Inference Using Simulated Neural Moments

Michael Creel

November 10, 2020

Universitat Autònoma de Barcelona, Barcelona GSE, and MOVE, Bellaterra (Barcelona) 08193, Spain. michael.creel@uab.cat.

Abstract

This paper studies Laplace-type estimators that are based on simulated moments. It shows that confidence intervals using these methods may have coverage which is far from the nominal level. A neural network may be used to reduce the dimension of an initial set of moments to the minimum number that maintains identification. When Laplace-type estimation and inference is based on these neural moments, confidence intervals have statistically correct coverage in most cases studied, with only small departures from correct coverage. The methods are illustrated by an application to a jump diffusion model for returns of the S&P 500 index.

Keywords: neural networks; Laplace type estimators; simulated moments; approximate Bayesian computing
JEL codes: C11, C12, C13, C45

1 Introduction

It has long been known that classical inference methods based on first-order asymptotic theory, when applied to the generalized method of moments estimator, may lead to unreliable results, in the form of substantial finite sample biases and variances, and incorrect coverage of confidence intervals, especially when the model is overidentified (Tauchen (1986), Hall and Horowitz (1996), Hansen, Heaton, and Yaron (1996), Donald, Imbens, and Newey (2009)). In another strand of the literature, Chernozhukov and Hong (2003) introduced Laplace type estimators, which allow for estimation and inference with classical statistical methods (those which are defined by optimization of an objective function) to be done by working with the elements of a tuned Markov chain, so that potentially difficult or unreliable steps such as optimization or computation of asymptotic standard errors, etc., may be avoided. A third important strand of literature is simulation-based estimation. The strands of moment-based estimation, simulation, and Laplace type methods meet in certain applications. The code by Gallant and Tauchen (Gallant and Tauchen (2002)) for efficient method of moments estimation (Gallant and Tauchen (1996)), which has been used in numerous papers, is an example. Another is Christiano, Trabandt, and Walentin (2010) (see also Christiano, Eichenbaum, and Trabandt (2016)), which proposes a Laplace type estimation methodology that uses simulated moments which are defined in terms of impulse response functions for estimation of macroeconomic modes. Very similar methodologies may be found in the very broad Approximate Bayesian Computing literature, some of which uses MCMC methods and criterion functions that involve simulated moments (*e.g.,* Marjoram et al. (2003)).

Given the uneven performance of inference in classical GMM applications, one may wonder how reliable are inferences made using the combination of Laplace type methods and simulated moments. This paper provides evidence that confidence intervals derived from such estimators may have poor coverage in some cases, and it provides evidence that a dimension reduction technique based computing simulated moments using a trained neural net can cause inferences to become much more reliable. The paper concludes with an example that uses the methods to estimate a jump-diffusion model for returns of the S&P 500 index.

The next section reviews how Laplace type methods may be used with simulated moments, and Section 3 then discusses how neural networks may be used to reduce the dimension of the moment conditions. Section 4 presents four simple models, and Section 5 gives results for the simple models. Section 6 illustrates the methods in the context of an empirical analysis of a model of more complexity, and a final section summarizes the conclusions.

2 Results and Discussion

2.1 Simulated Moments, Indirect Likelihood, and Laplace Type Inference

This section relies on results from the part of the simulation-based estimation literature that bases estimation on a statistic, including McFadden (1989), C. Gouriéroux, Monfort, and Renault (1993), Smith (1993) and Gallant and Tauchen (1996), among others, which is reviewed in Jiang and Turnbull (2004)). Suppose there is a model $M(\theta)$ which generates data from a probability distribution $P^{(\theta)}$ which depends on the unknown parameter vector θ. $M(\theta)$ is fully known up to θ, so that we can make draws of the data from the model, given θ. Let $Y = Y(\theta)$ be a sample drawn at the parameter vector θ, where $\theta \in \Theta \subset \mathbb{R}^k$ and Θ is a known parameter space. Suppose we have selected a finite dimensional statistic $Z = Z(\theta) = Z(Y(\theta))$ upon which to base estimation, and assume that the statistic satisfies a central limit theorem, uniformly, for all values of θ of interest:

$$\sqrt{n}\,(Z - E_\theta Z) \to^d N(0, \bar{\Sigma}(\theta)) \tag{1}$$

Let $Z^s(\theta) = Z(Y^s(\theta))$ be the statistic evaluated using an artificial sample drawn from the model at the parameter value θ. This statistic has the same asymptotic distribution as does $Z(\theta)$, and furthermore, the two statistics are independent of one another. With S such simulated statistics, define $m(\theta) = Z(\theta) - S^{-1}\sum_s Z^s(\theta)$ and $\bar{V}(\theta) = (1 + S^{-1})\bar{\Sigma}(\theta)$. We can easily obtain

$$\sqrt{n} m(\theta) \to^d N(0, \bar{V}(\theta)). \tag{2}$$

Now, suppose we have a real sample which was generated at the unknown true parameter value θ_0, and let $\hat{Z}$ be the associated value of the statistic. Define $\hat{m}(\theta) = \hat{Z} - S^{-1}\sum_s Z^s(\theta)$. With this, and eqn. 2,we can define the indirect likelihood function[1]

$$L = L(\theta|\hat{Z}) = \left|2\pi\hat{\bar{V}}(\theta)\right|^{-1/2} \exp(-\frac{1}{2}H) \tag{3}$$

where

$$H = H(\theta|\hat{Z}) = n \cdot \hat{m}(\theta)^T \hat{\bar{V}}^{-1}(\theta)\hat{m}(\theta), \tag{4}$$

where $\hat{\bar{V}}(\theta)$ is a consistent estimate of $\bar{V}(\theta)$.

