Estimating Gaussian Mixture Autoregressive model with Sequential Monte Carlo algorithm: A parallel GPU implementation

Yin, Ming

University of Helsinki, Helsinki Center of Economic Research (HECER)

December 2015

Estimating Gaussian Mixture Autoregressive model with Sequential Monte Carlo algorithm: A parallel GPU implementation

Ming Yin
University of Helsinki

Abstract

In this paper, we propose using Bayesian sequential Monte Carlo (SMC) algorithm to estimate the univariate Gaussian mixture autoregressive (GMAR) model. The prominent benefit of the Bayesian approach is that the stationarity restriction required by the GAMR model can be straightforwardly imposed via prior distribution. In addition, compared to MCMC (Markov Chain Monte Carlo) and other simulation based algorithms, the SMC is robust to multimodal posteriors, and capable of providing fast on-line estimation when new data is available. Furthermore, it has a linear computational complexity and is ready for parallelism. To demostrate the SMC, an empirical application with US GDP growth data is considered. After estimation, we conduct the Bayesian model selection to evaluate the empirical evidence for different GMAR models. To facilitate the realization of this compute-intensive estimation, we parallelize the SMC algorithm on a nVidia CUDA compatible Graphical Process Unit (GPU) card.

Keywords: *Nonlinear Time Series, Gaussian mixture autoregressive, Sequential Monte Carlo, Particle Filter, Bayesian Inference, GPGPU, Parallel Computing.*

JEL Classification: C11, C32, C52, C88

1 Introduction

Mixture autoregressive model is a recent development in nonlinear time series, it first appeared in Martin (1992) as multipredictor autoregressive time series (MATS) model. Later, Le, Martin and Raftery (1996) introduced the Gaussian mixture transition distribution (GMTD) model. Later, Le, Martin and Raftery (1996) introduced the Gaussian mixture transition distribution (GMTD) model. Wong and Li (2000, 2001) further generalized the GMTD model to the mixture autoregressive (MAR, hereafter) model. Glasbey (2001) also considered a first order mixture autoregressive model and applied it to solar radiation data. Lanne and Saikkonen (2003), Gourieroux and Robert (2006), Dueker, Sola and Spagnolo (2007) discussed similar models and their applications. Recently, Kalliovirta, Meitz, and Saikkonen (2015) extended the model of Glasbey (2001) to general p-th order, and by choosing mixing weights of the MAR model based on Gaussian assumption, they formed the Gaussian mixture autoregressive (GMAR, hereafter) model.

The MAR model provides a flexible and intuitive framework for conducting statistical inference. In particular, its attractiveness comes from three aspects: first, the MAR allows the possibility to obtain a stationary process by combining stationary AR processes with nonstationary AR processes. Second, given past history, the conditional distribution of underlying time series can be multimodal. Third, the MAR is capable of capturing the conditional heteroscedasticity, which is common in many nonlinear time series. These features make the MAR an ideal candidate for modeling nonlinear time series. On the other hand, as a special form of the MAR model, the GMAR model offering several appealing properties: it is defined in such a way that guarantees the stationarity and ergodicity conditions. In addition, for a p-th order model, the $p+1$ dimensional stationary distribution can be expressed explicitly by a mixture of Gaussian distributions with constant mixing weights.

Our major motivation using Bayesian method stems from the formulation of the GMAR model, which defines the mixing weights to follow a Gaussian AR process and imposes stationarity restrictions in its definition. Therefore, the Bayesian approach we propose here provides a natural treatment for the GMAR model as the stationarity restrictions required by the GAMR model can be straightforwardly imposed via prior distribution. In the statistics liter-

ature, mixture models can be estimated by various Bayesian approaches. The Markov Chain Monte Carlo (MCMC) algorithm is one of the most commonly used algorithms. In theory, it is able to provide a complete picture of the posterior distribution, but the biggest challenge in practice comes from the complexity of the posterior distribution $\pi_t(\theta)$, which tends to be highly multimodal and asymmetric. Standard MCMC algorithms can be easily trapped in local suboptimal modes or other subspace (see Celeux et al. (2000) and Jasra et al. (2005)), and therefore do not converge in this scenario.

In this paper, we consider sequential Monte Carlo (SMC) algorithm as an alternative solution to estimate the GMAR model. In contrary to aforementioned algorithms, the SMC has several advantages: first, as a sequential importance sampling based algorithm, it is robust to multimodal posteriors and can be used in high dimension scenarios. In addition, the SMC is capable of producing fast on-line updating of the estimation when new data is available. For general state space time series models, it can be used not only for parameter (and state) estimation (filtering) but also for smoothing and forecasting. Furthermore, compared to MCMC algorithms, SMC significantly simplified the tuning process. Finally, the SMC admits a competitive linear computational complexity of[1] $O(N)$ at each time and the algorithm is ready for parallelism.

To the best of our knowledge, the literature in nonlinear time series econometrics using SMC to estimate mixture models is scarce. A close work is Carvalho, Lopes, Polson and Taddy (2010) which develops a particle learning algorithm to estimate general Dirichlet process mixture models. However, their framework is an auxiliary particle filter (APF, see Pitt and Shephard (1999)) variation of the early work by MacEachern and Muller (1998).

The rest of this paper is organized as follows: in next section we briefly review the MAR model and the GMAR model. In the third section, we introduce the SMC algorithm in general and describe our estimation procedures. The fourth section is dedicated to an empirical example where we use SMC algorithm to estimate the GMAR model of US GDP growth rate. We provide three technical appendices: the first one discusses the details of resampling procedure and the second one discusses the general idea of parallelization based on CUDA platform, the last appendix provides a flowchart of our SMC algorithm.

[1] $O(\cdot)$ *is the Bachmann–Landau notation for complexities. In particular,* $O(N)$ *indicates linear complexity.*

2 Model

We consider an M-component mixture autoregressive model for the time series of interest y_t. The model can be described by the conditional density of y_t given its past information:

$$f(y_t|\mathscr{F}_{t-1}) = \sum_{m=1}^{M} \alpha_{m,t} \frac{1}{\sigma_m} \phi\left(\frac{y_t - \mu_{m,t}}{\sigma_m}\right), \tag{1}$$

where $\mathscr{F}_{t-1}$ denotes the σ-algebra generated by $\{y_{t-j},\, j > 0\}$, and $\alpha_{m,t}$ $(m = 1, \ldots, M)$ are positive time varying mixing weights that satisfy $\sum_{m=1}^{M} \alpha_{m,t} = 1$ for all t. $\phi(\cdot)$ denotes the probability density function of a standard normal random variable, and σ_m^2 $(m = 1, \ldots, M)$ is the variance of the mth component of the mixture. The quantity $\mu_{m,t}$ in (1) is given by:

$$\mu_{m,t} = \varphi_{m,0} + \sum_{i=1}^{p} \varphi_{m_t,i} y_{t-i}\,, \qquad (m = 1, \ldots, M) \tag{2}$$

where $\varphi_{m,0}$ is a constant term and $\varphi_{m,1}, \cdots, \varphi_{m,p}$ are unknown autoregressive coefficients. It is worth mentioning that if the mixture weights are assumed to be time invariant (i.e., $\alpha_{m,t} = \alpha_m$), then (1) becomes the MAR model of Wong and Li (2000).

From (1) and (2), it can be seen that the conditional expectation of y_t is the weighted average of $\mu_{m,t}$:

$$\mathbb{E}(y_t|\mathscr{F}_{t-1}) = \sum_{m=1}^{M} \alpha_{m,t}\mu_{m,t} = \sum_{m=1}^{M} \alpha_{m,t} \left(\varphi_{m,0} + \sum_{i=1}^{p} \varphi_{m,i} y_{t-i}\right). \tag{3}$$

Similarly, the conditional variance of y_t can be expressed as:

$$Var(y_t|\mathscr{F}_{t-1}) = \sum_{m=1}^{M} \alpha_{m,t}\sigma_m^2 + \sum_{m=1}^{M} \alpha_{m,t} \left(\mu_{m,t} - \sum_{n=1}^{N} \alpha_{n,t}\mu_{n,t}\right)^2. \tag{4}$$

This equals the weighted average of σ_m^2 plus an extra term. The extra term equals 0 when $\mu_{1,t} = \cdots = \mu_{m,t}$. Therefore, the conditional variance in (4) is the smallest in this case. Otherwise, the more $\mu_{m,t}$ differs from each other,

the larger is the conditional variance.

The GMAR model proposed by Kalliovirta, Meitz, and Saikkonen (2015) is based on a particular choice of the time varying mixing weights $\alpha_{m,t}$ $(m = 1, \ldots, M)$ in (1). To provide a formula for the mixing weights, we first define an M-component auxiliary Gaussian AR(p) processes:

$$v_{m,t} = \varphi_{m,0} + \sum_{i=1}^{p} \varphi_{m,i} v_{m,t-i} + \sigma_m \epsilon_t, \quad (m = 1, \ldots, M) \tag{5}$$

where ϵ_t is a standard normal random variable which is independent of $\{y_{t-j}, j > 0\}$, and the autoregressive coefficients $\varphi_m = (\varphi_{m,1}, \cdots, \varphi_{m,p})$ are assumed to satisfy the stationarity condition:

$$\varphi_m(z) = 1 - \sum_{i=1}^{p} \varphi_{m,i} z^i \neq 0 \quad for\, |z| \leq 1. \quad (m = 1, \ldots, M) \tag{6}$$

We proceed by defining a normally distributed random vector $\mathbf{v}_{m,t} = (v_{m,t}, \cdots, v_{m,t-p+1})'$ with density (cf. (5)):

$$n_p\left(\mathbf{v}_{m,t}|\vartheta_m\right) = (2\pi)^{-q/2} det(\Gamma_{m,p})^{-1/2}$$

$$\times exp\left\{-\frac{1}{2}\left(\nu_{m,t} - \mu_m 1_p\right)' \Gamma_{m,p}^{-1}\left(\nu_{m,t} - \mu_m 1_p\right)\right\}, \tag{7}$$

where $\vartheta_m = (\varphi_{m,0}, \varphi_m, \sigma_m^2)'$, and $\mu_m 1_p$ is the mean vector of $\mathbf{v}_{m,t}$ $(m = 1, \ldots, M)$. Specifically, $\mu_m = \frac{\varphi_{m,0}}{\varphi_m(1)}$, $\varphi_m(1) = 1 - \sum_{i=1}^{p} \varphi_{m,i}$, and $1_p = (1, \cdots, 1)'_{p\times 1}$. The covariance matrix $\Gamma_{m,p}$ $(m = 1, \ldots, M)$ in (7) is a $p \times p$ Toeplitz matrix with $\gamma_{m,0} = Cov(v_{m,t}, v_{m,t})$ along the main diagonal and $\gamma_{m,i} = Cov(v_{m,t}, v_{m,t-i})$ $(i = 1, \ldots, p-1)$ on the diagonal above and below the main diagonal.

Using (7), the time varying mixing weights $\alpha_{m,t}$ in (1) can be expressed as

$$\alpha_{m,t} = \frac{\alpha_m n_p\left(\mathbf{y}_{t-1}|\vartheta_m\right)}{\sum_{n=1}^{M} \alpha_n n_p\left(\mathbf{y}_{t-1}|\vartheta_n\right)}, \tag{8}$$

where $\mathbf{y}_{t-1} = (y_{t-1}, \cdots, y_{t-p})^T$, and $\alpha_m \in (0,1)$ $(m = 1, 2, \ldots, M)$ are unknown time invariant mixing weights that satisfy $\sum_{m=1}^{M} \alpha_m = 1$.

Equations (1), (5) and (8) define a GMAR(p, M) model. As shown in Theorem 1 and its proof of Kalliovirta, Meitz and Saikkonen (2015), $\mathbf{y}_t = (y_t, \cdots, y_{t-p+1})'$ is an ergodic Markov chain on $\mathbb{R}^p$ with a stationary distribution characterized by the density:

$$f(\mathbf{y}_t|\theta) = \sum_{m=1}^{M} \alpha_m n_p\left(\mathbf{y}_t|\vartheta_m\right), \tag{9}$$

where $\theta = (\vartheta_1, \cdots, \vartheta_m, \alpha_1, \cdots, \alpha_{M-1})$.

The above equation states the fact that the stationary distribution of $\mathbf{y}_t$ is a mixture of M multivariate normal distributions with time invariant mixing weights α_m. Moreover, the stationary distribution of the $(p+1)$-dimensional random vector $(y_t, \mathbf{y}_t')'$ is also a Gaussian mixture with density

$$f(y_t, \mathbf{y}_{t-1}|\theta) = \sum_{m=1}^{M} \alpha_m n_{p+1}\left(y_t, \mathbf{y}_{t-1}|\vartheta_m\right). \tag{10}$$

Obviously, (9) and (10) are of the same parametric form, but (10) is $(p+1)$-dimensional. Consequently, the marginal distributions of the elements of the vector $(y_t, \mathbf{y}_{t-1})$ belong to the same family.

Under the stationarity assumption in (6), the time invariant mixing weight α_m can be interpreted as the unconditional probability of the random vector $\mathbf{y}_t$ being generated from the mth component of the Gaussian mixture described in (9). Likewise, α_m represents the unconditional probability of y being generated from the mth component of Gaussian mixture density $\sum_{m=1}^{M} \alpha_m n_1\left(y|\vartheta_m\right)$, where $n_1\left(\cdot\right)$ is a normal density with mean μ_m and variance $\gamma_{m,0}$.

