Deep Equilibrium Nets for Solving Dynamic Stochastic Models

University of Lugano, 25-26 June 2024
Lecture 3: DEQN

Aleksandra Friedl

https://github.com/alexmalova/DEQN_lectures
friedl@ifo.de

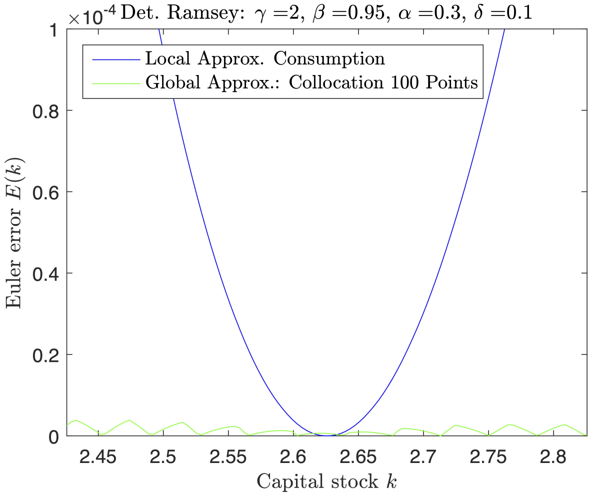
This course is inspired by Simon Scheidegger and based on his various teaching materials that I was relying on as a student and those which came as a result of our scientific collaboration

Outline of the presentation

Why we are doing what we are doing

- Modern economic models tend to be extremely rich in structure
- We want to capture:
  - **Heterogeneity**: distributional effects of the policies
  - Interaction of **climate and economy**
  - **Shocks**: aggregate and idiosyncratic
  - Borrowing constrains: **highly non-linear policies** potentially with kinks
- **Solution methods** (see Judd 1998):
  - **Local**: perturbation methods
  - **Global**: projection methods

Why a global solution method?



From Gerzensee lectures on Numerical methods, 2021, Felix Kuebler

Dynamic Stochastic Model: global solution method

- x: point in state space that describes the modeled system
    - State-space potentially irregularly-shaped and high-dimensional
- $p(x)$: time-invariant policy function
- traditional global solution: high-dimensional functions on which one interpolates
    - N^d points in ordinary discretization schemes
    - Curse of dimensionality
    - Usually: solve many non-linear systems of equations by invoking a solver

What is high-dimensional?

# State Variables	# Points	Time-to-solution
1	10	10 sec
2	100	1.6 min
3	1000	16 min
4	10'000	2.7 hrs
5	100'000	1.1 day
6	1'000'000	1.6 weeks
...	...	...
20	1e20	3 trillion years
Dimension reduction Exploit symmetries, e.g., via the active subspace method	**Deal with number of points** e.g., via Smolyak or adaptive sparse grids	**HPC** Exploit compute power, e.g., via parallelisation

What we are dealing with

Problem

- High-dimensional problem
- Non-linear irregularly shaped state spaces
- Stochastic nature of the problem

Solution

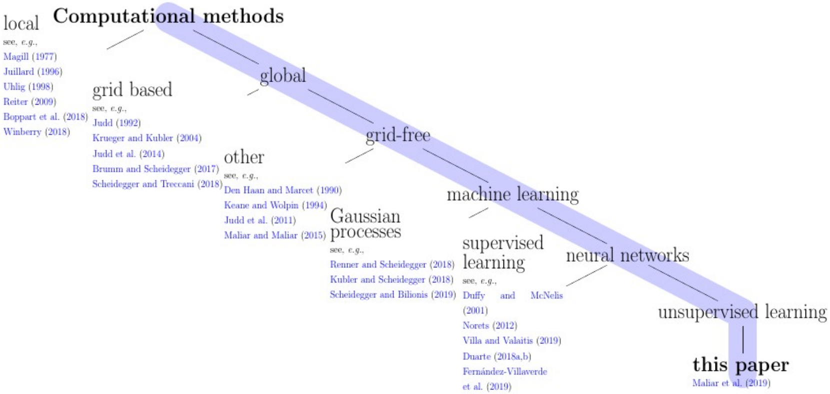
We offer a generic solution framework based on **neural networks** to solve highly-complex dynamic stochastic models.