To estimate $\bar{V}(\theta)$, one possibility is to use a fixed sample-based estimate that does not rely on an estimate of θ_0 (see, for example, Christiano, Trabandt, and Walentin (2010) and Christiano, Eichenbaum, and Trabandt (2016)). Another possibility is to (1) compute the

[1] These definitions and notation are loosely based on Jiang and Turnbull (2004).

estimate $\hat{\hat{\Sigma}}(\theta)$ of the covariance matrix in 1 as the sample covariance of R draws of $\sqrt{n}Z^s(\theta)$:

$$\hat{\hat{\Sigma}}(\theta) = \frac{1}{R}\sum_{r=1}^{R}(\sqrt{n}Z^r(\theta) - M)(\sqrt{n}Z^r(\theta) - M)', \tag{5}$$

where $M = \frac{1}{R}\sum_r \sqrt{n}Z^r(\theta)$ is the sample mean of the draws, and then (2) multiply the result by $1 + S^{-1}$ to obtain the estimate

$$\hat{\hat{V}}(\theta) = (1 + S^{-1})\hat{\hat{\Sigma}}(\theta). \tag{6}$$

This estimator may be used in a continuously updating fashion, by updating $\hat{\hat{V}}(\theta)$ in eqns. 3 or 4 every time the respective function is evaluated. Alternatively, if we obtain an initial consistent estimator of θ_0, then $\hat{\hat{V}}(\theta)$ can be computed at this estimate, and kept fixed in subsequent computations, in the usual two-step manner. Note that, if a fixed covariance estimator is used, then the maximizer of L is the same as the minimizer of H.

Extremum estimators may be obtained by maximizing $\log L$, or minimizing H. Laplace type estimators, as defined by Chernozhukov and Hong (2003), may be defined by setting their general criterion function, $L_n(\theta)$, as defined in their Section 3.1, to either $\log L$, or $-\frac{1}{2}H$. Once this is done, then the practical methodology is to use Markov chain Monte Carlo (MCMC) methods to draw a chain $C = \{\theta^r\}$, $r = 1, 2, ..., R$, given the sample statistic $\hat{Z}$, where acceptance/rejection is determined using the chosen $L_n(\theta)$, along with a prior, and standard proposal methods[2]. This paper will rely directly on the theory and methods of Chernozhukov and Hong (2003), just using the criterion functions presented above to define the specific Laplace type estimators. In the following, a primary use of the Chernozhukov and Hong (2003) methodology will be in order to obtain confidence intervals. For a function $f(\theta)$, Theorem 3 of Chernozhukov and Hong (2003) proves that a valid confidence interval can be obtained by using the quantiles of $\{f(\theta^r)\}_{r=1,2,..R}$, based on the final chain $C = \{\theta^r\}$, $r = 1, 2, ..., R$. For example, a 95% confidence interval for a parameter θ_j is given by the interval $(Q_{\theta_j}(0.025), Q_{\theta_j}(0.975))$, where $Q_{\theta_j}(\tau)$ is the τth quantile of the R values of the parameter θ_j in the chain C.

2.2 Neural Moments

The dimension of the statistics used for estimation, Z, can be made minimal (equal to the dimension of the parameter to estimate, θ) by filtering an initial set of statistics, say, W, through a trained neural net. Details of this process are explained in Creel (2017) and references cited

[2] It may be noted that methods other than MCMC may be used to generate the set of draws from the posterior, C. For example, one might use sequential Monte Carlo. Point estimation and inference using C remains the same regardless of how C is generated.

therein, and the process is made explicit in the code archive which accompanies this paper[3]. A summary of this process is: Suppose that W is a p vector of statistics $W = W(Y)$, with $p \geq k$, where $k = \dim \theta$. We may generate a large sample of (W, θ) pairs, following:

1. draw θ^s from the parameter space Θ, using some prior distribution (*e.g.,* a uniform distribution over Θ).

2. draw a sample Y^s from the model $M(\theta)$ at θ^s

3. compute the vector of raw statistics $W(Y^s)$.

We can repeat this process to generate a large data set $\{\theta^s, W^s\}$, $s = 1, 2, ..., S$, which can be used to train a neural network which predicts θ, given W. This process can be done without knowledge of the real sample data, and can in fact be done before the real sample data is gathered. The prediction from the net will of the same dimension as θ, and if the net is of an appropriate configuration and is well-trained using a squared error loss function, the output of the net will be a very accurate approximation to the posterior mean of θ conditional on W. The output of the net may be represented as $\hat{\theta} = f(W, \hat{\phi})$, where $f(W, \phi) : R^p \to R^k$ is the neural net, with parameters ϕ, that takes as inputs the p statistics W,and has $k = \dim \theta$ outputs. The parameters of the net, ϕ, are adjusted using standard training methods from the neural net literature to obtain the trained parameters, $\hat{\phi}$. Then we can think of $\hat{\theta} = f(W, \hat{\phi})$ as a $k-$dimensional statistic which can be computed essentially instantaneously once provided with W. We will use this statistic $\hat{\theta}$ as the Z of the previous section. Because the statistic is an accurate approximation to the posterior mean conditional on W (supposing the net was well trained), it has two virtues: it is informative for θ (supposing that the initial statistics W contain information on θ) and it has the minimal dimension needed to identify θ. From the related GMM literature, GMM methods are known to lead to inaccurate inference when the dimension of the moments is large relative to the dimension of the parameter vector (Donald, Imbens, and Newey (2009)). Use of a neural net as described here reduces the dimension of the statistic to the minimum required for identification.

When the statistic Z is the output of a neural net $f(W, \phi)$, where the parameter vector of the net, ϕ, can have a very high dimension (hundreds or thousands of parameters are not uncommon) the simulated likelihood of eqn. 3 will be a wavy function, with many local maxima. This will occur even if the net is trained using regularization methods. Because of this waviness, gradient-based methods will not be effective when attempting to maximize $\log L$ or to minimize H (eqns. 3 and 4), and attempts to compute the covariance matrix of the estimator that rely on derivatives of the log likelihood function will also fail. However, derivative free methods[4] can be used to compute extremum estimators, to obtain point estimators

[3] See https://github.com/mcreel/SNM. The function which specifies and trains the neural net is https://github.com/mcreel/SNM/blob/master/src/MakeNeuralMoments.jl

[4] Simulated annealing (Goffe, Ferrier, and Rogers (1994)) is used in what follows.

or to initialize a MCMC chain, and the simulation-based estimator of the covariance matrix $\bar{\Sigma}(\theta)$ of eqn. 1 discussed in the previous section does not depend on derivatives. A major motivation of using Laplace-type estimators in the first place is to overcome problems of local extrema, as Chernozhukov and Hong (2003) emphasize. It is worth noting that the output of the net evaluated at the real sample statistic, $\hat{\theta}$, will also provide an excellent starting value for computing extremum estimators, or for initializing a MCMC chain. Likewise, the covariance estimator of eqn. 6 can be used to define an very effective random walk multivariate normal proposal density for MCMC, by drawing the trial value θ^{s+1} from $N(\theta^s, \hat{\bar{V}})$, where θ^s is the current value of the chain.

