number of MCMC runs	
			GMM (SMM)	moment conditions	
				choice of weighting matrix	

B.2 Library Building and DNN Training Details

Table B.2 shows the solution library for optimal retirement models. Table B.3 reports the performance of DNN with three measures. Figure B.1 plots the true thresholds (in x-axis) against the predicted values from DNN (in y-axis) along with 45 degree line. Figure B.2 shows the true thresholds (red solid lines) for teachers with initial age 50 and the predicted threshold (blue dashed lines) from DNN.

Table B.2: structure of the library

	beta	gamma	kappa	kappa_1	rho	sigma	a	e	t	f(a, e, t)
0	0.9383	0.4805	0.8252	0.9042	0.3536	2229.598	49	10	0	-3217.5
1	0.9383	0.4805	0.8252	0.9042	0.3536	2229.598	49	10	1	-3198.26
2	0.9383	0.4805	0.8252	0.9042	0.3536	2229.598	49	10	2	-3175.25
3	0.9383	0.4805	0.8252	0.9042	0.3536	2229.598	49	10	3	-3147.83
4	0.9383	0.4805	0.8252	0.9042	0.3536	2229.598	49	10	4	-3115.26
...	...	...	...	...	...	...	...	...	...	...
20995	0.9726	0.8172	0.7357	1.0832	0.4281	2664.682	50	27	2	-32786.6
20996	0.9726	0.8172	0.7357	1.0832	0.4281	2664.682	50	27	3	-23351.5
20997	0.9726	0.8172	0.7357	1.0832	0.4281	2664.682	50	27	4	-17057.8
20998	0.9726	0.8172	0.7357	1.0832	0.4281	2664.682	50	27	5	-10094.1
20999	0.9726	0.8172	0.7357	1.0832	0.4281	2664.682	50	27	6	-7353.06

Table B.3: Training DNN with weighted sum loss

	sample size	weighted squared loss	mean squared error	mean absolute error
		$(\times 10^6)$	$(\times 10^6)$	$(\times 10^3)$
train set	139565	0.0020	0.0518	0.0732
test set	46522	0.0038	0.0601	0.0806

note: the inputs are scaled to [0, 1] by max-min scaler, while the outputs are not. we use $\sigma = 3,000$ for the weighted squared loss.

Figure B.1: scatter plot for true versus predicted thresholds

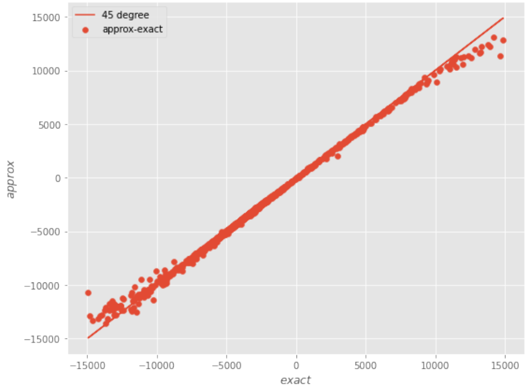
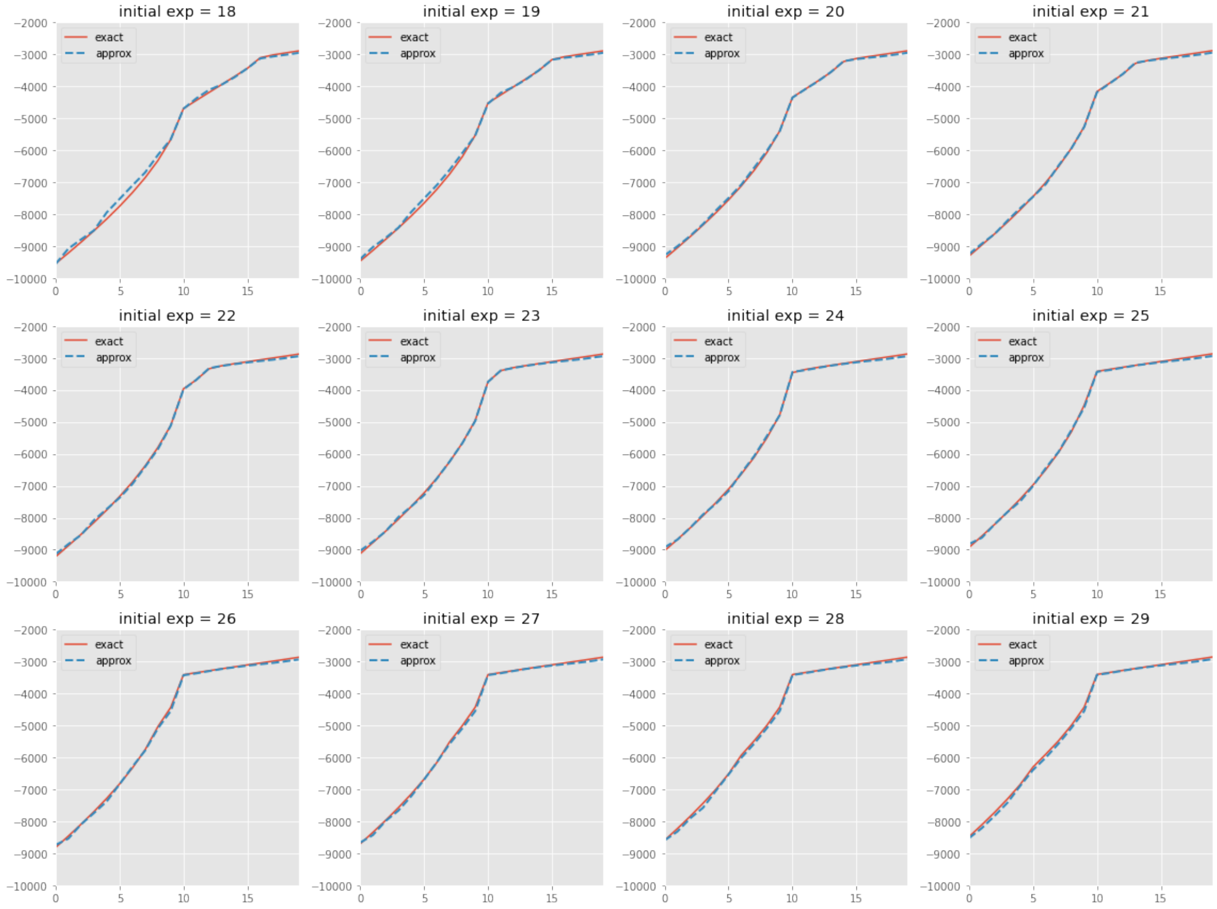


