Question 4: Machine Learning for Economics

1. The project proposal of this part simply consists of replication of the results of Childers(22) in Tensorflow and Tensorflow Probability.
   a. Are there any typos and errors in the paper?
   b. Are there any parts that are unclear?
   c. Other than replicating their results, can you write a short note to explain some of the difficult points that may be unclear to readers unfamiliar with the subject?
2. For the DSGE model in the paper, suppose the utility function of the households is an Epstein and Zin function, like the one in this paper: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=1995735:
   a. Can we apply the method in Childers(22)? If yes, can you write a program to show the results? If no, what can we do?
   b. Is the Caldara, Fernandez-Villaverde, Rubio-Ramirez and Yao (2012) paper above useful?
   c. Do you think the neural network method in this paper https://web.stanford.edu/~maliars/Files/JME2021.pdf is helpful? Can we train a neural network solution which takes the parameters of the DSGE model as input so that we do not have to retrain the neural network during estimation?

Literature review

1. George Tauchen (1986) : Finite State Markov-Chain approximation to Univariate and Vector Autoregressions
   a. Summary and significance
   - Revolutionary paper by Tauchen (Nobel winner Thomas Sargent's student) on providing analytically closed form solution of Markov-Chain to continuous process such as
     - $u^{'}(h)p(h) = \beta \int u^{'}(h^{'})[p(h^{'}) + h^{'}]F\left(\frac{dh^{'}}{h}\right)$, where $u^{'}(h)$ is the marginal utility of consumption, (h) is the dividend, β is the subjective rate of discount, and $F\left(\frac{dh^{'}}{h}\right)$ is the conditional distribution of the dividend.
   - It develops a method for choosing values for the state variables and the transition probabilities so that the resulting finite-state Markov chain mimics closely an underlying continuous valued autoregression like one above
   - **First ever theoretical paper on a method for finding a discrete Markov chain that approximates in the sense of weak convergence a continuous-valued univariate or multivariate autoregression. It is important to find discrete state spaces are used for finding numerical solutions to integral equations.**

   b. Development of theory
      1. With typical AR process $y_t = \lambda y_{t-1} + \epsilon_t$, where ϵ_t is a white noise process with variance σ_ϵ^2. The distribution function of ϵ_t be $\Pr[\epsilon_t \ll u] = F(\frac{u}{\sigma_\epsilon})$ where F is a cumulative distribution with unit variance.

2. $\tilde{y}_t$ denote the discrete-valued process that approximates the continuous process above, and $\bar{y}^1 < \bar{y}^2, \ldots, < \bar{y}^N$ denote the values that $\tilde{y}_t$ takes on.
3. $\bar{y}^j$ is chosen with a multiple of m of the unconditional standard deviation $\sigma_y = (\sigma_\epsilon^2/(1-\lambda^2)^{1/2}, \bar{y}^1 = \bar{y}^{-N}$, then, equispaced from $1\ to\ N$
4. The transition matrix is calculated as follow $p_{jk} = Pr[\tilde{y}_t = \bar{y}^k | \tilde{y}_{t-1} = \bar{y}^j]$. $w = \bar{y}^k - \bar{y}^{k-1}$. For each j, if k is between $2\ and\ N-1$, $p_{jk} = Pr\left[\bar{y}^k - \frac{w}{2} \leq \lambda\bar{y}^j + \epsilon_t \leq \bar{y}^k + \frac{w}{2}\right] = F\left(\frac{\bar{y}^k - \lambda\bar{y}^j + w/2}{\sigma_\epsilon}\right) - F\left(\frac{\bar{y}^k - \lambda\bar{y}^j - w/2}{\sigma_\epsilon}\right)$,
$p_{j1} = F\left(\frac{\bar{y}^1 - \lambda\bar{y}^j + w/2}{\sigma_\epsilon}\right), p_{jN} = F\left(\frac{\bar{y}^N - \lambda\bar{y}^j - w/2}{\sigma_\epsilon}\right)$

The rationale for this transition probabilities: Given a random variable of $\nu = \lambda\bar{y}^j + \epsilon$ where $\lambda\bar{y}^j$ is fixed and ϵ is distributed as ϵ_t, then the equations in 4 for p_{jk} make the distribution of $\tilde{y}_t$ conditional on $\tilde{y}_{t-1} = \bar{y}^j$ be a discrete approximation of the random variable ν. If $\bar{y}^k$ forms the reasonably fine real line, then the conditional distribution of $\tilde{y}_t$ given $\tilde{y}_{t-1} = \bar{y}^j$ will approximate closely in the sense of weak convergence that of y_t given $y_{t-1} = \lambda\bar{y}^j$.

Therefore

The discrete process $\tilde{y}_t \sim \tilde{y}_t - \bar{\lambda}\tilde{y}_{t-1} = \tilde{\epsilon}_t$, with $\mathrm{cov}(\tilde{y}_{t-1}, \tilde{\epsilon}_t) = 0$. Where $\bar{\lambda} = \mathrm{cov}(\tilde{y}_t, \tilde{y}_{t-1})/\mathrm{var}(\tilde{y}_t)$ and $\sigma_{\bar{\epsilon}}^2 = (1 - \overline{\overline{\lambda^2}})\mathrm{var}(\tilde{y}_t)$

Parameters $\bar{\lambda}, \sigma_{\bar{\epsilon}}^2$ are functions of the second moment of $\tilde{y}_t$ process and these moments can be computed from the transaction matrix in equations 4 and the $\{\bar{y}^j\}$.

Transition probability calculated with ϵ_t with normality assumption, the approximation of discrete $\bar{\lambda}, \sigma_{\bar{\lambda}}$ and continuous λ, σ_λ is good except for λ is very close to unity. It is obvious because it is Markov process with unity.