Creel (2017) used neural moments to compute a Laplace-type estimator, similarly to what is done here. That paper used nonparametric regression quantiles applied to the set of draws from the Laplace-type posterior draws in order to compute confidence intervals, and the posterior draws were generated by a procedure similar to sequential Monte Carlo, rather than MCMC. Also, the metric used for selection of particles was different from the GMM criterion, which is what is used here. The use of nonparametric regression quantiles is very costly to study by Monte Carlo. Thus, this paper focuses on straightforward use of the methods that Chernozhukov and Hong (2003) focus on: traditional MCMC using the GMM criterion function, and confidence intervals are computed using the direct quantiles from the posterior sample. These simplifications give a simpler and more tractable procedure that can reasonably be studied and verified by Monte Carlo.

2.3 Examples

This section presents five simple example models that are used to investigate the performance of the proposed methods. For all models, the code used (for the Julia language[5]) is available in an archive[6], where the details of each example may be consulted. The example models also serve as templates that may be used to apply to proposed methods to models of the reader's interest: one simply needs to provide similar functions to what is found in the directory for each example, for the model of interest. These are, fundamentally, 1) a prior from which to draw the parameters; 2) code to simulate the model given the parameter value, and finally, 3) code to compute the initial statistics, W, given the data generated from the model.

[5] https://julialang.org/

[6] https://github.com/mcreel/SNM

2.3.1 Stochastic Volatility

The simple stochastic volatility (SV) model is

$$\begin{aligned} y_t &= \phi \exp(h_t/2)\epsilon_t \\ h_t &= \rho h_{t-1} + \sigma u_t \end{aligned}$$

where ϵ_t and u_t are independent standard normal random variables. We use a sample size of 500 observations, and the true parameter values are $\theta_0 = (\phi_0, \rho_0, \sigma_0) = (0.692,\ 0.9,\ 0.363)$. These parameter values have been chosen to facilitate comparison with results of a number of previous studies that have used the SV model to check properties of estimators. For estimation, 11 statistics are used to form the initial set, W, which include moments of y and of $|y|$, as well as the estimated parameters of a HAR auxiliary model (Corsi (2009)) fit to $|y|$.[7]

2.3.2 Dynamic Panel Data

The dynamic panel data (DPD) model is borrowed from Forneron and Ng (2018), who adapted the model of Christian Gouriéroux, Phillips, and Yu (2010). The model is

$$y_{it} = \alpha_i + \rho y_{it-1} + \beta x_{it} + \sigma\epsilon_{it}$$

where α_i, x_{it}, and ϵ_{it} are all mutually independent standard normal random variables, for $i = 1, 2, ..., n$, and $t = 1, 2, ..., T$. We set $T = 5$ and $n = 100$. For initialization, y_{i0} is generated as a draw from it's unconditional distribution, for each i. We estimate the parameter vector $\theta_0 = (\rho_0, \beta_0, \sigma_0^2)$, where the true values are 0.6, 1.0 and 2.0, respectively. This is the same design as is used by Forneron and Ng (2018) in their Table 3, to facilitate comparison. Eight statistics are included in the initial set, W, for estimation of the three parameters. The first four are the three ordinary least squares estimates of the regression $y_{it} = \phi_1 + \phi_2 y_{it-1} + \phi_3 x_{it} + \eta_{it}$, which simply ignores the panel data structure, along with the estimated variance of the error term. The next four statistics are the fixed effects estimator, obtained by subtracting cross sectional means from all variables.[8]

2.3.3 ARMA

The next example is a simple ARMA(1,1) model

$$\begin{aligned} x_t &= \alpha x_{t-1} + f_t - \beta f_{t-1} \\ f_t &\sim IIN(0, \sigma^2), \end{aligned}$$

[7] See the file https://github.com/mcreel/SNM/blob/master/examples/SV/SVlib.jl for details.

[8] Details are in the file https://github.com/mcreel/SNM/blob/master/examples/DPD/DPDlib.jl.

with true values $\theta_0 = (\alpha_0, \beta_0, \sigma_0^2) = (0.95,\ 0.5,\ 1.0)$. The sample size is $n = 300$. The 13 statistics used to define the initial set, W, include sample moments and correlations, OLS estimates of an AR(1) auxiliary model fit to x_t, as well as an another AR(1) model fit to the residuals of the first model, plus partial autocorrelations of x_t[9]

2.3.4 Mixture of Normals

The final example model is a mixture of normals (MN). The variable y is drawn from the distribution $N(\mu_1, \sigma_1^2)$ with probability p and from $N(\mu_1 - \mu_2, \sigma_1^2 + \sigma_2^2)$ with probability $1 - p$. Samples of 1000 observations are drawn. The true parameter values are $\theta_0 = (\mu_1, \sigma_1, \mu_2, \sigma_2, p) = (1.0, 1.0, 0.2, 1.8, 0.4)$, and the prior restricts all parameters to be positive. Thus, the parameterization and the prior together impose that the first component has a larger mean and a lower variance than does the second component, in order to ensure identification. Also, the probability that either component is sampled is restricted to be at least 0.05. The 15 auxiliary statistics are the sample mean, standard deviation, skewness, kurtosis,, and 11 quantiles of y.[10]

2.3.5 Auction Model

Li (2010) proposes to use II for estimation of structural econometric models, and illustrates with a Monte Carlo example of estimation of the parameters of a Dutch auction, where only the winning bid is observed. We use the the same data generating process as in Li's paper, for comparability. In particular, we observe a sample of n i.i.d. auctions. At each auction $i = 1, 2, ..., n$, the quality x_i of the item being auctioned is observed as $x_i = 4u_i^2$, where u_i follows a uniform $(0, 1)$ distribution. Given this signal, N agents make a bid based on their private value of the item. Their privates values are mutually independent and come from a common exponential distribution with mean $\exp(\theta_0 + \theta_1 x_i)$. The equilibrium strategy for the winning bid is then $b_i^* = v_i^* - \int_0^{v_i^*} F^{N-1}(u|x_i)du / F^{N-1}(v_i^*|x_i)$ where v_i^* is the highest private valuation, and $F(\cdot|x_i)$ is the exponential distribution function. For a given value of N, symbolic computation software can be used to obtain an analytic solution for the winning bid, so simulations can be generated very quickly. The observed data are the n values of $\{x_i,\ b_i^*\}$, and we seek to estimate θ_0 and θ_1. The Monte Carlo results are for the following design: $n = 100$, $N = 6$, and the true parameter values to $\theta_0 = 1.0$ and $\theta_1 = 0.5$. The auxiliary statistics are the regression coefficients and estimated standard deviation of the error of the auxiliary model $\log b_i^* = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \epsilon_i$, plus the mean of winning bids and the standard deviation, skewness and kurtosis of the residuals of the auxiliary regression. Thus, we have seven statistics to identify the two parameters[11]

[9] Details are in the file https://github.com/mcreel/SNM/blob/master/examples/ARMA/ARMAlib.jl.