Figure B.2: Thresholds for teacher with initial age = 50



B.3 Discretization of AR(1) Process

Adda et al. (2003) modify the Tauchen method for discretization by assuming equal probability over all discretized intervals under stationary distribution. Consider an AR(1) process

$$z_t = \rho z_{t-1} + \epsilon_t$$

where Φ is standard normal cdf and ϵ_t is i.i.d. $N(0, \sigma^2)$.

We would like to discretize the real line into N intervals by points $z_1, ..., z_{N+1}$ where $z_1 = -\infty$ and $z_{N+1} = \infty$. The equal probability assumption requires

$$\Phi(\frac{z_{i+1}}{\sigma_z}) - \Phi(\frac{z_i}{\sigma_z}) = \frac{1}{N}$$

where $\sigma_z = \sigma/\sqrt{1-\rho^2}$. Hence we have

$$z_i = \sigma_z \Phi^{-1}(\frac{i-1}{N})$$

Now, we wish to find the mean point $\hat{z}_i$ for each interval $[z_i, z_{i+1}]$, which is the conditional expectation:

$$\hat{z}_i = \mathbb{E}[z_t | z_t \in [z_i, z_{i+1}]] = \sigma_z \frac{\phi(z_i/\sigma_z) - \phi(z_{i+1}/\sigma_z)}{\Phi(z_i/\sigma_z) - \Phi(z_{i+1}/\sigma_z)} = N\sigma_z(\phi(\frac{z_i}{\sigma_z}) - \phi(\frac{z_{i+1}}{\sigma_z}))$$

where ϕ is standard normal density, and the second equality is from equal probability assumptions.

Then, the transition probability is

$$\pi_{ij} = \Pr(z_t \in [z_j, z_{j+1}] | z_{t-1} \in [z_i, z_{i+1}])$$

which can be written as

$$\pi_{ij} = \frac{N}{\sqrt{2\pi}\sigma_z} \int_{z_i}^{z_{i+1}} e^{-u^2/(2\sigma_z^2)} [\Phi(\frac{z_{j+1}}{\sigma}) - \Phi(\frac{z_j}{\sigma})] du$$

Now, we can define a Markov process $\hat{z}_t$ taking values in $\hat{Z} = \hat{z}_1, ..., \hat{z}_N$ with transition matrix $\Pi = [\pi_{ij}]$

B.3.1 From Critical Values to Retirement Probabilities

We have two methods for deriving the unconditional retirement probabilities. One is based on GHK algorithm and detailed in the theoretical note. Here we present the other one.

