Automated Sensitivity Analysis for Bayesian Inference via Markov Chain Monte Carlo: Applications to Gibbs Sampling

LIANA JACOBI
Department of Economics, University of Melbourne,
MARK JOSHI † [1]
Department of Economics, University of Melbourne,
DAN ZHU
Department of Business Statistics and Econometrics, Monash University

February 9, 2018

Summary

Bayesian inference relies heavily on numerical Markov chain Monte carlo (MCMC) methods for the estimation of the typically intractable high-dimensional posterior distributions and requires specific inputs. In this paper we introduce a new general and efficient numerical approach to address important robustness concerns of MCMC analysis with respect to prior input assumptions, a major obstacle to wider acceptance of Bayesian inference, including MCMC algorithm performance (convergence) reflected in the dependence on the chain starting values. The approach builds on recent developments in sensitivity analysis of high-dimensional numerical integrals for classical simulation methods using automatic numerical differentiation methods to compute first order derivatives of algorithmic output with respect to all inputs. We introduce a range of new robustness measures based on Jacobian matrices of MCMC output w.r.t. to the two sets of input parameters, prior parameters and chain starting values, to enable researchers to routinely undertake a comprehensive sensitivity analysis of their MCMC results. The methods are implemented for a range of Gibbs samplers and illustrated using both simulated and real data examples.

1 Introduction

Bayesian inference is an increasingly popular alternative approach to classical statistical inference that is based on the estimation of the posterior distribution of the model parameters. Since the posterior distribution involves high dimensional integrals, Bayesian analysis relies heavily on Markov Chain Monte Carlo (MCMC) methods to estimate the posterior distribution. While recent advances in computational power have made the Bayesian approach a feasible and attractive alternative tool to the classical estimation approach to analyze complex problems in many disciplines, including many areas in economics, it is often applied hesitantly. Two main reasons, aside from the computational complexity, are concerns regarding the robustness of

[1] Mark Joshi passed away October 8, 2017.

the results with respect to the prior assumptions that are required for all model parameters and concerns regarding algorithm performance, in particular convergence. In this paper we introduce the first general and efficient numerical approach to compute sensitivities of Bayesian MCMC ouptut with respect to different input parameters. The approach opens up the possibility for a comprehensive assessment of prior robustness and algorithm performance that is currently not possible.

Firstly, different to frequentist analysis, the Bayesian approach requires the specification of prior assumptions for all model parameters. This extra information introduces an element of subjectivity (Lopes and Tobias 2011) that has been long discussed in the literature preceding the MCMC revolution (Berger et al 1994). While the choice of the prior is now typically taken within a parametric family of distributions guided by computational concerns, researchers still have to specify a potentially very large set of these prior input parameters. The parameters of these prior distributions are known as hyperparameters and are often difficult to specify reliably (O'Hagan and Luce 2003).

The importance of a comprehensive local prior sensitivity analysis is often acknowledged. However, it is a non-trivial task given the complex form of the posterior distribution and, currently, it is not part of standard Bayesian analysis. If any hyperparameter sensitivity analysis is undertaken, it typically is done via a bumping-type approach by rerunning the model with a small set of different inputs and observing whether or how the outputs change. Other recent work, such as Chib and Jacobi (2009, 2015) use a scenario-based method in connection with a prior simulation (Del Negro and Schorfheide 2008, Chib and Ergashev 2009). The bumping approach is very costly, as it requires re-running the MCMC estimation many times. It is also restrictive as it only provides the effects of the particular bumps chosen and it is very unstable if the MCMC algorithm contains discontinuities, such as common updates from gamma random variables, where a very small change in inputs can cause a large change to the output.

Secondly, the convergence of the MCMC output to the limiting posterior distribution is key to obtaining reliable estimates of the model parameters. Several measures of non-convergence have been put forward, but knowing when a change has converged is "as much of an art as a science" Greenberg (2012). The estimated sensitivities with respect to the starting values provide crucial information for the assessment of the MCMC algorithm. As the chain evolves the dependence of the MCMC draws on the starting values should disappear since Markov chain theory implies that after a chain has converged draws no longer depend on the starting values. Current checks for convergence and related issue of algorithm efficiency applied in the literature include re-running the MCMC chain with different starting values and diagnostics based on the autcorrelation of the draws, such as the examination of trace plots of the draws. However, most models studied by Bayesians are computational intensive, re-running the samplers many times is too costly to assess the convergence of the chain.

A comprehensive sensitivity analysis based on first order derivatives of MCMC output with respect to the hyper-parameters and starting values can provide essential information regarding both robustness concerns. In this paper we develop methods and show how such a comprehensive robustness analysis can be implemented, for example to obtain the Jacobian of posterior estimate, including the posterior mean and variance of the model parameters, with respect to all input parameters, and the Jacobian of parameters draws with respect to the vector

of starting values at each iteration of the chain.

In order to compute these sensitivities for MCMC output arising from the high-dimensional integral of the posterior distribution, we exploit the Automatic Differentiation approach and extend it to the setting of dependent sampling. The basic idea consists of using automatic algorithms to differentiate the *algorithm* used to compute the integral, in our case the MCMC algorithm to evaluate the integral to obtain the posterior distribution. In the case where the algorithm is continuous, this approach can be viewed as the small limit of finite differencing approximations. It is therefore often called *infinitesimal perturbation analysis (IPA)* or the *pathwise method*. The differentiation can be carried out in many ways and was originally done analytically for derivatives pricing problems. When an automatic algorithm is used for the differentiation due to the complexity of the computation, we shall refer to it as the automatic pathwise method or Automatic Differentiation (AD).

AD is quite distinct from both numerical finite differencing and symbolic differentiation. In particular, finite differencing gives a numerical approximation that has a bias depending on the bump size whereas AD is exact up to floating point error. Symbolic differentiation is exact, but requires the value to be differentiated to be input as a formula, rather than as an algorithm, and it outputs a formula that then requires evaluation separately, and so does not use the computation already performed to achieve acceleration whereas AD does.

Similar to the work on numerical sensitivity analysis for classical simulation output, we apply automatic differentiation as suggested by Giles and Glasserman (2006) to compute the Jacobian matrices of parameter draws and statistics based on these draws with respect to the two sets of prior inputs, hyperparameters and starting values. We address a number of challenges that arise when applying AD techniques to MCMC algorithms. Firstly, automatic differentiation methods can be memory intensive. MCMC algorithms already generate a very large number of outputs which prohibits the use of some known AD methods. In addition, MCMC samplers generate samples of dependent draws in contrast to the independent draws of classical simulation methods. This further increases the computational demands to obtain the gradients as we cannot exploit parallel computation. Further, Bayesian MCMC analysis relies on a set of specific random variables that are not common in classical simulation. Finally, it is necessary to assess which AD techniques are applicable and likely to be most effective numerically in the context of Bayesian MCMC analysis as any numerical technique is an approximation and some approximations are faster to converge to the true value than others.

The AD approach enables us to introduce a range of sensitivity measures based on Jacobian matrices that summarize robustness of draws and statistics, such as the posterior mean, with respect to the different input parameters to allow researchers to assess robustness of results to prior inputs as well as the convergence behaviour of the MCMC chain. We implement AD-based robustness approach for MCMC analysis for a range of models and Gibbs samplers. AD-based sensitivity analysis for samplers with Metropolis-Hastings updates is discussed in a separate paper as the inherent discontinuity of the accept-reject method it raises a different set of issues. To illustrate the new approach we apply the methods to assess the sensitivities in various simulated and real data examples. To benchmark our sensitivity estimates we compare our results with the sensitivities obtained via an alternative approach limited to sensitivities

with respect to prior hyper-parameters based on the likelihood ratio-based technique that was introduced for MCMC output in Perez et al (2006) and applied in Müller (2012) to compute a limited set of Jacobian matrices for the posterior parameter means with respect to prior means in the context of macroeconomic DSGE models. Our experiments show that sensitivity estimates obtained via the AD methods are more stable and faster to converge, a finding consistent with the large evidence from the performance of these methods in the context of classical simulation.

The remainder of the paper is organized as follows. Section 2 introduces the AD based robustness analysis for Bayesian inference via MCMC algorithms. Section 3 shows the implementation of the AD approach for a range of Gibbs samplers with commonly used Gibbs updates and in the context of data augmentation and discusses methods for discontinuous random variables. Section 4 compares the performance of the AD approach to the likelihood ratio approach for hyperparameter sensitivity for a range of Gibbs sampler using simulated data and Section 5 illustrates the new robustness measures in a real data examples. Section 6 concludes.

2 Robustness Analysis via Sensitivities of MCMC Ouptput

2.1 MCMC Inference

Consider the general setting where data Y has been generated by some data generation process that depends on a vector of parameters $\boldsymbol{\theta} \in \mathbb{R}^K$. Under the Bayesian approach, inference about $\boldsymbol{\theta}$ is based on the (joint) posterior distribution $\pi(\boldsymbol{\theta}|Y)$. The posterior distribution combines information from the data Y via the model likelihood $f(Y|\boldsymbol{\theta})$ and the prior distribution for all model parameters, $\pi(\boldsymbol{\theta}|\boldsymbol{\eta}_0)$. Here, we have the hyperparamters $\boldsymbol{\eta}_0 \in \mathbb{R}^p$. According to Bayes Theorem, the posterior distribution is derived from the expression

$$\pi(\boldsymbol{\theta}|Y, \boldsymbol{\eta}_0) = \frac{\pi(\boldsymbol{\theta}|\boldsymbol{\eta}_0) f(Y|\boldsymbol{\theta})}{\int f(Y|\boldsymbol{\theta})\pi(\boldsymbol{\theta}|\boldsymbol{\eta}_0) d\boldsymbol{\theta}} \tag{2.1}$$

which involves a K-dimensional integral. The posterior distribution will depend on the set of prior hyper-parameters which have to be specified without reference to the data used for the current analysis.

Since the posterior distribution is with a very few exceptions of an unknown form, the estimation relies on numerical MCMC simulation methods that have become feasible also for complex problems as a result of recent advances in computing power. The key idea behind Bayesian MCMC-based inference is the construction of a Markov Chain with a transition kernel that has the posterior distribution as its limiting distribution. While Bayesian MCMC methods employ some classical simulation techniques, they differ significantly from classical simulation methods as they generate dependent samples. Given a set of values $\boldsymbol{\theta}^0 \in \mathbb{R}^l$, where $l < K$, initializing the algorithm, the Markov-chain $\boldsymbol{\theta}^g$ evolves forward with a stationary transition kernel $p(\boldsymbol{\theta}^{g+1}|\boldsymbol{\theta}^g, \boldsymbol{\eta}_0, Y)$.

Once the Markov chain has converged it yields a sample of dependent draws from the posterior distribution. There is therefore a burn-in period of size B. The draws following this

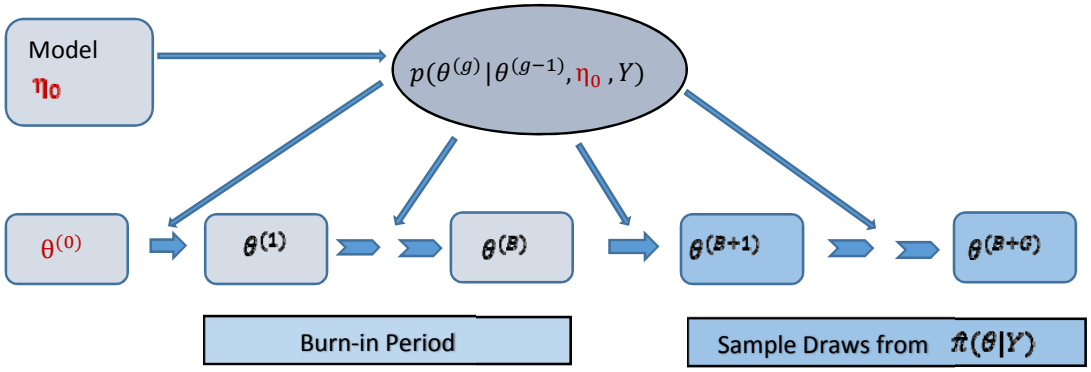


Figure 1: Algorithm based on MCMC chain.

burn-in period, $\{\boldsymbol{\theta}^{B+1}, \boldsymbol{\theta}^{B+2}, ..., \boldsymbol{\theta}^{B+G}\}$ form the basis for the computation of the reported estimation outputs (statistics), that include the parameter means, variances and quantiles,

$$\frac{1}{G}\sum_{g=B+1}^{B+G} S(\boldsymbol{\theta}^g) \xrightarrow{p} \mathbb{E}_{\hat{\pi}}[S(\boldsymbol{\theta})|Y, \boldsymbol{\theta}^0, \boldsymbol{\eta}_0] = \int S(\boldsymbol{\theta})\hat{\pi}(\boldsymbol{\theta}|Y, \boldsymbol{\theta}^0, \boldsymbol{\eta}_0)d\boldsymbol{\theta} \tag{2.2}$$

where the limit is as $G \to \infty$ and $\hat{\pi}(\boldsymbol{\theta}|Y, \boldsymbol{\theta}^0, \boldsymbol{\eta}_0)$ is the estimate of $\pi(\boldsymbol{\theta}|Y, \boldsymbol{\eta}_0)$ obtained from the MCMC algorithm for a fixed B. The MCMC sampler is constructed based on the model likelihood, the transition kernel and the prior parametric distribution.

From an algorithmic point of view, an MCMC algorithm is simply a function taking inputs (prior parameters, starting values) and returning a set of outputs (parameter draws, sample statistics). Sensitivities will inform us how the numerical output depends on the input of the sampler given a specific algorithm/transition kernel and will thus reveal key information about the prior robustness of the output and the performance of the chain in terms of convergence to the posterior distribution. In the next sections we introduce an approach to compute a complete set of sensitivities via the Jacobian matrices using Automatic Forward Differentiation.

2.2 Sensitivities via Infinitesimal Perturbation Analysis (Jacobians)

Without exhausting the notation, we shall denote $\boldsymbol{\theta}^g$ as both the gth iteration draw and the MCMC algorithm that computes it, i.e.

$$\boldsymbol{\theta}^g : \mathbb{R}^{p+l} \to \mathbb{R}^K$$

We can compute the effect of a small change in each element from the prior parameter vector $\eta_0^j \in \boldsymbol{\eta}_0$ on the gth draw of the parameter vector $\boldsymbol{\theta}$ by "bumping" η_0^j by a small value $h > 0$ and rerun the chain to obtain

$$\frac{\boldsymbol{\theta}^g(\boldsymbol{\eta}_0 + h\mathbb{I}^p(j), \boldsymbol{\theta}^0) - \boldsymbol{\theta}^g(\boldsymbol{\eta}_0, \boldsymbol{\theta}^0)}{h} \tag{2.3}$$

where $\mathbb{I}^p(j)$ is the p-dimensional indicator vector

$$\mathbb{I}^p_k(j) = \begin{cases} 1 \text{ if } j = k \\ 0 \text{ otherwise.} \end{cases} \qquad \text{for } k = 1, ..., p.$$

If we are interested in the sensitivity of a statistic such as the posterior mean we can compute it based on all draws after the burning period

$$\frac{1}{G} \sum_{g=B+1}^{B+G} \frac{\boldsymbol{\theta}^g(\eta_0 + h\mathbb{I}^p(j), \boldsymbol{\theta}^0) - \boldsymbol{\theta}^g(\boldsymbol{\eta}_0, \boldsymbol{\theta}^0)}{h} \tag{2.4}$$

Similarly, sensitivities with respect to the starting values can be computed by bumping the starting values $\boldsymbol{\theta}^0$ and rerunning the MCMC chain. This approach to assess sensitivities is computationally very demanding as it involves rerunning the chain for every single parameter that we are interested in. Hence, it is then only used in practice by researchers for key statistics to compute the effects of a discrete change a small subset of elements in $\boldsymbol{\eta}_0$ and/or $\boldsymbol{\theta}^0$ to gage robustness with respect to hyperparamters and starting values.

