Advanced Differential Geometry

Natural Gradient Descent and Real Business Cycle Model

Longye Tian 8095061

Mathematical Sciences Institute
Australian National University
July 8, 2025

Natural Gradient Descent and Real Business Cycle Model

Longye Tian u809061

Australian National University

July 8, 2025

Abstract

This research project aims at exploring the intersection between differential geometry, deep neural network and macroeconomic modelling. It focuses using natural policy gradient descent algorithm to solve the standard Real Business Cycle (RBC) model via deep neural network. In this project, we introduce the basic definitions for neural network and motivate the use of natural gradient methods. We also conduct a brief literature review to identity possible gaps in the literature. Last, we present our solutions of RBC model using natural gradient method and compared with the standard gradient algorithm. The final result is mixed and we discussed this mixed result at the end.

Keywords: Natural gradient descent, deep neural network, Riemannian manifolds, Fisher information metric, Real Business Cycle models, Information geometry

1 Introduction

Deep learning (DL) has becoming a popular tool to solve economic models. Recent studies by [1], [2] have used DL tools to tackle high-dimensional problems in economics. And the key algorithm to solve the neural network is the gradient descent algorithm see [3].

One problem when using the standard gradient descent algorithm is when the output is a probability distribution, the standard gradient descent algorithm is not working efficiently. This is because the geometry of the parameter space is not Euclidean.

This motivates the development of natural gradient method by [4]. Natural gradient descent lies at the intersection of differential geometry and optimization. In this research report, we explore this natural gradient descent algorithm and apply it onto a standard RBC model.

Section 2 presents the required information about neural network, gradient descent algorithm. Section 3 conducts a brief literature review to motivate this research topic. Section 4 presents the RBC model and solution using natural gradient descent. Section 5 concludes.

2 Neural Network and Gradient Descent

A standard DL macroeconomic problem as in [5] is to approximate the solution of the macroeconomic model. In general we are approximating a function given a set of optimization constraint.

Let σ denote the solution of the model, we assume it can be parameterized by θ. Then we compare the result from the approximated solution with the desired output and measure this discrepancy using a loss function

$$L(x, y, \theta) = (\sigma(x, \theta) - y)^2 \tag{1}$$

where x, y denote the input and output respectively. This gives us an expected error for different value of parameters

$$Error(\theta) = \int L(x, y, \theta)\, dP(x, y) \tag{2}$$

where $P(x, y)$ is the joint-distribution of input and output.

To solve for the solution is equivalent to find the parameters that minimizes the expected error, i.e.,

$$\theta^* = \arg\min_{\theta} Error(\theta) \tag{3}$$

Moreover, to make sure we can approximate our solution well, we use a deep neual network structure. As in [3], the basic unit in neural networks are called neurons. A neuron is a function mapping from $\mathbb{R}^n$ to $\mathbb{R}$:

$$f(x; \lambda) = g\left(\lambda_0 + \lambda_1 x_1 + \cdots + \lambda_n x_n\right) \tag{4}$$

where g is called activation function. Usually it is a nonlinear function chosen by the designer of the neural network.

By stacking the neurons together, we get a layer of the neural network, i.e.,

$$F(x; \Lambda) = \begin{pmatrix} f_1(x, \lambda^1) \\ f_2(x, \lambda^2) \\ \cdots \\ f_p(x, \lambda^p) \end{pmatrix} \tag{5}$$

where $\Lambda = (\lambda^1, \cdots, \lambda^p)$. This vector function F stacks p neurons together and p is called the width of this layer.

Finally, a neural network is a composition of layers, i.e.,

$$NN(x, \theta) = (F_k \circ F_{k-1} \circ \cdots \circ F_1)(x) \tag{6}$$

where θ denote all parameters involved in the neural network.

To minimize the objective function $Error(\theta)$, most of the current literature are using the standard gradient descent algorithm as shown in [3].

The gradient of the loss function points towards the direction of the steepest ascent, hence, the algorithm updates the parameter vector θ in the opposite direction of the gradient, i.e.,

$$\theta_{t+1} = \theta_t - \alpha \nabla_\theta L(\theta_t) \tag{7}$$

where α is the stepsize, called the learning rate.