Suppose we already have the critical values $\mathbf{f_t^\dagger}$ as well as parameters for the AR(1) process

$$\nu_t = \rho \nu_{t-1} + \epsilon_t$$

we wish to calculate the probability of retirement at year $t+m$ where $m = 0, 1, ..., T-t$.

First, for a chosen N, we discretize the process ν_t as detailed in Appendix A to get discretized points $\hat{\nu}$ and transition matrix Π. Second, we find the index of the smallest $\hat{\nu}$ which is greater than or equal to $-f^\dagger_{t+m}$:

$$\hat{i}_{t+m} = \inf_i \{\hat{\nu}_i | \hat{\nu}_i \geq -f^\dagger_{t+m}\}$$

Then the probability of retiring at period $t+m$ is

$$\Pr(f^\dagger_t > -\nu_t, ..., f^\dagger_{t+m-1} > -\nu_{t+m-1}, f^\dagger_{t+m} \leq -\nu_{t+m})$$

i.e.,

$$\Pr(\nu_t > -f^\dagger_t, ..., \nu_{t+m-1} > -f^\dagger_{t+m-1}, \nu_{t+m} \leq -f^\dagger_{t+m})$$

which can be approximated by

$$\Pr(\hat{\nu}_t > -f^\dagger_t, ..., \hat{\nu}_{t+m-1} > -f^\dagger_{t+m-1}, \hat{\nu}_{t+m} \leq -f^\dagger_{t+m})$$

shorthand the diagonal matrix where first $\hat{i}_{t+m}$ values are 1 and the left are 0 as

$$\text{diag}(\hat{i}_{t+m}) = \text{diag}(\{\overbrace{1, 1..., 1}^{\hat{i}_{t+m}\text{ times}}, 0, .., 0\})$$

and denote the stationary probability as

$$\pi_t = [\frac{1}{N}, \frac{1}{N}, ..., \frac{1}{N}]'$$

which is also the initial distribution of ν. then at time $t + m$, the unconditional retiring probability is

$$\Pr(\text{retire at } t + m) = \text{sum}\{\text{diag}(\hat{i}_{t+m})\pi_{t+m}\}$$

where I denotes the identity matrix, and the distribution evolves as

$$\pi_{t+m+1} = \Pi'[I - \text{diag}(\hat{i}_{t+m})]\pi_{t+m}$$

the probability of not retiring at the last observable period is

$$\Pr(\text{not retire at } T) = \text{sum}\{[I - \text{diag}(\hat{i}_T)]\pi_T\}$$

Appendix C

Additional Results for Chapter 3

C.1 Additional Descriptive Tables

Table C.1: Claim types of retired teachers

Claim Type	Numbers of teachers
Regular 2.2	7496
2.2 ERO Member Pay	2082
2.2 ERO Employer Pay	1205
Normal - Actuarial calculation	830
1 2/3% Formula - Graduated	419
Rule of 85 - 2.2	69
Regular 2.2 - Disability	36
35% of Last Salary - No Max	24
ERO Member Pay	5
Rule of 85 - Formula	4
Rule of 85 - Actuarial	3
ERO Employer Pay	2
Age Type Formula Calculation	2
Total	12177

Note: Retirement Claims by teachers. Total sample number = 27299. The number of officially recorded retired claims is different from our calculation of retired teachers, since they were recorded as of Feb 2014.

C.2 SMM Results

C.2.1 SMM Estimation

Denote the observed count for teachers retired in year t with initial age and experience (a, e) as $c_{(a,e)}(t)$ for $t = 0, 1, ..., T$, and those not retired as $c_{(a,e)}(T + 1)$. They sum up to the total number of teachers with the same initial age and experience (a, e):

$$\sum_{t=0}^{T} c_{(a,e)}(t) + c_{(a,e)}(T + 1) = C_{(a,e)}$$

For a set of parameters, $\theta = (\beta, \gamma, \kappa, \kappa_1, \rho, \sigma)$, we calculate the thresholds either through DP or SW. Suppose we proceed with DP thresholds $\nu^*_{a,e}(t; \theta)$, where we add θ to emphasis its dependence on the choice of parameters. Next, we simulate exactly $C_{(a,e)}$ AR(1) sequences with parameter ρ and σ for teachers with initial age and experience (a, e). Comparing the realization of preference shocks with the thresholds $\nu^*_{(a,e)}(t; \theta)$, we obtain the retirement status and timing for each teacher. Summing them up to derive the simulated count for $\hat{c}_{a,e}(t; \theta)$ and $\hat{c}_{(a,e)}(T + 1; \theta)$. We repeat for all (a, e) combinations and get a full simulated sample. The objective function is a distance metric of the simulated counts and observed counts:

$$J(\theta) = \sum_{(a,e)} \sum_{t=0}^{T+1} [\hat{c}_{(a,e)}(t; \theta) - c_{(a,e)}(t)]^2$$

And we wish to find θ to minimize such distance.