c. The vector case
1. The vector process $y_t = Ay_{t-1} + \epsilon_t$, var($\epsilon_t$) = Σ_t, a diagonal matrix, y_t is an *M* x 1 vector, A is an *M* x *M* matrix, ϵ_t is an *M* x 1 vector white noise process.
2. Discrete process of $\tilde{y}_t$, $\tilde{y}_{it}$ takes on one of N_i values: $\bar{y}^1 < \bar{y}^2, \ldots, < \bar{y}^N$. Similar to univariate case, multivariate case $\bar{y}_i^1, \bar{y}_i^N$ are set to minus and plus a small integer m times the unconditional standard deviation of y_{it}. The remaining $\bar{y}_i^l$ satisfy $\bar{y}_i^{l+1} = \bar{y}_i^l + w_i$, $l = 1,2, \ldots, N_i - 1$, $w_i = 2m\sigma(y_i)/(N_i - 1)$. The $\sigma(y_i)$ are the square roots of the diagonal elements of the matrix Σ_y that satisfies $\Sigma_y = \mathrm{A}\Sigma_y\mathrm{A}' + \Sigma_\epsilon$, which can be found by iterating $\Sigma_y(r) = \mathrm{A}\Sigma_y(r-1)\mathrm{A}' + \Sigma_\epsilon$, with convergence as $r \to \infty$ guaranteed as long as equation in 1 is stationary.
3. Transition probabilities: There are $N^* = N_1.N_2 \ldots N_M$ possible states for the system. Let index $j = 1,2, \ldots, N^*$ for a label for the states. And $\bar{l}(j)$ be an *M* x 1 vector of integers associated with state j such that when the system is in state j at time $t-1$

the components $\tilde{y}_{i,t-1}$ assumes $\tilde{y}_{i,t-1} = \bar{y}_i^p$, where $p = \bar{l}_i(j)$, for $i = 1,2, \ldots, M$. Then calculate p_{jk} , let $\tilde{y}(j)$ be the M x 1 vector of values for the $\tilde{y}$,s when the system is in state j, and put $\mu = A\tilde{y}(j)$. For each i let $h_i(j,l) = \Pr[\tilde{y}_{it} = \bar{y}_i^l | \text{state } j$ at $t-1]$ for $1 \leq l \leq N$, which is analogous to univariate case (4), is given as

$$
\begin{aligned}
h_i(j,l) &= F_i\left(\bar{y}_i^l - \mu_i + \frac{w_i}{2}\right) - F_i\left(\bar{y}_i^l - \mu_i - \frac{w_i}{2}\right) && \text{if } 2 \ll l \ll N_i - 1 , \\
&= F_i\left(\bar{y}_i^1 - \mu_i + \frac{w_i}{2}\right) && \text{if } l = 1 \\
&= 1 - F_i\left(\bar{y}_i^{N_i} - \mu_i - \frac{w_i}{2}\right) && \text{if } l = N_i
\end{aligned}
$$

Given the $h's$ transition probabilities $p(j,k) = \Pr[in\ state\ k | in\ state\ j]$ are, with independence of the $\epsilon's$, the products of the appropriate $h's$,

$$p(j,k) = \prod_{i=1}^{M} h_i\left(j, \bar{l}_i(k)\right) \text{ for } j,k = 1,2, \ldots, N^*$$

Numerical example

Theoretical VAR(1) process with two variables

$y_{1t} = 0.70y_{1,t-1} + 0.30y_{2,t-1} + \epsilon_{1t},, y_{2t} = 0.20y_{1,t-1} + 0.50y_{2,t-1} + \epsilon_{2t}$, $\epsilon_{1t}\ \epsilon_{2t}$ are iid normal (0,0.1) random variables.

The unconditional covariance matrix of this process can be calculated theoretically

$\Sigma_y = \begin{bmatrix} 0.332 & 0.126 \\ 0.126 & 0.185 \end{bmatrix}$ The N_i were each set to nine, yielding 9 x 9 = 81 states for $\tilde{y}_t$.The values $\bar{y}_i^l$ were computed with $m = 3$.

The induced representation $\tilde{y}_t = \bar{A}\tilde{y}_{t-1} + \tilde{\epsilon}_t$, where $\bar{A} = (E[\tilde{y}_t \tilde{y}'_{t-1}])(E[\tilde{y}_{t-1}\tilde{y}'_{t-1}])^{-1}$ with expectations computed using the transition matrix.

$$\bar{A} = \begin{bmatrix} 0.699 & 0.299 \\ 0.200 & 0.499 \end{bmatrix}, \ \Sigma_{\tilde{y}} = \begin{bmatrix} 0.373 & 0.139 \\ 0.139 & 0.200 \end{bmatrix}$$

The elements of $\bar{A}$ are very close to the original parameters of VAR(1) above, though covariance matrix are not very close to (though quite close) original theoretical covariance matrix. Intuitively, increasing moderately the number of grid points N_i can improve the accuracy of covariance matrix $\Sigma_{\tilde{y}}$.

Generating 51 observations on this $\tilde{y}_t$and fitting a VAR to these data gives

$\tilde{y}_{1t} = 0.11 + 0.61\tilde{y}_{1,t-1} + 0.34\tilde{y}_{2,t-1} \quad s^2 = 0.15, \tilde{y}_{2t} = 0.07 + 0.16\tilde{y}_{1,t-1} + 0.44\tilde{y}_{2,t-1} \quad s^2 = 0.09$

The discrete Markov chain imitates quite well the statistical properties underlying VAR(1) process.

2. Epstein and Zin (1989) : Substitution, Risk Aversion, and the Temporal Behavior of Consumption and Asset Returns: A Theoretical Framework
    a. Summary and significance
    1. Develops a class of recursive preference over intertemporal consumption lotteries. They permit risk attitudes to be disentangled from the degree of intertemporal substitutability. In an infinite horizon, representative agent context these preference specifications lead to a model of asset returns for both the temporal CAPM and the intertemporal consumption CAPM as special cases.
    2. Intertwined relative risk aversion and intertemporal substitutability lead to inflexiblility of the expected utility specification and subsequently lead to poor empirical performance of expected utility, representative agent, optimizing models.

3. Childers (2022) : Differentiable state-space models and Hamiltonian Monte Carlo estimation

A. Intro: Problem and proposal

1. DSGE(Dynamic Stochastic General Equilibrium) models rarely have a closed-form solution. Therefore, only numerical approximations to evaluate the moments or likelihood functions and posterior distributions of parameters
2. RWMH (Random Walk Metroplois-Hastings) algorithm, which is the most popular MCMC sample suffers from high autocorrelation across draws (if run 100,000 times, only 1,000 samples from a hypothetical independent MC sampler. Plus, sampling results can be sensitive to the choice of starting points.
3. Evaluating the likelihood function implied by the solution of a DSGE model is usually done by some filer, such as Kalman filter. Unfortunately, these filters are either restrictive in their requirements or computationally costly, non-differentiable, and difficult to tune
4. Proposed solution to 2 and 3:
   1. Apply HMC to the estimation of DSGE models. HMC works well in high-dimensional cases (HMC as an alternative to RWMH)
   2. Sample both parameter and latent variables simultaneously, instead of sampling the posterior of the model parameter by marginalizing out the latent variable. By doing so, avoiding filtering and both examine estimated latent variables and easily marginalize on samples ex-post by ignoring the estimated latent variables.
5. Idea of simultaneous sampling parameters and latent states has been around for decades, but Implementation was difficult. Just a few observations, high dimensional inference problem (RWMH cannot handle)
6. Gradient-based approaches such as HMC are limited by difficult geometry and the cost of gradient calculations, but not by the dimensionality of the problem.