Deep Equilibrium Nets (DEQN)

DEQN key ideas

1. Use the the implied error in the optimality conditions, as loss function.
2. Learn the equilibrium functions with stochastic gradient descent.
3. Take the (training) data points from a simulated path $\rightarrow$ can be generated at virtually zero cost.

An incomplete list of related literature



from Azinovic, 2022

Outline of the presentation

What is a deep neural network?

- Neural networks are universal function approximators
- A neural net is characterized by its parameters ρ
- Given a parameter vector ρ and an input vector x, denote the neural net as $\mathcal{N}_\rho$ and some desired function with f:

$$\mathcal{N}_\rho : \mathbb{R}^{N_{in}} \to \mathbb{R}^{N_{out}}, \ \mathcal{N}_\rho : x \to \mathcal{N}_\rho(x)$$
$$f : \mathbb{R}^{N_{in}} \to \mathbb{R}^{N_{out}}, \ f : x \to f(x)$$

We desire parameters ρ, such that

$$||\mathcal{N}_\rho - f||_{\text{some norm}} = 0$$

What is a deep neural network?

- **Neural network** is *the universal function approximator*
- A neural net $\mathcal{N}_\rho$ is **characterized by its parameters** ρ

$$p : \mathbb{R}^{N_{in}} \to \mathbb{R}^{N_{out}}, \; p : x \to p(x)$$
$$\mathcal{N}_\rho : \mathbb{R}^{N_{in}} \to \mathbb{R}^{N_{out}}, \; \mathcal{N}_\rho : x \to \mathcal{N}_\rho(x)$$

We desire parameters ρ, such that **loss function**

$$||\mathcal{N}_\rho - p||_{\text{some norm}} = 0.$$

DEQN technical details

- DEQN relies on **feedforward neural network (FNN)**
- FNN: L layers, each layer has N_l neurons and an activation function
- E.g., the neural network with **2** hidden **layers, 12 neurons** each, activated by *relu*.