C.2.2 Correction for Measurement Error

The experience credited for retirement of a teacher increases by one at the end of the AY (say July 1) for all. Age for retirement accounting depends on the birth date.

When birth year is observed but birth date is unavailable to researchers.

We denote the values by the age and experience in the initial period, (a, e). $y_{(a,e)}(t)$ is the salary of teacher in year t with (a, e) in the initial year 0 (hence with age $a + t$ and experience $e + t$ in year t.

The age relevant for pension benefit is conditioning on the unobserved birth month index M, $M = 0 : 1 \leq month \leq 12$. We assume that if an TRS teacher has a birthday with $M = 1 : 7 \leq month \leq 12$) she can count an additional year in age.

The year t value of pension wealth of a teachers with (a, e) is $W_{(a,e)}(t) = \sum_{s=t}^{T} G(a+s, a+s+1)\beta^{s-t}(B_{(a,e)}(s,t))^{\gamma}$. where $B_{(a,e)}(s,t)$ is the year-t value of annual retirement benefit received in year s by the teacher with initial (a, e) retired in year t.

So the pension benefit for retiring in current year is

$W_{(a,e)}(t)$ if $M = 0 : 1 \leq month \leq 6$;

$W_{(a+1,e)}(t)$ if $M = 1 : 7 \leq month \leq 12$.

The Dynamic Programming Problem

The utility function for period t is $[(\kappa_t y_{(a,e)}(t))^{\gamma} + \nu_t]$, where $\kappa_t = \kappa(\frac{60}{a+t})^{\kappa_1}$ is an age-dependent parameter of leisure, with $0 < \kappa \leq 1$ during working years and captures the disutility of working. The unobserved innovations in preferences are AR(1): $\nu_t = \rho\nu_{t-1} + \epsilon_t$. We assume ϵ_t is iid $N(0, \sigma^2)$.

The DP problem with accurately measured age and experience is as follows:

The value function of current teacher with preference error ν_t is $V_{(a,e)}(t, \nu_t)$, is

$$V_{(a,e)}(t, \nu_t) = max\{U_{(a,e)}(t, \nu_t) + \nu_t, \quad W_{(a,e)}(t)\}$$

where $U_{(a,e)}(t, \nu_t)$ is the expected value function of continuing teaching:

$$U_{(a,e)}(t, \nu_t) = [\kappa_t y_{(a,e)}(t)]^{\gamma} + \beta G(a + t, a + t + 1)\mathbb{E}_{\epsilon} V_{(a,e)}(t + 1, \nu_{t+1}). \tag{C.1}$$

The teacher chooses to retire when $\nu_t < \nu^*$ where $V_{(a,e)}(t, \nu^*) = W_{(a,e)}(t)$; i.e., $U_{(a,e)}(t, \nu^*) + \nu^* \leq W_{(a,e)}(t)$. The threshold of preference ν^* depends on (a, e, t) and we make the dependence explicit by denoting as $\nu^*(a, e, t)$.