2.1 Natural Gradient Descent

The standard gradient descent algorithm implicitly assumes that the parameter space is Euclidean, which may not be true in some setting. In particular, when parameters represent probability distributions. When optimizing over probability distributions, the parameter space forms a statistical manifold with a non-Euclidean geometry. This geometric insight, pioneered by Amari [4], leads to the natural gradient method.

2.1.1 Geometric Motivation

Following [4], we consider a parametric family of probability distributions $\{p(x|\theta) : \theta \in \Theta \subseteq \mathbb{R}^n\}$. The standard gradient descent updates parameters according to Equation 7, However, this update assumes that the distance between θ and $\theta + d\theta$ is measured by the Euclidean metric $||d\theta||^2 = \sum_i (d\theta_i)^2$.

In the context of probability distributions, a more appropriate distance measure is the Kullback-Leibler (KL) divergence:

$$D_{KL}(p(x|\theta)||p(x|\theta + d\theta)) = \int p(x|\theta) \log \frac{p(x|\theta)}{p(x|\theta + d\theta)} dx \tag{8}$$

For small $d\theta$, the KL divergence can be approximated using a second-order Taylor expansion:

$$D_{KL}(p(x|\theta)||p(x|\theta + d\theta)) \approx \frac{1}{2} d\theta^T G(\theta) d\theta \tag{9}$$

where $G(\theta)$ is the Fisher information matrix.

2.1.2 Fisher Information Metric

As shown in [4], the Fisher information matrix serves as a Riemannian metric on the statistical manifold:

$$g_{ij}(\theta) = \mathbb{E}_{p(x|\theta)} \left[\frac{\partial \log p(x|\theta)}{\partial \theta_i} \frac{\partial \log p(x|\theta)}{\partial \theta_j} \right] \tag{10}$$

According to [6], this metric is invariant under reparameterization of θ, it is unique Riemannian metric, up to scaling, that is invariant under sufficient statistics. Moreover, it provides a quadratic approximation to the KL divergence as discussed before.

2.1.3 Natural Gradient Update Rule

The natural gradient descent algorithm discusse in [7] modifies the gradient descent update to account for the geometry of the parameter space. Instead of moving in the direction of steepest descent in Euclidean space, we move in the direction of steepest descent on the statistical manifold, i.e.,

$$\tilde{\nabla}_\theta L(\theta) = G^{-1}(\theta)\nabla_\theta L(\theta) \tag{11}$$

where $\tilde{\nabla}_\theta L(\theta)$ is called the natural gradient by [4]. Hence, the natural gradient descent update becomes:

$$\theta_{t+1} = \theta_t - \alpha_t G^{-1}(\theta_t)\nabla_\theta L(\theta_t) \tag{12}$$

3 Literature Review

This section presents a short review of two research strands: the development of natural gradient methods and the application of deep learning techniques to macroeconomic modeling. One research gap is that most DL solution method in macroeconomics does not take into account the geometry of probability distribution.

3.1 Natural Gradient Methods in Machine Learning

3.1.1 Theoretical Foundations

Amari [4] developed the natural gradient method by treating the parameter spaces as Riemannian manifolds equipped with the Fisher information metric. This creates a coordinate-invariant optimization method that respects the underlying geometry of parameter space. In [6], he proves how the Fisher information matrix serves as the natural metric tensor on statistical manifolds.

This geometric perspective enables more efficient navigation of the loss landscape, particularly in regions where standard gradient methods struggle, such as plateaus and saddle points as shown in [7].

3.1.2 Practical Limitations

From the perspective of computational complexity, calculating the Fisher information matrix is computationally consuming. In other words, it takes more computational power compared to the standard gradient descent. This is because we need to calculate the Fisher information matrix first. This limits the practical applications of natural gradient methods.

To address this computational difficulty, Martens and Grosse [8] introduced Kronecker-Factored Approximate Curvature (K-FAC) , a scalable approximation that use the structure of neural networks to make natural gradient methods computationally viable for deep learning.