7. Two ideas in 4 can be illustrated by estimating three models
   1. A canonical real business cycle (RBC) model
      a. Simulate data from RBC model solved with a first-order perturbation and use this simulated sample to evaluate the associated likelihood using the Kalman-filter. Then implement RWMH and HMC samplers. Sampling in RWMH with only 3 parameters, the fraction of effective sample per RWMH ranges btw 0.4% and 6.2%, but HMC sampler gets 49~56% depending on the parameter. (10 to 100 times more/effective)
      b. Generate data from the same RBC model, solved using a second-order perturbation. Then estimate the model with the HMC sampler and the joint likelihood function of parameters and latent states. 3 parameters and 200 latent states, sample along 203 dimension over all. HMC effective samples of around 4%,the RWMC samplers 0.3%. HMC robust to the starting point of the Markov chain.
   2. A real small open economy model based on Schmitt-Grohe and Uribe (2003)

   a. Do same experiments as 1-a, 1-b above with simulated date from a version of the model in Schmitt-Grohe and Uribe (2003) augmented with additional shocks, with three latent states, three observables, and seven estimated parameters. HMC outperforms RWMH with 10-40 times faster in speed.
3. The medium-scale New Keynesian model in Fernandez-VIllaverde and Guerron-Quintana(2021)
   a. Estimate Keynesian model with 22 parameters and 240 latent states. HMC yields good MCMC mixing, RWMH with the particle filter fails to escape from the neighborhood of the starting point.
4. Open source program DifferentiableStateSpaceModels.jl, etc

8. Differentiable Programming

1. The computation of gradients of state-space models and their likelihoods (ML) – calculating Jacobians and the training of neural networks.
2. Basic idea is applying a method that can analyze a program can also differentiate arbitrary functions by recursively applying the chain rule until it hits primitive gradients. It refers as Automatic Differentiation (AD) in this paper.
3. Forward-mode AD applies the chain rule forward starting from x^0 and calculate $f(x^0)$ and $\nabla f(x^0)$
4. Reverse-mode AD takes the x^0 and applies the functions nested inside of f until it eventually has calculated $f(x^0)$. Then it perturbs $f(x^0)$ and applies chain rule backward until it has calculated $\nabla f(x^0)$.
5. In fact, all ML program principle is reverse-mode AD that program language either analyzes a function statically or traces the function during execution (Pytorch, Tensorflow, JAX) by managing the application of chain rule, compiling both the function calculation and its gradients, and accessing libraries of primitive gradients (black boxes to the AD package) → the right level of abstraction for providing gradients is crucial
6. This paper implementation: determine the appropriate level by providing gradients and implementing two orthogonal and composable components for gradient-based algo like HMC. (HMC is the perturbation solution of 1. the DSGE and 2. the state-space likelihood)
7. This paper avoids 1. standard DSGE solvers like generalized Schur decomposition (Klein 2000; Schmitt-Grohe and Uribe 2004), differentiable and requires custom gradients, and 2. The reordering of eigenvalues, (maybe not even differentiable at all). Instead, this paper derives an implicit gradient formula for the whole calculation of the DSGE solution (bypassing problems of 1 and 20), which provides a component that both solves the DSGE models and provides gradients which respect to the parameters of interest.
8. State-space models: stochastic difference equation that simulate solutions and accumulate likelihoods (must be differentiated with respect to the state-space

model primitives and the underlying "shocks". "Scientific machine learning" (Rackauckas 2022) develops AD systems to incorporate efficient and scalable primitives for scientific computing application to solve nonlinear and differential equation.

9. **Literature Review**

   1. Farkas and Tatar (2020) – estimating linear-Gaussian DSGE models with HMC methods on the marginal likelihood of parameters, using Smets and Wouters (2007) as an example. This paper is a special case of Farkas and Tatar (2020). Downside of Farkas and Tatar is non-customizable AD therefore step-by-step differentiation, which does not allow Schur decomposition methods, currently dominant approach to solve linear DSGE. This paper instead applies a general framework to implement HMC in perturbation solutions of DSGE models by passing the required derivatives to the downstream sampler with customized AD.
   2. The use of the joint likelihood for sampling from state-space models have been known for decades but not widely adapted due to the poor scaling with dimension of more traditional MCMC (Sarkka 2013). But with HMC, performance becomes competitive and the method has been applied to ARCH and ARMA models (Stan Development Team, 2022), stochastic vol models (Hoffman and Gelman 2014) and diffusion processes (Graham et al., 2022)
   3. HMC requires gradients of the likelihood and model solution. Derivs formulas have been derived for first-order perturbations of DSGE model (Iskev, 2010) and the marginal likelihood of linear-Gaussian state-space models (Watson, 1989). Nonetheless, higher-order case is novel.

B. Likelihood Function and Evaluation

1. **The State-Space Model**
   1. Let x_t be the vector of state variables with shock ϵ i.i.d. Let θ be the vector of parameters. In DSGE model, time-invariant θ determines the preferences, technology, information sets, and government policy rules. θ can be correlated or time-varying with shocks with extra notation.

      The initial distribution of the state variables is $p(x_0|\theta)$

      The optimization behavior of the agents is a policy function $g(.)$ that links state variables x_t with the control variables y_t ;

$$y_t = g(x_t; \theta)$$

      Given $g(x_t; \theta)$, other equilibrium conditions (such as resource constraints and exogenous laws of motion), and $p(x_0|\theta)$, a state-space representation with a transition equation $h(.)$

$$x_{t+1} = h(x_t;\theta) + \eta\epsilon_{t+1}$$

The conditional density $p(x_{t+1}|x_t,\theta)$ with distribution of ϵ_{t+1} and given x_t. ϵ_{t+1} assumes to be linear without loss of generality.

The second component of the state-space representation is a measurement equation $q(.)$ that links states and controls with the observables z_t ;

$z_t = q(x_t, v_t;\theta)$ where v_t is either a measurement error or a shock that hits observables but not the states of the model

As a result, the measurement equation defines a conditional density $p(z_t|x_t,\theta)$ given by the distribution of v_t. One or more states may be part of the observables z_t . In that case, $q(.)$ is the density function along the relevant dimensions and the conditional density is a Dirac (Dirac delta distribution over the real numbers whose value is zero everywhere except at zero, and whose integral over the entire real line is equal to one)

$$\delta(x) = \lim_{b\to 0}\frac{1}{|b|\sqrt{\pi}}e^{-(x/b)^2}$$

$z^T = \{z_t\}_{t=1,\dots,T}$ is the observation sequence, $x^T = \{x_t\}_{t=0,\dots,T}$ the state sequence, and $p(z^T|\theta)$ the likelihood function of the model.