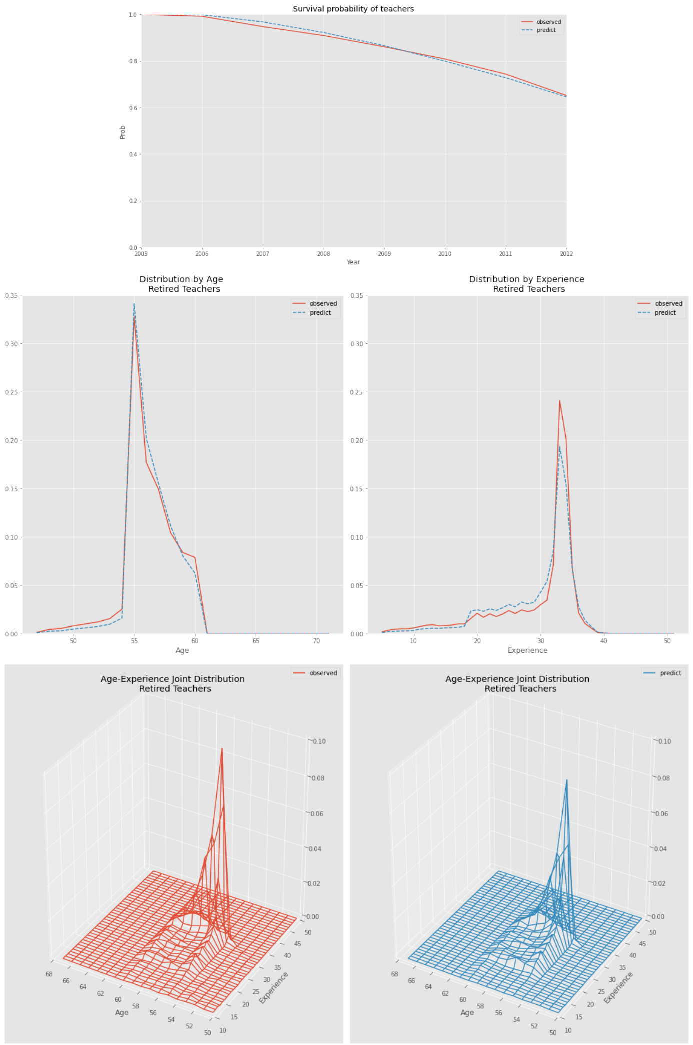
With uniformly distributed birthdays the probability of the unobserved index $M = 0$ is 0.5 and probability of $M = 1$ is 0.5. The threshold for a teacher (a, e, M) in t is $\nu(M, a, e, t)$, with $\nu(M = 0, a, e, t) = \nu^*(a, e, t)$, and $\nu(M = 1, a, e, t) = \nu^*(a + 1, e, t)$. The likely case is that as the teacher nearing eligibility for retirement $\nu(M = 0, a, e, t) > \nu(M = 1, a, e, t)$, i.e., a teachers with birthday $M = 1$ is likely to retire earlier because an extra year in age pushes her over the eligibility line she is more likely to retire with a given preference error ν_t. A teacher with a given initial (a, e) and not retiring in t may be because of a high ν_t or $M = 0$.

Table C.2: SMM parameter estimation results

model	SW	DP	SW	DP
			with ME	with ME
β	0.9648	0.9452	0.9681	0.9330
γ	0.6558	0.7773	0.6918	0.8242
κ	0.6515	0.2789	0.5800	0.4542
κ_1	1.0673	1.0269	1.0688	1.0615
ρ	0.3446	0.3129	0.3263	0.3115
σ	3224.0553	2950.8957	3597.4202	3934.2339

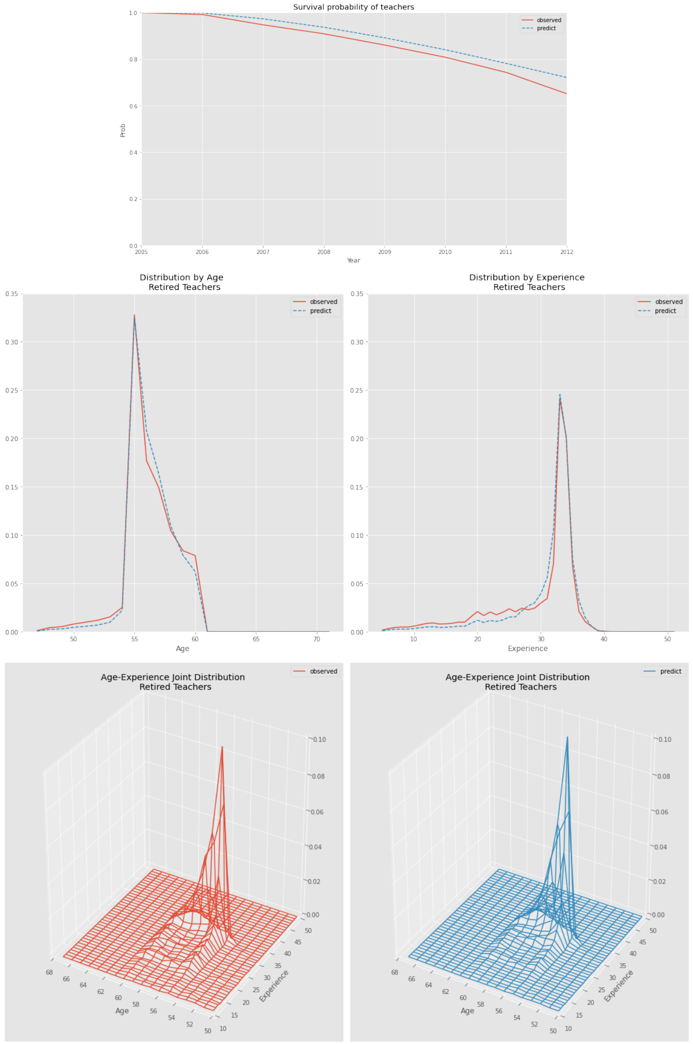
Note: SW means Stock Wise, DP means Dynamic Programming. with ME means with correction for measurement error in age.