In addition to its computational demand, this method is constrained by the choice of the bump size. Theoretically, we are interested in the case of letting $h \to 0$, to obtain the Jacobian matrix

$$J_{\boldsymbol{\theta}^g}(\boldsymbol{\eta}_0, \boldsymbol{\theta}^0) \in \mathbb{R}^{K \times (p+l)}$$

where the jth column for $1 \leq j \leq p$ is the first-order partial derivatives for $\boldsymbol{\theta}^g$ with respect to η_0^j

$$\frac{\partial \boldsymbol{\theta}^g(\boldsymbol{\eta}_0, \boldsymbol{\theta}^0)}{\partial \eta_0^j} = \lim_{h \to 0} \frac{\boldsymbol{\theta}^g(\boldsymbol{\eta}_0 + h\mathbb{I}^p(j), \boldsymbol{\theta}^0) - \boldsymbol{\theta}^g(\boldsymbol{\eta}_0, \boldsymbol{\theta}^0)}{h}. \tag{2.5}$$

From the Jacobian matrices of the draws $\{J_{\boldsymbol{\theta}^g}(\boldsymbol{\eta}_0, \boldsymbol{\theta}^0) : g = B + 1, ..., B + G\}$ we can obtain an estimate of the Jacobian matrix for the posterior mean of $\boldsymbol{\theta}$, $\mathbb{E}_{\hat{\pi}}[\boldsymbol{\theta}|Y]$, as

$$\frac{1}{G} \sum_{g=B+1}^{B+G} J_{\boldsymbol{\theta}^g}(\boldsymbol{\eta}_0, \boldsymbol{\theta}^0) \xrightarrow{p} J_{\mathbb{E}_{\hat{\pi}}[\boldsymbol{\theta}|Y, \boldsymbol{\theta}^0, \boldsymbol{\eta}_0]}. \tag{2.6}$$

as $G \to \infty$ by the law of large numbers. More generally, applying the chain rule, an estimate of the Jacobian matrix of any function S of the parameters can be obtained from

$$\frac{1}{G} \sum_{g=B+1}^{B+G} J_S(\boldsymbol{\theta}^g) J_{\boldsymbol{\theta}^g}(\boldsymbol{\eta}_0, \boldsymbol{\theta}^0) \xrightarrow{p} J_{\mathbb{E}_{\hat{\pi}}[S(\boldsymbol{\theta})|Y, \boldsymbol{\theta}^0, \boldsymbol{\eta}_0]} \tag{2.7}$$

where $J_S(\boldsymbol{\theta}^g)$ is the Jacobian of the statistic function S evaluated as $\boldsymbol{\theta}^g$. These Jacobian matrices that does not rely on bumping the parameters can be used for a comprehensive robustness analysis.

The AD method allows us to efficiently compute

$$J_{\boldsymbol{\theta}^g}(\boldsymbol{\eta}_0, \boldsymbol{\theta}^0) = \begin{pmatrix} \frac{\partial \boldsymbol{\theta}_1^g}{\partial \boldsymbol{\eta}_{0,1}} & \cdots & \frac{\partial \boldsymbol{\theta}_1^g}{\partial \boldsymbol{\eta}_{0,p}} & \frac{\partial \boldsymbol{\theta}_1^g}{\partial \boldsymbol{\theta}_1^0} & \cdots & \frac{\partial \boldsymbol{\theta}_1^g}{\partial \boldsymbol{\theta}_l^0} \\ \cdots & & & & & \\ \cdots & & & & & \\ \frac{\partial \boldsymbol{\theta}_K^g}{\partial \boldsymbol{\eta}_{0,1}} & \cdots & \frac{\partial \boldsymbol{\theta}_K^g}{\partial \boldsymbol{\eta}_{0,p}} & \frac{\partial \boldsymbol{\theta}_K^g}{\partial \boldsymbol{\theta}_K^0} & \cdots & \frac{\partial \boldsymbol{\theta}_K^g}{\partial \boldsymbol{\theta}_l^0} \end{pmatrix}$$

the Jacobian of each single parameter draw with respect to the hyper-parameters in the first p columns and with respect to the starting value in the last l columns as shown in the above matrix. While some of these sensitivities are of direct interest in Bayesian sensitivity analysis, most of these sensitivities are used to compute the sensitivities of the summary statistics of the posterior based on the set of draws after the Burn-in period. The sensitivity computations/algorithms we propose will therefore encompass the computation of Jacobian matrices for both the parameter draws and statistics of the parameters, such as the posterior mean.

$$J_{\hat{S}(\boldsymbol{\theta})}(\boldsymbol{\eta}_0, \boldsymbol{\theta}^0) = \frac{1}{G} \sum_{g=B+1}^{B+G} \begin{pmatrix} \frac{\partial \mathbb{S}_1(\boldsymbol{\theta}^g)}{\partial \boldsymbol{\theta}_1^g} & \cdots & \frac{\partial \mathbb{S}_1(\boldsymbol{\theta}^g)}{\partial \boldsymbol{\theta}_K^g} \\ \cdots & & \\ \frac{\partial \mathbb{S}_M(\boldsymbol{\theta}^g)}{\partial \boldsymbol{\theta}_1^g} & \cdots & \frac{\partial \mathbb{S}_M(\boldsymbol{\theta}^g)}{\partial \boldsymbol{\theta}_K^g} \end{pmatrix} J_{\boldsymbol{\theta}^g}(\boldsymbol{\eta}_0, \boldsymbol{\theta}^0) \tag{2.8}$$

for $S : \mathbb{R}^K \to \mathbb{R}^M$.

2.3 Jacobian Matrices via Automatic Forward Differentiation

Automatic (or algorithmic) Differentiation (AD) takes an algorithm for computing a value, E, and produces a new algorithm that computes the derivative of E with respect to one (or many) of its inputs. Thus we have an *algorithm* that turns an algorithm into another algorithm. AD has revolutionized the field of derivatives pricing, where classical Monte Carlo simulation is widely used to evaluate integrals numerically. Since the sensitivities express risk and hedging ratios, a large amount of work has been done on sensitivity analysis. Giles and Glasserman (2006) first suggested using automatic differentiation to accelerate methods previously developed. Homescu (2011) provides a good survey article which demonstrates the explosive impact of these techniques in finance. Of particular relevance for applications of AD to MCMC sampling is the introduction of partial proxy methods (OPP) by Chan and Joshi (2015), that combine the best aspects of the pathwise methods for smooth functions and likelihood ratio methods for discontinuous functions (Glasserman 2004). Joshi and Zhu (2016a, 2016b) provide important extensions of these methods for algorithms involving Gamma random variables and rejection sampling. Both of these feature prominently in Bayesian MCMC via the most commonly used Gibbs and Metropolis-Hastings algorithms.

AD relies on two fundamental results from computer science relating to the adjoint mode and forward mode of differentiation.

Theorem 1 Forward mode differentiation: *If a function F from $\mathbb{R}^n$ to $\mathbb{R}^k$ can be computed with L elementary operations then its gradient with respect to l of its inputs can be computed with $3lL$ elementary operations, and with no more than twice the memory required for the original computation.*

The theorem requires a class of elementary operations. These must be differentiable with simple derivatives. They will typically have one or two inputs. The final value has to be computable using only these operations. A typical class would be

$$+, *, -, \mathsf{rec}, \mathsf{exp}, \mathsf{log}, \mathsf{sin}, \mathsf{cos}, \mathsf{tan},$$

where $\mathsf{rec}\, z = 1/z$, and we shall work with this class. These elementary operations have to be continuous in a neighbourhood of where they are used and are usually differentiable everywhere. In practice, one can handle functions such as max which are differentiable almost everywhere and continuous everywhere. Since our program has to be a composition of such continuous functions, the output must be a continuous function of the inputs. Appendix 7.1 sketches out the general principles of construction a differentiated algorithm for any continuous algorithm. We also refer the reader to Griewank–Walther (2008) for a detailed exposition on Automatic Differentiation.

The forward mode theorem we have stated has two strong virtues: the additional time taken does not depend on the number of outputs and it takes no more than double the memory. The time taken, however, does scale linearly with the number of sensitivities to be computed. There is a second dual approach known as the adjoint or reverse mode. While the approach is faster, it requires storing the entire computation and so can have much greater memory costs than the original computation and the forward mode. For most more advanced models beyond the standard linear regression model with independent normal errors, the memory requirements of the adjoint method prohibit its application to computing the sensitivities for MCMC output. In the remainder of the paper we therefore focus on the forward approach.

2.4 Implementation of Automatic Differentiation via Forward Mode

To illustrate the basic idea for the implementation of AD for MCMC algorithm, let

$$F : \mathbb{R}^{p+K} \to \mathbb{R}^K \text{ such that}$$

$$\boldsymbol{\theta}^g = F(\boldsymbol{\eta}_0, \boldsymbol{\theta}^{g-1})$$

denote the MCMC algorithm with transition kernel $p(\boldsymbol{\theta}^g|\boldsymbol{\theta}^{g-1}, \boldsymbol{\eta}_0, Y)$ and starting values $\boldsymbol{\theta}^0$ that generates draws $\boldsymbol{\theta}^g$ given $\boldsymbol{\theta}^{g-1}$. Before starting the algorithm, one can initialize the sensitivities with respecting to the starting value by setting

$$J_{\boldsymbol{\theta}^0}(\boldsymbol{\theta}^0) = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & 1 & \dots & 0 \\ \dots & \dots & 0 & 1 & \dots \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & 0 & 1 \end{bmatrix} \in \mathbb{R}^{k \times l}$$

i.e. a sparse matrix with only the last l diagonal terms are nonzero, and initialize the sensitivities with respect to the hyper-parameters $J_{\boldsymbol{\theta}^0}(\boldsymbol{\eta}_0)$ as a zero matrix of dimension $K \times p$.

In a typical MCMC algorithm the transition kernel consists of a number of steps to draw several parameter blocks where each step consists of first updating the state parameter $\mathbf{s}$ of the transition kernel (such as means and variances of conditional distribution) and then drawing the model parameters from the transitional kernel with the updated state parameters.

For the clarity of the discussion we focus on the case of a one block MCMC algorithm where all elements in $\boldsymbol{\theta}$ are updated in one step and decompose this algorithm into two sub-functions $F = f^2 \circ f^1$ where

- $f^1 : \mathbb{R}^{p+K} \to \mathbb{R}^d$ calculates the state parameters $\mathbf{s}^g$ of the transition kernel, $\mathbf{s}^g \in \mathbb{R}^d$, i.e.

$$\mathbf{s}^g = f^1(\boldsymbol{\eta}_0, \boldsymbol{\theta}^{g-1})$$

  where f^1 only depend on the hyper-parameters and the dimension d is typically larger than K

- $f^2 : \mathbb{R}^d \to \mathbb{R}^K$ generates the draws given the parameters of the transition kernel, i.e.

$$\boldsymbol{\theta}^g = f^2(\mathbf{s}^g).$$

The aim is to compute the Jacobian $J_{\boldsymbol{\theta}^g}(\boldsymbol{\eta}_0, \boldsymbol{\theta}^0)$ of dimension $K \times (p+l)$ that contains the partial derivatives of the parameter draws with respect to input vectors $\boldsymbol{\eta}_0$ and $\boldsymbol{\theta}^0$. Notice that we have

$$J_{\boldsymbol{\theta}^g}(\boldsymbol{\eta}_0, \boldsymbol{\theta}^0) = \Big[J_{\boldsymbol{\theta}^g}(\boldsymbol{\eta}_0), J_{\boldsymbol{\theta}^g}(\boldsymbol{\theta}^0) \Big]$$

with $J_{\boldsymbol{\theta}^g}(\boldsymbol{\eta}_0) \in \mathbb{R}^{K \times p}$ the sensitivities of $\boldsymbol{\theta}^g$ with respect to the hyper-parameters and $J_{\boldsymbol{\theta}^g}(\boldsymbol{\theta}^0) \in \mathbb{R}^{k \times l}$ with respect to the starting values. Like the parameters, the sensitivities are also updated via the chain, starting from $g = 1$. Firstly, we apply the chain rule to compute $J_{\boldsymbol{\theta}^g}(\boldsymbol{\eta}_0)$, via

$$J_{\boldsymbol{\theta}^g}(\boldsymbol{\eta}_0) = J_{f^2}(\boldsymbol{\eta}^g) J_{f^1}(\boldsymbol{\eta}_0) + J_{f^2}(\boldsymbol{\eta}^g) J_{f^1}(\boldsymbol{\theta}^{g-1}) J_{\boldsymbol{\theta}^{g-1}}(\boldsymbol{\eta}_0) \tag{2.9}$$

where

- $J_{f^2}(\boldsymbol{\eta}^g) \in \mathbb{R}^{K \times d}$ is the Jacobian of f^2
- $J_{f^1}(\boldsymbol{\eta}_0) \in \mathbb{R}^{d \times p}$ are the first p columns of the Jacobian of f^1, $J_{f^1}(\boldsymbol{\eta}_0, \boldsymbol{\theta}^{g-1})$
- $J_{f^1}(\boldsymbol{\theta}^{g-1}) \in \mathbb{R}^{d \times K}$ are the last K columns of the Jacobian of f^1.
- $J_{\boldsymbol{\theta}^{g-1}(\boldsymbol{\eta}_0)} \in \mathbb{R}^{K \times p}$ has already been computed in the last iteration $g-1$, and it is initialized the beginning of the chain as a zero matrix.

Note that the expression (2.9) contains a path-wise and a chain-rule component. The path-wise component arises from the *direct* influence of hyper-parameters on the transition kernel, $p(\boldsymbol{\theta}^{g+1} | \boldsymbol{\theta}^g, \boldsymbol{\eta}_0, Y)$, i.e. the random number generation algorithm is a function of the hyper-parameters. At the same time, they also affect the previous draw at $(g-1)$ which in term determines the state parameters of the transition kernel at the current iteration (g).

We again apply the chain rule to compute the derivatives of $\boldsymbol{\theta}^g$ with respects to the starting values as

$$J_{\boldsymbol{\theta}^g}(\boldsymbol{\theta}^0) = J_{f^2}(\boldsymbol{\eta}^g) J_{f^1}(\boldsymbol{\theta}^{g-1}) J_{\boldsymbol{\theta}^{g-1}}(\boldsymbol{\theta}^0) \tag{2.10}$$

Note that the Jacobians $J_{f^2}(\boldsymbol{\eta}^g)$ and $J_{f^1}(\boldsymbol{\theta}^{g-1})$ appear in both the calculations in (2.9) and (2.10) so only have to be computed once to compute the Jacobians for both sets of input parameters. This substantially reduces the computational burden, contributing to the efficiency of the AD approach.