To provider further intuition, we consider an alternative expression of the GMAR model. Let $P_{t-1}(\cdot)$ denote the conditional probability of an event given past information $\mathscr{F}_{t-1}$. For each time t, let $s_t = (s_{t,1}, \cdots, s_{t,M})'$ be an unobserved M-dimensional random vector such that s_t and ϵ_t are independent conditional on $\mathscr{F}_{t-1}$. The conditional probability (condition on $\mathscr{F}_{t-1}$) that an element of the vector s_t takes the value one while the other elements equal zero is given by

$$P_{t-1}(s_{t,1}=0,\cdots,s_{t,m}=1,\cdots,s_{t,M}=0)=\alpha_{m,t}. \qquad (m=1,\ldots,M) \quad (11)$$

Thus, the mixing weights $\alpha_{m,t}$ can be interpreted as the probabilities that determine which of the M components generates the observation y_t. As a result, the GMAR model can be rewritten as:

$$y_t=\sum_{m=1}^{M} s_{t,m}\left(v_{m,t}+\sigma_m\epsilon_t\right)=\sum_{m=1}^{M} s_{t,m}\left(\varphi_{m,0}+\sum_{i=1}^{p}\varphi_{m,i}v_{m,t-i}+\sigma_m\epsilon_t\right). \quad (12)$$

It can be seen from (8), (11) and (12) that α_m in (8) also represents the unconditional probability of y_t being generated from the mth AR component in (12), where the time varying mixing weight $\alpha_{m,t}$ represents the corresponding conditional probability $\alpha_{m,t}$. In particular, $\alpha_{m,t}$ depends on the numerator of (8) which is a product of α_m and $n_p\left(\mathbf{y}_{t-1}|\vartheta_m\right)$. The latter part of the product, $n_p\left(\mathbf{y}_{t-1}|\vartheta_m\right)$, can be interpreted as the likelihood of the mth AR component in (12). The larger the likelihood is, the more likely y_t is generated from the mth AR component of (12) . However, the conditional probability $\alpha_{m,t}$ is also affected by α_m, which is the weight of $n_p\left(\mathbf{y}_{t-1}|\vartheta_m\right)$ in (9). In other words, α_m can offset a large value of $n_p\left(\mathbf{y}_{t-1}|\vartheta_m\right)$ making $\alpha_{m,t}$ small.

3 Bayesian Inference of the GMAR model

3.1 Framework

The posterior distribution $\pi(\theta|y)$ is proportional to the product of the likelihood function and the prior distribution:

$$\pi(\theta|y)\propto\pi(\theta)f(y|\theta), \quad (13)$$

where θ is the vector containing unknown parameters (see the discussion following (9)), $\pi(\theta)$ is the prior distribution of the parameters θ, and $f(y|\theta)$ is the likelihood function.

As discussed in Kalliovirta, Meitz, and Saikkonen (2015), the stationary distribution of the GMAR process (9) is known under the assumption that

the autoregressive coefficients $\varphi_m = (\varphi_{m,1}, \cdots, \varphi_{m,p})$ $(m = 1, \ldots, M)$ in (5) satisfied the stationarity condition (6). To the best of our knowledge, the GMAR model is the only mixture model that can admit the exact stationary distribution. As an advantage of Bayesian methods, such stationarity restrictions can be straightforwardly imposed via prior distribution.

As long as the stationary condition (6) holds, we can make use of initial values to construct the exact likelihood function. In particular, given observed data y_1,$\cdots$,y_T, the likelihood function of the GMAR model takes the following form:

$$f(y|\theta) = \left(\sum_{m=1}^{M} \alpha_m n_p \left(\mathbf{y}_0 | \vartheta_m \right) \right) \prod_{t=1}^{T} L_t(\theta), \tag{14}$$

where

$$L_t(\theta) = \sum_{m=1}^{M} \alpha_{m,t}(\theta)(2\pi\sigma_m^2)^{-1/2} exp\left(-\frac{\left(y_t - \mu_{m,t}(\vartheta_m)\right)^2}{2\sigma_m^2} \right).$$

Notice that the time-varying mixing weight $\alpha_{m,t}(\theta)$ and the conditional expectation $\mu_{m,t}(\vartheta_m)$, $(\vartheta_m \subset \theta)$ are both functions of the parameters (cf. (5) and (8)).

The posterior distribution of the parameters $\pi(\theta|y)$ is obtained by combining (14) and the prior distribution $\pi(\theta)$.

3.2 SMC estimation

In the Bayesian context, summary statistics (e.g. mean, variance, etc.) of the posterior distribution serves as an estimate of the parameters. However, as the posterior distribution (13) is analytically intractable, we estimate it using the SMC algorithm. The summaries of the posterior distribution can be then calculated by Monte Carlo methods. In this section we briefly discuss how the SMC algorithm can be effectively implemented to estimate the posterior distribution of the parameters of the GMAR model.

The SMC is an iterative algorithm that produces a sequence of particle system. We define the collection of N duplets (θ_t^i, w_t^i) $(i \in N)$ in the space of interest $\Theta_t \times \mathbb{R}^+$ as a particle system $\{\theta_t^i, w_t^i\}_{i \in N}$, $t \in \mathcal{L} = \{1, \cdots, L\}$ $(L \leq T)$. The variable θ_t^i $(i \in N)$ refers to a particle and it is associated

with a corresponding weight denoted by w_t^i $(i \in N)$. The particle system $\{\theta_t^i, w_t^i\}_{i\in N}$ targets a given distribution π_t in such a way that

$$\sum_{i=1}^{N} w_t^i \psi(\theta_t^i) \to \mathbb{E}_{\pi_t}(\psi), \tag{15}$$

almost surely as $N \to \infty$, for any π_t-integrable function ψ.

Since the target distributions considered in this paper are defined on the common space $\Theta_t = \Theta$, the target distribution π_t is the posterior density of θ given data up to time t $(t \in \mathcal{L})$: $\pi_t(\theta) = \pi(\theta|\mathbf{y_t})$. In particular, let integers τ_t $(t \in \mathcal{L})$ denote the dates of observations, such that $\tau_0 = 0 < \tau_1 < \cdots < \tau_L = T$. Then, from (14), the kernel of the posterior can be expressed as

$$\pi(\theta|\mathbf{y}_{\tau_t}) \propto \pi(\theta) \prod_{n=1}^{\tau_t} \sum_{m=1}^{M} \alpha_{m,n}(\theta)(2\pi\sigma_m^2)^{-1/2} exp\left(-\frac{(y_n - \mu_{m,n}(\vartheta_m))^2}{2\sigma_m^2}\right), \tag{16}$$

where $t \in \mathcal{L} = \{1, \cdots, L\}$ $(L \leq T)$.

We now briefly discuss the SMC algorithm which uses a sequence of particle system $\{\theta_t^i, w_t^i\}_{i\in N}$ to estimate the target distribution. We initialize the algorithm by sampling N particles $\{\theta_0^i\}_{i\in N}$ from the prior distribution $\pi_0(\theta)$. Then following Chopin (2004), we drive our particle system $\{\theta_t^i, w_t^i\}_{i\in N}$ toward the target distribution $\pi_T(\theta)$ by repeating three steps of Correction, Resampling and Mutation defined below:

1. Correction: The correction step is used to add observations into the particle system to update the importance weights that reflect the density of the particles in the current iteration. Due to the fact that for the GMAR model, target distributions are defined on the common space $(\Theta_t = \Theta)$, the importance weights are given by $w_t(\theta) = \pi_t(\theta)/\pi_{t-1}(\theta)$. In particular, from (14), the kernel of the weight function can be expressed as

$$\widetilde{w}_t^i(\theta) = \prod_{n=\tau_{t-1}+1}^{\tau_t} \sum_{m=1}^{M} \alpha_{m,n}^i \frac{1}{\sigma_m^i} \phi\left(\frac{y_n - \mu_{m,n}^i}{\sigma_m^i}\right), \quad (i \in N) \tag{17}$$

where $\widetilde{w}_t^i(\theta)$ are unnormalized particle weights calculated using $\alpha_{m,n}^i$, σ_m^i, $\mu_{m,n}^i$ $(i \in N)$. These weights are then normalized as

$$w_t^i = \frac{\widetilde{w}_t^i}{\sum_{i=1}^{N} \widetilde{w}_t^i}, \quad (i \in N). \tag{18}$$

2. Resampling: The resampling step combines the normalized particle weights (18) and particles $\left\{\theta_{t-1}^i\right\}_{i\in N}$ in a collection $\left\{\theta_{t-1}^i, w_t^i\right\}_{i\in N}$. Then, the residual resampling method is applied to simulate $\left\{\hat{\theta}_{t-1}^i\right\}_{i\in N}$. The residual resampling first removes particles with low weights, then replicates particles with high weights multiple times and finally assigns the same weights to all resampled particles. After resampling, the new particle system $\left\{\hat{\theta}_{t-1}^i, 1\right\}_{i\in N}$ approximates $\pi_t(\theta)$.

3. Mutation: The simulated particles $\left\{\hat{\theta}_{t-1}^i\right\}_{i\in N}$ are mutated according to a random-walk Metropolis-Hasting kernel $\hat{\theta}_t^i \sim p(\hat{\theta}_t|\mathbf{y}_{\tau_t}, c^2 Cov(\hat{\theta}_{t-1}))$ $(i \in N)$, where the p.d.f admits $\pi_t(\theta)$ as an invariance density.

For more elaborate discussion of this algorithm we refer to Del Moral, Doucet and Jasra (2006), which also provides the convergence results for the algorithm. It is worth mentioning that in the aforementioned algorithm the particles are mutated after the resampling step. This is for the concern of accuracy of the particle approximation $\{\theta_t^i, 1\}_{i\in N}$ of $\pi_t(\theta)$, in terms of forecasting. Note that, the accuracy can be readily improved by increasing the number of Markov Chain Monte Carlo (MCMC) iterations in the mutation phase.

A practical issue of the algorithm is that the discrepancy between the sequential importance distribution and the target distribution tends to increase with t, which leads to a degeneracy of the particle system. Although, resampling T times (i.e., $\tau_1 = 1, \cdots, \tau_L = T$) can keep the successive posterior distributions as close to each other as possible, this approach is usually not ideal as the resampling increases the variance of the estimates and reduce the number of distinct particles (see Chopin (2004) and Del Moral, Doucet and Jasra (2012)). Therefore, we resample only when necessary for preventing the degeneracy of the particles. More specifically, we adopt the adaptive procedure of Durham and Geweke (2014) to produce $\tau_1, \cdots, \tau_L$. In particular, at each cycle $t \in \mathcal{L} = \{1, \cdots, L\}$ $(L \leq T)$, conditional on the previous cycles, the posterior density $\pi(\theta|\mathbf{y}_{\tau_t})$ in (16) is obtained by introducing one new data observation at a time into the system, until a specific stopping criteria is met. We use the effective sample size (ESS, hereafter) proposed by Liu and Chen(1998) to monitor the degeneracy of the particles.

$ESS \stackrel{def}{=} \left[\sum_{i=1}^{N} \left(w_t(\theta_{t-1}^i)\right)^2\right]^{-1} \in [1, N]$. The smaller the ESS is, the higher is the degeneracy, and we use $N/2$ as the stopping threshold (see Del Moral, Doucet and Jasra (2012)). The convergence results presented in Del Moral, Doucet and Jasra (2006) suggest that (15) holds almost surely for the particle system generated by this adaptive algorithm.

Regarding the Mutation step, we use the random walk Metropolis-Hasting algorithm. The covariance matrix of the proposal distribution is constructed from the associated elements of the sample covariance matrix of the current population of the particles $Cov(\theta_{r-1,t})$, where $r = 1, \cdots, R$, and R is the maximum number of iteration. The covariance matrix is further multiplied by an adaptive tuning parameter c, $(0.1 \leq c \leq 1)$, that used to keep the MH acceptance rate at 0.25 (see Lanne and Luoto (2015), Durham and Geweke (2014), and Herbst and Schorfheide (2014)). In particular, c is set to be $c + 0.01$ if the acceptance rate is greater than 0.25, and set to be $c - 0.01$ otherwise. This procedure is repeated independently for each particle θ_t^i $(i \in N)$ until the particles are clearly distinct. Following Durham and Geweke (2014), we use relative numerical efficiency (RNE) as a measure of particle divergence (also see Geweke(2005,276)). We use the autocovariance of the predictive likelihood to calculate the RNE, and terminate the particle mutation when the RNE value exceeds a certain threshold. We set the maximum number of MCMC iteration in our algorithm at 200.

Below, is the summary of our SMC algorithm:

Algorithm 1 (pseudo-code): SMC for the GMAR model

1 Set number of particles$\leftarrow N$*, maximum number of iteration* $\leftarrow T$
2 $t \leftarrow 0$
3 $\{\theta_0^i\}_{i\in N} \sim N(\theta|\mu_\Theta, \sigma_\Theta^2)$ *//Initialize the particles*
4 $\{w_0^i\}_{i\in N} \leftarrow 1/N$ *//Initialize particle weights*
5 *while RNE condition not true*
6 *while ESS condition not true*
7 $t \leftarrow t+1$
8 $\left\{w_{t+1}^i\right\}_{i\in N} \propto \{w_t^i\}_{i\in N} \cdot f(y_t|\mathbf{y}_{t-1}, \theta)$ *//Updating weights*
9 *end (ESS)*
10 $\left\{\hat{\theta}_{t-1}^i, 1\right\}_{i\in N} \leftarrow \left\{\theta_{t-1}^i, w_t^i\right\}_{i\in N}$ *//Resampling*
11 $\{\theta_t^i\}_{i\in N} \sim f(\hat{\theta}_t|\hat{\theta}_{t-1})$ *//Propagate particles*
12 $\{w_t^i\}_{i\in N} \propto f(y_t|\mathbf{y}_{t-1}, \theta)$ *//Weight particles*
13 $\hat{\theta}_t^i \sim p(\hat{\theta}_t|\mathbf{y}_t, c^2 Cov(\hat{\theta}_{t-1}))$ *//Mutation*
14 *end (RNE)*

4 Empirical example

In this section, we estimate GMAR models for the U.S. Gross Domestic Product (GDP) growth data using the SMC algorithm explained in previous section. We first motivate the use of the GMAR model. Then we report our posterior results for the GMAR models with different number of states (M) and lags (p). Finally, we conduct Bayesian model selection to evaluate empirical evidence for different GMAR models.