Figure C.1: The in-sample goodness of fit for SW with SMM (corrected for measurement error)



Note: the upper panel plots the survival rate. The middle panels plot the marginal distribution of age (left) and experience (right) of retired teachers (at the time of retirement). We use red solid line for the observed data and blue dashed lines for the predicted data with estimated parameters. The lower panels plot the joint distribution of age and experience of retired teachers (at the time of retirement), where the red lines in the left is for the observed data and the blue lines in the right for the predicted data.

Figure C.2: The in-sample goodness of fit for DP with SMM (corrected for measurement error)



Note: the upper panel plots the survival rate. The middle panels plot the marginal distribution of age (left) and experience (right) of retired teachers (at the time of retirement). We use red solid line for the observed data and blue dashed lines for the predicted data with estimated parameters. The lower panels plot the joint distribution of age and experience of retired teachers (at the time of retirement), where the red lines in the left is for the observed data and the blue lines in the right for the predicted data.

C.3 Willingness-to-pay for Simple Pension Enhancement

Consider a small increase of δ to the annual pension benefit for a teacher with initial (a, e) and retiring in year t (By "small", we mean such increase does not change her retirement behavior). That is, for each year $s = t, ...T$, the benefit changes from $B_{(a,e)}(s, t)$ to $B_{(a,e)}(s, t) + \delta$. The increase in year t monetary value pension wealth is

$$\begin{aligned}\Delta PW_{(a,e)}(t) &= \sum_{s=t}^{T} G(a+s, a+s+1)\beta^{s-t}(B_{(a,e)}(s,t)+\delta) \\ &- \sum_{s=t}^{T} G(a+s, a+s+1)\beta^{s-t}(B_{(a,e)}(s,t)) \\ &= \delta \sum_{s=t}^{T} G(a+s, a+s+1)\beta^{s-t}.\end{aligned}$$

And the increase in the year t subjective value of pension wealth (discounted utility from pension) is

$$\begin{aligned}\Delta W_{(a,e)}(t) &= \sum_{s=t}^{T} G(a+s, a+s+1)\beta^{s-t}(B_{(a,e)}(s,t)+\delta)^{\gamma} \\ &- \sum_{s=t}^{T} G(a+s, a+s+1)\beta^{s-t}(B_{(a,e)}(s,t))^{\gamma} \\ &= \sum_{s=t}^{T} G(a+s, a+s+1)\beta^{s-t}[(B_{(a,e)}(s,t)+\delta)^{\gamma} - (B_{(a,e)}(s,t))^{\gamma}].\end{aligned}$$

Now, fix $\nu = \nu_t^*$, the threshold value for period t, we would like to calculate a one-time payment which equates the change in the discounted utility of future pension benefits and the change in current salary:

$$-\Delta U_{(a,e)}(t, \nu_t^*) + \nu_t^* = \Delta W_{(a,e)}(t).$$

that is, the maximum amount the teacher is willing to pay for such pension enhancement. Assume the teacher would retire at t given ν_t^*, we have:

$$[\kappa_t y_{(a,e)}(t)]^\gamma - [\kappa_t y_{(a,e)}(t) - \Delta y_{(a,e)}(t)]^\gamma = \Delta W_{(a,e)}(t).$$

from which we can solve for $\Delta y_{(a,e)}(t)$.[1] Then the WTP estimate is

$$WTP_{(a,e)}(t) = \frac{\Delta y_{(a,e)}(t)}{\Delta PW_{(a,e)}(t)}$$

Table C.3: WTP estimates for a female teacher with $a = 55, e = 28$

year t	$\delta = 1$	$\delta = 10$	$\delta = 100$
0	0.9580	0.9522	0.9486
1	0.9490	0.9432	0.9397
2	0.9404	0.9345	0.9311
3	0.9249	0.9189	0.9155
4	0.9169	0.9109	0.9076
5	0.9094	0.9032	0.9000
6	0.9019	0.8957	0.8926
7	0.8969	0.8906	0.8874
8	0.8931	0.8866	0.8836
9	0.8898	0.8834	0.8803
10	0.8874	0.8808	0.8777

Note: assume retiring under the 2.2 formula and ERO paid by the employer.

Using the estimated parameters for DP model with measurement error, we calculate the WTP for a female teacher with $a = 55, e = 28$ for different enhancement in pension benefit ($\delta = 1, 10, 100$) and report them in Table 3.5. The WTPs are around 0.9, decreases as the teacher getting older, and decreases when we increase the amount of enhancement due to the curvature of the utility function.[2]

[1] Note that the payment term $\Delta y_{(a,e)}(t)$ is not multiplied by the disutility of teaching, κ_t. otherwise, the WTP is greater than one due to consumption smoothing incentives.

