Will Artificial Intelligence Replace Computational Economists Any Time Soon?

Maliar, Maliar, and Winant

January 13, 2020

Table of contents

Goal

- Using AI to solve high-dimensional dynamic economic models.
- Solving (1) lifetime reward, (2) Bellman equation and (3) Euler equation.
- Introduce all-in-one integration technique that makes the stochastic gradient unbiased for the constructed objective functions.

Conceptually, you have seen this before: Use deep neural networks as an approximating functions, and alleviate the curse of dimensionality.

Strategy for solving these high dimensional models

- Defining an objective (loss) function to be minimized.
- Adapting a stochastic gradient descent method to train the constructed objective functions.
- Introducing integration methods that are suitable for the constructed objective functions in the context of deep learning based simulation.

Learning: Set up

- $\{x_i, y_i\}_{i=1}^n$ (iid) is observed, $x_i \in \mathbb{R}^{d_x}$, $y_i \in \mathbb{R}^{d_y}$.
- The goal of the machine(or sometimes the econometrician): finding a function $\phi : \mathbb{R}^{d_x} \to \mathbb{R}^{d_y}$ that provides the best prediction, given $\{x_i, y_i\}$ is observed.
- (Depends on who you ask) a parametric family of functions $\{\phi(., \theta) | \theta \in \mathbb{R}^{d_\theta}\}$.
- We need a loss function to minimize to find the best $\phi(., \theta)$. $l : \mathbb{R}^{d_x} \times \mathbb{R}^{d_y} \times \mathbb{R}^{d_\theta} \to \mathbb{R}$
- Define *expected risk* as:

$$\Xi(\theta) \equiv \int l\big(\phi(x; \theta), y\big) dP(x, y)$$

Not feasible!

Learning : Set up

- $\Xi^n(\theta) \equiv \frac{1}{n}\sum_{i=1}^{n} l\big(\phi(x_i;\theta), y_i\big)$
  Problem:
  $$\theta_{min} = \text{argmin}_{\theta\in\Theta}\Xi^n(\theta). \tag{1}$$
- Then $y = \phi(x;\theta_{min})$.
- OLS: $\phi(x;\theta) = \theta x$, $l(\phi(x;\theta), y) = (y-\theta x)^2$.
- This is called *supervised learning* because for each data point x_i, the machine is given correct output y_i to check its prediction $\phi(x_i,\theta)$.
- Not good for a computational economist, we dont get to see the correct y_i (think of ϕ as a policy function).
- $w \equiv (x, y)$,

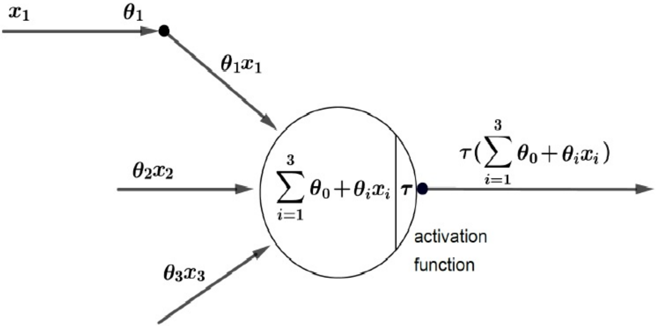
$$\Xi(\theta) = \mathbb{E}_w\big[\xi(w;\theta)\big] \to \frac{1}{n}\sum_{i=1}^{n}\xi(w_i;\theta). \tag{2}$$

Parametric family of functions: Multi-layer neural nets

A neural network consists of connected nodes, called *artificial neurons*. An artificial neuron consists of

- An ith input (a received signal) $(x_{i,0}, ..., x_{i,n})$, $x_{i,0} = 1$ by conventions
- Weighting an input bat parameters $\theta = (\theta_0, ..., \theta_n) \in \mathbb{R}^{n+1}$ (non-activated output)
- Sends an activated output $\in \mathbb{R}$: $\tau(\theta' x)$. τ is called an activation function:
    1. Sigmoid : $\tau(x) = \frac{1}{1+e^{-x}}$
    2. Heaviside: $\tau(x) = 1(x \geq 0)$
    3. relu : $\tau(x) = \max\{0, x\}$
    4. leaky relu : $\tau(x) = \max\{\kappa x, x\}$, $\kappa \leq 0$