4.1 Background

One well-known characteristics of the U.S. business cycle is the asymmetry of real output across different business cycles. Nelson and Plosser (1982), Cochrane (1988) have documented the stylized fact of the U.S. economy that the output growth is positively autocorrelated over short horizons and has

weak and possible insignificant negative autocorrelation over long horizons. Cogley and Nason (1995) investigated the difficulties of using real business cycle (RBC) models to replicate this recognized pattern and suggested that standard RBC models have weak propagation mechanism and must rely on exogenous sources of dynamics.

Among the extensive literature, Hamilton (1989) is a distinguished example. In his seminal paper, Hamilton used a two-regime Markov-switching autoregressive (MSAR, hereafter) model to study the U.S. real GNP growth, and successfully captured the asymmetry in the business cycle and the estimated shifts between the two phases accord well with the National Bureau of Economic Research (NBER) chronology of U.S. business cycle. Hamilton's paper also stimulated the applications of Markov-switching class models in describing the dynamics of many macroeconomic time series (See Hamilton (2008)). Similar evidence obtained in Kim et al. (2005), and Camacho and Perez-Quiros (2007), inter alia, further confirmed that the output growth dynamics might be better captured by shifts between business cycle states rather than by the traditional linear autoregressive models.

Despite its popularity, there are two major drawbacks in the Hamilton's model. On the one hand, it assumes that the Markov state variable governing the switch of the regime is strictly exogenous, and thus independent of the regression disturbance at all leads and lags. This exogenous assumption is not generally realistic as the aggregate output may simultaneously affect the current state of the business cycle. On the other hand, Hamilton's model is limited to the case of two regimes, even it can capture the short and steep pattern of recessions relative to expansions, it ignores another important feature of the business cycle which has been documented during the sample period: Typically, recessions were entailed with high growth recovery phases that bring output back to its per-recession level. Sichel (1994) and Boldin (1996) extended Hamilton's model to a three-regime Markov-switching model, and applied it to capture a high growth recovery state. In line with these studies, Kim and Murray (2002) and Kim and Piger (2000) suggest dividing the business cycle into three phases: recession, high growth and normal growth.

Inasmuch as these limitations of the MSAR model have been discovered, we propose to use the GMAR model to study the US GDP data. One obvious reason for this choice is that the GMAR can be viewed as a time inhomogeneous MSAR model[2], which has endogenously determined state variables.

[2] *See discussion in section 2.4 of Kalliovirta, Meitz, and Saikkonen (2015).*

In the following parts of this section, we discuss empirical evidence that lend support of using the GMAR model as a novel alternative model for the U.S. GDP growth dynamics.

4.2 Estimation results

We estimate GMAR models with different number of states ($M \in \{2, 3\}$) and lags ($p \in \{2, \cdots, 5\}$). Figure 2 depicts the data[3] used in this empirical analysis, which consists of the quarterly U.S. GDP growth series from 1947:Q1 to 2015:Q1.

To present our empirical results, we start by checking the plots of the marginal parameter posterior distributions. We first look at the GMAR model with three states and two lags ($M = 3, p = 2$). Figures 3 to 8 plot the scatter plot and histogram of simulated $log\sigma_1$, $log\sigma_2$, $log\sigma_3$, respectively. We observe two distinguish modes in both figures. Figure 9 to 14 provide the scatterplot of $(log\sigma_1, log\sigma_2)$, $(log\sigma_1, log\sigma_3)$, $(log\sigma_2, log\sigma_3)$ and their joint histogram respectively. As a common pattern of these joint histograms, there are at least four identifiable modes, two of them are quite obvious, whereas the rest are not very clear. This observation is not surprising, because according to Celeux, Hurn and Robert (2000), an M-component mixture model may have up to $M!$ possible (local) submodes[4]. We also observe similar bimodality pattern in the two-state, two-lag ($M = 2$, $p = 2$) GMAR model. Figures 16 to 19 display the histograms of $log\sigma_1$ and $log\sigma_2$. Figures 20, 21 provide the scatterplot of $(log\sigma_1, log\sigma_2)$ and their joint histogram, where we see only two distinguished modes, with almost no outlier. However, the observed bimodality in parameter σ warrant further examination as we notice all other parameters in these models are unimodally distributed (see Figure 15 and Figure 22). A potential issue here is the so-called label switching problem, which is common in Bayesian estimation of mixture model. It arises when sample from the unconstrained posterior distribution, since the label of the mixture component is switching over time, it is therefore unknown that the sampled parameter corresponds to which of the labeled subspace. Ignore this label switching problem may result the unconstrained posterior distribution to be sensitive to the permutation of the components of the mixture model.

To examine if the posterior distribution of σ_i ($i \in \{1, \cdots, M\}$) is invariant

[3] *Data are obtained from: https://research.stlouisfed.org/fred2/series/GDP/*

[4] *For instance, when $M = 3$ there will be $3! = 6$ local submodals.*

to label switching, also to identify different states of the US GDP within the sample period, following Frühwirth-Schnatte (2001), we consider to impose a labeling constraint to the unconstrained posterior distribution. There are two major criteria for the selection of this labeling constraint: first, it should serve the purpose of identify different states of the economy. To this extend, constraints like $\sigma_1 < \cdots < \sigma_M$ are excluded as it can not be used to distinguish recessions from expansions. Second, as discussed in Frühwirth-Schnatte (2001), the labeling constraint should take the geometry (for example, the multimodality pattern) of the interested marginal parameter posterior distribution into account. In our case, as shown above, only parameters σ_i, $(i \in \{1, \cdots, M\})$ exhibiting bimodality. If we use $\sigma_1 < \cdots < \sigma_M$ as our constraint, we are actually introduce a bimodal bias toward those unimodally distributed parameters.

In particular, we opt to impose constraint $\mu_1 < \mu_2 \cdots < \mu_M$ on the unconstrained posterior distribution, where μ_m is the unconditional mean of state m, $m \in (1, \cdots, M)$. We then permute all particles according to this constraint, and calculate the SMC estimates. Table 1 reports the estimates of four different GMAR models. As a general observation, for three-lag GMAR models[5], the estimate of the first two autoregressive components are all positive and significantly different from zero, whereas the estimates for the third autoregressive component are all negative and very close to zero. This finding verifies the well known asymmetry of real output across different business cycles (see Nelson and Plosser (1982), and Cochrane (1988)). Since similar asymmetry is not observed in two-lag GMAR models, we believe that lag length of three is the threshold of capturing this important feature of the US economy. Table 1 also reveals another interesting finding: for three-state GMAR models, after imposing the labeling constraint, we identify three states with completely different unconditional means. We label the state with negative mean as the recession state, and the state with small positive mean (around 1) and bigger positive mean (around 3) as the normal and the fast growth state, respectively. This finding conforms with the results reported in Sichel (1994) and Boldin (1996), and reinforce the argument by Kim and Piger (2000) and Kim and Murray (2002) that suggest to divide the business cycle into three phases: recession, fast growth and normal growth.

To further examine the effects of the labeling constraint on the geometry

[5] *Similar result can also be found in the estimates of GMAR models with more than three lags.*

of marginal posterior distribution of σ, we consider a GMAR(3,3) model and plot the corresponding histograms of σ in Figure 23. The first row displays the histograms of the unconstrained posterior distributions of σ, where we can see bimodality in both regimes. The second row displays the histograms of σ after imposing the labeling constraint. Surprisingly, the labeling constraint almost completely removes the bimodality of σ in the recession (left column) and fast growth (right column) regimes. The resulting unique mode of σ in the recession regime is about 0.5, smaller than 1.2, which is the unique mode of the fast growth regime. Intuitively, this implies that the economy in the recession regime is less volatile than in the fast growth regime. It also suggests that possible regime switching has no effect on the uncertainty level of these two regimes. Meanwhile, for the identified normal growth regime, the bi-modes of σ have been reshaped by the labeling constraint to be more asymmetric: the dominating mode is around 0.5, and the dominated mode is around 1.2. We tend to interpret 0.5, the dominating mode, as the general uncertainty level of this regime, and ascribe the increase in the second mode to the uncertainty entailed with possible regime switching.

We have the impression, among the four candidate models presented in Table 1, the GMAR(3,3) apparently is the most appropriate model. It does not only capture the asymmetry of output across different business cycles, but also identify three distinguish states of the US GDP growth. To formally evaluate evidence for GMAR models with different specifications, we adopt Bayesian model selection framework.

4.3 Model selection

The Bayesian model selection is based on the marginal likelihoods of data given GMAR models with different states and lags. Thanks to the powerlessness of the SMC algorithm, we obtain these nontrivial model likelihoods as by-products of our estimation. Table 2 lists log marginal likelihoods of data given different GMAR models. For computational reason, these quantities are calculated from the unconstrained[6] parameter posterior distribution. The bigger the model likelihood is, the better the model fits the data. As we noted, in general, GMAR models with more states and longer lags tend to perform better.

[6] *According to Bayesian theory, the inclusion of the identification constraint does not change the model likelihoods.*

To offer direct comparison, Table 3 combines the results of log Bayesian Factor ($2lnK$) for different $GMAR(p, M)$ models, where $p \in (2, \cdots, 5)$, $M \in (2, 3)$. These quantities are calculated[7] from the log marginal likelihoods presented in Table 2. For simplicity, we use the notation $\mathbf{G}(p, M)$ in Table 3 to represent the GMAR model with p lags and M states. The first column of Table 3 lists GMAR models with different lags and number of regimes. Each of these models is used as the benchmark model to compare with alternative models listed in the first row of Table 3. The resulting log Bayesian Factors may take different signs, where the positive sign indicates the benchmark GMAR model (in the column) is supported against the alternative GMAR model (in the row), and the negative sign indicates there is evidence against the benchmark model. The numerical value indicates the strength of evidence. We use the Kass and Raftery (1995) scale as the interpretation of the strength of the evidence, Table 4 provides the scale.

Based on the results reported in Table 3, we find: given lag length, three-state GMAR models always outperform two-state alternatives. On the other hand, given number of states, the model likelihood is increasing with the lag. In particular, evidence confirms that lag length of three is a threshold for the GMAR model: when the lag is below this threshold, the model likelihood is very low. Otherwise, the performance of the GMAR model is dramatically improved. This substantial improvement in model likelihoods implies that the asymmetry pattern of the US GDP across different economic states is too important to be ignored. Therefore, while using the GMAR model to study the US GDP data, we need to include at least three lags to capture this pattern.

Last but not least, from Table 2 and Table 3, we notice that the model likelihood is keep increasing with the number of states and the lag length. This monotonicity proves that the GMAR model has the ability to extract information from more states and lags. However, after a certain threshold (number of states and lags), the marginal improvement in model evidence is decreasing[8]. It is therefore not always optimal to assume more states and longer lags for the GMAR model. Owing to the fact that, including more states and lags entails the estimation of more parameters. For the quarterly

[7] *The log Bayesian Factor is calculated as two times the difference between the log marginal likelihood of the benchmark model and the log marginal likelihood of the alternative model.*

[8] *According to Kass and Raftery Scale, after the threshold* $(M = 3, p = 5)$*, the evidence improvement from using more complex specification is becoming ignoble.*

US GDP data of concern, given less than three hundred observations, including too many (eg, more than three) states and too long (eg, more than five) lags may lead to overfitting the model (see Hamilton (2015)). Moreover, formal economic justification of using more (than three) regimes still deserve further investigation.

5 Conclusion

In this paper, we investigated using SMC algorithm to estimate the GMAR model proposed by Kalliovirta, Meitz, and Saikkonen (2015). After presenting our SMC estimation procedure, we considered an empirical example regarding the GMAR model of the US GDP growth data. To examine the potential label switching problem discussed in Frühwirth-Schnatte (2001), also to efficiently identify different states of the US GDP within the sample period, after estimation, we imposed a labeling constraint to the unconstrained posterior distribution. We also conducted Bayesian model selection based on the marginal likelihoods of data given different specifications of the GMAR model to evaluate their appropriateness.

The contribution of this paper is fourfold: first, we estimate the GMAR model using SMC algorithm, which has several attractive properties: It is robust to multimodal posteriors; It has a linear computational complexity $O(N)$ and ready for parallelism; It is capable of providing fast on-line estimation when new data is available. Second, to identify different states of the US GDP growth, following Frühwirth-Schnatte (2001), we imposed a labeling constraint to the unconstrained parameter posterior distribution. This constraint also helps us to recognize the bimodality in the posterior distribution of σ_i ($i \in \{1, \cdots, M\}$). Third, we calculate marginal likelihood of data for different GMAR models as by-products of our SMC estimation, and conduct Bayesian model selection based on these model likelihoods. Comparing to information based criteria (like AIC, or BIC), Bayesian model selection provides a unified and easy-to-use framework for direct comparison of GMAR models with different setting. Fourth, we implement our SMC algorithm on the nVidia CUDA GPU parallel computing platform, which further facilitates the realization of the compute-intensive algorithm. The inexpensive parallel strategy discussed in this paper also provides guideline for future computation-based researches.