3.2 Deep Learning in Macroeconomic Modeling

Macroeconomic models, particularly dynamic stochastic general equilibrium (DSGE) models, have traditionally relied on linearization techniques and perturbation methods to approximate solutions to complex systems [2]. These approaches face significant computational challenges when dealing with high-dimensional state spaces, occasionally binding constraints, or non-normal shock distributions [5].

Recent research has explored deep learning as an alternative solution method for macroeconomic models. Maliar et al. [1] use deep neural networks to approximate value functions and decision rules in dynamic economic models. Their work empirically demonstrates that neural networks can capture nonlinearities and handle high-dimensional state spaces that challenge traditional solution methods.

Building on this foundation, Beck et al. [5] provides a complete tutorial on how to use deep neural network to solve a DSGE model. In particular, they use standard RBC model as an example.

3.2.1 Reserach Gap

The literature review reveals a clear research gap at the intersection of natural gradient methods and macroeconomic modeling. While deep learning has been successfully applied to solve complex economic models, researchers have primarily relied on standard stochastic gradient descent and its variants.

To date, no research has systematically investigated whether the theoretical advantages of natural gradient methods translate to practical benefits when applied to deep learning solutions for macroeconomic models.

4 Applications

4.1 The RBC Model

We consider a standard real business cycle model with a representative household and a representative firm following [9]. Time is discrete and indexed by $t = 0, 1, 2, \ldots$.

4.2 Households

Following [9], the standard representative household chooses sequences of consumption $\{C_t\}_{t=0}^{\infty}$ and labor supply $\{N_t\}_{t=0}^{\infty}$ to maximize expected lifetime utility:

$$\mathbb{E}_0 \sum_{t=0}^{\infty} \beta^t U(C_t, N_t) \tag{13}$$

where $\beta \in (0,1)$ is discount factor, with constant-relative risk averse utility function

$$U(C_t, N_t) = \frac{C_t^{1-\sigma}}{1-\sigma} - \chi \frac{N_t^{1+\psi}}{1+\psi} \tag{14}$$

where $\sigma > 0$ is the coefficient of relative risk aversion, $\psi \geq 0$ is the inverse of the Frisch elasticity of labor supply, and $\chi > 0$ is a scaling parameter for the disutility of labor.

The household faces the following budget constraint:

$$C_t + I_t = W_t N_t + R_t K_t + \Pi_t \tag{15}$$

where W_t is the real wage, R_t is the rental rate of capital, K_t is the capital stock at the beginning of period t, I_t is investment, and Π_t represents firm profits. Moreover, capital accumulates according to:

$$K_{t+1} = (1-\delta)K_t + I_t \tag{16}$$

where $\delta \in (0,1)$ is the depreciation rate of capital.

4.3 Firms

Following [9], the standard representative firm operates a Cobb-Douglas production function

$$Y_t = A_t K_t^{\alpha} N_t^{1-\alpha} \tag{17}$$

where $\alpha \in (0,1)$ is the capital share of output, and A_t is total factor productivity (TFP) that follows an AR(1) process:

$$\log(A_t) = \rho \log(A_{t-1}) + \varepsilon_t \tag{18}$$

where $\rho \in [0,1)$ is the persistence parameter and $\varepsilon_t \sim N(0, \sigma_\varepsilon^2)$ is an i.i.d. productivity shock. The firm maximizes profits:

$$\Pi_t = Y_t - W_t N_t - R_t K_t \tag{19}$$

taking prices as given.