The simpler expression for starting value sensitivities results from the fact that we only have a chain-rule component. The influence of the starting value here is only secondary. It is only passed through via the dependence on the last draw, since it is not a direct parameter of the transition kernel. For practitioners, this set of sensitivities is critical, as it numerically assesses the dependence of the chain on the starting point over the *whole* parameter space, i.e. shed lights on when to stop the burning period and start to collect samples of the empirical posterior distribution. This is not computationally feasible in the past via numerical differentiation. Automatic Differentiation allows us to algorithmic capture these effects in an efficient manner, that is computing them simultaneously through elementary operations.

Since f^1 computes the parameters of a transition kernel, such as means and variances, it is typically straight forward to decompose the subfunction into elementary operations and to compute the required Jacobians. Special care have to taken with matrix inversions and Cholesky's decompositions in multidimensional contexts (as shown in Appendix 7.2).

The potentially more challenging component is f^2 which involves the draw of random variables. In the case of a Gibbs sampler the distribution of the random variables would be determined by the conditional posterior distributions. If a random variable follows a standard continuous distributions where we can apply the inverse transform method. In this case the Jacobians can be computed in a straight forward manner, for example for Normal Random variables. In section 3.1 we illustrate the implementation of these calculations in the context of a 2-step Gibbs sampler. Two notable exceptions that commonly arise in Gibbs samplers are Gamma and Wishart random variable. In Section 3 we discuss details regarding the differentiation of such random variables. Another important exception is the case of the Metropolis algorithm that is typically applied when the conditional distribution is not available in closed form that we address in a different paper.

For Gibbs algorithms with more than one block, the logic behind AD is essentially the same as what we demonstrated via $F = f^2 \circ f^1$. One can always decompose into the mapping for computing the state parameters and the mapping for generating the draws as demonstrates in Section 3.1. The only subtlety lies in the different dimensions of the mappings, typically determined by the dimensions of each blocks. However, we believe this is a minor issue which can be solved by an adequate programer, hence we shall not exhaust the length of the paper in explaining the exact details of the decomposition.

2.5 Sensitivity Measures for Prior Robustness and Convergence

A key feature of the algorithmic differentiation approach is that it provides a general framework to obtain the sensitivities of all MCMC output, including draws and statistics, with re-

spect to any of the input parameters. These sensitivities computed via AD provide us important information about two critical issue in Bayesian inference, namely prior robustness and convergence.

2.5.1 Sensitivities for Prior Robustness

The first objective of Bayesian sensitivity analysis is to address the issue of subjectivity and assess the robustness of the MCMC output with respect to the complete set of subjective prior hyper-parameters set by the researcher. In general, we want to compute the sensitivities of any function of the parameters with respect to the prior hyper-parameters

$$\frac{\partial \mathbb{E}_{\hat{\pi}}[S(\boldsymbol{\theta})|Y, \boldsymbol{\theta}^0, \boldsymbol{\eta}_0]}{\partial \boldsymbol{\eta}_0}$$

where $\mathbb{E}_{\hat{\pi}}$ reflects that the expectation of S is obtained from the estimated posterior distribution. Sensitivities of posterior mean estimates are of particular interest in many empirical applications. To assess the robustness for the posterior mean estimate

$$\hat{\theta}_k = \frac{1}{G} \sum_{g=B+1}^{B+G} \theta_k^g \xrightarrow{p} \mathbb{E}_{\hat{\pi}}[\theta_k|Y, \boldsymbol{\theta}^0, \boldsymbol{\eta}_0]$$

we compute the gradient vector $\nabla \hat{\theta}_k(\boldsymbol{\eta}_0)$ that contains the set of partial derivatives $\partial \hat{\theta}_k / \partial \eta_{0,p}$ with respect to all hyper-parameters $\eta_{0,p} \in \boldsymbol{\eta}_0$. The Jacobian matrix $J_{\hat{\boldsymbol{\theta}}}$ summarizes the complete set of sensitivities for

$$J_{\hat{\boldsymbol{\theta}}} = \begin{pmatrix} \nabla \hat{\boldsymbol{\theta}}_1(\boldsymbol{\eta}_0) \\ \vdots \\ \nabla \hat{\boldsymbol{\theta}}_K(\boldsymbol{\eta}_0) \end{pmatrix} \tag{2.11}$$

As shown further below obtaining this Jacobian matrix requires the estimation of the sensitivities of all draws, i.e. the draws after Burn-in period for any element k in the parameter vector, summarized in the gradient vectors $\nabla \theta_k^{(g)}(\boldsymbol{\eta}_0)$ for g=B+1,...,B+G.

2.5.2 Sensitivities for Convergence

A key advantage of sensitivity analysis via the AD approach is that it opens up the possibility to compute the first order derivatives of the draws and also statistics with respect to the starting values, which is not possible via likelihood ratio type methods. As Markov chain theory implies that after a chain has converged draws no longer depend on the starting values, we propose a set of measures to assess to convergence directly based on the sensitivity of the draws to the starting values. In other words, convergence would imply that derivatives of draws, and hence also statistics, with respect to θ^0 should equal to zero. Sensitivity analysis based on AD enables us to assess the evolution of the dependence draws of the chain with respect to the starting values by computing the sensitivities of the vector draws of draws at

each iteration g with respect to the

$$\left(\frac{\partial \boldsymbol{\theta}^{(g)}}{\partial \theta_j^0}\right) \quad \forall\ \theta_j^0 \in \boldsymbol{\theta}^0$$

starting at $g = 1$. As convergence requires that these sensitivities should be tending to zero, the required number of burn-in periods can be determined based on the $(\partial \boldsymbol{\theta}^{(g)}/\partial \theta_j^0)$'s. For example, we propose one measure based on a threshold for the maximum observed starting value sensitivities at each iterations.

2.5.3 Summary Robustness Measures

The AD approach enables us to compute a very large set of prior robustness and convergence measures. For convergence we compute the first order derivatives of each element in the $K \times 1$ parameter vector $\boldsymbol{\theta}^g$ in $J_{\boldsymbol{\theta}^g}(\boldsymbol{\theta}^0)$ and several statistics based on $\boldsymbol{\theta}^g$ with respect to the $l \times 1$ dimensional vector of starting values at each iteration of the chain. It is therefore useful to compute some summary convergence measure that report for example the maximum of average of the set of absolute starting values sensitivities at each iteration of the chain and monitor their evolution.In Section 5.1 we discuss a few possible summary convergence measure in the context of our illustrative example.

For prior robustness measure the main focus in on the sensitivities of parameter based statistics. The AD allows us to compute the sensitivities of each element in the $K \times 1$ posterior mean vector $\mathbb{E}_{\hat{\pi}}[g(\boldsymbol{\theta})|Y, \boldsymbol{\theta}^0, \boldsymbol{\eta}_0]$ with respect to the p elements in the prior hyper parameter vector. In this case the researcher might be interested in the complete set of sensitivities to see how important each of the prior assumptions was. However, in most models we have a large number of prior hyper-parameters, which leads to high-dimensional gradient vectors. We therefore also propose a summary measures of overall dependence on the prior hyper-parameters based Euclidean norm $\|\nabla \hat{\boldsymbol{\theta}}_k(\boldsymbol{\eta}_0)\|$ of the gradient vectors as discussed in the context of the illustrative example in Section 5.2.

3 Sensitivities via AD for Gibbs Output

In this paper we focus on the sensitivity analysis of Bayesian output obtained from a Markov chain where the transition kernel $p(\boldsymbol{\theta}^g|\boldsymbol{\theta}^{g-1}, \boldsymbol{\eta}_0, Y)$ is based on the Gibbs sampler. The Gibbs algorithm is the most efficient and most commonly applied method in MCMC analysis and the easiest to implement as it is simply a sequence of draws from full conditional distributions. In many problems it is possible to choose suitable prior distributions, so-called conjugate priors, that ensure that given the likelihood the full conditional posterior distributions are of a known form.

Suppose for now that the parameter vector $\boldsymbol{\theta}$ can be divided in two blocks $(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)$. If the full-conditional distributions $\pi(\boldsymbol{\theta}_1|Y, \boldsymbol{\theta}_2)$ and $\pi(\boldsymbol{\theta}_2|Y, \boldsymbol{\theta}_1)$ are of a known type and can be sampled directly, then the transition kernel consists of two steps. After initializing $\boldsymbol{\theta}_2^0$, we draw θ_1^g from $\pi(\boldsymbol{\theta}_1^g|Y, \boldsymbol{\theta}_2^{g-1})$ and then $\boldsymbol{\theta}_2^g$ from $\pi(\boldsymbol{\theta}_2^g|Y, \boldsymbol{\theta}_1^g)$. The number of blocks and

specific form of the full conditional distributions will be problem specific. Common updates in Gibbs algorithms are based on the Normal distribution for coefficient vectors, the Gamma distribution for variance parameters and the Wishart distribution for covariance matrices.

3.1 Linear Model Output from Gibbs Algorithm

We introduce the concept of sensitivity analysis first for Gibbs output in the context of Linear model with standard priors estimated via a 2-step sampler by implementing forward mode AD as introduced in Section 2.4. Under the conjugate choices of a Normal prior distribution for $\boldsymbol{\beta} \in \mathbb{R}^k$ and a Gamma prior for $h = \sigma^{-2} \in \mathbb{R}$ the posterior distribution of the parameter vector $\boldsymbol{\theta} = (\boldsymbol{\beta}, h)$, given by

$$\pi(\boldsymbol{\beta}, h|Y, \boldsymbol{\eta}_0) \propto N_k(\boldsymbol{\beta}|\mathbf{b}_0, \mathbf{B}_0)\, G\left(h \mid \frac{\alpha_0}{2}, \frac{\delta_0}{2}\right) N_n(Y|X\boldsymbol{\beta}, h^{-1} \times I). \tag{3.1}$$

The posterior distribution depends on the vector of hyper-parameters

$$\boldsymbol{\eta}_0 = (\mathbf{b}_0, \mathbf{B}_0, \alpha_0, \delta_0) \in \mathbb{R}^p, \text{ where } p = k + \frac{k(k+1)}{2} + 3,$$

and can be estimated via a two-step Gibbs sampler. We initialize the chain with a value h^0 so that

$$\theta^0 = h^0 \in \mathbb{R}, l = 1.$$

Estimation. At each iteration, g, of the chain, $\boldsymbol{\beta}$ is drawn from a Normal distribution and h from a Gamma distribution. We again decomposing the algorithm in the two type of subfunctions to compute the state values of the transition kernel (f^1) and the parameter draw from the transition kernel (f^2) as follows:

1. Draw $\boldsymbol{\beta}^g$ via algorithm F^1 such that

   - $f^{1,1}$: Compute $\mathbf{s}_1^g = (\mathbf{b}^g, \mathbf{B}^g) \in \mathbb{R}^{k+\frac{k(k+1)}{2}}$, where $\mathbf{B}^{(g)} = \left[h^{g-1}X^TX + \mathbf{B}_0^{-1}\right]^{-1}, \mathbf{b}^{(g)} = \mathbf{B}^g\left[h^{(g-1)}X^TY + \mathbf{B}_0^{-1}\mathbf{b}_0\right]$.
   - $f^{1,2}$: draw $\boldsymbol{\beta}^g \sim N(\mathbf{b}^g, \mathbf{B}^g)$ via

     $$\boldsymbol{\beta}^g = \mathbf{b}^g + Chol(\mathbf{B}^g)Z \text{ for } Z \sim N(0, \mathbb{I})$$

     where $\mathbb{I}$ is the $k \times k$ identity matrix.

2. Draw h^g from F^2:

   - $f^{2,1}$: Compute $\mathbf{s}_2^g = (\alpha_1, \delta^g) \in \mathbb{R}^2$, where $\alpha_1 = \alpha_0 + n$ and

     $$\begin{aligned}\delta^{(g)} &= \delta_0 + (Y - X\boldsymbol{\beta}^g)^T(Y - X\boldsymbol{\beta}^g) \\ &= \delta_0 + Y^TY - 2Y^TX\boldsymbol{\beta}^g + (\boldsymbol{\beta}^g)^TX^TX\boldsymbol{\beta}^g\end{aligned}$$

- $f^{2,2}$: Draw $h^g \sim G\Big(\frac{\alpha_1}{2}, \frac{\delta^{(g)}}{2}\Big)$ from $h^g = \frac{2}{\delta^g} G$ with $G \sim G(\frac{\alpha_1}{2}, 1)$.

The inverse variance $h = \sigma^{-2}$ is Gamma distributed with hyperparameters α and δ using the following parameterization:

$$\frac{\delta^\alpha}{\Gamma(\alpha)} h^{\alpha-1} e^{-\delta h}$$

where $E[h] = \alpha/\delta$ and $\mathsf{Var}[h] = \alpha/\delta^2$ (see Greenberg 2012 for detailed discussion). The Gibbs sampler described above has the following chain:

$$h^{(0)} \to \beta^{(1)} \to h^{(1)} \to \beta^{(2)} \to h^{(2)} \dots \to \beta^{(g)} \to h^{(g)} \to \beta^{(g+1)} \to \dots \to h^{(B+G)}.$$

Sensitivity Computations. We shall illustrate the computation of the sensitivities with respect to the two sets of inputs, the prior hyper-parameters and the starting value. We initialize the gradient vector $\bigtriangledown_{h^0}(\boldsymbol{\eta}_0, h^0) = [0, 0, \dots, 0, 1]$ and use the following algorithm to compute the sensitivities $J_{\boldsymbol{\beta}^g}(\boldsymbol{\eta}_0, h^0)$ and $\bigtriangledown_{h^g}(\boldsymbol{\eta}_0, h^0)$:

1. Obtain $J_{\boldsymbol{\beta}^g}(\boldsymbol{\eta}_0, h^0) = [J_{\boldsymbol{\beta}^g}(\boldsymbol{\eta}_0), J_{\boldsymbol{\beta}^g}(h^0)]$ by decomposing into sub-mappings [2]

   - Operation $f^{1,1}$: Obtain the distributional parameters of the normal update
     - $\mathbf{B}^* = h^{g-1} X^T X + \mathbf{B}_0^{-1}$ [3]

       $$J_{\mathbf{B}^*}(\boldsymbol{\eta}_0, h^0) = [J_{\mathbf{B}^*}(\boldsymbol{\eta}_0), J_{\mathbf{b}^*}(h^0)] = [X^T X . * \bigtriangledown_{h^{g-1}}(\boldsymbol{\eta}_0) + J_{\mathbf{B}_0^{-1}}(\eta_0), \frac{\partial h^{g-1}}{\partial h^0} X^T X]$$