Appendix A: Residual resampling scheme

Algorithm (pseudo-code): Residual resampling

1 Calculate the integer part: $Int \leftarrow \Sigma\lfloor N. \times w_t \rfloor$
2 Calculate the Residual $\leftarrow N - Int$
3 Calculate the modified weights: $\tilde{w}_t \leftarrow \dfrac{N. \times w_t - Int}{Residual}$
4 *for* $j \leftarrow 1 : N$
5 *for* $i \leftarrow 1 : Int(j)$
6 $index(i) \leftarrow j$
7 $i \leftarrow i + 1$
8 *end*
9 *end*
10 Calculate the Cumsum of weights: $\Sigma \leftarrow cumsum(w_t)$
11 *while* $i \leq N$
12 *draw* $\lambda \sim U(0, 1]$
13 $j \leftarrow 1$
14 *while* $\Sigma(j) < \lambda$
15 $j \leftarrow j + 1$
16 *end*
17 $index(i) \leftarrow j$
18 $i \leftarrow i + 1$
16 *end*

Appendix B: Parallel considerations

Parallel computing is a computation strategy in which numerous calculations are carried out simultaneously, operating on the principle that large problems can often be divided into smaller ones, and then solved concurrently. In high-performance computing industry, parallel computing has been employed for many years and already become the dominant paradigm. In order to facilitate the realization of the compute-intensive GMAR estimation, in light of the general-purpose computing on graphics processing units (GPGPU) philosophy, we tailor our SMC algorithm and parallelize it on a consume level nVidia CUDA compatible Graphical Process Unit (GPU) card. In this section, we briefly discuss our parallel implementation of the SMC algorithm on the nVidia CUDA platform. In addition, we provide a technical appendix describing the details of parallelization at pseudo code level.

In the past decade, the highly parallel graphics processing unit (GPU) has rapidly gained maturity as a powerful engine for computationally demanding applications, and the General Purpose Computing on Graphics Processing Units (GPGPU[9]) based parallelism is becoming popular. Unlike other parallel computing strategies that require expensive, dedicated hardware, GPUs are very common in today's hardware. GPGPU compatible GPUs can be found in almost every new computer (laptops, PCs, workstations, even clusters). Powerful yet easy to access, GPGPU is potentially the most cost effective high performance computing platform.

CUDA[10], also known as the Compute Unified Device Architecture, is a parallel GPGPU computing platform developed by nVidia. It is designed jointly at software and hardware levels to enable the use of the GPU in general-purpose. At hardware level, any GPU card equipped with the nVidia CUDA compatible chips can be used for general-purpose computation. Most of these cards are capable for conducting billions of floating point[11] operations per second, some high performance computation dedicated cards even have supercomputer level performance[12]. At software level, the CUDA provides application programming interface (APIs) that gives programmer direct

[9] *GPGPU is a computation solution to use the graphics processing unit (GPU) to perform computation.*

[10] *http://www.nvidia.com/object/cuda_home_new.html*

[11] *Floating point is a formulaic representation used in computing that approximates a real number so as to support a trade-off between range and precision.*

[12] *For example, the nVidia GeForce Titan Black, one of the GPU cards we used to*

access to the GPU's virtual instruction set and parallel computational elements. This interface makes the CUDA an ideal abstraction for programmers to achieve parallel speed-up from parallel data operations without taking care of too much details. Typically, CUDA GPUs can execute our algorithms from 10 to 50 times faster than standard CPUs.

Although the CUDA platform is both powerful and promising, the GPU programming differs significantly from traditional CPU programming. It requires the architecture to be exploited by algorithms. In practice, transplant existing codes to GPU platform for acceleration is more demanding than simply move from one CPU platform to another. Significant changes must be made to the code, not only on algorithms but also on data structures. To achieve a high performance parallel computing on the CUDA GPU platform, we need to consider both the hardware and the software aspects.

With respect to the hardware, modern computers have a typical heterogeneous architecture which consists of one or more multicore CPUs and GPUs. Under this architecture, instead of being standalone device, GPUs must operate in conjunction with a CPU based host through a PCI express bus[13]. Since CPUs are optimized for instruction intensive tasks and GPUs are designed for data intensive operations. Their complementary attributions enable the best performance by using both of them.

---

conduct the estimation, has 5.1 TFLOPS (Tera floating-point operations per second) for single-precision calculation, and 1.3 TFLOPS for double-precision calculation. 1 TFLOPS $= 10^{12}$ *FLPOS.*

[13] *In terms of GPU programming, we refer the CPU as the 'host' and the GPU as the 'device'.*

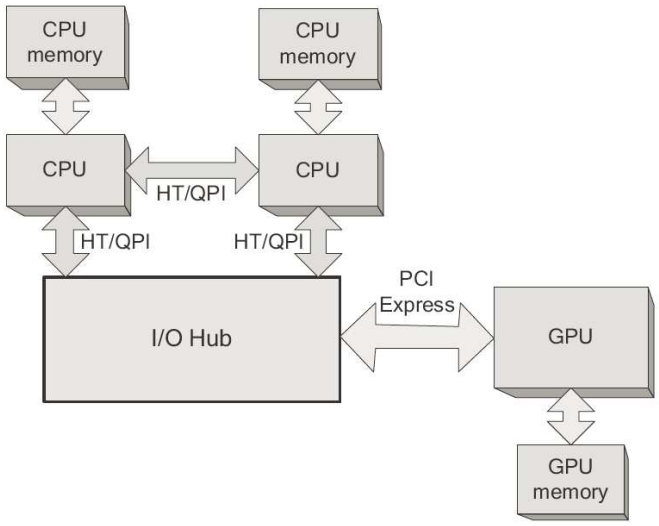

Figure 1: Illustration of the heterogeneous (CPU/GPU) architecture.

According to Flynn's Taxonomy, a widely used classification of computer architecture, GPU belongs to the Single Instruction Multiple Data (SIMD) parallel architecture. It means there are multiple cores (thread processors), all of them independently execute the same instruction stream at any time. The CUDA using a similar Single instruction, multiple thread (SIMT) architecture, where multi-threading is simulated by GPU processors. These processors, say a number n of them, are capable to execute many more than n tasks. This is achieved as each processor has multiple 'threads', which execute in lock-step, and are analogous to SIMD 'lanes'.

With respect to the software, GPGPU is especially well-suited for algorithms exhibiting two properties: data parallel[14] and throughput intensive. The former means that a processor can execute the operation on different data elements simultaneously. It focuses on mapping data elements to parallel threads and operate at the same time. The later means that the algorithm is designed to process massive data elements, so there should be plenty of them to operate on in parallel. Taking advantage of these two properties, GPUs achieve extreme performance by incorporating thousands of relatively simple processing units to operate on many data elements simultaneously.

[14] *In general, there are two fundamental types of parallelism model, task-based parallelism and data-based parallelism. Task-based parallelism focuses on distributing instructions across multiple cores and mainly implemented in CPU and operating system.*

Appendix C: Flow chart of the SMC algorithm

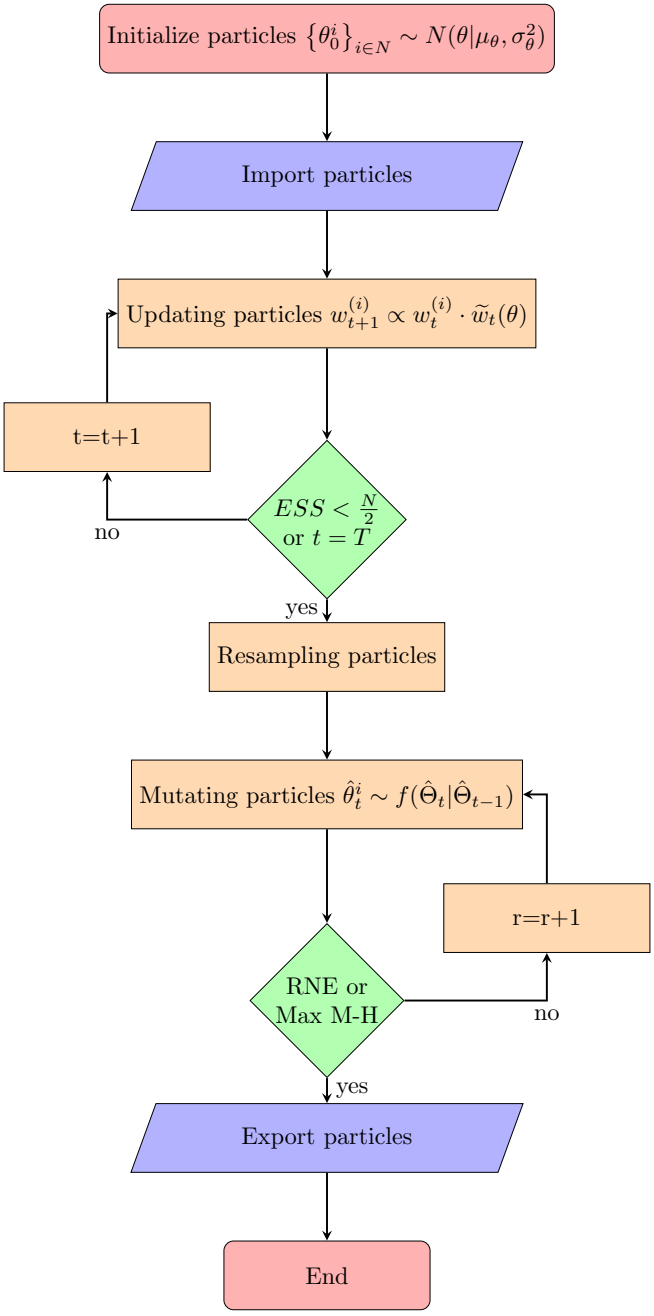

Table 1: SMC Estimates (labeling constraint: $\mu_1 < \cdots < \mu_M$)

	$GMAR(2,2)$	$GMAR(3,2)$	$GMAR(2,3)$	$GMAR(3,3)$
σ_1	0.7324	0.5515	0.9614	1.3817
σ_2	1.1110	1.1691	0.9026	0.7580
σ_3			0.9962	1.1987
α_1	0.4880	0.4904	0.2468	0.2109
α_2	0.5120	0.5096	0.3971	0.4060
α_3			0.3562	0.3831
μ_1	1.3805	1.3489	-1.1175	-1.6580
μ_2	1.7416	1.7298	1.4947	1.4337
μ_3			3.4533	3.4731
$\varphi_{1,0}$	0.6795	0.7115	0.4193	0.2006
$\varphi_{2,0}$	0.7373	0.8254	0.7408	0.7031
$\varphi_{3,0}$			1.0215	0.8392
$\varphi_{1,1}$	0.2842	0.2703	0.3256	0.3719
$\varphi_{1,2}$	0.2222	0.3258	0.0549	0.0806
$\varphi_{1,3}$		-0.1227		-0.0407
$\varphi_{2,1}$	0.4223	0.4497	0.3279	0.3443
$\varphi_{2,2}$	0.1529	0.1692	0.1688	0.2726
$\varphi_{2,3}$		-0.0973		-0.1101
$\varphi_{3,1}$			0.3954	0.4555
$\varphi_{3,2}$			0.1036	0.1601
$\varphi_{3,3}$				-0.0711

Table 1: SMC estimates for four different GMAR models. In particular, let **GMAR**(p, M) *denote the GMAR model with* p *lags and* M *regimes. These estimates are calculated by Monte Carlo approximation. To identify different states, following Frühwirth-Schnatte (2001), we impose a labeling constraint on* μ_m, $m \in (1, \cdots, M)$, *such that* $\mu_1 < \mu_2 \cdots < \mu_M$. *All particles then permuted according to this constraint.*

Table 2: log marginal likelihoods of data given model

	$M = 2$	$M = 3$	$M = 4$	$M = 5$
$p = 2$	-362.7964	-362.5500	-361.7417	-361.6124
$p = 3$	-353.6275	-353.3204	-351.7975	-351.9863
$p = 4$	-349.9917	-349.7407	-348.2111	-348.5294
$p = 5$	-347.8468	-347.3192	-346.9334	-347.0751

Table 2: log marginal likelihoods of data given different GMAR models. The bigger the log marginal likelihood is, the better the model is.

Table 3: log Bayesian Factor ($2lnK$)

	$G(2,2)$	$G(3,2)$	$G(4,2)$	$G(5,2)$	$G(2,3)$	$G(3,3)$	$G(4,3)$	$G(5,3)$
$G(2,2)$	0	-18.3378	-25.6094	-29.8992	-0.4928	-18.9520	-26.1114	-30.9544
$G(3,2)$	18.3378	0	-7.2716	-11.5614	17.8450	-0.6142	-7.7736	-12.6166
$G(4,2)$	25.6094	7.2716	0	-4.2898	25.1166	6.6574	-0.5020	-5.3450
$G(5,2)$	29.8992	11.5614	4.2898	0	29.4064	10.9472	3.7878	-1.0552
$G(2,3)$	0.4928	-17.8450	-25.1166	-29.4064	0	-18.4592	-25.6186	-30.4616
$G(3,3)$	18.9520	0.6142	-6.6574	-10.9472	18.4592	0	-7.1594	-12.0024
$G(4,3)$	26.1114	7.7736	0.5020	-3.7878	25.6186	7.1594	0	-4.8430
$G(5,3)$	30.9544	12.6166	5.3450	1.0552	30.4616	12.0024	4.8430	0

Table 3: log Bayesian Factors ($2lnK$) for different GMAR models, where the notation $\mathbf{G}(p, M)$ represents a GMAR model with p lags and M regimes. The first column of this table lists different GMAR models used as the benchmark models to compare with alternative GMAR models listed in the first row. These log Bayesian Factors are calculated as two times the difference of the log marginal likelihood of the benchmark model and the log marginal likelihood of the alternative model. The resulting values may take different signs, where the positive sign indicates the benchmark GMAR model (in the column) is supported against the alternative GMAR model (in the row), and the negative sign indicates there is evidence against the benchmark.