4.4 Equilibrium Conditions

4.4.1 First-Order Conditions

The household's optimization problem yields the following first-order conditions:

Labor-Leisure Choice:

$$\chi N_t^{\psi} = C_t^{-\sigma} W_t \tag{20}$$

Euler Equation:

$$C_t^{-\sigma} = \beta \mathbb{E}_t \left[C_{t+1}^{-\sigma} (R_{t+1} + 1 - \delta) \right] \tag{21}$$

The firm's profit maximization yields:

Labor Demand:

$$W_t = (1-\alpha) A_t K_t^{\alpha} N_t^{-\alpha} \tag{22}$$

Capital Demand:

$$R_t = \alpha A_t K_t^{\alpha-1} N_t^{1-\alpha} \tag{23}$$

4.4.2 Market Clearing and Resource Constraint

Following [9], in equilibrium, all markets clear. The resource constraint of the economy is:

$$Y_t = C_t + I_t \tag{24}$$

Combining this with the capital accumulation equation:

$$Y_t = C_t + K_{t+1} - (1-\delta)K_t \tag{25}$$

4.4.3 Complete System of Equilibrium Conditions

The competitive equilibrium is characterized by sequences $\{C_t, N_t, K_{t+1}, Y_t, W_t, R_t, I_t\}_{t=0}^{\infty}$ that satisfy:

$$\chi N_t^{\psi} = C_t^{-\sigma} W_t \tag{26}$$
$$C_t^{-\sigma} = \beta \mathbb{E}_t \left[C_{t+1}^{-\sigma} (R_{t+1} + 1 - \delta) \right] \tag{27}$$
$$W_t = (1-\alpha) A_t K_t^{\alpha} N_t^{-\alpha} \tag{28}$$
$$R_t = \alpha A_t K_t^{\alpha-1} N_t^{1-\alpha} \tag{29}$$
$$Y_t = A_t K_t^{\alpha} N_t^{1-\alpha} \tag{30}$$
$$Y_t = C_t + I_t \tag{31}$$
$$K_{t+1} = (1-\delta) K_t + I_t \tag{32}$$
$$\log(A_t) = \rho \log(A_{t-1}) + \varepsilon_t \tag{33}$$

given initial conditions K_0 and A_0, and the stochastic process for productivity shocks.

4.5 Application

Following [5], we use a neural network to approximate consumption as a function of capital. And we put the entire complete system of equilibrium conditions into the loss function. This forms a Economics-Informed Neural Network.

Moreover, to calibrate using US data, we choose the parameters as follows

- Discount factor $\beta = 0.99$
- Relative risk aversion $\sigma = 2.0$
- Capital share in production $\alpha = 0.33$
- Depreciation rate $\delta = 0.025$
- Persistence of productivity $\rho = 0.95$
- Standard deviation of productivity shock $\sigma_{\epsilon} = 0.01$