     - $\mathbf{B}^g = (\mathbf{B}^*)^{-1}$

       $$J_{\mathbf{B}^g}(\boldsymbol{\eta}_0, h^0) = [J_{\mathbf{B}^g}(\boldsymbol{\eta}_0), J_{\mathbf{B}^g}(h^0)] = [-\mathbf{B}^* . * J_{\mathbf{B}^*}(\boldsymbol{\eta}_0) . * \mathbf{B}^*, -\mathbf{B}^* . * J_{\mathbf{B}^*}(h^0) . * \mathbf{B}^*] \tag{3.2}$$

       See the appendix for the details of this formula.
     - $\mathbf{b}^* = h^{(g-1)} X^T Y + \mathbf{B}_0^{-1} \mathbf{b}_0$

       $$J_{\mathbf{b}^*}(\boldsymbol{\eta}_0, h^0) = [J_{\mathbf{b}^*}(\boldsymbol{\eta}_0), J_{\mathbf{b}^*}(h^0)] = [X^T Y . * \bigtriangledown_{h^{g-1}}(\boldsymbol{\eta}_0) + J_{\mathbf{B}_0^{-1}\mathbf{b}_0}(\eta_0), \frac{\partial h^{g-1}}{\partial h^0} X^T Y]$$

     - $\mathbf{b}^g = \mathbf{B}^g \mathbf{b}^*$

       $$J_{\mathbf{b}^g}(\boldsymbol{\eta}_0, h^0) = \mathbf{B}^g J_{\mathbf{b}^*}(\boldsymbol{\eta}_0, h^0) + \mathbf{b}^* . * J_{\mathbf{B}^g}(\boldsymbol{\eta}_0, h^0)$$

   - Operation $f^{1,2}$: First pass the Jacobian operator through the Cholesky decomposition and obtian $J_{chol(\mathbf{B}^g)}$, which is a quite involved process. There are Matlab packages computes such derivative (see Appendix 7.2 for more details). then

     $$J_{\boldsymbol{\beta}^g}(\boldsymbol{\eta}_0, h^0) = J_{\mathbf{b}^g}(\boldsymbol{\eta}_0, h^0) + Z . * J_{chol(\mathbf{B}^g)}(\boldsymbol{\eta}_0, h^0)$$

[2] To save computational time, $\mathbf{B}_0^{-1}$ and $\mathbf{B}_0^{-1}\mathbf{b}_0$ as well as their Jacobians $J_{\mathbf{B}_0^{-1}}(\boldsymbol{\eta}_0) \in \mathbb{R}^{(K\times K)\times p}$ and $J_{\mathbf{B}_0^{-1}\mathbf{b}_0}(\eta_0) \in \mathbb{R}^{K\times p}$ can be pre-computed and stored.

[3] Strictly speaking this is creating a 3-D tensor, the $.*$ operation is multiplying every element of $\bigtriangledown_{h^{g-1}}(\boldsymbol{\eta}_0)$ by $X^T X$.

2. Obtain $\bigtriangledown_{h^g}(\boldsymbol{\eta}_0, h^0)$

   - Operation $f^{2,1}$: update δ^g

   $$\bigtriangledown_{\delta^g}(\boldsymbol{\eta}_0, h^0) = \mathbb{I}^{p+l}(p) + 2((\mathbf{b}^g)^T(X^TX) - Y^TX)J_{\boldsymbol{\beta}^g}(\boldsymbol{\eta}_0, h^0).$$

   - Operation $f^{2,2}$: given G

   $$\bigtriangledown_{h^g}(\boldsymbol{\eta}_0, h^0) = -\frac{2}{(\delta^g)^2} G \bigtriangledown_{\delta^g}(\boldsymbol{\eta}_0, h^0) + \frac{2}{\delta^g} J_G(\boldsymbol{\eta}_0, h^0). \tag{3.3}$$

   Here the second part of the equation involves taking derivatives of a gamma random variable with unit dispersion parameter, it is not trivial and we discuss it in section 3.2.

3. Store sensitivities of interest from $\bigtriangledown_{h^g}(\boldsymbol{\eta}_0, h^0)$ and $J_{\boldsymbol{\beta}^g}(\boldsymbol{\eta}_0, h^0)$, in particular starting values sensitivities, and accumulate sums for prior input parameter sensitivities to compute Jacobians following (2.7).

4. After repeating steps 1 - 3 for $B + G$ iterations of the algorithm, we can compute the basic prior robustness measures, for example for the posterior means of $\boldsymbol{\beta}$

$$\frac{1}{G}\sum_{g=B+1}^{B+G} J_{\boldsymbol{\beta}^g}(\boldsymbol{\eta}_0).$$

3.2 Automatic Differentiation of Discontinuous Random Variables

The automatic differentiation approach requires the algorithm to be continuous. Typically, if we use a Gibbs sampler to estimate $\pi(\boldsymbol{\theta}|Y)$ that is continuous and differentiable, i.e. a sequence of random variables generated by the inverse cumulative methods, then we can compute the sensitivities of the MCMC output or any statistic $g(\boldsymbol{\theta})$ estimated based on $\hat{\pi}(\boldsymbol{\theta}|Y)$ from the Gibbs algorithm. Given a random uniform $u \sim U(0,1)$, we obtain $X_\theta = F^{-1}(u, \theta)$ from the inverse of the target cumulative density function provided that F^{-1} can be computed analytically. A classic example is the generation of random normal variates for the $\boldsymbol{\beta}$ update in the Gibbs sampler for the linear model, where the standard random Normal z is generated via the inverse transform method as $Z = \Phi^{-1}(U)$, so that the $\boldsymbol{\beta}$ really is generated in the computer algorithm as

$$\beta^{(g)} = \mathbf{b}^{(g)} + Chol(\mathbf{B}^{(g)})\Phi^{-1}(u), \tag{3.4}$$

where Φ is the standard normal cumulative density function. Given $U \in [0,1]^K$, this mapping is smooth with respect to both $\mathbf{b}^{(g)}$ and $\mathbf{B}^{(g)}$ which allows the direct application of automatic differentiation as shown above.

Problems occur when the random variable generators have discontinuities, such as in the case of Gamma and Wishart random variables. For these distributions, F^{-1} does not exist or is too cumbersome to work with, alternative methods such as the acceptance-rejection, the

transform-with-multiple-root and the ratio-of-uniform methods were introduced to simulate these variates. There are inherent discontinuities in these algorithms since a candidate outcome is accepted as a variate from the target distribution depending on passing certain criteria that are set by the parameter of interest. Thus, a small change in the parameter of interest can result in a big change in the simulated variate that is a pathwise discontinuity in the simulation algorithm. And so the application of automatic differentiation is problematic.

For our simple model, $h^{(g)}$ is drawn from gamma distributions with parameter α_1 and $\delta^{(g)}$ and the inverse of its cumulative density function is not available analytically. Rejection techniques for simulating $\gamma(\alpha, 1)$ random variates have been studied extensively and vary according to the value of α since the qualitative properties of the distribution vary greatly with it (Tanizaki, 2008). For our examples, α is always greater than 1. The GKM1 method (Cheng and Feast, 1979) simulates gamma random variables with $\alpha > 1$ by using a computing a ratio of uniforms and then accepting or rejecting according to parameter values. We give the algorithm in the appendix 7.2.

Joshi and Zhu (2016) introduced a method to resolve this pathwise discontinuity in the classical simulation setting. They effectively use a measure change at each acceptance-rejection point, to ensure the pathwise discontinuity is removed and the regularised pathwise estimator satisfied the application of automatic differentiation to obtain unbiased derivative estimates. Unfortunately, the method is not directly applicable to the dependent MCMC settings. The induced likelihood ratio weights resulting from each measure change are carried through the markov chain, and easily explode the variances of the derivative estimates. On the other hand, Suri (1983) proposed a generic approach to compute derivatives of random variates X with respect to its distributional parameters θ based on

$$F(F^{-1}(u, \theta), \theta) = u.$$

One can differentiate this expression from both sides with respect to θ and obtain

$$\frac{\partial}{\partial \theta} F(X, \theta) + f(X, \theta) \frac{\partial X_\theta}{\partial \theta} = 0,$$

where f is the probability density function of the distribution. Thus, we have

$$\frac{\partial X_\theta}{\partial \theta} = -\frac{\frac{\partial}{\partial \theta} F(X, \theta)}{f(X, \theta)}.$$

The method depends on the differentiability of the cumulative density function.

Glasserman and Liu(2010) applied the method for computing the derivative price sensitivities when the underlying stock process is Lévy driven. In particular, they used it for computing derivatives of the Gamma random variate, $\Gamma(\alpha, 1)$, with respect to its shape parameter, α, using the following formula

$$\frac{\partial \Gamma(\alpha, 1)}{\partial \alpha} = \frac{F_{(\alpha-1,1)}(\Gamma(\alpha, 1)) - \psi(\Gamma(\alpha, 1)) F_{(\alpha,1)}(\Gamma(\alpha, 1))}{f_{(\alpha,1)}(\Gamma(\alpha, 1))} \tag{3.5}$$

where

- $F_{(\alpha,1)}$ and $F_{(\alpha-1,1)}$ are cumulative density functions gamma distribution with dispersion parameter one, and shape parameters α and $\alpha - 1$ respectively,
- $\psi(x)$ is the digamma function,
- $f_(\alpha, 1)$ is the probability density function of gamma distribution with shape parameter α and dispersion parameter one.

We use expression (3.5) to compute $J_G(\boldsymbol{\eta}_0, h^0)$ in (3.3).

Another random variable that is common in Bayesian Gibbs analysis, but not in classical simulation analysis where AD methods have been applied, is the Wishart random variable. It is the conjugate prior choice for the inverse of the unrestricted covariance matrices in many models, $\boldsymbol{\Sigma}^{-1} \sim W(\nu_0, \mathbf{R}_0)$, with degree of freedoms ν_0 and scale matrix $\mathbf{R}_0$ which is defined as follows

$$f(\boldsymbol{\Sigma}^{-1}|\nu_0, \mathbf{R}_0) \propto \frac{|\boldsymbol{\Sigma}|^{-\frac{(\nu_0-k-1)}{2}}}{|\mathbf{R}_0|^{\nu_0/1}} \exp\left\{-\frac{1}{2}tr(\mathbf{R}_0^{-1}\boldsymbol{\Sigma}^{-1})\right\}$$

where tr refers to the trace of the matrix defined as the sum of the diagonal elements sot hat the expression in the exponential is a scalar. This conjugate prior implies a Gibbs update from a Wishart random variable. For the derivation of the Wishart random variables, we first reduce it to the Gamma case via a transformation. In particular, we generate the Wishart random matrix $X \sim \text{Wishart}(p, R)$ via the Bartlett decomposition,

$$X = LAA^TL^T$$

where L is the Cholesky decomposition factor of the scale matrix R, and A is an $p \times p$ lower triangle matrix that its off-diagonal terms are $N(0, 1)$ and its diagonal terms are chi-squared random variables. Since the chi-squared random variables can be treated as a special case of Gamma random variables, the distributional derivative method is directly applicable to their generation. The sensitivities of the Wishart random variables then follow by chaining.

3.3 Implementation and Run-time

Next we illustrate the computation time of running the MCMC algorithm and sensitivity algorithm time for computing first-order sensitivities for the Normal model with a sample of size 1,000 using the methods described in the previous two sections. Theorem 1 on the forward model of AD implies that the computational cost of the forward mode AD increases linearly with the number of input parameters of interest and the number of operations in the original algorithm for estimation. As some operations are very fast increases in computation time are often less than linear.

First we focus on the change in computational time when we increase the number of inputs. We generate a linear increase in the number of inputs for sensitivity computations by increasing the dimension of $\boldsymbol{\beta} \in \mathbb{R}^k$ and focusing on the computation of the sensitivities with respect to the prior mean vectors $\mathbf{b}_0 \in \mathbb{R}^k$. Table A shows the changes in computational time for estimation and the sensitivity computations for $J_{\beta^g}(\mathbf{b}_0)$ for $k = 1, ..., 7$. From the case

with the basic Normal model with an intercept in the first row to the case with additional 6 covariates in the last row the run-time for the sensitivity compuations increases from 6 to 9 seconds. In Table B we investigate the change in run-time as the number of MCMC itera-

Table A: Run-time and Parameter dimension		
k	Estimation	Sensitivity
1	1.016	6.316
2	1.085	6.771
3	1.085	7.483
4	1.059	7.291
5	1.108	7.671
6	1.151	8.032
7	1.169	9.074

Table B: Run-time and Iterations		
G	Estimation	Sensitivity
1000	0.110	0.736
2000	0.203	1.333
3000	0.301	1.936
4000	0.394	2.602
5000	0.555	3.205
6000	0.672	3.687
7000	0.734	4.361

Table 1: Run-time for Estimation and Sensitivity Estimation Components for different dimensions of the input vector (Table A with $B = 1000$, $G = 10000$) and different iteration numbers (Table B with $B = 0$, $k = 3$).

tions increase to reflect an increase in the number of iterations. Run-time of the sensitivity computations increases from 0.7 to 4.3 seconds as iterations increase from 1000 to 7000. As we increase iterations/operations in original estimation algorithm by a factor of 7, the time to compute sensitivities increases by a factor around 6.

The results are consistent with the theoretical computational cost of the forward mode AD, with computation times increasing at most linearly with the number of input parameters of interest and the number of operations in the original algorithm for estimation. This is important for a wide application of an AD based sensitivities as models are typically more complex. It also shows that while the associated computational cost for applying the bumping method prevents the assessment of sensitivities in practice, using AD the substantial amount of information embedded in sensitivity estimates can be obtained without exhausting the computational capacity.

4 Performance and Application of AD methods for Gibbs

4.1 Settings

We consider three different set-ups to demonstrate the performance of the AD approach and to compare the AD approach to a likelihood ratio approach for sensitivity computations of hyper-parameters. We start out with the standard linear regression model with Normal errors

$$y_i = x_i'\boldsymbol{\beta} + \epsilon_i \ , \epsilon_i \sim N(0, \sigma^2 = h^{-1}) \ \text{ with } \boldsymbol{\beta} \sim N_k(\mathbf{b}_0, \mathbf{B}_0) \ , \ h = \sigma^{-2} \sim G\left(\frac{\alpha_0}{2}, \frac{\delta_0}{2}\right).$$

Next we consider the extension of the linear regression model with student-t errors. The fat-tailed student-t distribution is often applied in modeling of income data and finance data (Chib

and Jacobi 2016, Martin et al 2005, Geweke 1993) when the Normality assumption fails. In Bayesian analysis the student-t distribution is commonly represented as a scale mixture of normals

$$\epsilon_i \sim N(0, \lambda_i^{-1}\sigma^2) \ , \ \lambda_i \sim G\Big(\frac{\nu}{2}, \frac{\nu}{2}\Big)$$

where the scale parameters $\{\lambda_i\}$ are added to set of model parameters. This type of data augmentation in common in Bayesian MCMC estimation. While it leads to a large increase in the dimension of the parameter space, it is applied in many cases to improve the structure of the likelihood to allow for an estimation via Gibbs samplers.