Table 4: Kass and Raftery (1995) Scale of Bayesian Factor

Bayesian Factor ($2lnK$)	**Strength of evidence**
0 to 2	not worth more than a bare mention
2 to 6	positive
6 to 10	strong
$>$10	very strong

References

[1] Boldin, M. (1992). Using Switching Models to Study Business Cycle Asymmetries: Overview of Methodology and Applications. Federal Reserve Bank of New York Research Paper, 1992-11.

[2] Boldin, Michael. (1996). A Check on the Robustness of Hamilton's Markov Switching Model Approach to the Economic Analysis of the Business Cycle, Studies in Nonlinear Dynamics & Econometrics. 1(1), 1-14.

[3] Camacho, M., & Perez, Q. G. (2007). Jump-and-Rest Effect of U.S. Business Cycles. Studies in Nonlinear Dynamics & Econometrics. De Gruyter. 11(4), 1-39.

[4] Carvalho, C. M. Johannes, M. Lopes, H. F., & Polson, N. (2010). Particle learning and smoothing. Statistical Science 25, 88-106. 710-730.

[5] Celeux, G. Hurn, M., & Robert, C.P. (2000). Computational and Inferential Difficulties with Mixture Posterior Distributions. Journal of the American Statistical Association, 95(451), 957-970.

[6] Chopin, N. (2004). Central limit theorem for sequential Monte Carlo and its application to Bayesian inference. The Annals of Statistics, 32(6), 2385–2411.

[7] Chopin, N. Lelievre,T., & Stoltz, G. (2012). Free energy methods for Bayesian statistics: Efficient exploration of univariate Gaussian mixture posteriors. Statistics and Computing, 22(4), 897-916.

[8] Cogley, T., & Nason, J. M. (1995). Output Dynamics in Real-Business-Cycle Models. American Economic Review, American Economic . 85(3), 492-511.

[9] Del Moral, P., A. Doucet, & A. Jasra (2008). On adaptive resampling procedures for sequential Monte Carlo methods. Unpublished manuscript, Insitut National de Recherche en Informatique et en Automatique, Bordeaux, France.

[10] Doucet, A., N. de Freitas., & N. Gordon (2001). Sequential Monte Carlo Methods in Practice. New York, NY: Springer-Verlag.

[11] Dueker, M. J. Sola, M., & Spagnolo, F. (2007). Contemporaneous threshold autoregressive models: estimation, testing and forecasting. Journal of Econometrics, 141, 517-547.

[12] Durham,G. & Geweke, J. (2014). Adaptive sequential posterior simulators for massively parallel computing environments. Bayesian model comparison, 1-44.

[13] Fruhwirth-Schnatter, S. (2001). Markov chain Monte Carlo estimation of classical and dynamic switching and mixture models. Journal of the American Statistical Association, 96, 194-209.

[14] Geweke, J. (1989). Bayesian inference in econometric models using Monte Carlo integration. Econometrica, 57, 1317-1339.

[15] Geweke, J. (2007). Interpretation and inference in mixture models: Simple MCMC works. Computational Statistics and Data Analysis, 51(7), 3529 - 3550.

[16] Gordon, N. J. Salmond, D. J., & Smith, A. F. M. (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation. IEEE Proceedings F on Radar and Signal Processing, 140(2), 107–113.

[17] Gourieroux, C., & Robert, C. Y. (2006). Stochastic unit root models. Econometric Theory, 22, 1052-1090.

[18] Hamilton, J., (1989). A New Approach to Economic Analysis of Nonstationary Time Series. Econometrica, 57, 357–384.

[19] Herbst, E., & F. Schorfheide. (2014). Sequential Monte Carlo sampling for DSGE models. Journal of Applied Econometrics, 29, 1073–1098.

[20] Kass, R., & Raftery, A. (1995). Bayes Factors. Journal of the American Statistical Association, 90(430), 773-795.

[21] Kalliovirta, L., & Meitz, M and Saikkonen, P. (2015). A Gaussian mixture autoregressive model for univariate time series. Journal of Time Series Analysis, 36(2), 247-266.

[22] Kim, C.-J., & C. J. Murray (2002). Permanent and transitory components of recessions. Empirical Economics 27,(2), 163-183.

[23] Kim, C. J., Morley, J., & J. Piger. (2005). Nonlinearity and the Permanent Effects of Recessions, Journal of Applied Econometrics 20, 291-309.

[24] Kim, C.-J., Piger, J., & R. Startz. (2008). Estimation of markov regime-switching regressions with endogenous switching. Journal of Econometrics 143, (2), 263-273.

[25] Lanne, M., & J. Luoto. (2015). Estimation of DSGE Models under Diffuse Priors and Data-Driven Identification Constraints, CREATES Research Paper 2015-37.

[26] Lanne, M., & Saikkonen, P. (2003). Modeling the U.S. short-term interest rate by mixture autoregressive processes. Journal of Financial Econometrics, 1, 96-125.

[27] Liu J.S., & Chen R (1998). Sequential Monte Carlo Methods for Dynamic Systems. Journal of the American Statistical Association, 93(443), 1032-1044.

[28] Martin, V. L. (1992). Threshold time series models as multimodal distribution jump process. Journal of Time Series Analysis, 13(1), 79-94.

[29] Morozov, S., & Mathur. S. (2011). Massively Parallel Computation Using Graphics Processors with Application to Optimal Experimentation in Dynamic Control. Computational Economics, 1 –32.

[30] Morton, K. D., Torrione, P. A., & Collins, L.M. (2011). Variational Bayesian Learning for Mixture Autoregressive Models With Uncertain-Order. IEEE Transactions on Signal Processing, 59(6), 2614 - 2627.

[31] Nelson, C., & Plosser, C. (1982). Trends and random walks in macroeconmic time series: Some evidence and implications. Journal of Monetary Economics. 10(2). 139-162.

[32] Pitt, M., & Shephard, N. (1999). Filtering via Simulation: Auxiliary Particle Filters. Journal of the American Statistical Association, 94, 590-599.

[33] Rubin, D. (1988). Using the SIR algorithm to simulate posterior distributions. In Bayesian Statistics 3 (J. M. Bernardo, M. H. DeGroot, D. V. Lindley and A. F. M. Smith, eds.) 395–402. Oxford University Press.

[34] Sichel, D. E. (1994). Inventories and the three phases of the business cycle. Journal of Business and Economic Statistics, 12 (3), 269-277.

[35] Venkataramana, K., & Sekhar, C.C. (2009). Bayesian mixture of AR models for time series clustering. Advances in Pattern Recognition, 2009. ICAPR '09. 35-38

[36] Wong, C. S., & W. K. Li. (2000). On a mixture autoregressive model. Journal of the Royal Statistical Society: Series B, 62, 95-115.

[37] Wong, C. S., & W. K. Li. (2001). On a mixture autoregressive conditional heteroscedastic model. Journal of the American Statistical Association, 96, 982-995.

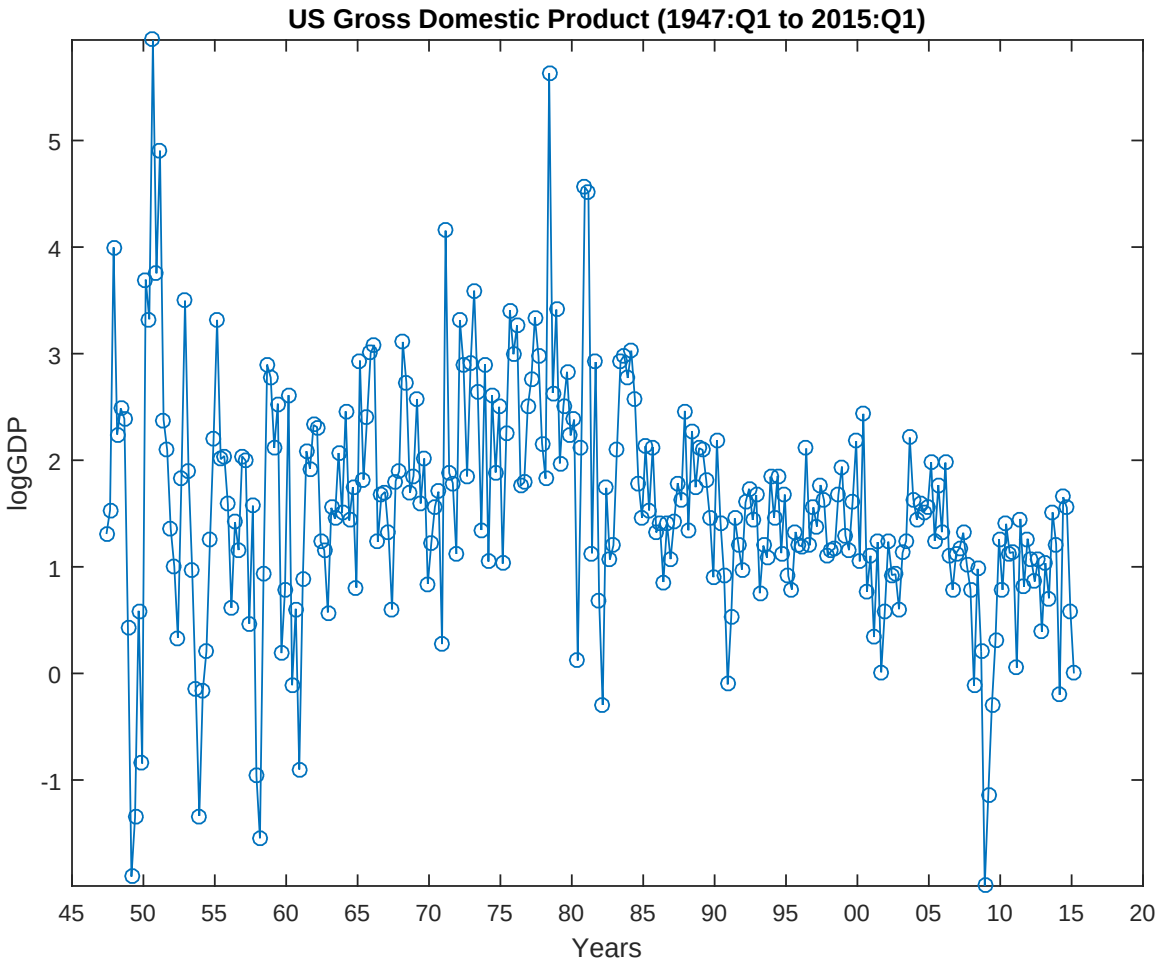

Figure 2 : US GDP growth (1947:Q1 to 2015:Q1)

Three regimes ($M = 3$):