2. **Marginal vs. Joint Likelihood**
   1. **The Marginal Likelihood**: A prior distribution $p(\theta)$ for the parameters of the model, by Bayes' theorem
      $$\ln p(\theta|z^T) = \ln p(\theta, z^T) + C = \ln p(\theta) + \ln p(z^T|\theta) + C$$
      $$= \ln p(\theta) + \sum_{t=1}^{T}\ln p(z_t|z^{t-1},\theta) + C$$
      The log-posterior density is (up to a constant C) the sum of the log prior and the conditional density of the data, which also the sum of the log prior and the sum of the conditional densities for z_t across all T periods. And recursively construct the sequence of $\{p(z_t|z^{t-1},\theta)\}_{t=1,\dots,T}$ by filtering:
      1. The initial density at $t = 1$:
         $p(x_{t-1}|z^{t-1},\theta) = p(x_0|\theta)$
      2. Chapman-Kolmogorov equation to combine $p(x_{t-1}|z^{t-1},\theta)$ with the conditional distribution $p(x_t|x_{t-1},\theta)$ and integrate over the states to get:
         $$p(x_t|z^{t-1},\theta) = \int p(x_t|x_{t-1},\theta)\,p(x_{t-1}|z^{t-1},\theta)dx_{t-1}$$
      3. $p(x_t|z^{t-1},\theta)$, the measurement z_t and $p(z_t|x_t,\theta)$, then by Bayes' theorem
         $p(x_t|z^t,\theta) = p(z_t|x_t,\theta)p(x_t|z^{t-1},\theta)/p(z_t|z^{t-1},\theta)$

4. $p(z_t|z^{t-1},\theta) = \int p(z_t|x_t,\theta)p(x_t|z^{t-1})dx_t$
5. Loop over steps 2-4. For all all T periods of observations.

$p(z_t|z^{t-1},\theta)$ is "marginal likelihood" because the states x_t is integrated out. Steps 2 and 3 are conceptually simple but very hard to implement. When $h(.), q(.)$ are linear and ϵ, v follow a Gaussian distribution, equations in 2 and 3 can be evaluated exactly with Kalman filter. In nonlinear or non-Gaussian, a closed form for the marginal likelihood is rarely available, therefore numerical approximation has to be used.

2. **Joint Likelihood**: Applying Bayes' theorem to the likelihood of the data Z^T jointly in the parameters θ and the states $x^T = \{x_t\}_{t=0,\ldots,T}$, $p(z^T|x^T,\theta)$, to compute the joint posterior $p(\theta, x^T|z^T)$ without first marginalizing out the states:

$$p(\theta, x^T|z^T) = p(z^T|\theta, x^T)p(x^T|\theta)p(\theta)/\iint p(x^T|\,\theta)p(\theta)dx^T d\theta$$

Taking logs and ignoring constant terms, the posterior can be decomposed into

$$\ln p(\theta, x^T|z^T) = \ln p(\theta, x^T) + \ln p(z^T|x^T,\theta) + C$$

$$= \ln p(\theta) + \ln p(x^T|\theta) + \textstyle\sum_{t=1}^T \ln p(z_t|x^t,\theta) + C \text{ (log-prior +log-likelihood)}$$

$$= \ln p(\theta) + \textstyle\sum_{t=1}^T \ln p(z_t|x^t,\theta) + \sum_{t=1}^T \ln p(x_t|x_{t-1},\theta) + \ln p(x_0|\theta) + C$$

$$= \ln p(\theta) + \textstyle\sum_{t=1}^T \ln p(z_t|\epsilon^t, x_0,\theta) + \sum_{t=1}^T \ln p(\epsilon_t|\theta) + \ln p(x_0|\theta) + C$$

Evaluating the log-likelihood sequentially with Markov structure of the state-space model, x_t only depends on x_{t-1} given θ. Equations interprets as the log-posterior using both the data and the history of states which propagates with initial state x_0, the sequence of shocks $\epsilon^T = \{\epsilon_t\}_{t=1,\ldots,T}$ and the model parameters θ. Eventually, final equation is coming from $q(.), h(.)$. It can be clearly seen from the equations above.

$$\textstyle\sum_{t=1}^T \ln p(z_t|x^t,\theta) + \sum_{t=1}^T \ln p(x_t|x_{t-1},\theta) = \sum_{t=1}^T \ln p(z_t|\epsilon^t, x_0,\theta) + \sum_{t=1}^T \ln p(\epsilon_t|\theta)$$

As a result, the joint likelihood equation can be rewritten as an exogenous series of shocks ϵ_t. Computation does not need to infer the latent state series x^T or posterior distributions, but rather retrieve a series of x^T from the shock series.

Important: as this is the whole point of this paper using HMC !!!
Instead of sampling just θ to compute a marginal likelihood $\ln p(\theta|z^T)$, by sampling θ, ϵ^T and x_0, compute the joint likelihood $\ln p(\theta, x^T|z^T)$ which is same as $\ln p(\theta, \epsilon^T, x_0|z^T)$. The marginal likelihood can be obtained from the joint likelihood by integrating over ϵ^T and x_0:

$$\boldsymbol{p(\theta|z^T) = \int p(\theta, x^T|z^T)\, dx^T = \iint p(\theta, \epsilon^T, x_0|z^T) d\epsilon^T dx_0}$$

3. **An AR(1) Example with Measurement Noise**

To illustrate the difference between the marginal and joint likelihoods, consider dynamic equilibrium model AR(1) process with a measurement noise:

$x_t = \rho x_{t-1} + \epsilon_t, z_t = x_t + v_t, \{\epsilon_t\} \sim i.i.d.$ with a standard Gaussian distribution,

$\{v_t\} \sim i.i.d.\ N(0, \Omega)$. The initial x_0 is drawn from the invariant distribution (stationary distribution of Markov chain). $|\rho| < 1$ with $N(0,1/(1-\rho^2))$. The probability density function of a standard Gaussian distribution as $\varphi(.)$

Goal: Estimating $\boldsymbol{\rho}$, the persistence parameter of the states, given data $\boldsymbol{z^T}$.