[10] Details are in the file https://github.com/mcreel/SNM/blob/master/examples/MN/MNlib.jl.

[11] Details are in the file https://github.com/mcreel/SNM/blob/master/examples/Auction/Auctionlib.jl.

2.4 Monte Carlo Results

This section reports results for Laplace type MCMC estimation of each of the test models, using the GMM-like criterion function H (see eqn. 4) as the L_n of Chernozhukov and Hong (2003). Results using the criterion L (see eqn. 3) are qualitatively very similar, and are thus not reported. For three of the test models (SV, DPD and ARMA) the covariance matrix $\bar{V}(\theta)$ was estimated using eqn. 6 in a continuous updating fashion, at each parameter trial. For the MN and Auction models, the traditional two-step procedure was used. Concretely, the parameter vector θ was consistently estimated using $\arg\min H(\theta)$, where $\hat{\bar{V}}(\theta)$ in eqn. 4 was set to an identity matrix, using simulated annealing to do the minimization. Then, the resulting estimate $\hat{\theta}$ was used to compute $\hat{\bar{V}}(\hat{\theta})$ using eqn. 6, and this was kept fixed for the remainder of the MCMC computations. This makes the MCMC iterations faster, as the simulation-based update of the covariance estimate at each MCMC step is avoided. For all of the test models, the number of artificial samples used to train the neural net was 20,000 times the number of parameters of the model. This is actually a fairly small number, given that generating the samples and training the nets is an operation that takes only 10 minutes or less for the test models. The reason that a larger number of samples was not used is that it was desired to obtain results that may be more relevant for cases where it is more costly to simulate from the model, as is the case of the jump diffusion model studied in the next section.

For the SV, DPD and ARMA models, two versions are reported: first, using the initial statistics, W, and, second, using the statistics Z which are the output of the trained neural net. For the MN and Auction models, the MCMC estimators using W were not computed, as this is quite time consuming when the dimension of the statistics is large, and, from the previous models, it was already clear that using the neural statistics dominated using the full vector of statistics. Results include root mean squared error (RMSE) and bias for the extremum estimator which minimizes H, as this was found to have slightly better (but qualitatively very similar) performance than the posterior mean or median of the tuned MCMC chain, and coverage of 90, 95 and 99% confidence intervals computed using the appropriate quantiles of the final tuned MCMC chain, following Chernozhukov and Hong (2003). For the SV, DPD and ARMA models, 500 Monte Carlo replications were used. For the MN and Auction models, 1000 replications were used, as, for these models, the procedures were less costly due to the use of two step estimation and the omission of the computations using the full vector of statistics.

Table 1 reports RMSE for the test models. RMSE is in most cases not dramatically different between estimators that use the full set of statistics and those based on the neural net filtered statistics, and there is no clear pattern in the differences that do exist. In six of nine cases where the comparison may be made, the neural net filtered statistics lead to lower

RMSE, but in most instances, the difference is fairly small. There is no clear reason to prefer one version based on RMSE. Table 2 reports bias. In all cases, bias is low, relative to the true parameter values, and there is no clear pattern of differences between estimators based on W and those based on Z. In some cases, the results of these two tables may be compared with findings of other studies. For the SV model, Creel and Kristensen (2012), Table 3, collects results from a number of studies that report Monte Carlo results for the SV model. The RMSE and bias results reported here are very competitive with what is summarized there. The DPD results may be compared with Forneron and Ng (2018), Table 3. The present results are overall very similar to those they report, except that the bias in the estimation of the σ^2 parameter is considerably smaller using the methods proposed in this paper. The ARMA results may be compared with the middle panel of Table 1 of Fiorentini, Galesi, and Sentana (2018). The procedures used here lead to somewhat less bias, most notably for the β parameter, though the differences are not great. Finally, for the Auction model, bias and RMSE are improved compared to the results reported in Li (2010), Tables 1 and 2, especially for the θ_0 parameter. It should be noted that the objectives of the referenced studies did not necessarily include the identification of statistics (the W of this paper) that lead to the lowest RMSE or bias possible for a given method, but, rather to compare methods using a given set of statistics. This paper, in general, uses different raw statistics than do the cited studies, to highlight the fact that neural statistics can combine the information contained in a larger group of raw statistics to lead to a relatively efficient estimator. The reason for making these comparisons is simply to indicate that the methods proposed in this paper lead to results that are competitive, in terms of low bias and efficiency, with what is in the literature.

The main focus of this paper is, however, the reliability of inferences. Results for confidence interval coverage are presented in Tables 3, 4 and 5, for 90, 95 and 99 percent intervals, respectively. Monte Carlo confidence interval coverage is the proportion of times that the true parameter value is not rejected, at the chosen confidence level. Before interpreting results, we remind the reader that R Monte Carlo replications were done, where $R = 500$ for the SV, DPD and ARMA models, and $R = 1000$ for the MN and Auction models. Critical values for the hypothesis that a $100 \cdot p\%$ confidence interval has correct coverage can be found using the quantiles of a binomial(R,p) random variable. In Tables 3-5, cases where correct coverage is rejected at the 1% significance level are indicated by italic typeface. Here, we see some important differences. Coverage is poor for estimators that directly use the full set of moments, W, for the SV and ARMA models, where correct coverage is rejected in all cases except for one of 18 (the 90% interval for the ϕ parameter of the SV model). For the DPD model, the estimators based on W and on Z both perform well, as correct coverage is never rejected. For the five test models, when the neural net statistics are used, correct coverage is rejected in 10 of 48 cases. Eight of these cases correspond to the MN test model. When correct coverage is rejected, there are no cases where the departure from correct coverage