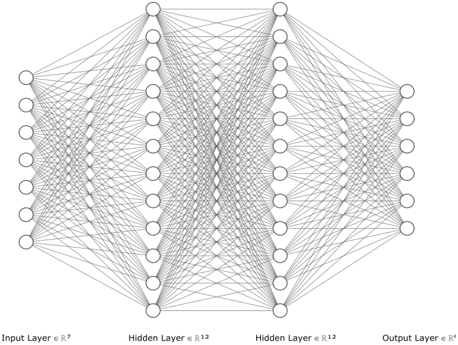
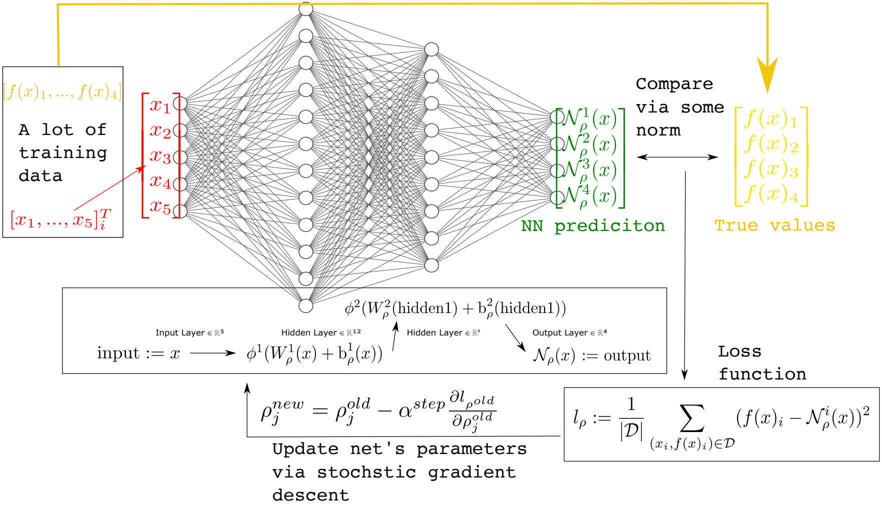
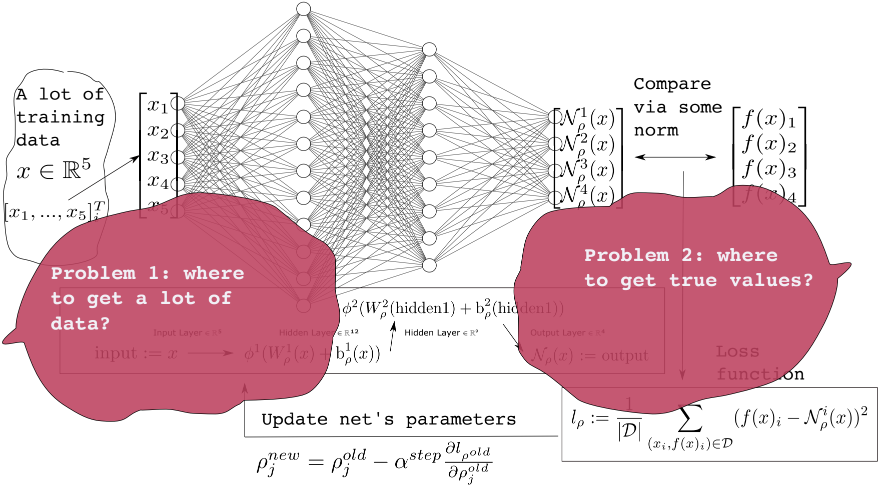


Figure: FNN, the input $\mathbf{x}$ is an 7−dimensional vector, two hidden layers with 12 neurons each, and $\mathbf{p}(\mathbf{x})$ is a 6−dimensional output.

Neural network at a glance



Neural network at a glance: problem



Economic loss function: [Azinovic, Gaegauf, and Scheidegger(2019)]

1. From **equilibrium conditions** we know the relationship between the state x_i and the policy function $p(x_i)$:

$$G(x_i, p(x_i)) = 0$$

2. We propose an **economic loss function**

$$l_\rho := \frac{1}{|\mathcal{D}|} \sum_{x_i \in \mathcal{D}} \left(G(x_i, \mathcal{N}_\rho(x_i))\right)^2$$

3. We train the net using **the stochastic gradient descent** such that the policy function produced by the net satisfies equilibrium conditions almost exactly.

How DEQN works

1. Formulate a **set of first-order conditions** $G(x_t, p(x_t)) = 0$
2. Activate **neural net** $\mathcal{N}_\rho(x_t)$ with the **random parameters** ρ
3. Take the starting state of the economy and **simulate** the evolution over time $N_{\text{path length}}$ with the $\mathcal{N}_\rho(x_t)$. We get: $G(x_t, \mathcal{N}_\rho(x_t)) \neq 0$
4. Use the loss function to **update** the neural net parameters ρ

$$l_\rho := \frac{1}{N_{\text{path length}}} \sum_{x_t \text{on sim. path}} \Big(G(x_t, \mathcal{N}_\rho(x_t))\Big)^2$$

5. **Repeat** steps 3 and 4 until $l_\rho \approx 0$.

Advantages of the DEQN approach

- **Alleviates the curse of dimensionality** due to the stochastic gradient descent procedure
- Capable of **approximating policy functions with strong nonlinearities** as the whole ergodic distribution of policy variables is being determined
- Allows to deal with **large state spaces** (up to several hundreds of variables), so we can have multiple-agents in the model.