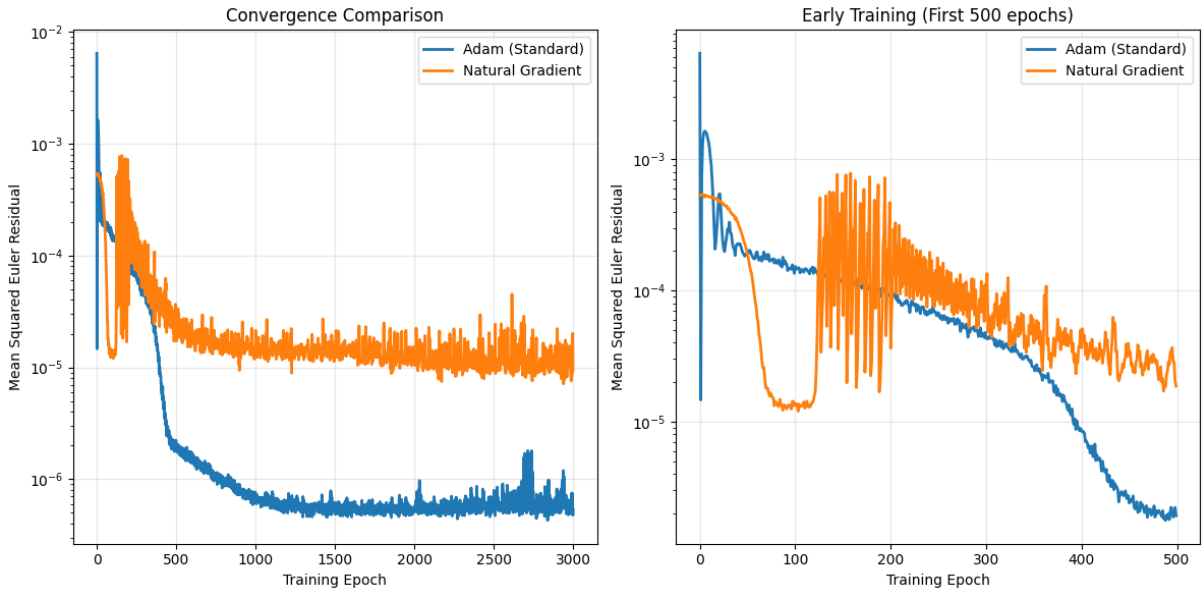

Figure 1: Neural Network Error Comparison

Calibrating the model using the above parameters is standard according [9]. In constrast to [5], we use a natural gradient descent approach as discussed before. As shown in Figure 1, the approximation error of natural gradient descent shrinks to very low level very quickly compared to the standard gradient descent. Although later on, the error jumps back compared to the standard gradient descent algorithm. This is a mixed result in the standard RBC model.

One way to interpret this mixed result is that the standard RBC model has a deterministic policy function which is a special case of stochastic policy function and standard gradient descent algorithm like Adam maybe more specialized in this particular version. This result, hence, motivates further investigation in this field.

5 Conclusion

This project focus on the intersection betweeen differential geometry, gradient descent algorithm in deep neural network and macroeconomic modelling. In particular, we demonstrate the procedure to apply natural gradient algorithm on a RBC model.

From our literature review, neural network solution in macroeconomics is new and no existing literature connect between differential geometry and computational economics. Following [7], we recognize neural network parameter spaces form statistical manifolds equipped with the Fisher information metric.

Our empirical results show rapid convergence with approximation errors decreasing to 10^{-5} within 5000 training epochs, confirming the theoretical advantages of respecting the intrinsic geometry of parameter spaces.

But at the same time, we find that standard gradient descent algorithm gives lower approximation error after 5000 epochs. This indicates a mixed result and motivates further investigations.
The natural gradient framework is particularly relevant for economic models where parameters represent probability distributions or where the loss landscape exhibits complex curvature. While computational costs remain a challenge—particularly the $\mathcal{O}(n^3)$ complexity of inverting the Fisher matrix—approximation methods like K-FAC offer promising scalability solutions.

Future research directions include: (i) extending to heterogeneous agent models where dimensionality challenges are more severe, (ii) developing adaptive learning rate schemes based on local curvature, (iii) applying natural gradient methods to policy optimization in monetary and fiscal settings

This research project demonstrates that the geometry of optimization matters in computational economics. As we tackle increasingly complex models with high-dimensional state spaces, geometric methods offer principled approaches to navigate these challenging landscapes. We hope this work can encourage broader adoption of differential geometric tools in computational economics.

References

[1] Lilia Maliar, Serguei Maliar, and Pablo Winant. "Deep learning for solving dynamic economic models." In: *Journal of Monetary Economics* 122 (2021), pp. 76–101.

[2] Jesús Fernández-Villaverde et al. "Solving high-dimensional dynamic programming problems using deep learning". In: *Unpublished working paper* (2020).

[3] Ian Goodfellow et al. *Deep learning*. Vol. 1. 2. MIT press Cambridge, 2016.

[4] Shun-Ichi Amari. "Natural gradient works efficiently in learning". In: *Neural computation* 10.2 (1998), pp. 251–276.

[5] Pierre Beck et al. *Deep learning solutions of DSGE models: a technical report*. Tech. rep. Central Bank of Luxembourg, 2024.

[6] Shun-ichi Amari and Hiroshi Nagaoka. *Methods of information geometry*. American Mathematical Society, 2000.

[7] Shun-ichi Amari. *Information geometry and its applications*. Vol. 194. Springer, 2016.

[8] James Martens and Roger Grosse. "Optimizing neural networks with kronecker-factored approximate curvature". In: *International conference on machine learning*. PMLR. 2015, pp. 2408–2417.

[9] Jordi Galí. *Monetary policy, inflation, and the business cycle: an introduction to the new Keynesian framework and its applications*. Princeton University Press, 2015.