A simple artificial neuron



Training a neural net

Assume $\{x_i, y_i\}_{i=1}^n = \{w_i\}_{i=1}^n$ is observed. Define:

$$\Xi^{n'}(\theta) = \frac{1}{n'} \sum_{i=1}^{n'} \xi(w_i; \theta), \;\; n' \leq n \tag{3}$$

Batch Gradient Descent: Choose an initial θ_0

$$\theta_{k+1} = \theta_k - \lambda_k \nabla_\theta \Xi^{n'}(\theta_k). \tag{4}$$

What is λ? Think of Newton-Raphson's method: We want to avoid the inverse of the Hessian.

Training a neural net: Summary

Algorithm 1. *DL algorithm for supervised learning.*

Step 1. Initialize the algorithm.

i). Set up an expected risk $\Xi(\theta) = E_\omega [\xi(\omega; \theta)]$.

ii). Define approximation $\varphi(\cdot, \theta)$ for φ, where $\theta \equiv [\vartheta, \lambda]$ and ϑ and λ are the approximation coefficients and hyperparameters of the algorithm, respectively.

iii). Define an empirical risk $\Xi_n(\theta) = \frac{1}{n} \sum_{i=1}^{n} \xi(\omega_i; \theta)$.

iv). Fix convergence criteria c_{inn} and c_{out} for inner and outer loops, respectively.

v). Split the data into 3 samples for constructing a solution (Sample 1), for validation (Sample 2) and for evaluating the accuracy (Sample 3).

Step 2. Train the machine, i.e., find θ that minimizes the empirical risk $\Xi_n(\theta)$.

Outer loop (validation on Sample 2): Fix the hyperparameters λ.

Inner loop (approximation on Sample 1): Fix the approximation coefficients ϑ.

Use data from Sample 1 to evaluate $\nabla_\vartheta \xi(\omega_i; \theta)$ (SGD or BGD) and update ϑ.

End the inner loop if the convergence criterion c_{inn} is reached.

Use data from Sample 2 for validation and update λ.

End the outer loop if if the convergence criterion c_{out} is reached.

Step 3. Assess the accuracy of constructed approximation $\varphi(\cdot, \theta)$ on Sample 3.

The economic problem of interest

Problem:

- Exogenous state in $\mathbb{R}^{n_m}$: $m_{t+1} = M(m_t, \epsilon_t)$
- Endogenous state in $\mathbb{R}$: $s_{t+1} = S(m_t, s_t, x_t, m_{t+1})$
- Choice variable $x_t \in \mathbb{R}^{n_x}$: $x_t \in X(m_t, s_t)$
- Period reward function: $r(m_t, s_t, x_t)$
- Agents problem:

$$\min_{\{x_t, s_{t+1}\}_{t=0}^{\infty}} \quad \mathbb{E}_0 \left[\sum_{t=0}^{\infty} \beta^t r\big(m_t, s_t, x_t\big) \right] \tag{5}$$

- Policy function: $x_t = \psi(m_t, s_t) \in X(m_t, s_t)$
- Approximating the policy function: $\phi(., \theta)$

Consumption-saving problem with four shocks

Problem:

- Agent's problem:

$$\begin{aligned} \min_{\{x_t, s_{t+1}\}_{t=0}^{\infty}} \quad & \mathbb{E}_0\left[\sum_{t=0}^{\infty} \beta^t e^{\chi_t} u(c_t)\right] \\ \text{s.t.} \quad & w_{t+1} = re^{\varrho_t}(w_t - c_t) + e^{y_t} e^{p_t(1-\mu)} \\ & c_t \leq w_t \end{aligned} \tag{6}$$

- $z_t \in \{y, p, \varrho, \chi\}$:

$$z_{j,t+1} = \rho_j z_{j,t} + \sigma_j \epsilon_{j,t}, \text{ and } \epsilon_{j,t} \sim \mathcal{N}(0,1).$$