Table 1: RMSE of $\arg\min H$, using raw (W) or neural net (Z) statistics

Model	Parameter	true value	W	Z
SV	ϕ	0.692	0.123	0.075
	ρ	0.90	0.086	0.072
	σ	0.363	0.138	0.133
DPD	ρ	0.6	0.035	0.034
	β	1.0	0.075	0.067
	σ^2	2.0	0.138	0.151
ARMA	α	0.95	0.030	0.046
	β	0.5	0.078	0.081
	σ^2	1.0	0.099	0.086
MN	μ_1	1.0	na	0.029
	σ_1	0.2	na	0.114
	μ_2	0.0	na	0.029
	σ_2	2.0	na	0.073
	p	0.4	na	0.034
Auction	θ_1	1.0	na	0.031
	θ_2	0.5	na	0.022

is large, and in all cases, confidence intervals contain the true parameter more often that what corresponds to the nominal level, so the error is one of conservatism, with Type I error occurring less frequently than what would be correct[12].

2.5 Application: A Jump Diffusion Model of S&P 500 Returns

The previous examples are all small models that are not costly to simulate. As an example of a more computationally challenging model, this section presents results for estimation of a jump diffusion model of S&P 500 returns. Solving and simulating[13] the model for each MCMC trial parameter acceptance/rejection decision takes about 13 seconds, so training a net and estimation by MCMC is somewhat costly, requiring about 2 days to complete using a moderate power workstation and threads-based parallelization, where possible. This example is intended to show that the methods are feasible for research projects where simulation from the model is costly, but not extremely so.

The jump diffusion model is

[12] The results for the MN model can be improved if a larger set of artificial data is used to train the neural net. When the training/testing size was increased 10 fold, the number of rejections of correct coverage reduced to 5 out of 15, instead of 8 out of 15, as reported in the tables.

[13] The model is solved and simulated using the SRIW1 strong order 1.5 solver from the DifferentialEquations.jl package for the Julia language.

Table 2: Bias of $\arg\min H$, using raw (W) or neural net (Z) statistics

Model	Parameter	true value	W	Z
SV	ϕ	0.692	0.011	-0.018
	ρ	0.90	0.001	0.003
	σ	0.363	0.012	-0.005
DPD	ρ	0.6	-0.001	0.002
	β	1.0	0.000	0.000
	σ^2	2.0	0.010	0.003
ARMA	α	0.95	-0.014	-0.015
	β	0.5	0.008	0.001
	σ^2	1.0	0.004	-0.006
MN	μ_1	1.0	na	0.008
	σ_1	0.2	na	0.034
	μ_2	0.0	na	0.006
	σ_2	2.0	na	-0.024
	p	0.4	na	0.005
Auction	θ_1	1.0	na	0.013
	θ_2	0.5	na	0.004

$$dp_t = \mu dt + \sqrt{\exp h_t} dW_{1t} + J_t dN_t$$
$$dh_t = \kappa(\alpha - h_t) + \sigma dW_{2t}$$

where p_t is log price, h_t is log volatility, J_t is jump size, and N_t is a Poisson process with jump intensity λ_0. W_{1t} and W_{2t} are two standard Brownian motions with correlation ρ. When a jump occurs, its size is $J_t = a\lambda_1\sqrt{\exp h_t}$, where a is 1 with probability 0.5 and -1 with probability 0.5. So, jump size depends on the current standard deviation, and jumps are positive or negative with equal probability. Log price, p_t, is simulated using 5 minute tics, and the observed log price adds a $N(0, \tau^2)$ measurement error to p_t. From this model, 1000 daily observations on returns, realized volatility (RV), and bipower variation (BV) are generated. Both RV and BV are informative about volatility , and, because BV is somewhat robust to jumps, while RV is not, the difference between the two can help to identify the frequency and size of jumps (Barndorff-Nielsen and Shephard (2002)). The model is simulated on a continuous 24 hour basis, and returns are computed using the change in daily log closing price, for trading days only. Overnight periods and weekends are simulated, but returns, RV and BV are recorded only at the close of trading days. In summary, the seven parameters are $\theta = (\mu, \kappa, \alpha, \sigma, \rho, \lambda_0, \lambda_1, \tau)$, and simulated data consists of 1000 daily observations on returns, RV and BV. The model studied here is quite similar to that studied in Creel and Kristensen (2015) and Creel (2017), except that the drift process is simplified to be constant,

Table 3: 90% confidence interval coverage using H to define the Laplace type estimator, using raw (W) or neural net (Z) statistics. Italic typeface indicates that correct coverage is rejected at the 1% level.

Model	Parameter	W	Z
SV	ϕ	0.876	*0.936*
	ρ	*0.732*	0.894
	σ	*0.762*	0.88
DPD	ρ	0.888	0.896
	β	0.900	0.914
	σ^2	0.908	0.898
ARMA	α	*0.786*	0.914
	β	*0.814*	*0.938*
	σ^2	*0.808*	0.908
MN	μ_1	na	*0.935*
	σ_1	na	0*.930*
	μ_2	na	*0.930*
	σ_2	na	*0.932*
	p	na	*0.947*
Auction	θ_1	na	0.915
	θ_2	na	0.921

and the jump process is modeled somewhat differently, with constant intensity, and with the magnitude of a jump depending on the current instantaneous volatility. These changes were motivated by the results of the previous papers, and by the better tractability of the present specification.

The raw statistics, W, which are used to train the net and to do estimation are a combination of coefficients from auxiliary regressions between the three observed variables, summary statistics, and functions of quantiles of the variables. It is possible to tune the choice of statistics through experimentation with artificially generated data, to ensure that the parameters are all well identified, and it is possible to check that statistics have a minimal importance, by examining the weights of the first layer of the neural net[14]. The details of the 25 statistics which are used are found in the file JDlib.jl (this same file also gives details of the priors, which are uniform over fairly broad supports, for all parameters). The importances of the statistics are seen in the figure ImportanceOfStatistics.svg.