Outline of the presentation

Model specification and equilibrium conditions

Model specification:

$$\max_{\{C_t\}_{t=0}^{\infty}} \sum_{t=0}^{\infty} \beta^t \frac{(C_t/L_t)^{1-1/\psi}}{1-1/\psi} L_t$$

$$\begin{aligned} \text{s.t. } K_{t+1} &= (1-\delta)K_t - C_t + A_t K_t^{\alpha} L_t^{1-\alpha} \quad [\lambda_t] \\ 0 &\leq C_t \\ 0 &\leq K_{t+1} \end{aligned}$$

where labor and TFP evolve exogenously

$$L_t = L_0 + (L_\infty - L_0)\left(1 - \exp\left(-\delta^L t\right)\right)$$

$$A_t = A_0 \exp\left(\frac{\alpha_1(1-\exp(-\alpha_2 t))}{\alpha_2}\right)$$

Functional rational expectations equilibrium (FREE)

- Let us denote the state of the economy by x, the policy by y, and the policy function by f, thus

$$x = \{k_t, \tau_t\} \in \mathbb{R}^1,\ \ y = \left\{\hat{k}_{t+1}, \hat{\lambda}_t, \hat{c}_t\right\} \in \mathbb{R}^3$$
$$f : x \mapsto y, \quad f : \mathbb{R}^2 \mapsto \mathbb{R}^3$$

- Equilibrium conditions:

$$(\hat{c}_t / L_t)^{-1/\psi} - \hat{\lambda}_t \stackrel{!}{=} 0$$
$$-\hat{\lambda}_t + \beta \hat{\lambda}_t \left(1 - \delta + A_t \alpha \hat{k}_{t+1}^{\alpha-1} L_t^{1-\alpha}\right) \stackrel{!}{=} 0$$
$$\hat{\lambda}_t \left((1-\delta) k_t - \hat{c}_t + A_t k_t^{\alpha} L_t^{1-\alpha} - \hat{k}_{t+1}\right) \stackrel{!}{=} 0$$

Nonstationary Ramsey problem and DEQN ([Azinovic, Gaegauf, and Scheidegger(2019)])

Initial state $[k_0, \tau_0]$

Randomly initialized net $\mathcal{N}_\rho^0$

$\mathcal{N}_\rho^{old} = \mathcal{N}_\rho^0$

Initial state $[k_0, \tau_0]$

Updated net $\mathcal{N}_\rho^{old} = \mathcal{N}_\rho^{new}$

`Generate a lot of data`

$$[\hat{k}_1, \hat{\lambda}_1, \hat{c}_1] = \mathcal{N}_\rho^{old}([k_0, \tau_0])$$
$$[\hat{k}_2, \hat{\lambda}_2, \hat{c}_2] = \mathcal{N}_\rho^{old}([\hat{k}_1, \tau_1])$$
$$\vdots$$
$$[\hat{k}_{t+1}, \hat{\lambda}_{t+1}, \hat{c}_{t+1}] = \mathcal{N}_\rho^{old}([\hat{k}_t, \tau_t])$$
$$\vdots$$
$$[\hat{k}_T, \hat{\lambda}_T, \hat{c}_T] = \mathcal{N}_\rho^{old}([\hat{k}_{T-1}, \tau_{T-1}])$$

`Euler Errors`

$$e^{EEc}(\rho) = \left(\frac{\hat{c}_t}{L_t}\right)^{-1/\psi} - \hat{\lambda}_t \stackrel{!}{=} 0$$
$$e^{EEk}(\rho) = -\hat{\lambda}_t + \beta \hat{\lambda}_{t+1} \left(1 - \delta + A_{t+1} \alpha \hat{k}_{t+1}^{\alpha-1} L_{t+1}^{1-\alpha}\right) \stackrel{!}{=} 0$$
$$e^{KKT}(\rho) = \hat{\lambda}_t \left((1-\delta)\hat{k}_t - \hat{c}_t + A_t \hat{k}_t^{\alpha} L_t^{1-\alpha} - \hat{k}_{t+1}\right) \stackrel{!}{=} 0$$