Finally, we consider a 2-equation model with joint Normal errors

$$y_{1,i} = x'_{1,i}\boldsymbol{\beta} + \epsilon_i \ \ , \ \ y_{2,i} = x'_{2,i}\boldsymbol{\gamma} + u_i \ , \ \ (\epsilon_i, u_i) \sim N_2(\mathbf{0}, \boldsymbol{\Sigma})$$

where the prior distribution for the covariance matrix is specified $\boldsymbol{\Sigma}^{-1} \sim W(\nu_0, \mathbf{R}_0)$. This type of model is the basis of many multivariate response models such as seemingly unrelated regressions or treatment effects models and estimation involves a common Gibbs update from a Wishart distribution.

For each model the posterior distribution can be estimated via a Gibbs Sampler (see Greenberg 2012 and appendices 7.3 and 7.4) based on the widely used Gibbs updates from Normal, Gamma and Wishart distributions. The complete set of input sensitivities are computed via embedded sensitivity algorithms based on the Automatic Differentiation methods discussed in Section 3.

4.2 Comparison with Likelihood Ratio Approach for Prior Mean Robustness

For a subset of the sensitivities obtained via the algorithmic approach it is possible to compute the derivatives based on the Likelihood Ratio Type approach for MCMC output introduced in Perez et al (2006). The approach is analogous to the likelihood ratio approach in financial mathematics (see Glasserman 2004). Müller applies the approach in the context of the exponential family to obtain the prior sensitivities of the $\boldsymbol{\beta}$ vector with respect to its prior mean vector $\mathbf{b}_0$.

Consider the posterior mean of some function of $\boldsymbol{\theta}$

$$E_\pi[g(\boldsymbol{\theta})|Y, \boldsymbol{\eta}_0] = \int_{\boldsymbol{\theta}} g(\boldsymbol{\theta})\, \pi(\boldsymbol{\theta} \,|\, Y, \boldsymbol{\eta}_0)\, d\boldsymbol{\theta}.$$

Here, without exhausting the amount of notations, we shall work with one element of the hyper-parameter. As shown in Perez et al (2006) the partial derivative with respect to the jth element in the prior parameter vector $\boldsymbol{\eta}_{0,j} \in \boldsymbol{\eta}_0$

$$\frac{\partial}{\partial \boldsymbol{\eta}_{0,j}} \mathbb{E}_\pi[g(\boldsymbol{\theta})|Y] = \frac{\partial}{\partial \boldsymbol{\eta}_{0,j}} \left[\frac{\int g(\boldsymbol{\theta}) f(Y|\boldsymbol{\theta})\, \pi(\boldsymbol{\theta}|\boldsymbol{\eta}_0)\, d\boldsymbol{\theta}}{\int f(Y|\boldsymbol{\theta})\, \pi(\boldsymbol{\theta}|\boldsymbol{\eta}_0) d\boldsymbol{\theta}} \right]$$

can be re-expressed using standard operations as

$$\frac{\int \left(g(\boldsymbol{\theta}) - E_{\pi}[g(\boldsymbol{\theta})|Y, \boldsymbol{\eta}_0]\right) \frac{\partial}{\partial \boldsymbol{\eta}_{0,j}} \log \pi(\boldsymbol{\theta}|\boldsymbol{\eta}_0) f(Y|\boldsymbol{\theta}) \pi(\boldsymbol{\theta}|\boldsymbol{\eta}_0)\, d\boldsymbol{\theta}}{\int f(Y|\boldsymbol{\theta})\, \pi(\boldsymbol{\theta}|\boldsymbol{\eta}_0) d\boldsymbol{\theta}}$$

where $\partial \log \pi(\boldsymbol{\theta}|\boldsymbol{\eta}_0)/\partial \boldsymbol{\eta}_{0,j}$ is also referred to as the score function. This sensitivity estimate can be computed using draws on $\boldsymbol{\theta}$ from the MC sampler as

$$\frac{1}{G} \sum_{g=B+1}^{B+G} [g(\boldsymbol{\theta}^{(g)}) - \hat{\boldsymbol{\theta}})] \frac{\partial}{\partial \boldsymbol{\eta}_{0,j}} \log \pi(\boldsymbol{\theta}^{(g)}|\boldsymbol{\eta}_0) \tag{4.1}$$

where $\hat{\boldsymbol{\theta}}$ is the sample mean estimate. For example, under this approach, which relies on computability of the score function of the prior distribution, the sensitivities of the posterior mean $\hat{\boldsymbol{\beta}}$ with respect to its prior mean vector $\mathbf{b}_0$ is obtained from the following expression

$$\frac{\partial}{\partial \mathbf{b}_0} \mathbb{E}[\beta|Y] = \Sigma_p^{-1} \Sigma_{\hat{\pi}}, \tag{4.2}$$

where Σ_p is the prior variance B_0 and $\Sigma_{\hat{\pi}}$ is the posterior covariance matrix of $\boldsymbol{\beta}$. The latter can be computed based on the MCMC draws after the burn-in.

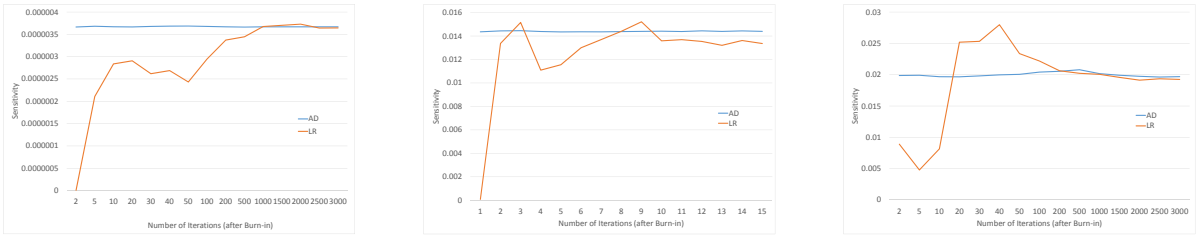
Table 2 presents the vector of sensitivity estimates of $\partial\hat{\beta}/\partial b_0$ for all three models, and also for the coefficient vector, $\frac{\partial\hat{\gamma}}{\partial g_0}$, in the second equation in the joint model. We observe that the two metholodologies give similar but not identical numbers – this difference most likely arising from the slowness of the LR method's convergence which we will see below.

Normal Model		Student-t Model		2-Equation Model			
AD	LR	AD	LR	AD	LR	AD	LR
$\frac{\partial \hat{E}[\beta_1\|Y]}{\partial \mathbf{b}_0}$	$\widehat{\frac{\partial E[\beta_1\|Y]}{\partial \mathbf{b}_0}}$	$\frac{\partial \hat{E}[\beta_1\|Y]}{\partial \mathbf{b}_0}$	$\widehat{\frac{\partial E[\beta_1\|Y]}{\partial \mathbf{b}_0}}$	$\frac{\partial \hat{E}[\beta_1\|Y]}{\partial \mathbf{b}_0}$	$\widehat{\frac{\partial E[\beta_1\|Y]}{\partial \mathbf{b}_0}}$	$\frac{\partial \hat{E}[\gamma_1\|Y]}{\partial \mathbf{g}_0}$	$\widehat{\frac{\partial E[\gamma_1\|Y]}{\partial \mathbf{g}_0}}$
3.67E-06	3.61-06	5.04E-06	5.02E-06	0.005970	0.005759	5.43E-06	5.32E-06
-2.88E-06	-2.82-06	-3.94E-06	-3.94E-06	-0.000376	-0.000363	-4.27E-06	-4.22E-06
-3.52E-07	-3.40E-07	-4.80E-07	-4.74E-07	7.52E-05	7.22E-05	-5.22E-07	-5.25E-07
-3.99E-07	-3.93-07	-5.57E-07	-5.43E-07	7.03E-05	6.83E-05	-5.91E-07	-5.52E-07

Table 2: Sensitivities of posterior mean of β_1 and γ_1 with respect to their prior mean vectors.

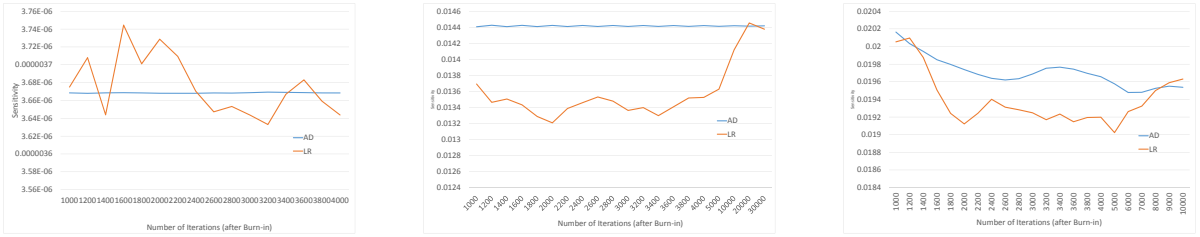
Expression (4.1), without the $\hat{\boldsymbol{\theta}}$ term, is well known in the financial math literature on sensitivity estimation as the Likelihood Ratio approach. There are a number of known issues with such Likelihood Ratio methods. Firstly, they tend to be more unstable and slower to converge. Secondly, they have higher variance. These are also borne out in the context of our MCMC analysis. Figures 2 and 3 show that sensitivities computed under the AD approach converge faster and remain more stable across iterations than those computed under the LR approach. Even in the case of the simple linear regression model the LR estimates converge more slowly and also remain less stable.

Unlike our method of assessing the sensitivities of the MCMC outputs over the evolution of the markov chain, the LR method focuses on the posterior distribution. It explore the dependence of the posterior distribution on the prior hyper-parameters given the sample obtained has converged asymptotically. On the other hand, AD allows us to focus on the dependence of the MCMC algorithmic output on its inputs including both hyper-parameter and starting value irregardless of convergence.



(a) Normal Model (b) Student-t Model (c) 2-Equation Model

Figure 2: Convergence AD and LR Estimates: Estimates for $\frac{\partial \hat{\beta}_1}{\partial b_{0,1}}$ at first iterations of the MCMC algorithms.



(a) Normal Model (b) Student-t Model (c) 2-Equation Model

Figure 3: Stability of AD and LR Estimates: Estimates for $\frac{\partial \hat{\beta}_1}{\partial b_{0,1}}$ at later iterations of the MCMC algorithms.

Note that we do not expect a large variation in the reported estimates $\partial \hat{\boldsymbol{\theta}}_k^{(g)} / \partial \boldsymbol{\eta}_0$ as the AD approach is known to produce sensitivity estimators with lower standard error compared to the bumping or likelihood ratio type approaches in the context of classical simulations. Here we also observe much lower variance in our sensitivity estimates compared to the sensitivities computed using the Perez-type LR approach even in the linear model with simulated data.

4.3 Prior Sensitivity and Sample Size

We can use the AD approach to illustrate how the effect of $\boldsymbol{\eta}_0$ changes with an increase in the sample size.As the posterior distribution combines information from the prior and the data

(likelihood), it is well known that the effect of prior assumptions depends on sample size. The more information about the model parameters is provided in the data, the less is the impact of the prior hyper-parameters on the posterior estimates. In order to demonstrate how the effect of η_0 on posterior inference changes with an increase data in each of the three models, we have estimated each model with sample sizes ranging from 100 to 1000. We consider a summary (overall) measure of the sensitivities of the posterior estimate $\widehat{\theta}_k$ with respect to all prior hyper-parameters in η_0 based on the Euclidean Norm of the gradient vector $\| \frac{\partial \theta_k}{\partial \eta_0} \|$. This

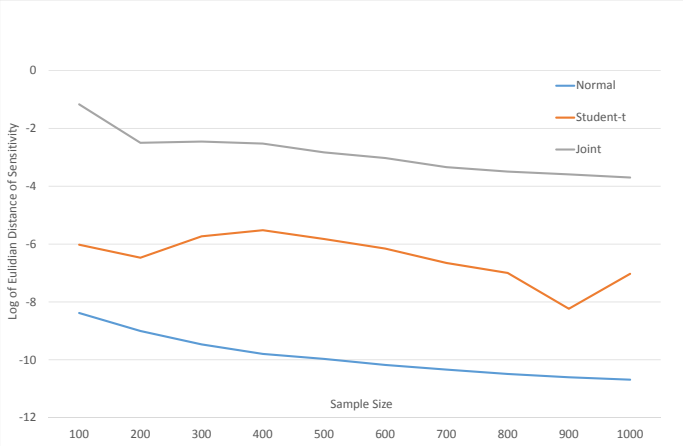


Figure 4: Log of Euclidean Distance of the sensitivities for β_1 under different sample sizes for different models.

log sensitivity measures for the intercept under different sample sizes are shown in Figure 4 for each of the three models. As expected, we observed the fastest decrease in these sensitivities in the simulated data experiment with the linear model. The joint model sensitivities decrease mostly until a sample size of a thousand. In the student-t model we do see an decreasing trend, but also observe some fluctuations in the sensitivities.

5 Illustrative Example

In this section we illustrate the use of the new methods for a comprehensive input sensitivity analysis for both prior robustness and convergence in the context of a widely studied problem in empirical labour economics regarding the earnings gain from an additional year of schooling on Earnings. It is not easy to identify the direct effects from schooling as it is well known that the schooling decision depends on factors unobserved in the data, such as ability and determination, that will also directly effect earnings. Thus, the true causal effect of the schooling decision without contamination from these unobserved confounder effects cannot be estimated by simply regressing schooling on earnings but the econometric analysis has to take into account this endogeneity of the schooling variable. Following previous work we rely on an identification strategy that exploits a so-called instrumental variable based on a change in the school leaving minimum age (Oreopoulos (2006), Oreopoulos (2008), Devereux and

Hart (2010), Chib and Jacobi (2016)) that affected schooling but had no direct effect on earnings. We follow previous work and exploit the 1947 increase in the school leaving minimum age from age 14 to age 15. The data for analysis comes from the UK General Household Surveys using a sample of 47016 males. 75% of the sample males turned 14 after the change in the school leaving minimum age. The average age at survey time is 47.

We proceed with Bayesian IV approach to control for endogeneity of the schooling decision that is based on a joint/simultaneous model for earnings and schooling

$$y_i = \boldsymbol{\beta}_1 + s_i\boldsymbol{\beta}_2 + x'_{yobi}\boldsymbol{\beta}_3 + x'_{agei}\boldsymbol{\beta}_4 + \epsilon_i$$
$$s_i = \gamma_1 + x_{li}\gamma_2 + x'_{yobi}\boldsymbol{\gamma}_3 + x'_{agei}\boldsymbol{\gamma}_4 + u_i$$

where y_i as the log weekly earnings, s_i the age a subject left school, x_{yobi} contains a set of cohort control variables based on the birth year, x_{agei} and finally x_{li} as an indicator for the increase in the compulsory school leaving age. To account for unobserved confounders the error terms follow a joint normal distribution

$$(\epsilon_i, u_i) \sim N_2\left(\mathbf{0}, \boldsymbol{\Sigma} = \begin{pmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{12} & \sigma_{22} \end{pmatrix}\right)$$

where $\boldsymbol{\Sigma}$ is a full covariance matrix. This ensures that α_1 captures the returns to an additional year of schooling.