The model was fit to S&P500 data[15] from 16 Dec. 2013 to 05 Dec. 2017, which is an interval of 1000 trading days, the same as was used to train the neural net. The data may be seen in Figure 1, where we observe typical volatility clusters and some jumps. For example, the Brexit drop of June, 2016 is clearly seen, and the more extreme spike in RV versus BV at

[14] See the file Importance.jl for how to do this, as well as discussion in Creel (2017).

[15] The data source is the Oxford-Man Institute's realized library, v. 0.2, https://realized.oxford-man.ox.ac.uk/images/oxfordmanrealizedvolatilityindices-0.2-final.zip

Table 4: 95% confidence interval coverage using H to define the Laplace type estimator, using raw (W) or neural net (Z) statistics. Italic typeface indicates that correct coverage is rejected at the 1% level.

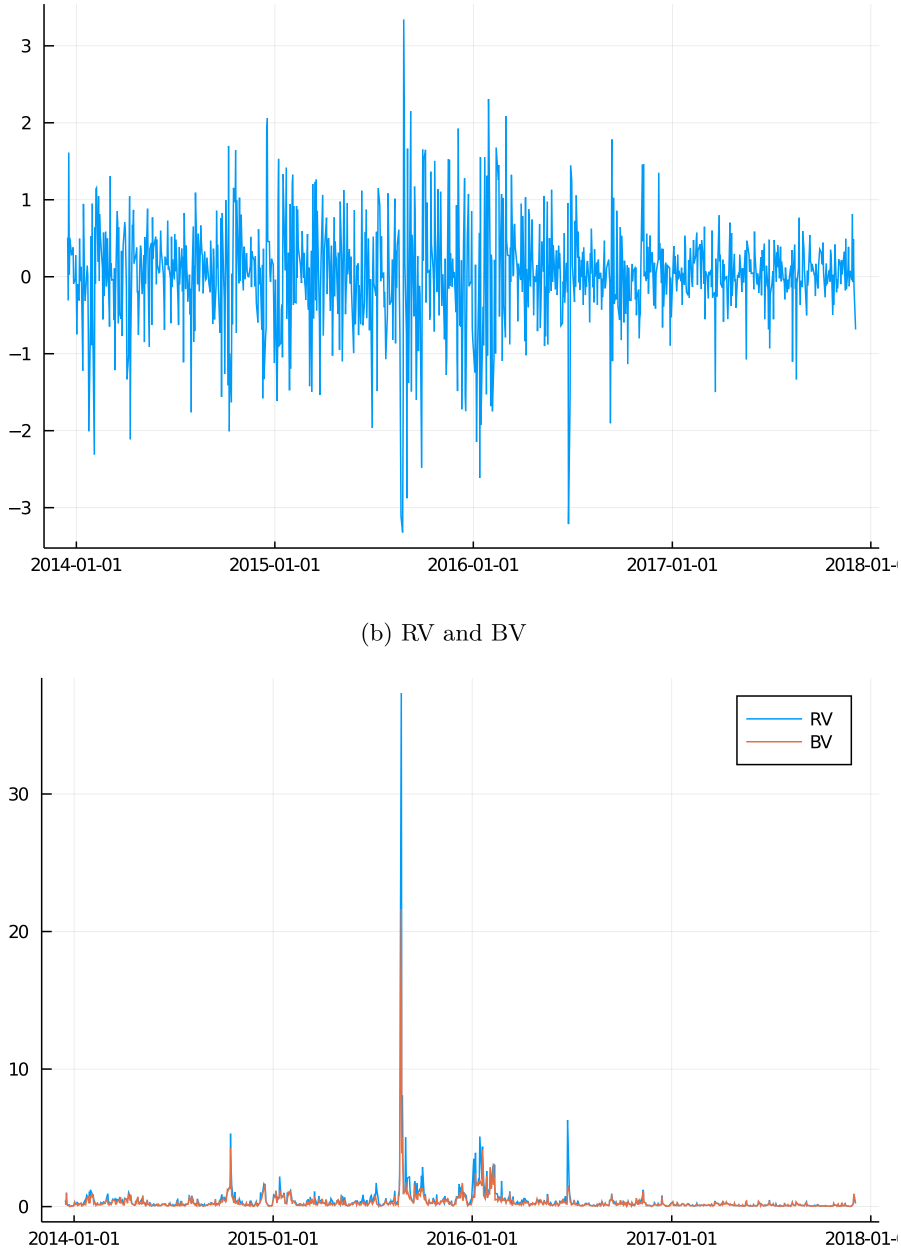
Model	Parameter	W	Z
SV	ϕ	*0.916*	0.966
	ρ	*0.796*	0.950
	σ	*0.824*	0.950
DPD	ρ	0.966	0.956
	β	0.966	0.960
	σ^2	0.954	0.946
ARMA	α	*0.838*	0.966
	β	*0.856*	0.966
	σ^2	*0.880*	0.954
MN	μ_1	na	0.963
	σ_1	na	0.962
	μ_2	na	*0.970*
	σ_2	na	*0.970*
	p	na	*0.977*
Auction	θ_1	na	0.962
	θ_2	na	0.960

this point illustrates the fact that jumps can be identified by comparing the two.

The estimation results are in Figure 2, which shows nonparametric plots of the marginal posterior density for each parameter, along with posterior means and medians, and 90% confidence intervals defined by the limits of the green areas. All posteriors are considerably more concentrated than are the priors. Drift (μ) is not significantly different from zero. There is quite a bit of persistence in volatility, as mean reversion, κ, is estimated to be between 0.09 and 0.13. Leverage (ρ) is quite strong, estimated between -0.85 and -0.65. The jump probability per day (λ_0) is estimated to be between 0.01 and 0.04, with the point estimate being approximately 0.02. So, jumps are a statistically important feature of the model. When a jump does occur, its magnitude (λ_1) is approximately 3.2 times the current instantaneous standard deviation. An interesting result is that the standard deviation of measurement error, τ, is estimated to be approximately 0.01. The hypothesis that this parameter is zero cannot be rejected at the 10% significance level, but most posterior probability is on positive values. Thus, it appears that it is a safer option to allow for measurement error in the model, as its omission could bias the estimates of the other parameters.

Creel (2017) uses similar methods to analyze the S&P 500 data over the Jan. 2015 - May 2016 interval. That paper found that mean volatility (α) was higher than the estimate here, the variance of volatility (σ) was lower than that found here, and that mean reversion (κ) was faster than that estimated here. These results are consistent with what is seen in Figure

Figure 1: Plot of returns, RV and BV, S&P 500, 16 Dec. 2013 - 05 Dec. 2017.