The Marginal Likelihood given $\boldsymbol{z^T}$, $\boldsymbol{\rho}$

1. Evaluate the log-prior at ρ and set $L = \ln p(\rho)$

2. Initialize the distribution of x_0 with $N(0,1/(1-\rho^2))$. Mean $x_{0|0} = 0$, Variance

$p_{0|0} = 1/(1-\rho^2)$

3.Loop over $t = 1, \dots, T$. For each t, apply the Kalman fitler to infer the posterior distribution of x_t, and the likelihood L:
$x_{t|t-1} = \rho x_{t-1|t-1}, p_{t|t-1} = \rho^2 p_{t-1|t-1} + 1, x_{t|t} = x_{t|t-1} + \frac{p_{t|t-1}}{p_{t|t-1}+\Omega}(z_t - x_{t|t-1})$
$p_{t|t} = p_{t|t-1} - \frac{p^2{}_{t|t-1}}{p_{t|t-1}+\Omega}, L = L + \ln\varphi\left(\frac{z_t - x_{t|t-1}}{\sqrt{p_{t|t-1}+\Omega}}\right)$

The Joint Likelihood given $\boldsymbol{z^T}$, $\boldsymbol{\rho}$, initial $\boldsymbol{x_0}$ and vector of $\boldsymbol{T}$ elements $\{\boldsymbol{\epsilon_t}\}$
1.Set L equal to the sum of the log-prior $\ln p(\rho)$ at ρ and the log-prior of the latent states $\sum_{t=1}^{T} \ln\varphi(\epsilon_t)$ at $\{\epsilon_t\}$
2.Log-density of x_0 in the initial distribution $L = L + \ln\varphi\left(x_0\sqrt{1-\rho^2}\right)$
3.Loop over $t = 1, \dots, T$. For each t, compute x_t and accrue the likelihood L
$x_t = \rho x_{t-1} + \epsilon_t, L = L + \ln\varphi\left(\frac{z_t - x_t}{\sqrt{\Omega}}\right)$

While in the end, the posterior of ρ should be the same, there are significant differences between two approaches.
The marginal: depends on a filter (Kalman) to infer the distribution of the latent states over time while it only samples the parameters that governs the model

The joint: filter free, because the state variables is deterministic given some shocks, but at a cost that it has to draw both the parameters and the shocks. Sampling from a very high-dimensional space consisting of both the parameters and shocks. Dealing with nonlinear or non-Gaussian DSGE models, the joint likelihood is attractive. → HMC, a scalable and efficient sampler that handles high-dimensional spaces.

3. Hamiltonian Monte Carlo

Standard MCMC sampler such as RWMH: straightforward to code, but spends much of time outside of typical sets (the part of the parameter space containing the relevant information to compute the expectations we care about in Bayesian analysis)

Definition of the typical set: For $\epsilon > 0$, and I, the typical set A_ϵ^I with respect to the target posterior $p(\theta|z^T)$ is

$A_\epsilon^I \equiv \left\{(\theta_0, \theta_1, \dots, \theta_I: | -\frac{1}{I+1}\sum_{i=0}^{I} \ln p(\theta_i|z^T) - b(\theta)| \leq \epsilon \right\},$
$b(\theta) = -\int p(\theta_i|z^T) \ln p(\theta_i|z^T) d\theta$ is the differential entropy of the parameters with respect to their posterior density. By a weak law of large numbers, $\Pr(A_\epsilon^I) > 1 - \epsilon$ if I is sufficiently large. That is to say, A_ϵ^I includes "most" sequences of θ_i's that are distributed according to the posterior $p(\theta_i|z^T)$.

Two properties of the typical set

- The typical set is not the region where the posterior density is the highest.
- The typical set narrows as the dimensionality of θ grows.

These two properties explain why RWMH wastes many iterations 1. The proposal density is blind regarding the typical set of the posterior, which most draws will be either far away from the typical set, or highly auto-correlated and the typical set will not be traversed efficiently. 2. Initiating the sampling process from the mode will barely help, as the mode is usually not in the typical set.

HMC improves sampling efficiency by exploiting information from the posterior's gradient. To avoid pushing the samples to the mode of the posterior, HMC augments the gradient information with an extra momentum force, in analogy to Hamiltonian mechanics in physics.

HMC augments that the vector of parameters θ with an auxiliary momentum vector $\mathbf{P}$ of the same dimensions and that follows a Gaussian distribution $N(0, M)$.

The Hamiltonian associated with the posterior of θ is:

$H(\theta,\mathbf{P}) = -\ln p(\theta) + \frac{1}{2}\mathbf{P}^{\mathbf{T}}M^{-1}\mathbf{P}$, $-\ln p(\theta)$ is the minus log-posterior, analog of a "potential energy" function and $\frac{1}{2}\mathbf{P}^{\mathbf{T}}M^{-1}\mathbf{P}$ acts as a "kinetic energy" term. The total energy of the Hamiltonian is preserved when $\{\theta,\mathbf{P}\}$ evolve over time with Hamilton's equation$\frac{d\theta}{dt} = \frac{\partial H}{\partial \mathbf{P}}, \frac{d\mathbf{P}}{dt} = -\frac{\partial H}{\partial \theta}$ which suggests a simple sampling scheme: generating a random momentum $\mathbf{P}\sim N(0,M)$, then move $\{\theta,\mathbf{P}\}$ using Hamilton's equation for some length of time to obtain the next draw (requiring gradient information on $H(.)$). This draw has a stationary marginal density $p(\theta)$, which provides a theoretical justification for HMC.

A particle moves on the manifold characterized by the log posterior of θ. Every time a new sample from the current position, randomly kicking the particle in an arbitrary direction and the particle moves following the Hamiltonian dynamics, then simply stop the particle after a fixed time and record the new θ.

Hamilton's equation$\frac{d\theta}{dt} = \frac{\partial H}{\partial \mathbf{P}}, \frac{d\mathbf{P}}{dt} = -\frac{\partial H}{\partial \theta}$, ODE solved as follows

1.Draw $\mathbf{P}$ from the Gaussian distribution $N(0,M)$

2.Run Verlet integrator (Numerical method to integrate Newton's equation of motion), a numerical ODE solver that maintains the energy-preserving property of the original system, with step ϵ for each of the L iterations, resulting in an integration time of $L\epsilon$. Both are tuning parameters, 1. ϵ controlling the step size of Hamiltonian approximation 2. L determining the number of steps in each iteration. Then at the end of each iteration, the move will be accepted or rejected based on Metropolis step (accept or reject based on candidate $x^{'}$, α ratio $f(x^{'})/f(x_t)$ by generating uniform $u > \alpha$ then reject so $x_{t+1} = x_t$, otherwise accept and $x_{t+1} = x'$)

Each iteration preserved density by definition, much higher proportion of iterations can be accepted than RWMH. Many methods of selecting hyper parameters of HMC introduced. The most popular is the No-U-Turn sampler (NUTS) propsed by Hoffman and Gelman (2014), which endogenously picks L, ϵ with sample adaptation. This article uses generalized NUTS in Betancourt(2018).

4. Gradients to DSGE Solutions

The derivations of the derivatives of DSGE first and second order solutions with respect to the model parameters. The downstream HMC sampler requires gradient information, computing gradient values by solving equations derived from the implicit function theorem. The first-order gradient result is documented in Iskrev, 2010, the second-order result in this paper is **novel extension and important contribution** that further generalizes higher-order perturbations.