- KKT:

$$c - w \leq 0, \; h \geq 0, \text{ and } (c - w)h = 0.$$

$$h \equiv u'(c)e^{\chi - \varrho} - \beta r \mathbb{E}_\epsilon\left[u'(c')e^{\chi'}\right], \; \frac{c_t}{w_t} \equiv \zeta_t = \sigma\big(\zeta_0 + \phi(z_t, w_t; \theta)\big).$$

Consumption-saving problem: Focusing on Euler equation

Problem:

- Fischer-Burmeister function : $\Psi^{\mathsf{FB}} = a + b - \sqrt{a^2 + b^2}$
- $\omega \equiv (z, w)$, the objective (loss, risk) function:

$$\Xi(\theta) = \mathbb{E}_\omega\big[\xi(\omega,\theta)\big] \equiv \mathbb{E}_\omega\left[\Psi^{\mathsf{FB}}\Big(w - c, u'(c) - \beta r e^{\varrho}\mathbb{E}_\epsilon[u'(c')]\Big)\right]^2 \quad (7)$$

Training results

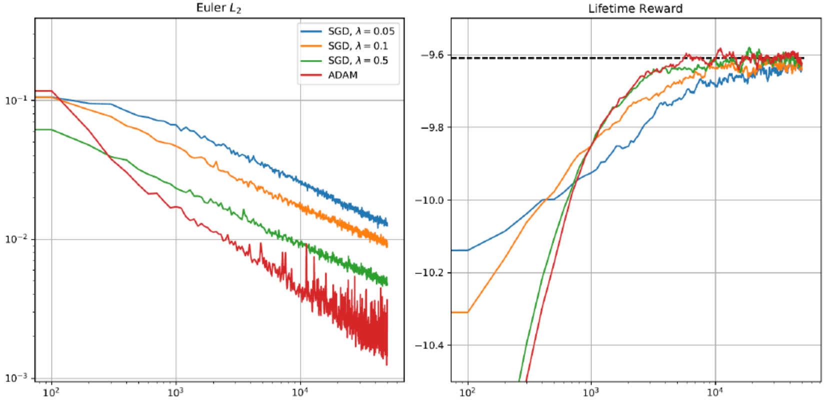


Figure 3. Training with a minimization of Kuhn-Tucker-conditions residuals in the baseline model.

Training results: Decision Rule

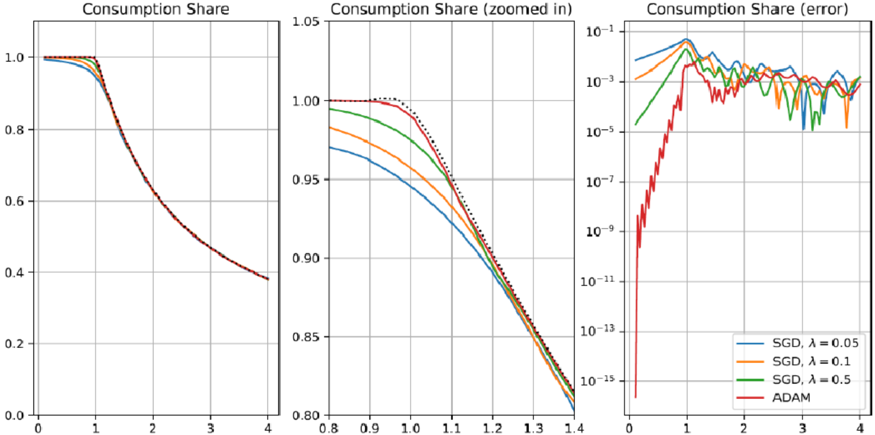


Figure 6. The consumption-share decision rule in the baseline model.

Conclusion

Problem:

- No!
- Maybe it is the time to move from model-specific methods to general-purpose AI technologies?
- In the paper, we propose one AI technology that makes economic models tractable: a deep learning method based on Monte Carlo simulation.
- No free lunch theorem: There's no such thing as a free lunch, unless you skip your dinner;)