(a) Returns

(b) RV and BV

Table 5: 99% confidence interval coverage using H to define the Laplace type estimator, using raw (W) or neural net (Z) statistics. Italic typeface indicates that correct coverage is rejected at the 1% level.

Model	Parameter	W	Z
SV	ϕ	*0.936*	0.984
	ρ	*0.848*	0.988
	σ	*0.888*	0.984
DPD	ρ	0.996	0.994
	β	0.988	0.986
	σ^2	0.996	0.978
ARMA	α	*0.898*	0.998
	β	*0.916*	0.988
	σ^2	*0.920*	0.992
MN	μ_1	na	0.996
	σ_1	na	0.989
	μ_2	na	0.993
	σ_2	na	0.995
	p	na	0.996
Auction	θ_1	na	0.991
	θ_2	na	0.987

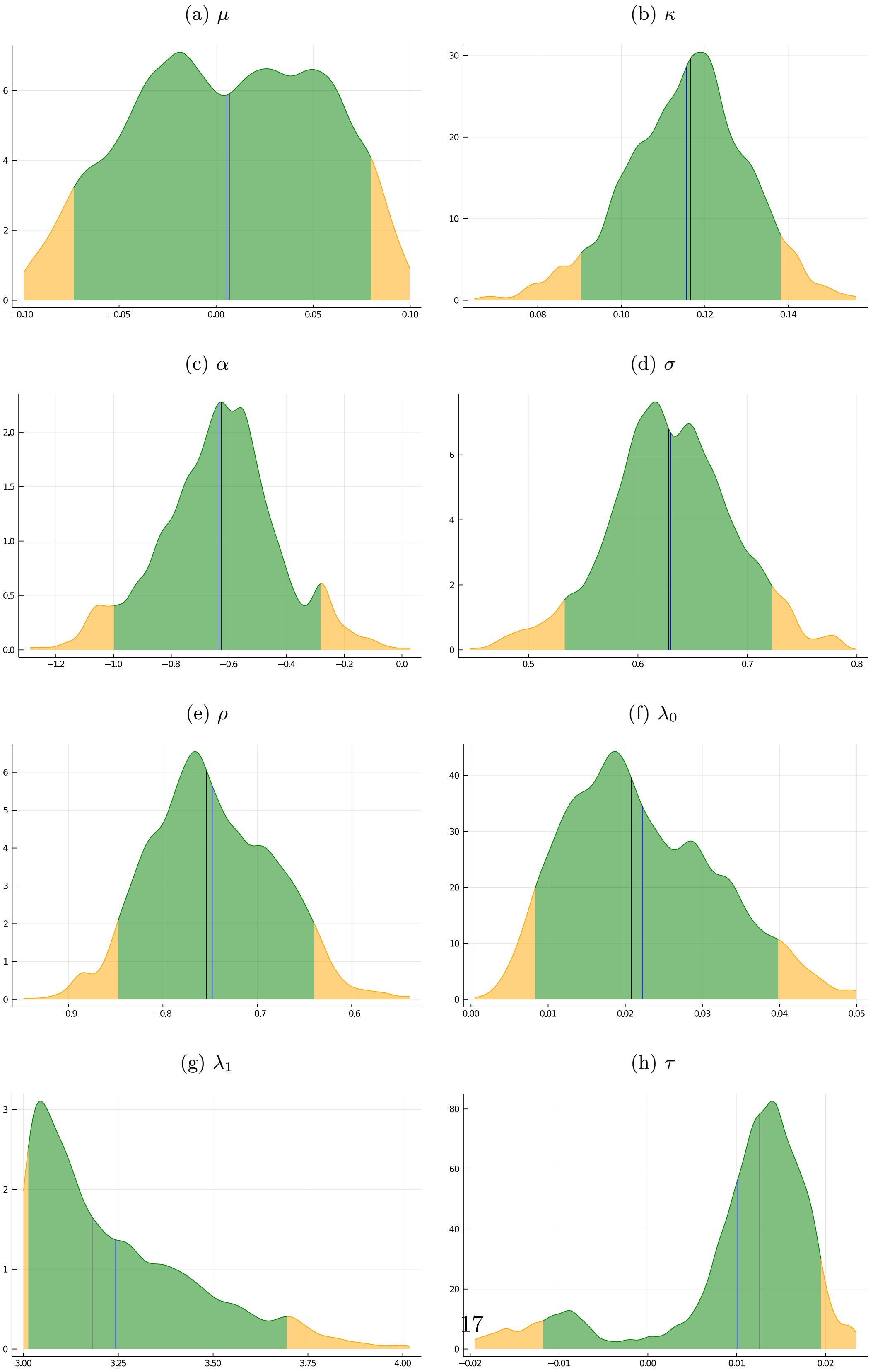
1. The Jan. 2015 - May 2016 interval is one of relatively high volatility, with less pronounced clusters. For this shorter data window, volatility is higher on average and less variable, with less clustering. The differences in the estimated parameters between the earlier paper and the results here are consistent with these facts.

3 Conclusions

This paper has shown, through Monte Carlo experimentation, that confidence intervals based upon quantiles of a tuned MCMC chain may have coverage which is far from the nominal level, even for simple models with few parameters. It has proposed to use neural networks to reduce the dimension of an initial set of moments to the minimum number of moments needed to maintain identification. When estimation and inference using well-known MCMC methods and the Laplace version of GMM is based on neural moments, confidence intervals have statistically correct coverage in most cases studied by Monte Carlo, and departures from correct coverage are small. The methods have been illustrated by the estimation of a jump diffusion model for S&P 500 data.

It is to be noted that the step of filtering moments though a neural net is very easy and quick to perform using modern deep learning software environments. The software archive that accompanies this paper provides a function for automatic training, requiring no human

Figure 2: MCMC results for the jump-diffusion model of S&P 500 data. Posterior mean in blue, posterior median in black. The green-yellow borders define the limits of a 90% confidence interval.



(a) μ

(b) κ

(c) α

(d) σ

(e) ρ

(f) λ_0

(g) λ_1

(h) τ

intervention. It only requires functions that provide simulated moments computed using data drawn from the model at parameter values drawn from the prior. Filtering moments through a neural net gives an informative, minimal dimension statistic as the output. This provides a convenient and automatic alternative to moment selection procedures. Uninformative moments are essentially removed, and correlated moments are combined.