1. Canonical form by Schmitt-Grohe and Uribe (2004) of a variety of DSGE models:

$\mathbf{E_t}H(y^{'},y,x^{'},x;\theta)=0, y=g(x;\theta), x^{'}=h(x;\theta)+\eta\epsilon'$, $h(.)$ is the law of motion of the states, $g(.)$ is the policy function. Deterministic version is $H(\bar{y}^{'},\bar{y},\bar{x}^{'},\bar{x};\theta)=0$ and approximate the solutions to g, h by perturbing around $\bar{x}$. Let $\hat{x}=x-\bar{x}$, $\hat{y}=y-\bar{y}$ be the difference between deterministic and true solution, then the first-order solution to the DSGE model is $\hat{y}=g_x\hat{x}$, $\hat{x}^{'}=h_x\hat{x}+\eta\epsilon'$, where ' denotes partial derivatives. $[\hat{y}]^i=[g_x\hat{x}]^i+\frac{1}{2}\hat{x}^T[g_{xx}]^i\hat{x}+\frac{1}{2}[g_{\sigma\sigma}]^i$, $[\hat{x}']^j=[h_x\hat{x}]^j+\frac{1}{2}\hat{x}^T[h_{xx}]^j\hat{x}+\frac{1}{2}[h_{\sigma\sigma}]^j+[\eta\epsilon']^j$. g_{xx}, h_{xx} are three dimensional.

2. Solution Gradients

The derivatives of the first and second order solutions with respect to the parameters θ via the implicit function theorem.

Steady State: differentiating $H(\bar{y}^{'},\bar{y},\bar{x}^{'},\bar{x};\theta)=0$, $H_x\frac{\partial\bar{x}}{\partial\theta}+H_y\frac{\partial\bar{y}}{\partial\theta}+H_{x'}\frac{\partial\bar{x}}{\partial\theta}+H_{y'}\frac{\partial\bar{y}}{\partial\theta}+H_\theta=0$. All the derivatives of H are evaluated at the deterministic point. This is linear equation system.

First-Order Solutions: The derivatives of the first-order solutions, g_x and h_x, with respect to the parameters θ, $\frac{\partial g_x}{\partial\theta},\frac{\partial h_x}{\partial\theta}$. Evaluating at DSS, g_x and h_x satisfy $H_x+H_yg_x+H_{x'}h_x+H_{y'}g_xh_x=0$. Differentiating with respect to θ:

$$\begin{bmatrix}\frac{dH_{y'}}{d\theta}\\ \frac{dH_y}{d\theta}\\ \frac{dH_{x'}}{d\theta}\\ \frac{dH_x}{d\theta}\end{bmatrix}^{\mathrm{T}}\begin{bmatrix}g_xh_x\\ g_x\\ h_x\\ I\end{bmatrix}+\begin{bmatrix}H_y & H_{x'}+H_{y'}g_x\end{bmatrix}\begin{bmatrix}\frac{\partial g_x}{\partial\theta}\\ \frac{\partial h_x}{\partial\theta}\end{bmatrix}+\begin{bmatrix}H_{y'} & 0\end{bmatrix}\begin{bmatrix}\frac{\partial g_x}{\partial\theta}\\ \frac{\partial h_x}{\partial\theta}\end{bmatrix}h_x=0.$$ Sylvester equation system (AXB+CXD + E = 0) in $\frac{\partial g_x}{\partial\theta}$ and $\frac{\partial h_x}{\partial\theta}$

Second-Order Solutions: The second-order solutions to the DSGE include four additional terms, $g_{xx}, h_{xx}, g_{\sigma\sigma}, h_{\sigma\sigma}$. And these derivatives $\frac{\partial g_{xx}}{\partial\theta},\frac{\partial h_{xx}}{\partial\theta},\frac{\partial g_{\sigma\sigma}}{\partial\theta},\frac{\partial h_{\sigma\sigma}}{\partial\theta}$. The solutions to $\frac{\partial g_{xx}}{\partial\theta},\frac{\partial h_{xx}}{\partial\theta}$ by differentiating first order solution with respect to x, and flattening the second and third dimensions of g_{xx}, h_{xx}

$\begin{bmatrix}H_{y'} & 0\end{bmatrix}\begin{bmatrix}g_{xx}\\ h_{xx}\end{bmatrix}h_x\otimes h_x+\begin{bmatrix}H_y & H_{y'}g_x+H_{x'}\end{bmatrix}\begin{bmatrix}g_{xx}\\ h_{xx}\end{bmatrix}+\mathrm{C}=0$, which is a Sylvester equation on $[g_{xx}, h_{xx}]'$. And further differentiating it with respect to θ, another Sylvester equation in $\frac{\partial g_{xx}}{\partial\theta},\frac{\partial h_{xx}}{\partial\theta}$.

The solutions to $\frac{\partial g_{\sigma\sigma}}{\partial\theta}, \frac{\partial h_{\sigma\sigma}}{\partial\theta}$ is firstly solving $[H_{y'} + H_y \quad H_{y'} g_x + H_{x'}] \begin{bmatrix} g_{\sigma\sigma} \\ h_{\sigma\sigma} \end{bmatrix} + \mathrm{B} = 0$ to get $g_{\sigma\sigma}$ and $h_{\sigma\sigma}$. Then further differentiating with respect to θ to get a linear equation system for $\frac{\partial g_{\sigma\sigma}}{\partial\theta}, \frac{\partial h_{\sigma\sigma}}{\partial\theta}$. Matrix B and C are constant given the second-order solution to the DSGE model.

5. Implementation

JP Morgan project requires this implementation done in Python and Tensorflow. Two generic parts, built to work with various economic models, and a third part that combines these together with problem-specific details required for Bayesian estimation.

1.DifferentiableStateSpaceModels.jl handles first- and second-order solutions to state-space models, gradients, and sensitivity analyses.

- Narrowly focused on calculating perturbation solutions and their derivatives
- First, developing a domain-specific language to specify state-space model variables and equations using Symbolics (Gowda et al., 2022). Users can also specify optional equations for steady-state valuation or initial conditions.
- Second, generating the symbolic derivatives for the model to use in the perturbation solution algorithms.(though AD or symbolic derivatives can be used, symbolic derivatives is chosen because AD might require a mixed reverse and forward approach, which makes more complicated and hard)
- The Symbolics.jl ecosystem allows to define a Dynare-like domain-specific language (A software for the simulation and estimation of rational expectation model), and the necessary functions are differentiated symbolically and exported as regular Julia functions.
- At that point, the connection to the previous symbolic language is severed, and the functions generated behave and perform identically to handcrafted functions and gradients in Julia.
- Implementing the first- and second-order solutions to the approximated state-space model using standard algorithms.
- Finding the gradient of those solutions with respect to the parameters.
- In summary, a perturbation solution as a function that takes a model and a set of parameters values and generates a state-space representation then providing a custom gradient for that function so that any downstream usage of the state-space (ex, matrices, vectors, and **tensors**) is differentiable with respect to the model parameters.