`Loss function`

$$l_{\rho^{old}} = \frac{1}{|\mathcal{D}|} \sum_{x \in \mathcal{D}} \left[\omega_1 \left(e_x^{EEc}\left(\rho^{old}\right)\right)^2 + \omega_2 \left(e_x^{EEk}\left(\rho^{old}\right)\right)^2 + \omega_3 \left(e_x^{KKT}\left(\rho^{old}\right)\right)^2\right]$$

`Update nets' parameters`

$$\rho_i^{new} = \rho_i^{old} - \alpha^{step} \frac{\partial l_{\rho^{old}}}{\partial \rho_i^{old}}$$

`Stochastic gradient descent`

Investment in the nonstationary Ramsey model dependent on capital

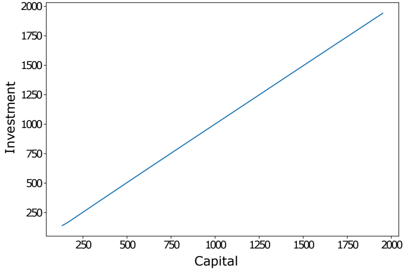


Figure: Investment DEQN

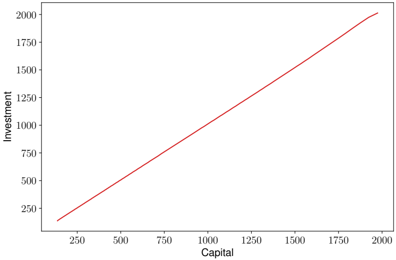


Figure: Investment VFI

Investment in the nonstationary Ramsey model dependent on year

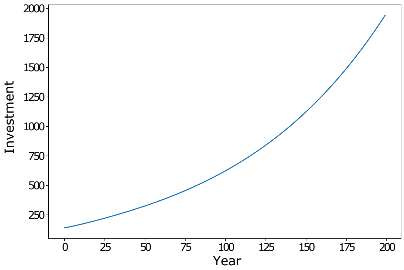


Figure: Investment DEQN

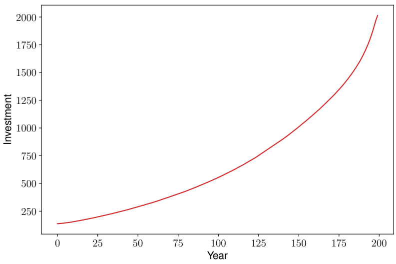


Figure: Investment VFI

Consumption in the nonstationary Ramsey model dependent on year

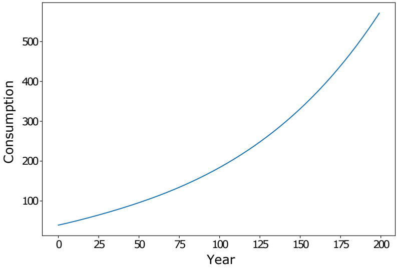


Figure: Consumption DEQN

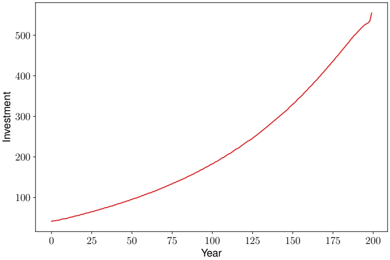


Figure: Consumption VFI

Capabilities of the DEQN

- Solve large non-linear models with stochasticity
- Solve the models for all the values of the parameters in one go (pseudo-states)
- Deep surrogates: solve the model and calibrate the parameters at the same time

Outline of the presentation

To do:

- Let's look at the Brock Mirman problem: 01_Brock_Mirman_1972_DEQN.ipynb
- Rewrite the code such that you change the equilibrium conditions (for example as in the slide 21).

Questions?