We proceed under standard assumptions of conjugate Normal priors for $\boldsymbol{\beta} = (\boldsymbol{\beta}_1, \boldsymbol{\beta}_1, \boldsymbol{\beta}_3, \boldsymbol{\beta}_4)$ and $\boldsymbol{\gamma} = (\gamma_1, \gamma_2, \boldsymbol{\gamma}_3, \boldsymbol{\gamma}_4)$ and a Wishart prior for the inverse covariance matrix $\boldsymbol{\Sigma}^{-1}$. The joint prior distribution for the model parameters $\boldsymbol{\theta} = (\boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\Sigma})$ is

$$\pi(\boldsymbol{\theta}|\boldsymbol{\eta}_0) = N_k(\boldsymbol{\beta}|\mathbf{b}_0, \mathbf{B}_0)N_p(\boldsymbol{\gamma}|\mathbf{g}_0, \mathbf{G}_0)W(\boldsymbol{\Sigma}^{-1}\nu_0, \mathbf{R}_0)$$

which depends on the vector of prior hyper-parameters $\boldsymbol{\eta}_0 = (\mathbf{b}_0, \mathbf{B}_0, \mathbf{g}_0, \mathbf{G}_0, \nu_0, \mathbf{R}_0)$. We set the prior mean vectors $\mathbf{b}_0$ and $\mathbf{g}_0$ as vectors of zero, which is a common choice in empirical work. The prior covariances $\mathbb{B}_0$ and G_0 are specified as diagonal matrices with diagonal elements set at 100 to give us fairly uninformative prior. Finally, we set R_0 as an identity matrix and $\nu = 5$.

The posterior distribution of the model parameters $\pi(\boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\Sigma}|Y)$ defined by the model and priors above can then be estimated via a 3-step Gibbs sampler (see Appendix 7.4) with starting values $\boldsymbol{\theta}^0 = (\boldsymbol{\Sigma}^{(0)}, \gamma^0)$.

1. Draw $\boldsymbol{\beta}^{(g)}$ from $\pi(\boldsymbol{\beta}|Y, \boldsymbol{\gamma}^{(g-1)}, \boldsymbol{\Sigma}^{(g-1)}) = N_k(\mathbf{b}^{(g)}, \mathbf{B}^g)$
2. Draw $\boldsymbol{\gamma}^{(g)}$ from $\pi(\boldsymbol{\gamma}|Y, \boldsymbol{\beta}^{(g)}, \boldsymbol{\Sigma}^{(g-1)}) = N_k(\mathbf{g}^{(g)}, \mathbf{G}^g)$
3. Draw $\boldsymbol{\Sigma}^{-1^{(g)}}$ from $\pi(\boldsymbol{\Sigma}^{-1}|Y, \boldsymbol{\beta}^{(g)}, \boldsymbol{\gamma}^{(g)}) = W(\nu_1^{(g)}, \mathbf{R}_1^{(g)})$

Sensitivities for the output from this Gibbs samplers are computed based on the methods described in Sections 3.1 and 3.2.

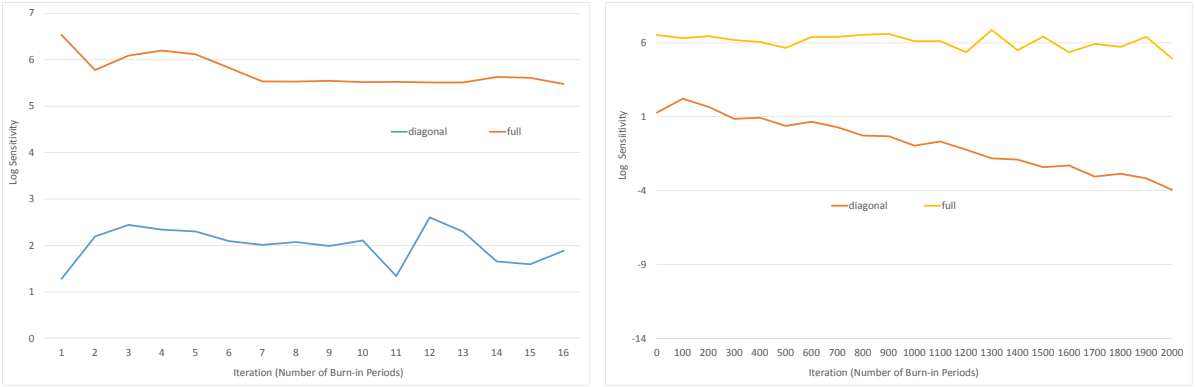
5.1 Convergence

We start the sensitivity analysis by looking at convergence via a range of measures based on the starting value sensitivities of the draws. Here we initialize γ^0 is initialized as zero vector. For the covariance matrix Σ^0 we consider two common starting value choices, a full matrix, where non-zero off-diagonal terms reflect confounding on unobservables, and a diagonal matrix

$$\Sigma^0 = \begin{pmatrix} 1 & 0.2 \\ 0.2 & 1 \end{pmatrix}, \ \Sigma^0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

to illustrate the potential effects of starting value choices. The first choice implies that the earnings and schooling decisions are both affected by some common unobserved factors that link the update the corresponding model parameters β and γ. Under the latter choice the decisions are unrelated and the two models not connected.

These two options lead to very different convergence behavior in the chain. The two graphs in Figure 5 below show the maximum absolute value of the elements in the Jacobian matrices of the starting value sensitivities $J_{\theta^{(g)}}(\theta^0)$ as the chain evolves. The left focuses on the initial iterations while the right graph looks at the first 2000 iterations. The graphs are in the natural logarithms to illustrate the very small elements. At the initial 15 iterations the largest observed



(a) Sensitivities at initial iterations

(b) Evolution of Sensitivities

Figure 5: Maximum of starting value sensitivities from Jacobian matrix of all partial derivatives of draw θ^g with respect to all starting values in θ^0 at iteration g under full and diagonal choices of Σ^0.

sensitivities are quite high across both MCMC chains. Staring with a full covariance matrix we observe log values above 5 which implies sensitivities around 250. Starting with a diagonal matrix, we observed much lower sensitivity values, but still not negligible with values above 7. Figure 5 (b) shows that in the latter case the sensitivities drop-off steadily over the first 2000 iterations to a log values of -4 (0.02). In the case of the diagonal matrix 2000 Burn-in Iterations would not be sufficient for the effects of the starting value to disappear. If we specify $\Sigma^{(0)}$ as a full covariance matrix, we observe very slow convergence in the case of the real data application with some very large sensitivities up to 10,000 iterations.

The reason for the faster convergence is that setting Σ^0 as a diagonal matrix implies no connection between the two equations (β and γ) at the starting of the sampler. In other words

$\boldsymbol{\beta}^1$ is updated without reference to the endogenous regressor model and therefore will not depend on $\boldsymbol{\gamma}^0$. If on the other hand we specify $\boldsymbol{\Sigma}^0$ as a full covariance matrix, this is not the case. The reason for the slow drop is that we "carry" along a term for the dependence from the first iteration, all though its impact will diminish as the chain continues (G increases) and as dependencies decrease. At iteration g, the starting value sensitivities for the parameter vector $\boldsymbol{\theta}$ with respect to the all starting values, $\boldsymbol{\theta}^0$, is given by expression

$$J_{\boldsymbol{\theta}^g}(\boldsymbol{\theta}^0) = \prod_{i=2}^{g} J_{\boldsymbol{\theta}^i}(\boldsymbol{\theta}^{i-1}) \times J_{\boldsymbol{\theta}^1}(\boldsymbol{\theta}^0)$$

where the last expression is a $K \times l$ Jacobian matrix that contains sensitivities of the first draw of $\boldsymbol{\theta}$ with respect to all starting values. The terms inside the product are $K \times K$ matrices, that represent the dependence structure embedded in the kernel. In the case of a diagonal $\boldsymbol{\Sigma}^0$ matrix, the matrix $J_{\boldsymbol{\theta}^1}(\boldsymbol{\theta}^0)$ will be sparse matrix. This immediately eliminates the dependence of the draws on the initial gamma values. More importantly, the diagonal matrix indicates the starting value is within a region of low auto-correlation, hence it allows for a faster mixing without clustering around the initial region. Figure 7 shows lower autocorrelations in the draws under the diagonal $\boldsymbol{\Sigma}^0$ compared to those under a full $\boldsymbol{\Sigma}^0$ in Figure 6.

If we proceed with the full version of the starting matrix, a longer Burn-in period is required. The first rows in Table 3 below show two summary measures of starting value dependence in term of the maximum and average starting values sensitivities with respect to the complete vector of starting values ($\boldsymbol{\gamma}^0$ and $\boldsymbol{\Sigma}^0$). Each measure is reported at the first iteration following Burn-in Periods between 2,000 and 30,000 for the complete parameter vector $\boldsymbol{\theta}$ and the first two elements in the $\boldsymbol{\beta}$ vector, the intercept and the coefficient on the endogenous regressor.

		Convergence Measures based on Starting Value Sensitivities						
		$B = 2,000$	$B = 4,000$	$B = 6,000$	$B = 8,000$	$B = 10,000$	$B = 20,000$	$B = 30,000$
$\boldsymbol{\theta}$	$max(\lvert J_{\boldsymbol{\theta}}^{(B+1)} \rvert)$	140.378	547.098	1049.58	97.924	0.13307	7.95221E-04	7.8078E-04
	$Aver(J_{\boldsymbol{\theta}}^{(B+1)})$	5.68456	17.9786	23.6727	3.0054	3.02134E-03	3.6818E-06	3.70973E-06
β_1	$\max(\lvert \frac{\partial \beta_1^{(B+1)}}{\partial \eta_0} \rvert)$	94.70852	227.1879	297.2609	0.197087	0.032546	9.39954E-07	3.92757E-06
	$Aver(\frac{\partial \beta_1^{(B+1)}}{\partial \eta_0})$	-18.1531	-43.7185	-50.3356	0.033583	0.00563	-6.71396E-08	-2.8054E-07
β_2	$\max(\lvert \frac{\partial \beta_2^{(B+1)}}{\partial \eta_0} \rvert)$	29.38193	130.1667	206.4557	83.95459	0.40879	1.33207E-05	7.89301E-06
	$Aver(\frac{\partial \beta_2^{(B+1)}}{\partial \eta_0})$	-5.66511	-24.1972	-35.1668	14.23689	0.070612	9.51476E-07	-5.63786E-07

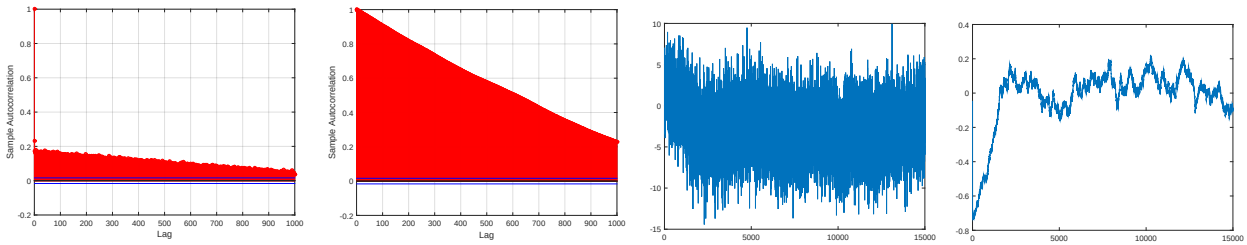
Table 3: Evolution of Starting Value Sensitivities (summary measures) between 2,000 and 30,000 Burn-in Periods with full $\boldsymbol{\Sigma}^{(0)}$ matrix.

All measures show that starting values sensitivities remain high and somewhat volatile over the first 8,000 iterations before they start to decrease. [4] Over 10,000 iterations the overall

[4]The fact that the sensitivity to the initial state can actually increase is a little curious. One possible explanation is that for certain densities and certain inputs, inverse cumulative sampling algorithms have very large dependence of output on input. So if the Markov chain enters a domain where such dependence is present this

sensitivities measures based on all elements in $\boldsymbol{\theta}$ are reasonably small with the average sensitivity at 0.003 and the maximum sensitivity close to 0.1. Both measures drop further, and drop further off to 3.7E-06 and 0.04 at 20,000 iterations respectively. Increasing the burn-in period further does not appear to yield any additional decrease in starting value sensitivities. The overall convergence measures presented in the remaining rows in Table 3 for β_1 and β_2 reflect large staring value sensitivities initially but we see a faster drop-off for β_1 with very low sensitivities after 6000 iterations. For β_2 we still see high values at 8000 iterations. Since both measures are reasonably small at 10,000 iterations, we set B to 10,000 for the posterior analysis reported in the next section.

While no overall convergence measure for the chain currently exists in the literature , we can look at the autocorrelation and the trace plots of particular draws to help assess the mixing of the chain and convergence. In Figure 6we present graphs of each for two elements in the $\boldsymbol{\beta}$ vector, the intercept and the coefficient on the endogenous schooling variable. The trace



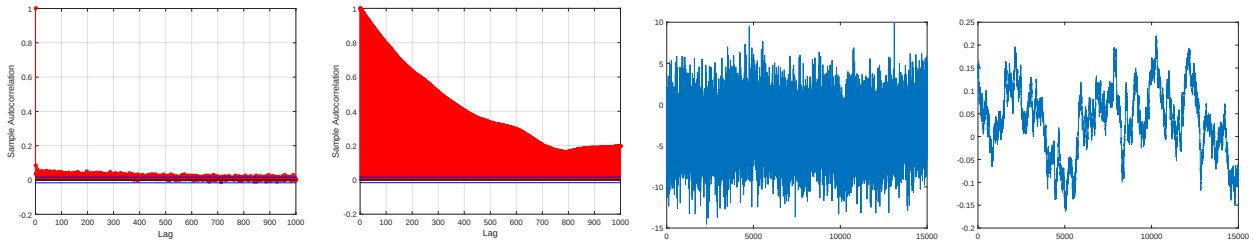
(a) Autocorrelation: β_1 (b) Autocorrelation: β_2 (c) Trace plot: β_1 (d) Trace plot: β_2

Figure 6: Autocorrelations up to lag 1000 and trace plots when $\boldsymbol{\Sigma}^0$ is set as full matrix.

plots also reflect the slow convergence of the chain, but would suggest convergence has been reached even before 5000 iterations. Both the trace plots and the autocorrelation plots show that have good mixing in the for β_1, but poor mixing for β_2. This is also reflected in the trace plots where we see lots of clustering in the β_2 draws.

For both parameters we observe lower autocorrelations and better mixing under the diagonal matrix choice for $\boldsymbol{\Sigma}^0$. This results reflects the pattern of the lower convergence measured as higher starting value sensitivities as presented in Figure 5. We would expect to see higher autocorrelation of draws for in the draws from the algorithm starting with full diagonal matrix as slower mixing of the chain reflected in high autocorrelation of the draws would lead to slower convergence of the chain. Figure 7 presents the trace plots and autocorrelations for β_1 and β_2. In each case the plots reflect better mixing and faster convergence than in the plots in Figure 6 for the algorithm run with the full matrix for $\boldsymbol{\Sigma}_0$. We observe inefficiency factors above 400 for β_1 and two elements in the Σ matrix, σ_{11} and σ_12. For the remaining parameters the inefficiency factors are close to 1 (complete set of results available upon request).

may increase the numbers. For an example of such large dependence in the context of the Heston model see Chan, Joshi, Zhu (2015).