2. DifferenceEquations.jl provides differentiable simulations and likelihoods given state-space models. Simulation and estimation algorithms.
   - The generated state-space model for a given set of parameters (from 1) is fed into DifferenceEquations.jl that provides a variety of standard features such as

simulations likelihood calculations- both using a marginalized approach with a Kalman filter and a joint approach with fixing the noise.

- These functions are given high-performance custom derivatives. A log-likelihood as a given state-space representation, observables, and noise, then providing a function that calculates the log-likelihood and finds its gradients with respect to the state-space matrices/tensors and possibly the high-dimensional latent variables.

3. Turing.jl combines these composable functions together with priors on parameters to use with HMC samplers.
   - The most easily replaceable component is the problem-specific implementation of the likelihood and priors for the HMC sampler. Used within this Turing.jl probabilistic programming language embedded within Julia. User only need to define the prior distributions of the parameters to draw, manipulate the observables as required, call DifferentiableStateSpaceModels.jl to calculate the perturbation given sampled parameters, then use the solution to the DSGE downstream in DifferenceEquations.jl to calculate log-likelihood.
   - Gradients are provided to the sampler by Turing using the underlying AD package (Zygote.jl), given the custom rules.
   - HMCExamples.jl has examples using 1, 2, 3

6. **Main Rersults**

First estimate a canonical real business cycle model and compare the efficiency and robustness of HMC against RWMH. Second, estimate a real small open economy model based on Schmitt-Grohe and Uribe(2003). Third, estimate a medium-scale New Keynesian DSGE model, Fernandez-Vollaverde and Guerron-Quintana(2021), close to the models used for real policy analysis.

1.Estimating the Real Business Cycle Model

c, k, y, z represent consumption, capital, outout and the TFP (total factor productivity: dividing output by the weighted geometric average of labor and capital input) Four equations: an intertemporal Euler equation, a resource constraint, a production function, and the law of motion for TFP:

$\frac{1}{c_t} - \beta\frac{\alpha e^{z_{t+1}}k_{t+1}^{\alpha-1}+(1-\delta)}{c_{t+1}} = 0, c_t + k_{t+1} - (1-\delta)k_t - y_t = 0, y_t - e^{z_t}k_t^{\alpha} = 0, z_{t+1} - \rho z_t - \sigma\epsilon_{t+1} = 0, \{\epsilon_t\}$~ i.i.d standard Gaussian distribution.

Set as pseudotrue parameter values $\alpha = 0.3, \beta = 0.998, \rho = 0.9, \delta = 0.025, \sigma = 0.1$, all in quarterly calibrated models. Simulate two sets of 200 artificial observations for consumption c_t and investment $i_t = k_{t+1} - (1-\delta)k_t$, one set coming from the first-order solution of the model and the other from the second-order solution. To avoid stochastic singularity, introduce s small measurement error (Gaussian with variance of $1e-5$ for each variable).

Estimate the model using the simulated data and assuming the researcher knows $\delta = 0.025, \sigma = 0.1$, and the variance of the measurement error, but need to find α, β, ρ. As β is close to 1, then sample $\beta_{draw} = 100(\frac{1}{\beta} - 1)$ instead. As priors, use a truncated normal for α with mean 0.3, stdev 0.025, 0.2 lower truncation bound, 0.5 upper truncation bound: for β_{draw} use a Gamma with mean 0.25 and stdev 0.1; and for ρ, use a Beta with mean 0.5 and stdev 0.2.

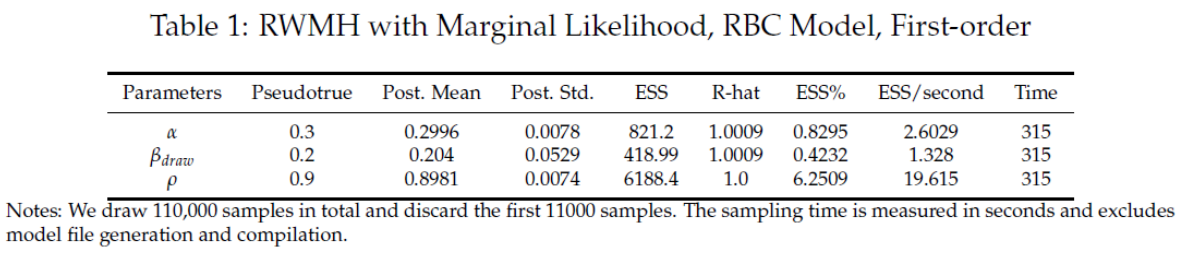

Table 1: RWMH with Marginal Likelihood, RBC Model, First-order

Parameters	Pseudotrue	Post. Mean	Post. Std.	ESS	R-hat	ESS%	ESS/second	Time
α	0.3	0.2996	0.0078	821.2	1.0009	0.8295	2.6029	315
β_{draw}	0.2	0.204	0.0529	418.99	1.0009	0.4232	1.328	315
ρ	0.9	0.8981	0.0074	6188.4	1.0	6.2509	19.615	315

Notes: We draw 110,000 samples in total and discard the first 11000 samples. The sampling time is measured in seconds and excludes model file generation and compilation.

Estimating the first-order RBC (real business cycle) model evaluating the marginal likelihood with a Kalman filter and sampling from the posterior with a RWMH. Dynare is a choice of software as it is the most popular software for the solution and estimation of DSGE models and a high quality, state-of-the-art alternative based on highly optimized c++ mode, incorporating several algorithmic advances into the solution and estimation algorithms. Therefore, represents a stronger benchmark relative to existing Julia implementation.

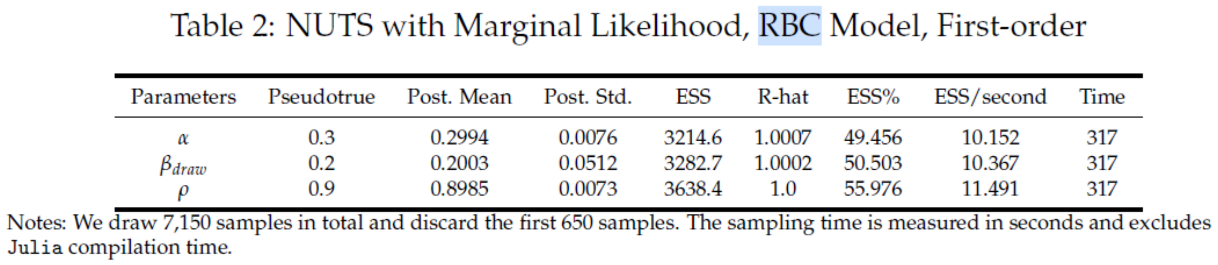

Table 2: NUTS with Marginal Likelihood, RBC Model, First-order

Parameters	Pseudotrue	Post. Mean	Post. Std.	ESS	R-hat	ESS%	ESS/second	Time
α	0.3	0.2994	0.0076	3214.6	1.0007	49.456	10.152	317
β_{draw}	0.2	0.2003	0.0512	3282.7	1.0002	50.503	10.367	317
ρ	0.9	0.8985	0.0073	3638.4	1.0	55.976	11.491	317

Notes: We draw 7,150 samples in total and discard the first 650 samples. The sampling time is measured in seconds and excludes `Julia` compilation time.