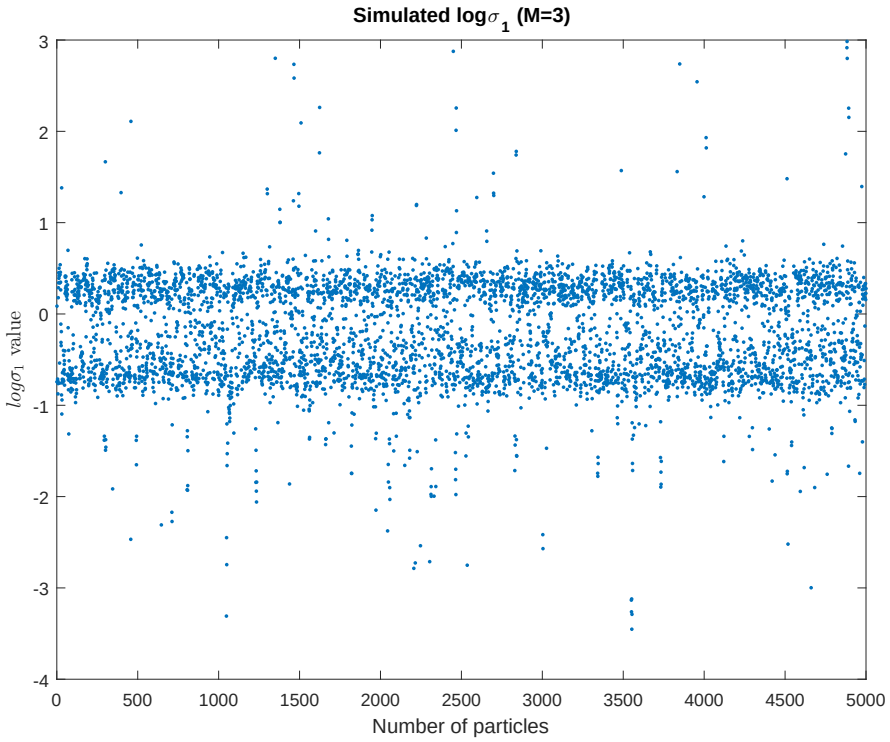

Figure 3: Simulated $log\sigma_1$

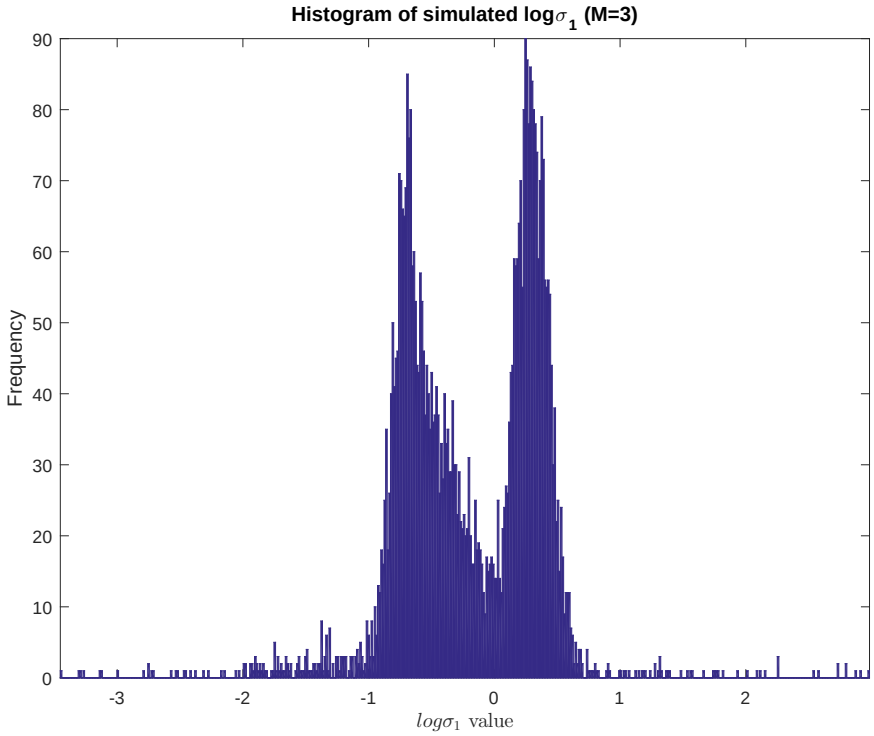

Figure 4: Histogram of simulated $log\sigma_1$

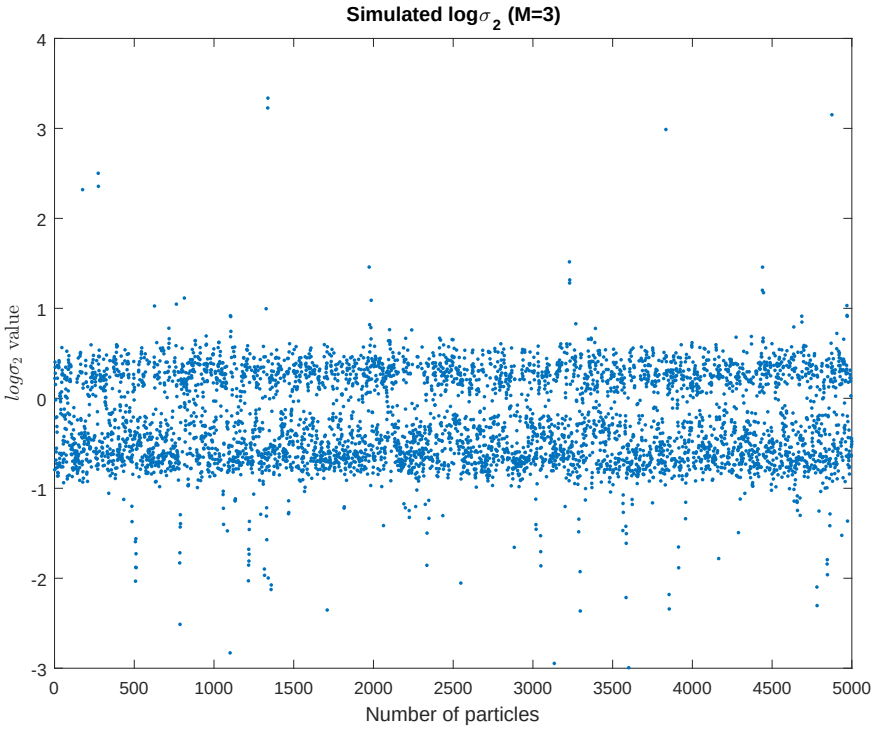

Figure 5: Simulated $log\sigma_2$

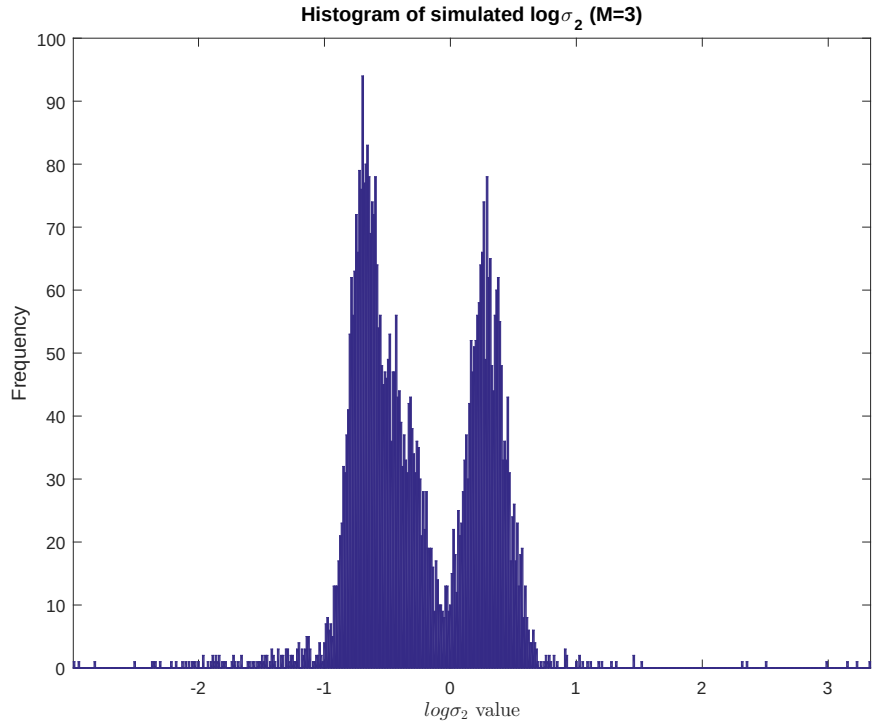

Figure 6: Histogram of simulated $log\sigma_2$

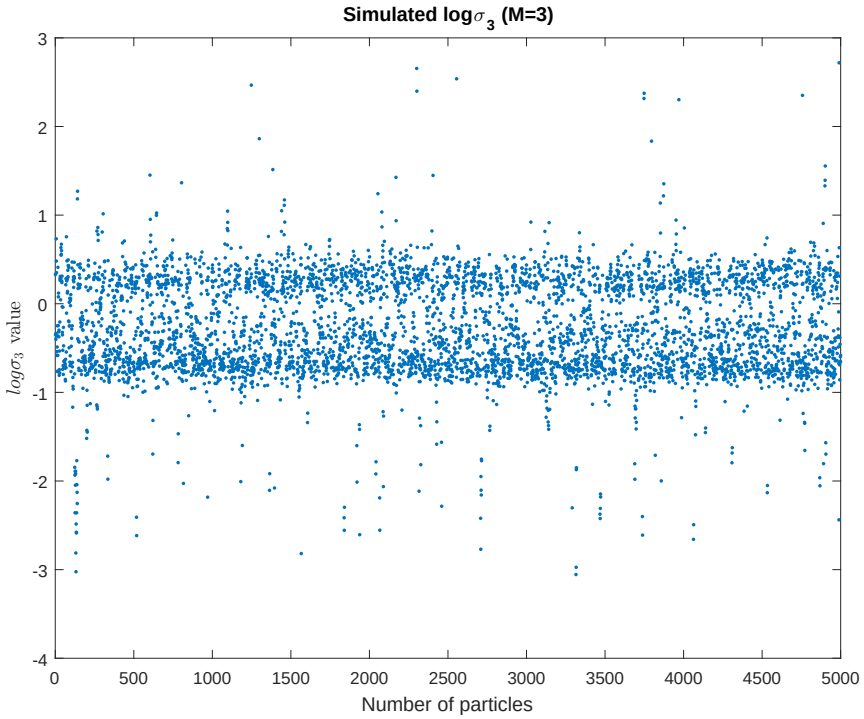

Figure 7: Simulated $log\sigma_3$

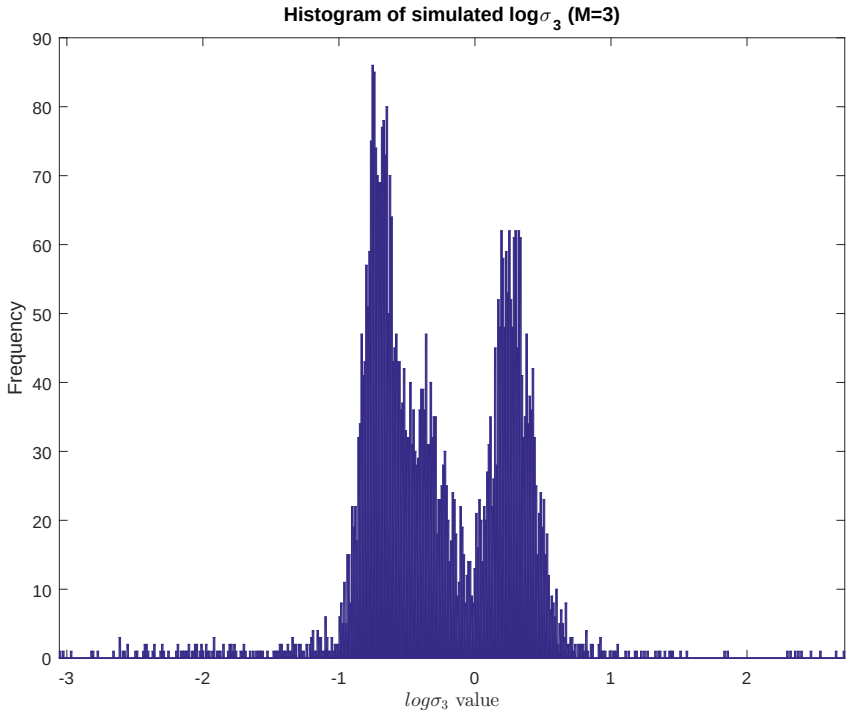

Figure 8: Histogram of simulated $log\sigma_3$

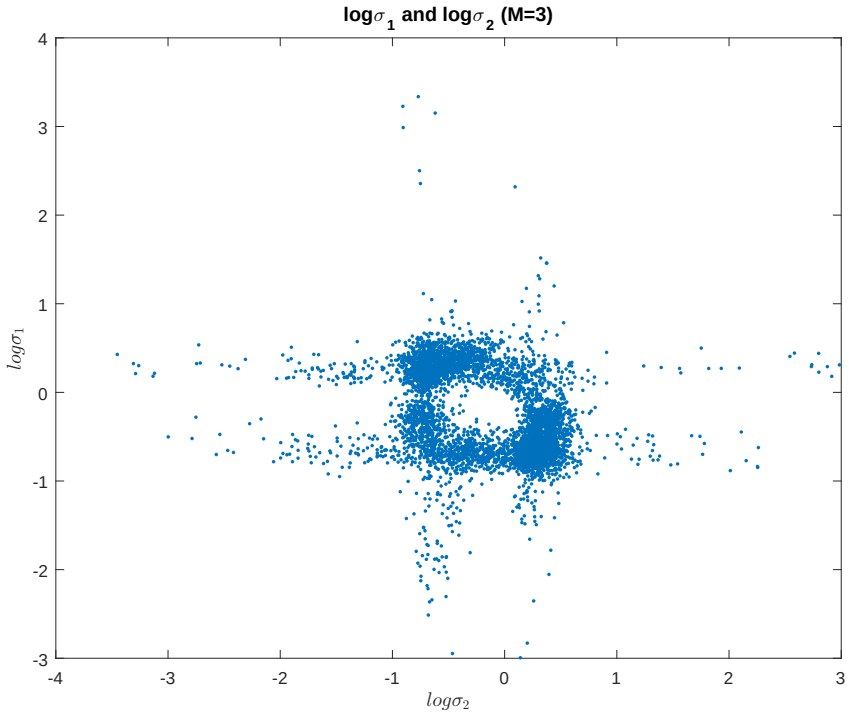

Figure 9: Simulated $log\sigma_1$ and $log\sigma_2$

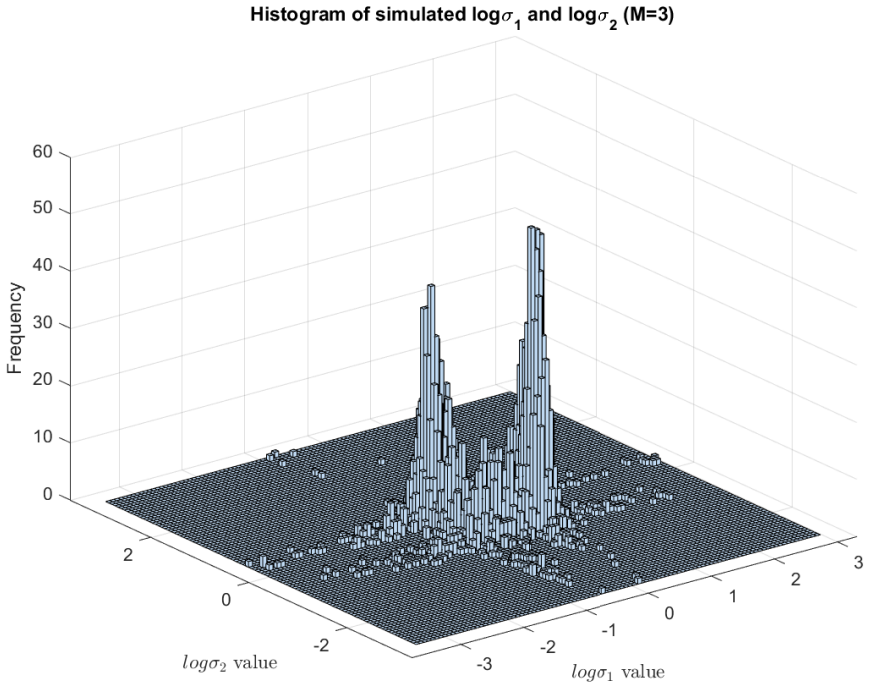