[2] We also experiment for some other teachers, and the WTP estimates are also close to 0.9.

Bibliography

Adda, J., R. Cooper, and R. W. Cooper (2003). Dynamic economics: quantitative methods and applications. MIT press.

Arcidiacono, P. and R. A. Miller (2011). Conditional choice probability estimation of dynamic discrete choice models with unobserved heterogeneity. Econometrica 79(6), 1823–1867.

Backes, B., D. Goldhaber, C. Grout, C. Koedel, S. Ni, M. Podgursky, P. B. Xiang, and Z. Xu (2016). Benefit or burden? on the intergenerational inequity of teacher pension plans. Educational Researcher 45(6), 367–377.

Bajari, P., C. L. Benkard, and J. Levin (2007). Estimating dynamic models of imperfect competition. Econometrica 75(5), 1331–1370.

Barron, A. R. (1993). Universal approximation bounds for superpositions of a sigmoidal function. IEEE Transactions on Information theory 39(3), 930–945.

Belloni, M. (2008). The option value model in the retirement literature: the trade-off between computational complexity and predictive validity, Volume 50. CEPS.

Berkovec, J. and S. Stern (1991). Job exit behavior of older men. Econometrica: Journal of the Econometric Society, 189–210.

Blevins, J. R. (2016). Sequential monte carlo methods for estimating dynamic microeconomic models. Journal of Applied Econometrics 31(5), 773–804.

Brown, K. M. (2013). The link between pensions and retirement timing: Lessons from California teachers. *Journal of Public Economics* *98*, 1–14.

Chen, H., A. Didisheim, and S. Scheidegger (2021). Deep structural estimation: With an application to option pricing. *Available at SSRN*.

Chernozhukov, V., D. Chetverikov, M. Demirer, E. Duflo, C. Hansen, W. Newey, and J. Robins (2018). Double/debiased machine learning for treatment and structural parameters.

Ching, A. T., S. Imai, M. Ishihara, and N. Jain (2012). A practitioner's guide to bayesian estimation of discrete choice dynamic programming models. *Quantitative Marketing and Economics* *10*(2), 151–196.

Cho, I.-K. and T. J. Sargent (1996). Neural networks for encoding and adapting in dynamic economies. *Handbook of computational economics* *1*, 441–470.

Costrell, R. M. and J. B. McGee (2010). Teacher pension incentives, retirement behavior, and potential for reform in arkansas. *Education Finance and Policy* *5*(4), 492–518.

Costrell, R. M. and M. Podgursky (2009). Peaks, cliffs, and valleys: The peculiar incentives in teacher retirement systems and their consequences for school staffing. *Education Finance and Policy* *4*(2), 175–211.

Doherty, K. M. (2012). *No one benefits: How teacher pension systems are failing both teachers and taxpayers*. National Council on Teacher Quality.

Farmer, L. E. (2021). The discretization filter: A simple way to estimate nonlinear state space models. *Quantitative Economics* *12*(1), 41–76.

Farrell, M. H., T. Liang, and S. Misra (2020). Deep learning for individual heterogeneity. *arXiv preprint arXiv:2010.14694*.

Farrell, M. H., T. Liang, and S. Misra (2021). Deep neural networks for estimation and inference. *Econometrica* *89*(1), 181–213.

Fitzpatrick, M. D. (2015). How much are public school teachers willing to pay for their retirement benefits? *American Economic Journal: Economic Policy* *7*(4), 165–88.

Fitzpatrick, M. D. and M. F. Lovenheim (2014). Early retirement incentives and student achievement. *American Economic Journal: Economic Policy* *6*(3), 120–54.

Friedberg, L. and S. Turner (2010). Labor market effects of pensions and implications for teachers. *Education Finance and Policy* *5*(4), 463–491.

Gustman, A. L. and T. L. Steinmeier (1986). A structural retirement model. *Econometrica* *54*(3), 555–584.

Hotz, V. J. and R. A. Miller (1993). Conditional choice probabilities and the estimation of dynamic models. *The Review of Economic Studies* *60*(3), 497–529.

Hotz, V. J., R. A. Miller, S. Sanders, and J. Smith (1994). A simulation estimator for dynamic models of discrete choice. *The Review of Economic Studies* *61*(2), 265–289.

Igami, M. (2020). Artificial intelligence as structural estimation: Deep blue, bonanza, and alphago. *The Econometrics Journal* *23*(3), S1–S24.