Sampling from the posterior using NUTS, Julia library. BUTS traverses the posterior more efficiently than RWMH. NUTS efficiency advantage comes with the overhead cost of gradient evaluation. The posterior drawn with both models are centered around pseudotrue parameter values and stdev are similar.

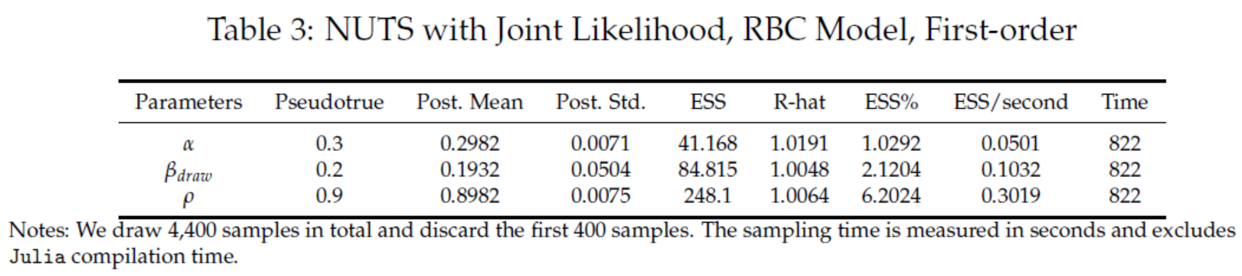

Table 3: NUTS with Joint Likelihood, RBC Model, First-order

Parameters	Pseudotrue	Post. Mean	Post. Std.	ESS	R-hat	ESS%	ESS/second	Time
α	0.3	0.2982	0.0071	41.168	1.0191	1.0292	0.0501	822
β_{draw}	0.2	0.1932	0.0504	84.815	1.0048	2.1204	0.1032	822
ρ	0.9	0.8982	0.0075	248.1	1.0064	6.2024	0.3019	822

Notes: We draw 4,400 samples in total and discard the first 400 samples. The sampling time is measured in seconds and excludes `Julia` compilation time.

The joint likelihood method using NUTS, the ESS% drops significantly compared to the marginal likelihood as simply a much higher dimensional parameter space: 3+200 = 203 and requires more time to get convergence.

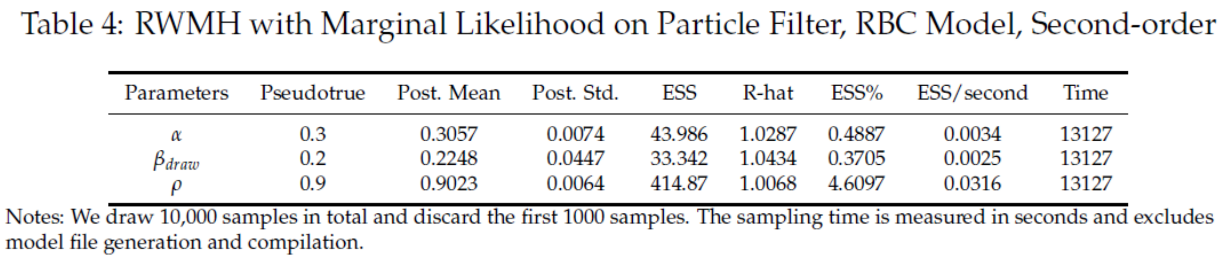

Table 4: RWMH with Marginal Likelihood on Particle Filter, RBC Model, Second-order

Parameters	Pseudotrue	Post. Mean	Post. Std.	ESS	R-hat	ESS%	ESS/second	Time
α	0.3	0.3057	0.0074	43.986	1.0287	0.4887	0.0034	13127
β_{draw}	0.2	0.2248	0.0447	33.342	1.0434	0.3705	0.0025	13127
ρ	0.9	0.9023	0.0064	414.87	1.0068	4.6097	0.0316	13127

Notes: We draw 10,000 samples in total and discard the first 1000 samples. The sampling time is measured in seconds and excludes model file generation and compilation.

In the second-order RBC model, data-generating process is no longer linear-Gaussian. Therefore, applying the particle filter to filter through the latent variables in the marginal likelihood approach.

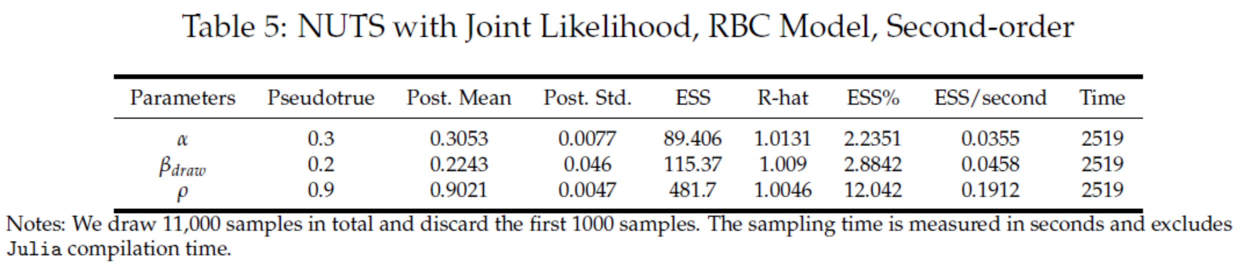

Table 5: NUTS with Joint Likelihood, RBC Model, Second-order

Parameters	Pseudotrue	Post. Mean	Post. Std.	ESS	R-hat	ESS%	ESS/second	Time
α	0.3	0.3053	0.0077	89.406	1.0131	2.2351	0.0355	2519
β_{draw}	0.2	0.2243	0.046	115.37	1.009	2.8842	0.0458	2519
ρ	0.9	0.9021	0.0047	481.7	1.0046	12.042	0.1912	2519

Notes: We draw 11,000 samples in total and discard the first 1000 samples. The sampling time is measured in seconds and excludes `Julia` compilation time.

Joint likelihood on RWMH and NUTS show clear better performance of NUTS.