This paper has examined how inference using quantiles of traditional MCMC chains may be improved when neural moments are used. It seems likely that other inference methods which are used with simulation-based estimators, such as Hamiltonian Monte Carlo and sequential Monte Carlo, among others, may be made more reliable if neural moments are used, as dimension reduction while maintaining relevant information is likely to be generally beneficial.

Acknowledgements

Support from the Spanish Ministry of Science, Innovation and Universities and FEDER through a Severo Ochoa Center of Excellence award to the Barcelona GSE and through grant PGC2018-094364-B-I00 is gratefully acknowledged.

References

Barndorff-Nielsen, Ole E and Neil Shephard (2002). "Econometric analysis of realized volatility and its use in estimating stochastic volatility models". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 64.2, pp. 253–280.

Chernozhukov, Victor and Han Hong (Aug. 2003). "An MCMC approach to classical estimation". In: *Journal of Econometrics* 115.2, pp. 293–346. DOI: 10.1016/s0304-4076(03)00100-3. URL: http://dx.doi.org/10.1016/s0304-4076(03)00100-3.

Christiano, Lawrence J, Martin S Eichenbaum, and Mathias Trabandt (2016). "Unemployment and business cycles". In: *Econometrica* 84.4, pp. 1523–1569.

Christiano, Lawrence J, Mathias Trabandt, and Karl Walentin (2010). "DSGE models for monetary policy analysis". In: *Handbook of Monetary Economics*. Vol. 3. Elsevier, pp. 285–367.

Corsi, Fulvio (2009). "A simple approximate long-memory model of realized volatility". In: *Journal of Financial Econometrics* 7.2, pp. 174–196.

Creel, Michael (2017). "Neural nets for indirect inference". In: *Econometrics and Statistics* 2, pp. 36–49. URL: https://doi.org/10.1016/j.ecosta.2016.11.008.

Creel, Michael and Dennis Kristensen (2012). "Estimation of dynamic latent variable models using simulated non-parametric moments". In: *The Econometrics Journal* 15.3, pp. 490–515. URL: https://doi.org/10.1111/j.1368-423X.2012.00387.x.

— (2015). "ABC of SV: Limited information likelihood inference in stochastic volatility jump-diffusion models". In: *Journal of Empirical Finance* 31, pp. 85–108. ISSN: 0927-5398. DOI: http://dx.doi.org/10.1016/j.jempfin.2015.01.002. URL: http://www.sciencedirect.com/science/article/pii/S0927539815000031.

Donald, Stephen G., Guido W. Imbens, and Whitney K. Newey (2009). "Choosing instrumental variables in conditional moment restriction models". In: *Journal of Econometrics* 152.1. Recent Adavances in Nonparametric and Semiparametric Econometrics: A Volume Honouring Peter M. Robinson, pp. 28–36. ISSN: 0304-4076. DOI: 10.1016/j.jeconom.2008.10.013. URL: http://www.sciencedirect.com/science/article/pii/S0304407609000566.

Fiorentini, Gabriele, Alessandro Galesi, and Enrique Sentana (2018). "A spectral EM algorithm for dynamic factor models". In: *Journal of Econometrics* 205.1, pp. 249–279.

Forneron, Jean-Jacques and Serena Ng (2018). "The ABC of simulation estimation with auxiliary statistics". In: *Journal of Econometrics* 205.1, pp. 112–139.

Gallant, A. Ronald and George Tauchen (1996). "Which moments to match?" In: *Econometric Theory* 12, pp. 363–390. DOI: 10.2139/ssrn.37760. URL: http://dx.doi.org/10.2139/ssrn.37760.

Gallant, A. Ronald and George Tauchen (2002). “EMM: A program for efficient method of moments estimation, Version 1.6, User’s Guide”. In: *Manuscript, University of North Carolina.*

Goffe, William L, Gary D Ferrier, and John Rogers (1994). “Global optimization of statistical functions with simulated annealing”. In: *Journal of Econometrics* 60.1-2, pp. 65–99.

Gouriéroux, C., A. Monfort, and E. Renault (1993). “Indirect inference”. In: *Journal of Applied Econometrics*, S85–S118. DOI: 10.1002/jae.3950080507. URL: http://dx.doi.org/10.1002/jae.3950080507.

Gouriéroux, Christian, Peter CB Phillips, and Jun Yu (2010). “Indirect inference for dynamic panel models”. In: *Journal of Econometrics* 157.1, pp. 68–77.

Hall, Peter and Joel L Horowitz (1996). “Bootstrap critical values for tests based on generalized-method-of-moments estimators”. In: *Econometrica*, pp. 891–916.

Hansen, Lars Peter, John Heaton, and Amir Yaron (July 1996). “Finite-sample properties of some alternative GMM estimators”. In: *Journal of Business & Economic Statistics* 14.3, pp. 262–280. DOI: 10.1080/07350015.1996.10524656. URL: http://dx.doi.org/10.1080/07350015.1996.10524656.

Jiang, Wenxin and Bruce Turnbull (2004). “The indirect method: inference based on intermediate statistics a synthesis and examples”. In: *Statistical Science* 19.2, pp. 239–263.

Li, Tong (2010). “Indirect inference in structural econometric models”. In: *Journal of Econometrics* 157.1, pp. 120–128.

Marjoram, Paul et al. (2003). “Markov chain Monte Carlo without likelihoods”. In: *Proceedings of the National Academy of Sciences* 100.26, pp. 15324–15328.

McFadden, Daniel (1989). “A method of simulated moments for estimation of discrete response models without numerical integration”. In: *Econometrica* 57.5, pp. 995–1026. ISSN: 00129682, 14680262. URL: http://www.jstor.org/stable/1913621.

Smith, Anthony A (1993). “Estimating nonlinear time-series models using simulated vector autoregressions”. In: *Journal of Applied Econometrics* 8.S1.

Tauchen, George (Oct. 1986). “Statistical properties of generalized method-of-moments estimators of structural parameters obtained from financial market data”. In: *Journal of Business & Economic Statistics* 4.4, p. 397. DOI: 10.2307/1391493. URL: http://dx.doi.org/10.2307/1391493.