Imai, S., N. Jain, and A. Ching (2009). Bayesian estimation of dynamic discrete choice models. *Econometrica* *77*(6), 1865–1899.

Iskhakov, F., J. Rust, and B. Schjerning (2020). Machine learning and structural econometrics: contrasts and synergies. *The Econometrics Journal* *23*(3), S81–S124.

Judd, K. L., L. Maliar, S. Maliar, and R. Valero (2014). Smolyak method for solving dynamic economic models: Lagrange interpolation, anisotropic grid and adaptive domain. Journal of Economic Dynamics and Control 44, 92–123.

Kim, D., C. Koedel, W. Kong, S. Ni, M. Podgursky, and W. Wu (2021). Pensions and late-career teacher retention. Education Finance and Policy 16(1), 42–65.

Knapp, D., K. Brown, J. Hosek, M. G. Mattock, and B. J. Asch (2016). Retirement Benefits and Teacher Retention. Rand Corporation.

Knapp, D., J. R. Hosek, M. G. Mattock, and B. J. Asch (2019). Predicting Retention Behavior: Ex Ante Prediction and Ex Post Realization of a Voluntary Retirement Incentive Offer. RAND.

LeCun, Y., Y. Bengio, and G. Hinton (2015). Deep learning. nature 521(7553), 436–444.

Lumsdaine, R. L., J. H. Stock, and D. A. Wise (1992). Three models of retirement: Computational complexity versus predictive validity. In Topics in the Economics of Aging, pp. 21–60. University of Chicago Press.

Malanga, S. and J. McGee (2018). Garden state crowd out: How new jersey's pension crisis threatens the state budget. New York: Manhattan Institute (January).

Ni, S. and M. Podgursky (2016). How teachers respond to pension system incentives: New estimates and policy applications. Journal of Labor Economics 34(4), 1075–1104.

Ni, S., M. Podgursky, and F. Wang (*forthcoming*, 2022). How much are public school teachers willing to pay for their retirement benefits? comment. American Economic Journal: Economic Policy.

Norets, A. (2009). Inference in dynamic discrete choice models with serially orrelated unobserved state variables. Econometrica 77(5), 1665–1682.

Norets, A. (2012). Estimation of dynamic discrete choice models using artificial neural network approximations. Econometric Reviews 31(1), 84–106.

Novy-Marx, R. and J. Rauh (2011). Public pension promises: how big are they and what are they worth? The Journal of Finance 66(4), 1211–1249.

Reich, G. (2018). Divide and conquer: Recursive likelihood function integration for hidden markov models with continuous latent variables. Operations Research 66(6), 1457–1470.

Rust, J. (1987). Optimal replacement of gmc bus engines: An empirical model of harold zurcher. Econometrica: Journal of the Econometric Society, 999–1033.

Rust, J. (1996). Numerical dynamic programming in economics. Handbook of computational economics 1, 619–729.

Rust, J. (1997). Using randomization to break the curse of dimensionality. Econometrica: Journal of the Econometric Society, 487–516.

Rust, J. (2000). Nested fixed point algorithm documentation manual. Unpublished Manuscript (6), 1–43.

Rust, J. and C. Phelan (1997). How social security and medicare affect retirement behavior in a world of incomplete markets. Econometrica, 781–831.

Semenova, V. (2018). Machine learning for dynamic discrete choice. arXiv preprint arXiv:1808.02569.

Stinebrickner, T. R. (2000). Serially correlated variables in dynamic, discrete choice models. Journal of Applied Econometrics 15(6), 595–624.

Stinebrickner, T. R. (2001). A dynamic model of teacher labor supply. Journal of Labor Economics 19(1), 196–230.

Stock, J. H. and D. A. Wise (1990). Pensions, the option value of work, and retirement. Econometrica 58(5), 1151–1180.

Su, C.-L. and K. L. Judd (2012). Constrained optimization approaches to estimation of structural models. Econometrica 80(5), 2213–2230.

Tauchen, G. (1986). Finite state markov-chain approximations to univariate and vector autoregressions. Economics letters 20(2), 177–181.

TRS (2018a). Evolution of the TRS benefit structure. Available at: `www.trsil.org/Evolution_of_TRS_Benefit_Structure`, retrieved 2018-09-19.

TRS (2018b). Tier 1 member guide. Available at: `www.trsil.org/members/retired/guide`, retrieved 2018-09-09.

VITA

Fangda Wang was born in Zhejiang, China. He majored economics at Renmin University of China and graduated in 2014. He was awarded a masters' degree in economics from Nanjing University in 2017, and went to University of Missouri-Columbia. He received a PhD in economics in 2022.