(a) Autocorrelation: β_1 (b) Autocorrelation: β_2 (c) Trace plot: β_1 (d) Trace plot: β_2

Figure 7: Autocorrelations up to lag 1000 and trace plots when $\mathbf{\Sigma}^0$ is set as diagonal matrix.

5.2 Posterior Estimates and their Prior Robustness

Table 4 presents the estimates of the posterior means and standard deviations. Of particular interest is the posterior mean estimate of β_2 which suggests a 5.6% earnings increase from an additional year of schooling. The estimate of 0.47 for γ_2 reflects a strong effect of the law change on the schooling intake.

To assess the sensitivity of the reported posterior mean estimates to the prior hyper-parameters, the table presents a new measure of the "overall" prior sensitivity of each posterior mean estimates, $\left\| \frac{\partial \boldsymbol{\theta}_k}{\partial \boldsymbol{\eta}_0} \right\|$ with respect to all prior hyper-parameters that is computed in terms of the Euclidean distance/norm as

$$\| \frac{\partial \boldsymbol{\theta}_k}{\partial \boldsymbol{\eta}_0} \| = \sqrt{\sum_{l=1}^{p} \left(\frac{\partial \boldsymbol{\theta}_k}{\partial \boldsymbol{\eta}_{0,l}} \right)^2}$$

The sensitivity measures range from 0.09 for the intercept γ_1, followed by 0.045 for the

	Posterior Mean Estimates, Standard Deviations and Total Sensitivities									
$\boldsymbol{\beta}$ (Mean)	-3.6448	0.0583	0.5866	-1.6811	0.2264	-0.0119	0.0350	-0.0253	0.0119	-0.0017
$\boldsymbol{\beta}$ (STD)	1.3992	0.0339	0.1269	0.4259	0.0625	0.0034	0.0099	0.0140	0.0062	9.0E-04
$\| \frac{\partial \boldsymbol{\beta}}{\partial \boldsymbol{\eta}_0} \|$	0.0488	0.0053	0.0034	0.0108	0.0015	7.92E-05	9.6E-04	0.0017	7.4E-04	1.1E-04
$\boldsymbol{\gamma}$ (Mean)	8.7855	0.4696	0.4589	-1.4171	0.1933	-0.0097	-0.0346	0.1058	-0.0495	0.0075
$\boldsymbol{\gamma}$ (STD)	2.9700	0.0381	0.2707	0.9095	0.1335	0.0072	0.0211	0.0265	0.0116	0.0017
$\| \frac{\partial \boldsymbol{\gamma}}{\partial \boldsymbol{\eta}_0} \|$	0.0928	0.0012	0.0084	0.0282	0.0041	2.2E-04	3.6E-04	5.2E-04	2.3E-04	3.1E-05
$\mathbf{\Sigma}$ (Mean)	0.2628	0.1245	1.3335							
$\mathbf{\Sigma}$ (STD)	0.0093	0.0453	0.0087							
$\| \frac{\partial \mathbf{\Sigma}}{\partial \boldsymbol{\eta}_0} \|$	0.0013	0.0070	2.1E-05							

Table 4: Posterior Estimates and Overall Prior Sensitivity Measures of Model Parameters

intercept β_1 to 2.1E-05, for the variance estimate σ_{22}. Both the magnitude of the overall sensitivity measures and the magnitude of the posterior estimates varies across the model parameters. These are absolute magnitudes. We can also look at relative numbers that we can scale the overall norm measure in terms of the absolute value of the posterior mean estimate. For example, we would obtain $\| \frac{\partial \beta_1}{\partial \boldsymbol{\eta}_0} \| / -3.6448 = -0.01339$ for the intercept term and

$\|\frac{\partial \beta_2}{\partial \boldsymbol{\eta}_0}\|/0.0583 = 0.0909$ for the coefficient on schooling.

Table 5 reports the Jacobian matrices $J_{E(\beta_1|Y)}$ and $J_{E(\beta_2|Y)}$ with the derivatives of the posterior mean estimates with respect to the prior hyper-parameters. The derivatives with respect to the off-diagonal elements in $\mathbf{B}_0$ and $\mathbf{G}_0$ have been omitted for space reasons. As

	Prior Hyper-Parameter Sensitivities for β_1								and β_2	
$\frac{\partial \beta_1}{\partial \mathbf{b}_0}$	2.0E-02	-1.1E-04	-1.6E-03	5.5E-03	-8.0E-04	4.3E-05	-1.5E-05	2.3E-05	-8.8E-06	1.1E-06
$\frac{\partial \beta_1}{\partial \mathbf{B}_{kk,0}}$	-7.1E-04	-6.2E-08	-9.6E-06	-9.2E-05	-1.8E-06	-5.1E-09	-5.2E-09	-5.8E-09	-1.0E-09	-1.8E-11
$\frac{\partial \beta_1}{\partial \mathbf{g}_0}$	8.0E-03	-2.3E-05	-7.3E-04	2.4E-03	-3.5E-04	1.9E-05	-5.1E-06	5.6E-06	-1.8E-06	1.4E-07
$\frac{\partial \beta_1}{\partial \mathbf{G}_{kk,0}}$	7.1E-04	-1.1E-07	-3.3E-06	-3.4E-05	-6.8E-07	-1.8E-09	1.8E-09	6.1E-09	8.7E-10	1.1E-11
$\frac{\partial \beta_1}{\partial \mathbf{\Sigma}_0}$	-1.4E-05	-4.3E-02	4.0E-03							
$\frac{\partial \beta_2}{\partial \mathbf{b}_0}$	-1.1E-04	1.3E-05	-6.9E-06	2.2E-05	-3.0E-06	1.5E-07	2.4E-06	-4.2E-06	1.8E-06	-2.6E-07
$\frac{\partial \beta_2}{\partial \mathbf{B}_{kk,0}}$	3.8E-06	7.5E-09	-4.1E-08	-3.7E-07	-6.9E-09	-1.9E-11	8.2E-10	1.0E-09	2.2E-10	4.4E-12
$\frac{\partial \beta_2}{\partial \mathbf{g}_0}$	-1.8E-06	2.8E-06	3.5E-08	-8.4E-08	9.3E-09	-3.8E-10	8.6E-07	-1.2E-06	5.4E-07	-7.4E-08
$\frac{\partial \beta_2}{\partial \mathbf{G}_{kk,0}}$	-1.6E-07	1.3E-08	1.3E-10	6.2E-10	-5.9E-13	-3.8E-14	-3.1E-10	-1.3E-09	-2.7E-10	-5.6E-12
$\frac{\partial \beta_2}{\partial \mathbf{\Sigma}_0}$	9.7E-07	5.2E-03	-4.9E-04							

Table 5: Posterior Estimates and Overall Prior Sensitivity Measures of Model Parameters

expected we see slightly larger sensitivities for β_1 than β_0 which is reflected in the larger overall sensitivity measure for the former reported in Table 4.

A different approach to assess the magnitude of the sensitivities in terms of predicted earnings y^p of a representative individual. Here we predict earnings from the model based on the sample mean values of the covariates and the posterior mean estimates. (Another approach would be to compute the predictions based on parameters draws from the posterior parameter distributions alongside the estimation algorithm.) In Table 6 below we present the "raw" sensitivities of predicted earnings with respect to $\mathbf{b}_0$ and $\mathbf{g}_0$ as well the implied changes from a 1% posterior mean change in the element. Note that the latter number is usually lower as we pre-multiply the derivative with a number below zero, i.e. $\triangle_{b_{0,1}} y^p = [0.01\hat{\beta}_1] \times (\partial y^p / \partial b_{0,1})$. The derivative is a local measure so we use to assess small changes. The reason for using a 1% change in the posterior mean rather than the prior hyper-parameter itself is that it is common to set those at zero, in particular for prior means. Among this set of results we see the largest effects from changes in the prior means for the intercept terms followed by prior means on the birth-cohort effects. We have also reported the estimates for the sensitivities with respect to the starting values of $\boldsymbol{\gamma}_0$ in Table 6. We note that some of these sensitivities are of non-trivial size.

6 Discussion

In this paper we have introduced a general and flexible numerical approach to undertake a comprehensive input sensitivity analysis for output from Bayesian MCMC analysis to assess robustness of (i) posterior estimates to all prior hyper-parameters and (ii) MCMC parameter

	Sensitivities for Predicted Earnings									
	Partial Derivatives									
$\frac{\partial y^p}{\partial \mathbf{b}_0}$	0.0226	-9.1E-05	-0.0019	0.0065	-0.0009	5.0E-05	-1.0E-05	1.6E-05	-5.2E-06	5.3E-07
$\frac{\partial y^p}{\partial \mathbf{g}_0}$	0.0094	-2.0-05	-0.0009	0.0029	-0.0004	2.2E-05	-3.4E-06	3.0E-06	-5.1E-07	-4.3E-08
$\frac{\partial y^p}{\partial \gamma^0}$	-1.7E-05	-4.5E-06	-0.0013	-0.0001	-0.0012	-0.0036	-0.0002	-0.0006	-0.0008	-0.0035
	Effects from 1% Posterior Mean Changes									
$\triangle_{\mathbf{b}_0} y^p$	-0.0008	-5.3E-08	-1.1E-05	-0.0001	-2.1E-06	-6.1E-09	-3.6E-09	-3.9E-09	-6.1E-10	-9.0E-12
$\triangle_{\mathbf{g}_0} y^p$	0.0008	-9.19E-08	-3.92E-06	-4.05E-05	-8.1E-07	-2.2E-09	1.2E-09	3.1E-09	2.5E-10	-3.2E-12
$\triangle_{\gamma^0} y^p$	-1.5E-06	-2.1E-08	-5.8E-06	1.6E-06	-2.2E-06	3.5E-07	6.5E-08	-6.3E-07	3.8E-07	-2.7E-07

Table 6: Partial derivatives and approximate changes from 1% changes in hyper-parameters.

draws with respect to all starting values (chain convergence). Under the approach sensitivities for all MCMC output, including draws and statistics, with respect to all inputs can be computed via embedded sensitivity algorithms that are based on algorithmic differentiation (AD) methods. This approach to estimate the Jacobian matrices of the first order derivatives of MCMC output can be viewed as a small-size limit of the bumping method. The extensions to higher order derivatives, such as the Hessian matrix of the posterior statistic, is clearly feasible (Joshi and Zhu, 2016b). Whilst it has some similarities with symbolic differentiation, it works by differentiating an algorithm at the level of elementary operations rather than by differentiating formulas.

The introduced set of methods based on the forward mode AD approach for sensitivity analysis offers researcher an new tool to improve current prior robustness and algorithm convergence analysis. Different from existing methods, the approach enables researchers to compute sensitivities with respect to both the complete set of prior hyper-parameters as well as the complete set of starting values required to initiate the MCMC chain. Prior robustness measures based on the Jacobian of posterior statistics with respect of prior hyper-parameters allow researchers to gain a better understanding of prior to posterior dependence within a parametric family. The extension to a wider range of robustness measures is possible via distributional derivatives which we will explore in future research.

As part of the algorithmic differentiation the complete set of partial derivatives (all inputs) is computed for all parameters (draws) at every iteration of the chain. These results are used to compute the partial derivatives/sensitivities of functions of the parameters, in particular statistics such as posterior means and variances. Importantly, the partial derivatives of the parameter draws with respect to the starting values allows us to monitor the convergence of the chain. The paper introduces a new set of direct measures of chain convergence based on the starting value sensitivities of all parameters that will help to assess chain convergence directly and set a reasonable burn-in period. The measures will complement existing measures of algorithm efficiency based on the autocorrelations of the draws.

An essential feature of the introduced methods is that the computational cost of the forward mode AD increases linearly with the number of input parameters of interest and the number of operations in the estimation algorithm. As some operations are very fast increases in computation time are often less than linear as illustrated in our examples. This makes the approach feasible for applications with high dimensional parameter and input parameter spaces. Clearly

the computational burden could be lowered by computing only a subset of the possible sensitivities that are particularly relevant in a specific context. In this paper we have aimed to introduce the approach for the general set of prior sensitivities that are of relevance for prior robustness and convergence to illustrate the potential of this approach. We have also proposed some summary sensitivity measures based on the large set of input sensitivities.

Finally, to test and illustrate the new methods, we have implemented sensitivity analysis both with respect to the prior hyper-parameters and MCMC chain starting values in the context of the Gibbs estimation of linear and student-t regression models as well as joint models in simulated and real data experiments. We also show that our sensitivity estimates of the posterior means with respect to the hyper-parameters are comparable to those obtained via the likelihood ratio approach, but are faster to converge and more stable. While overall prior parameters sensitivities are small in our real data applications, partially a results of large sample sizes, the sensitivities vary both across parameters within a model and across models. The developed convergence measures shows the fastest convergence of the Gibbs sampler for the linear model and the slowest convergence for the Gibbs sampler of the Joint Model. The latter has a particularly slow convergence if a full covariance matrix is chosen to start of the chain, requiring 10,000 iterations as a burn- in period.

A potential challenge of the AD based approach is the requirement of the continuity of the underlying MCMC algorithm. In this paper we show how to address issues of discontinuities that arise in the context of common random variable updates in Gibbs algorithms, the Gamma and Wishart updates. Discontinuities arising from a Metropolis Hastings updates will be addressed in a separate paper.

7 Technical Appendix

7.1 Automatic Differentiation via Forward Mode

We now discuss in more detail how the automatic differentiation is carried out. A program will start with a number of inputs, $z_1, z_2, \ldots, z_M$, for some M. In our context, these are all the starting parameters of the model including the hyper-parameters, the starting point of the chain and the data. So we could have

$$z_1 = \theta_1^{(0)},\ z_2 = \theta_2^{(0)},\ \ldots,\ z_p = \theta_p^{(0)},$$

and

$$z_{k+1} = \eta_0^1,\ \ldots,\ z_{k+j} = \eta_0^j,$$

and similarly for the data. Our program will start with these values $\{z_r\}_{r=1,\ldots,M}$ and applies elementary operations from the class introduced in Subsection 2.3 one at at time to already computed values to get a sequence of new values $\{z_r\}_{r=M+1,\ldots,L}$. Thus we have a sequence of operations F_j for $j = k+1, \ldots, L$, and

$$z_j = F_j(z_{j_1}, z_{j_2})$$

for some $j_1, j_2 < j$. For some operations such as exp, F_j will have no dependence on z_{j_2} but it is easier notationally to assume all operations have two inputs.