Figure 10: Histogram of simulated $log\sigma_1$ and $log\sigma_2$

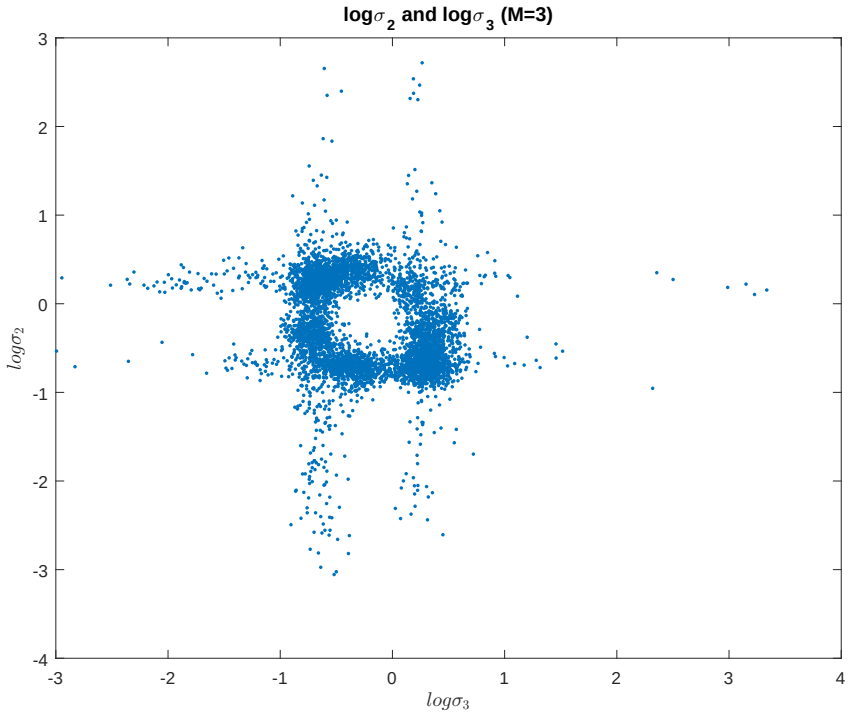

Figure 11: Simulated $log\sigma_2$ and $log\sigma_3$

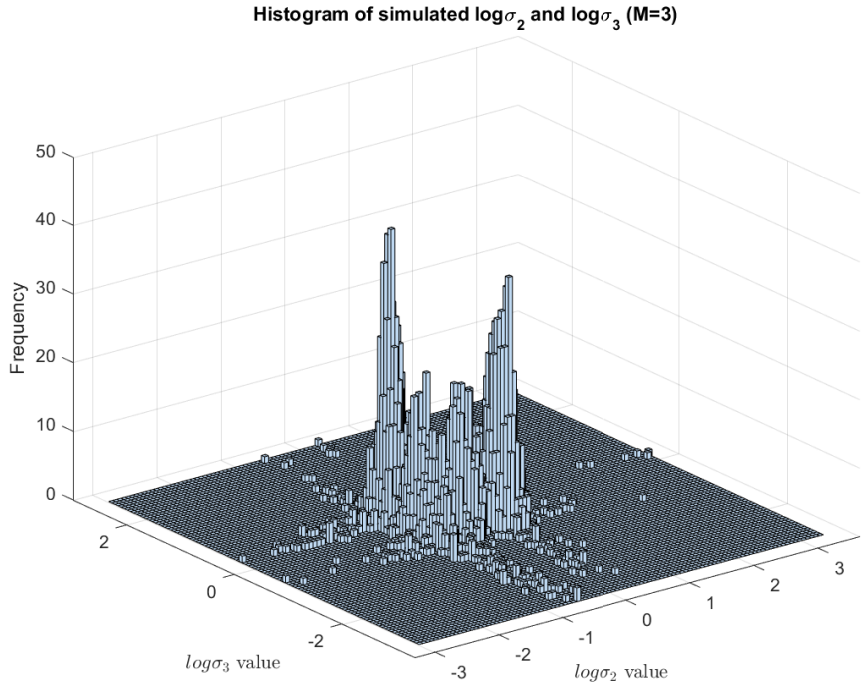

Figure 12: Histogram of simulated $log\sigma_2$ and $log\sigma_3$

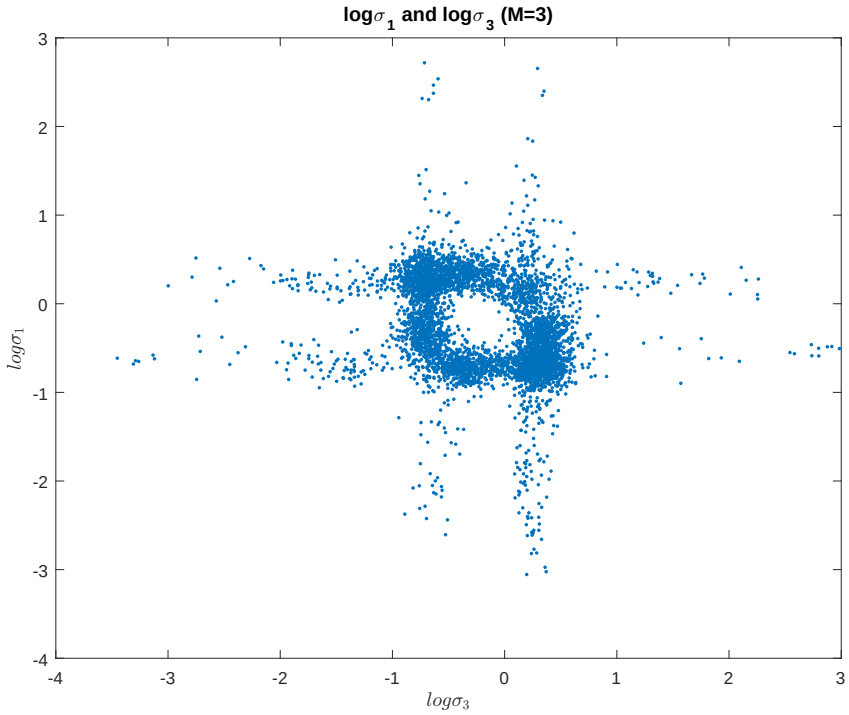

Figure 13: Simulated $log\sigma_1$ and $log\sigma_3$

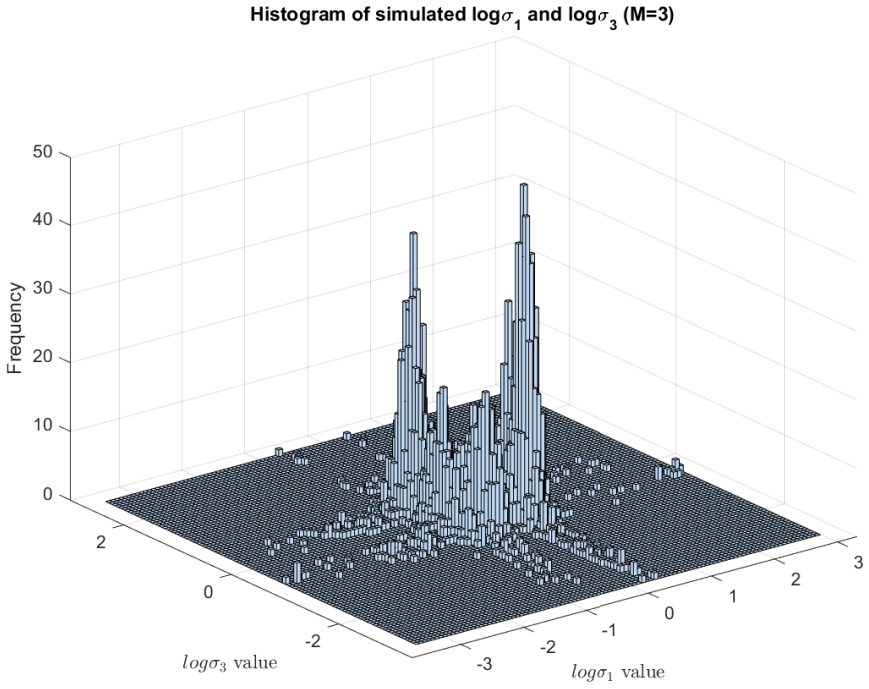

Figure 14: Histogram of simulated $log\sigma_1$ and $log\sigma_3$

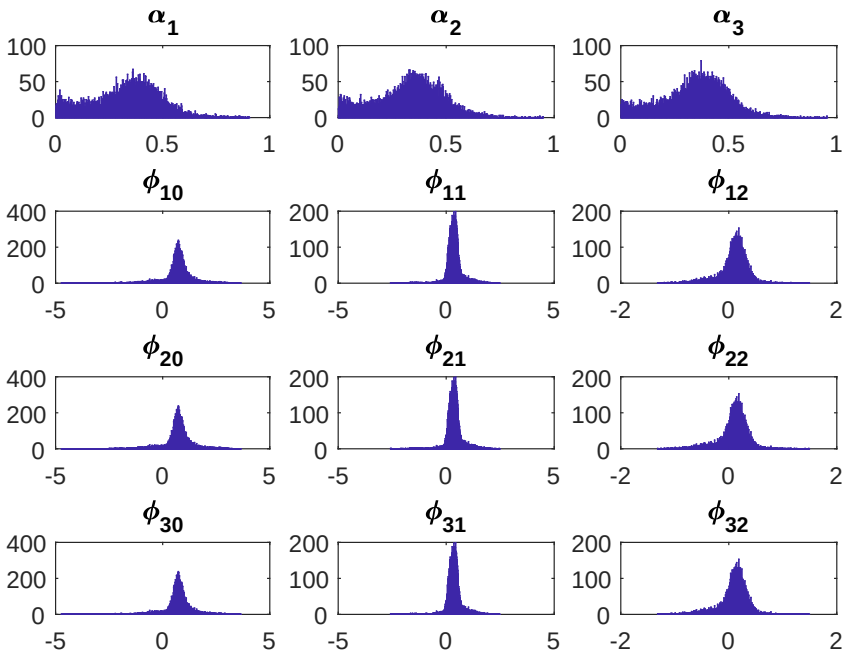

Figure 15: Histograms of simulated other (than σ) parameters ($M = 3, p = 2$)

Two regimes ($M = 2$):

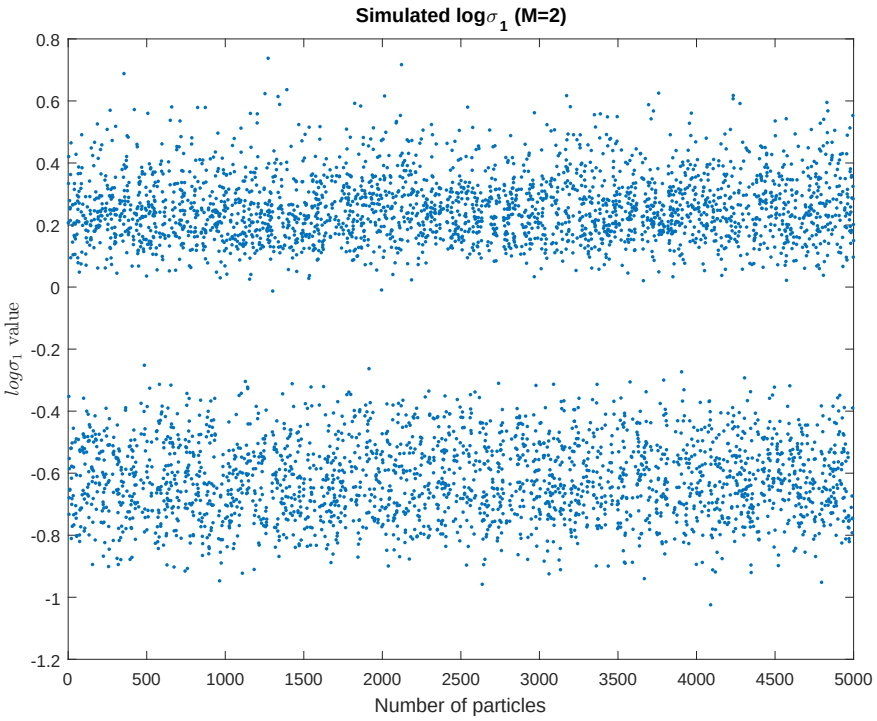

Figure 16: Simulated $log\sigma_1$

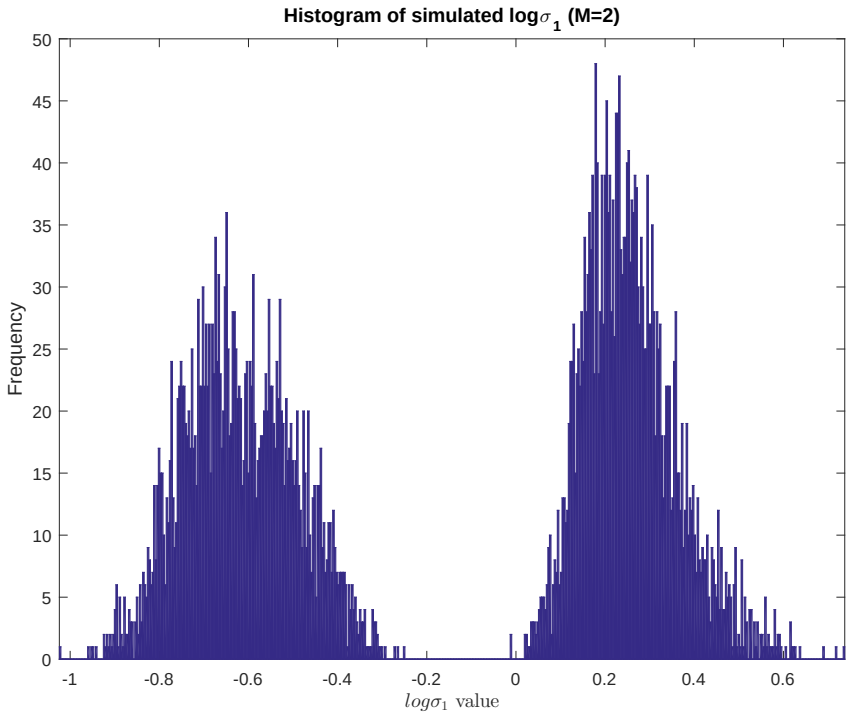

Figure 17: Histogram of simulated $log\sigma_1$

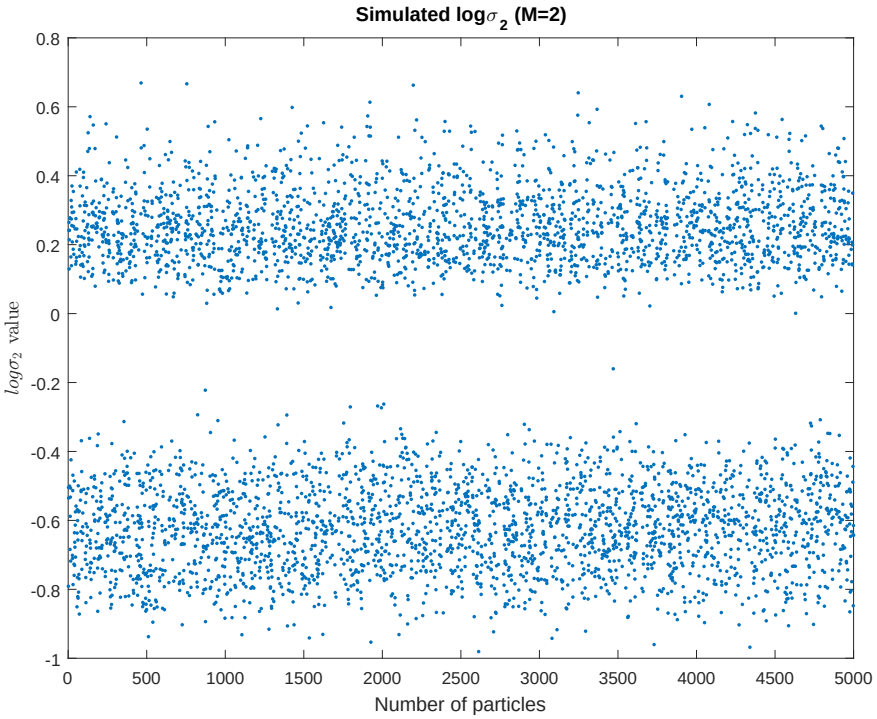

Figure 18: Simulated $log\sigma_1$

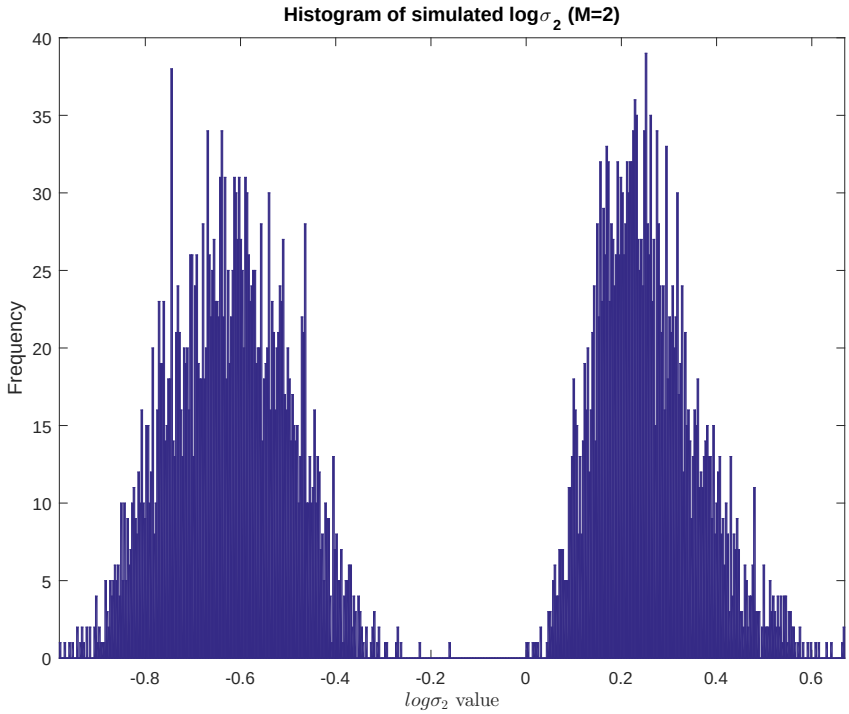

Figure 19: Histogram of simulated $log\sigma_2$

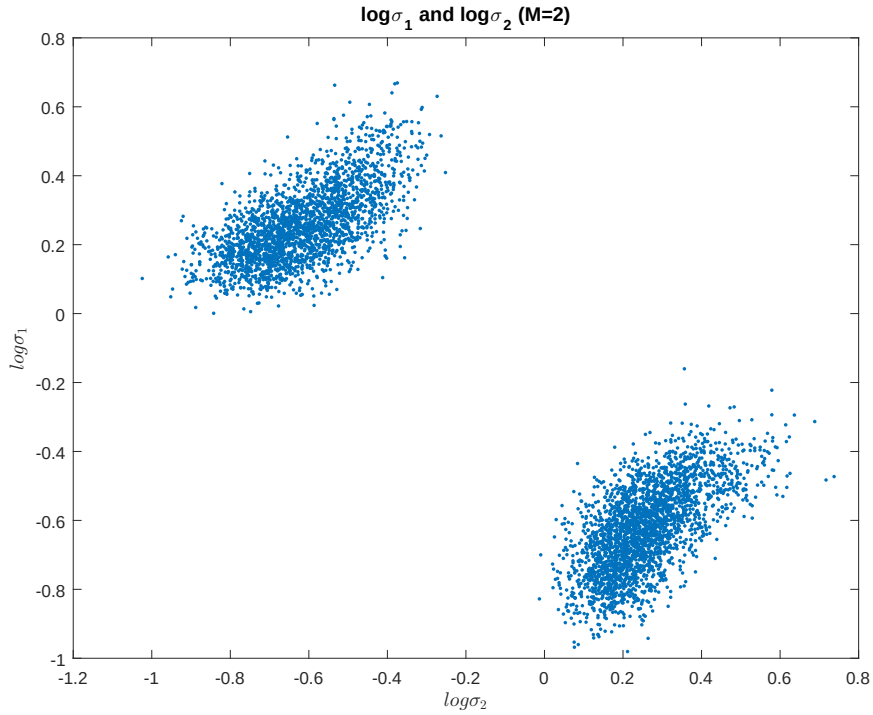

Figure 20: Simulated $log\sigma_1$ and $log\sigma_2$

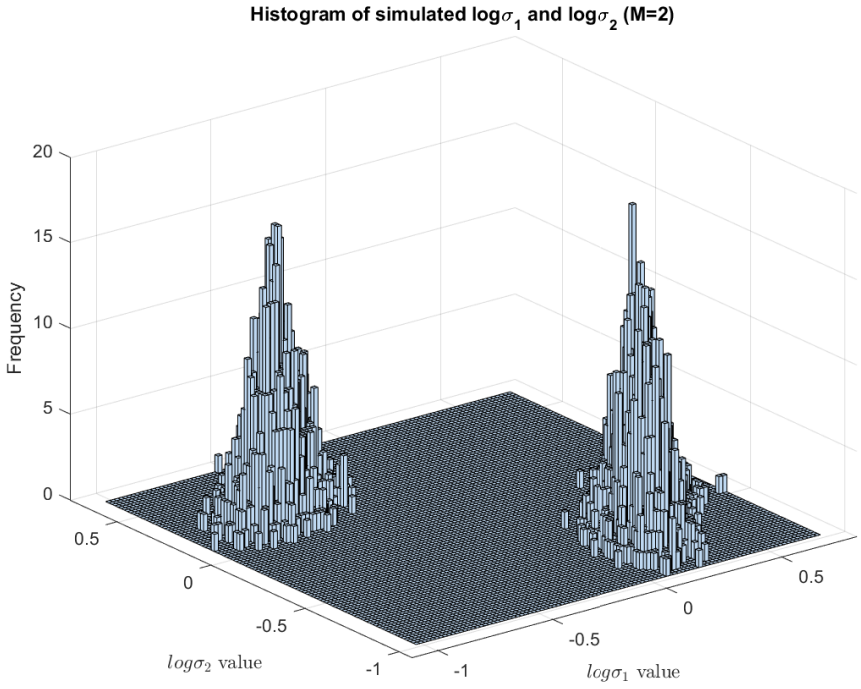

Figure 21: Histogram of simulated $log\sigma_1$ and $log\sigma_2$

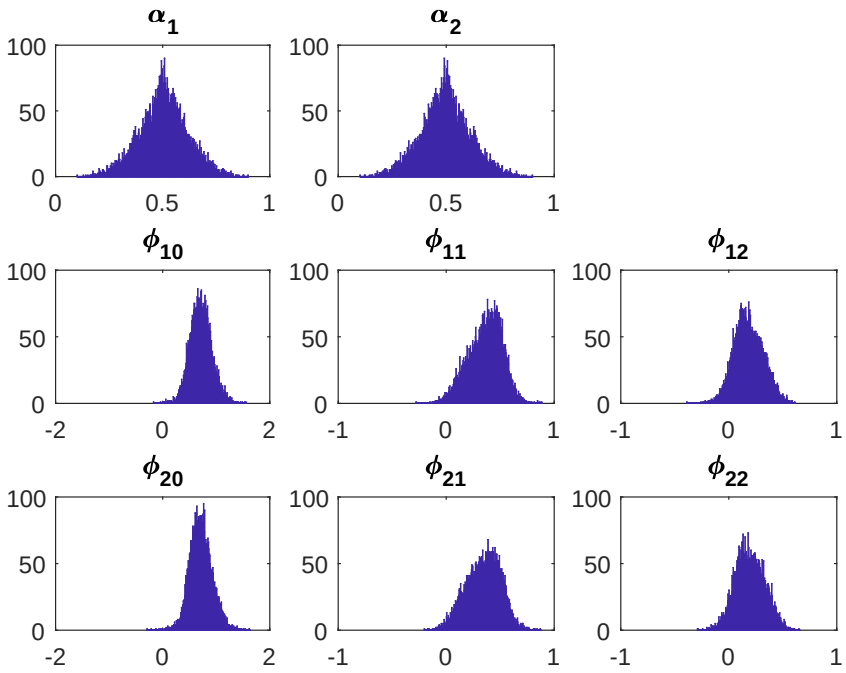

Figure 22: Histograms of simulated other (than σ) parameters ($M = 2, p = 2$)

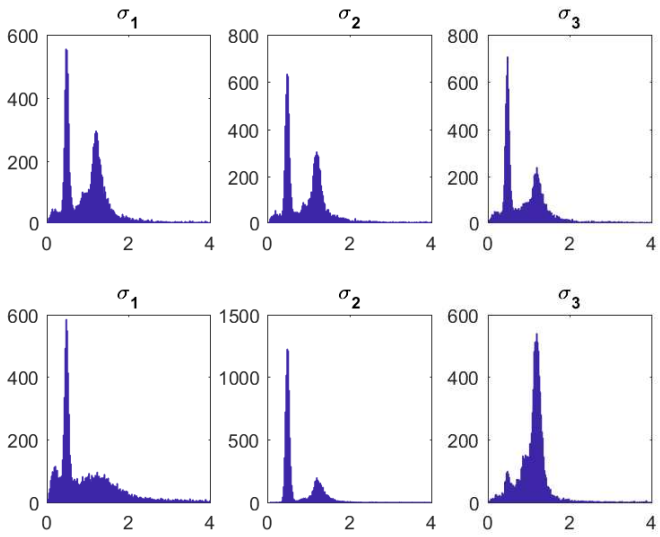

Figure 23: Histogram of σ, before and after impose labeling constraint ($M = 3, p = 3$)