We now fix a particular input, z_s, for which we want the output's sensitivity. The sensitivity of z_s to itself is, of course, 1 and the sensitivity of z_r to z_s for $r \leq M, r \neq s$, will be zero since they are separate inputs. We compute the sensitivity to z_s of each subsequent z_l for $l > M$ one by one starting with z_{M+1}. We emphasize that at each stage we are computing its numerical value for a given set of inputs. Now suppose we have computed $\frac{\partial z_r}{\partial z_s}$ for $r < j$, and we want to find $\frac{\partial z_j}{\partial z_s}$. We can write

$$\frac{\partial z_j}{\partial z_s} = \frac{\partial F_j}{\partial z_s}(z_s) = \frac{\partial F_j}{\partial z_{j_1}}\frac{\partial z_{j_1}}{\partial z_s} + \frac{\partial F_j}{\partial z_{j_2}}\frac{\partial z_{j_2}}{\partial z_s}. \tag{7.1}$$

The values $\partial F_j/\partial z_{j_1}$ and $\partial F_j/\partial z_{j2}$ are very easily computable since F_j is an elementary operation. For example, if F_j is $+$ then they are both 1 and

$$\frac{\partial z_j}{\partial z_s} = \frac{\partial z_{j_1}}{\partial z_s} + \frac{\partial z_{j_2}}{\partial z_s}. \tag{7.2}$$

If $F_j(z_{j_1}, z_{j_2}) = \mathsf{exp}(z_{j_1})$ then

$$\frac{\partial z_j}{\partial z_s} = \mathsf{exp}(z_{j_1})\frac{\partial z_{j_1}}{\partial z_s} = z_j\frac{\partial z_{j_1}}{\partial z_s}. \tag{7.3}$$

We therefore rewrite our computer program to compute the numerical value $\partial z_j/\partial z_s$ at the same time as z_j and to store the result alongside z_j. We drop its storage at the same that we drop the storage of z_j. Once z_j has been done, we move on to z_{j+1} and so on. When the

program terminates in a value z_L, it will have $\partial z_L/\partial z_s$ alongside and the derivative has been computed. Depending on the class of elementary operations, it is usually possible to automate this process.

Note that the analysis above does not require us to have only one output and if the original algorithm outputted several values $z_{L-w+1}, z_{L-w+2}, \ldots, z_L$ then it is simply a question of reading off the last w derivatives. If we want to compute the sensitivity for multiple inputs, then we can simply repeat the above process for each one of them.

We have sketched out how to construct the differentiated algorithm and it is important to realize that this is a practical result. Any continuous computation implementable in a computer can be broken down and differentiated in this way.

7.2 More on Implementing AD for Linear Models

Multidimensional setup

For the actual applications, β is often a vector. As a consequence, we need compute and update a gradient vector for each $g(\beta_k)$ with more state variables including each elements of β^g, $\mathbf{b}^{(g)}$ and $\mathbf{B}(g)$. We also generate $\beta^{(g)}$ from multi-variate normal distribution with parameter $\mathbf{b}^{(g)}$ and $\mathbf{B}^{(g)}$ by decomposing $\mathbf{B}^{(g)}$ via the Cholesky decomposition. Consider a matrix $\Sigma = AA^T$, we want to obtain the sensitive of A with respect to a real parameter θ

$$\frac{\partial \Sigma}{\partial \theta} = \frac{\partial A}{\partial \theta} A^T + A \frac{\partial A^T}{\partial \theta}$$

The trick is to left-multiply by A^{-1} and right-multiply by A^{-T}

$$A^{-1}\frac{\partial \Sigma}{\partial \theta} A^{-T} = A^{-1}\frac{\partial A}{\partial \theta} A^T A^{-T} + A^{-1} A \frac{\partial A^T}{\partial \theta} A^{-T} = A^{-1}\frac{\partial A}{\partial \theta} + \frac{\partial A^T}{\partial \theta} A^{-T}$$

The second term is the transpose of the first, meaning it is upper-triangular and has the same diagonal. We can therefore remove the second term by applying a function Φ to both sides, where Φ takes the lower-triangular part of a matrix and halves its diagonal:

$$\Phi_{i,j}(A) = \begin{cases} A_{i,j} \text{ if } i > j \\ 0.5 A_{i,j} \text{ if } i = j \\ 0 \text{ if } i < j. \end{cases}$$

Hence

$$\frac{\partial A}{\partial \theta} = A\Phi(A^{-1}\frac{\partial \Sigma}{\partial \theta} A^{-T}).$$

The proof of this method is in [20]. For the step where we need to compute derivatives of matrix inversion, we could also apply the automatic differentiation to the inversion algorithm, but there is a trick to save the computational cost. Since for a square matrix A depending on real parameter θ

$$A^{-1}A = \mathbb{I},$$

where $\mathbb{I}$ is the identity matrix, differentiating both sides gives us

$$\frac{dA^{-1}}{d\theta}A + A^{-1}\frac{dA}{d\theta} = \mathbf{0} \rightarrow \frac{dA^{-1}}{d\theta} = -A^{-1}\frac{dA}{d\theta}A^{-1}.$$

where $\mathbf{0}$ is the zero square matrix.

Generating Gamma Random Variable

1. *Setup:* $\hat{\alpha} = \alpha - 1$, $b = \frac{\alpha - \frac{1}{6\alpha}}{\hat{\alpha}}$, $m = \frac{2}{\hat{\alpha}}$ *and* $d = m + 2$.
2. *Repeat: generate* $V = U(0,1)$ *and* $V^D = U(0,1)$ *independently, set* $Y = b\frac{V^D}{V}$.
3. *If* $m_1 V - d + Y + \frac{1}{Y} \leq 0$ *accept,*
   else if $m\log(V) - \log(Y) + Y - 1 \leq 0$ *accept,*
4. *until accept,*
5. *return* $Z = \hat{\alpha}Y$.

Thus, a small change in the shape parameter α may induce changes in the above acceptance-rejection decisions.

1. *Setup:* $\hat{\alpha} = \alpha - 1$, $b = \frac{\alpha - \frac{1}{6\alpha}}{\hat{\alpha}}$, $m = \frac{2}{\hat{\alpha}}$ *and* $d = m + 2$.
2. *Repeat: generate* $V = U(0,1)$ *and* $V^D = U(0,1)$ *independently, set* $Y = b\frac{V^D}{V}$.
3. *If* $m_1 V - d + Y + \frac{1}{Y} \leq 0$ *accept,*
   else if $m\log(V) - \log(Y) + Y - 1 \leq 0$ *accept,*
4. *until accept,*
5. *return* $Z = \hat{\alpha}Y$.

Thus, a small change in the shape parameter α may induce changes in the above acceptance-rejection decisions.

7.3 Gibbs Algorithm for Student-t Model

For the linear regression model with student-t error from Section 1, we define $\lambda = \lambda_1, \lambda_2, ..., \lambda_n$ as the vector of the scale parameters for all observations, then we can define the likelihood of the model now also conditioned on the scale parameters as

$$p(y|\boldsymbol{\beta}, h, \lambda) = \prod_{i=1}^{n} N(y_i|x_i\boldsymbol{\beta}, \lambda_i^{-1}h^{-1}) = \prod_{i=1}^{n} \frac{h^{1/2}\lambda_i^{1/2}}{(2\pi)^{1/2}} \exp\{-\frac{1}{2}h\lambda_i(y_i - x_i\boldsymbol{\beta})^2\}$$

where $h = 1/\sigma^2$. To express the likelihood more compactly in matrix form we define the diagonal matrix of scale parameters

$$\Lambda = \begin{pmatrix} \lambda_1 & 0 & 0 & 0 \\ 0 & \lambda_2 & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \lambda_n \end{pmatrix}$$

The likelihood then can be expressed in vector form as the multivariate normal density

$$p(y|\boldsymbol{\beta}, h, \lambda) = N(y|X\boldsymbol{\beta}, h^{-1}\Lambda^{-1}) = \frac{h^{n/2}}{(2\pi)^{n/2}}|\Lambda|^{1/2}\exp\{-\frac{1}{2}h(y - X\boldsymbol{\beta})'\Lambda(y - X\boldsymbol{\beta})\}$$

where $|h\Lambda| = h^n|\Lambda|$ by matrix algebra.

The implied prior distribution of the model parameters is

$$\pi(\boldsymbol{\beta}, h, \lambda) = N(\boldsymbol{\beta}|\boldsymbol{\beta}_0, B_0)\; G(h|\frac{\alpha_0}{2}, \frac{\delta_0}{2})\; \prod_{i=1}^{n} G\left(\frac{\nu}{2}, \frac{\nu}{2}\right)$$

The posterior distribution of the model parameters is

$$\pi(\boldsymbol{\beta}, h, \lambda|Y) \propto \prod_{i=1}^{n} G(\lambda_i|\frac{\nu}{2}, \frac{\nu}{2})\; N_k(\boldsymbol{\beta}|\mathbf{b}_0, \mathbf{B}_0)\, G(h|\frac{\alpha_0}{2}, \frac{\delta_0}{2})\, N(y|X\boldsymbol{\beta}, h^{-1}\Lambda^{-1}).$$

The posterior distribution can be estimated via a Gibbs sampler since we stay within the normal regression framework, i.e. $\pi(\boldsymbol{\beta}|Y, h, \lambda$ is still of a normal form

$$p(\boldsymbol{\beta}|y, h, \lambda) = N(\mathbf{B}^{-1}[hX'\Lambda y + \mathbf{B}_0^{-1}\mathbf{b}_0], \mathbf{B} = [h(X'\Lambda X) + \mathbf{B}_0^{-1}]^{-1})$$

the conditional posterior distribution of the precision h is still a gamma distribution

$$p(h|\boldsymbol{\beta}, \lambda, y) = G\left(\frac{n + \alpha_0}{2}, \frac{(y - X\boldsymbol{\beta})'\Lambda(y - X\boldsymbol{\beta}) + \delta_0}{2}\right)$$

and the conditional posterior distribution of the scale parameters is just a product of independent gamma distribution]

$$p(\lambda|y, \boldsymbol{\beta}, h) = \prod_{i=1}^{n} p(\lambda_i|y_i, \boldsymbol{\beta}, h) = \prod_{i=1}^{n} G\left(\frac{\nu + 1}{2}, \frac{\nu + h(y_i - x_i\boldsymbol{\beta})^2}{2}\right)$$

We summarize the Gibbs' algorithm.

1. **Initialization**: The initial inputs of the algorithm:

   - $\mathbf{B}_0$, the prior covariance matrix of $\boldsymbol{\beta}$,
   - $\mathbf{b}_0$, the prior mean of $\boldsymbol{\beta}$,

- the prior δ_0 and α_0 of the Gamma,
- the prior for λ degree of freedom parameter ν
- $h^{(0)} = \frac{1}{\sigma^{2(0)}}$ and $\boldsymbol{\beta}^0$, the initial starting point of the algorithm
- set counter at $g = 1$

2. **Update vector** λ: Draw $\lambda_i^{(g)}$, $i = 1, ..., n$ from $\pi(\lambda_i|Y, h^{(g-1)}, \boldsymbol{\beta}^{(g-1)}) = G\Big(\frac{\nu_1}{2}, \frac{\nu_2^{(g)}}{2}\Big)$ where
$$\nu_2^{(g)} = \nu + h^{(g-1)}(y_i - x_i\boldsymbol{\beta}^{(g-1)})^2, \nu_1 = \nu + 1.$$

3. **Update vector** $\boldsymbol{\beta}$: Draw $\boldsymbol{\beta}^{(g)}$ from $\pi(\boldsymbol{\beta}|Y, h^{(g-1)}, \lambda^{(g)}) = N_k(\mathbf{b}^{(g)}, \mathbf{B}^g)$ where
$$\mathbf{B}^{(g)} = \Big[h^{(g-1)}X^T\Lambda^{(g)}X + \mathbf{B}_0^{-1}\Big]^{-1}, \mathbf{b}^{(g)} = \mathbf{B}^{(g)}\Big[h^{(g-1)}X^T\Lambda^{(g)}Y + \mathbf{B}_0^{-1}\mathbf{b}_0\Big].$$

4. **Update** h: Draw $h^{(g)}$ from $\pi(h|Y, \boldsymbol{\beta}^{(g)}, \lambda^{(g)}) = G\Big(\frac{\alpha_1}{2}, \frac{2}{\delta^{(g)}}\Big)$ where
$$\delta^{(g)} = \delta_0 + (Y - X\boldsymbol{\beta}_j)^T\Lambda^{(g)}(Y - X\boldsymbol{\beta}^{(g)}), \alpha_1 = \alpha_0 + n.$$

5. **Repeat** steps (2) and (4) for $B + G$ times, where B is the initial burn-in period required for the chain to converge which is discarded and G is the required simulation sample size.

7.4 MCMC Algorithm for Joint Model

Below are the details for the Gibbs algorithm used to fit the joint model defined in Section 4.1 and also used in the illustrative example in Section 5. To express the likelihood in a compact form suitable for an efficient gibbs update, we first define the covariate vectors for the earnings and schooling equations as $x_{yi} = \{const, s_i, x_{yobi}, x_{agei}\}$ and $x_{si} = \{const, x_{li}, x_{yobi}, x_{agei}\}$ then the Covariate Matrix X_i and coefficient vector

$$\mathbf{y}_i = \begin{pmatrix} y_i \\ s_i \end{pmatrix} \quad , \quad X_i = \begin{pmatrix} x'_{yi} & \mathbf{0} \\ \mathbf{0} & x'_{si} \end{pmatrix} \quad , \quad \boldsymbol{\delta} = \begin{pmatrix} \boldsymbol{\beta} \\ \boldsymbol{\gamma} \end{pmatrix}$$

where $\boldsymbol{\beta} = (\boldsymbol{\beta}_1, \boldsymbol{\beta}_1, \boldsymbol{\beta}_3, \boldsymbol{\beta}_4)$ is a $k \times 1$ vector and $\boldsymbol{\gamma} = (\gamma_1, \gamma_2, \boldsymbol{\gamma}_3, \boldsymbol{\gamma}_4)$ a $p \times 1$ vector. The likelihood under this model can then be compactly expressed as

$$f(y|\boldsymbol{\delta}, \boldsymbol{\Sigma}) = \prod_{i=1}^{n} N(X_i\boldsymbol{\delta}, \boldsymbol{\Sigma})$$

We assume that the inverse of the covariance matrix follows a prior Wishart distribution $\boldsymbol{\Sigma}^{-1} \sim W(\nu_0, \mathbf{R}_0)$ with degree of freedoms ν_0 and scale matrix $\mathbf{R}_0$ which is defined as follows

$$f(\boldsymbol{\Sigma}^{-1}|\nu_0, \mathbf{R}_0) \propto \frac{|\boldsymbol{\Sigma}|^{-\frac{(\nu_0 - k - 1)}{2}}}{|\mathbf{R}_0|^{\nu_0/1}} \exp\left\{-\frac{1}{2}tr(\mathbf{R}_0^{-1}\boldsymbol{\Sigma}^{-1})\right\}$$

where tr refers to the trace of the matrix defined as the sum of the diagonal elements sot hat the expression in the exponential is a scalar. This assumption of a conditionally conjugate prior for $\mathbf{\Sigma}$ allows for the estimation of the model parameters via a standard Gibbs sampler as the full condition distribution of $\mathbf{\Sigma}^{-1}$ is a Wishart distribution $f(\mathbf{\Sigma}^{-1}|\nu_1, R_1)$ with the updated parameter given by

$$\nu_1 = \nu_0 + n \ , \ \mathbf{R}_1 = \Big[\mathbf{R}_0^{-1} + \sum (\mathbf{y}_i - X_i\boldsymbol{\delta})(\mathbf{y}_i - X_i\boldsymbol{\delta})'\Big]^{-1}$$

Further, assuming normal priors for the coefficient vectors $\boldsymbol{\beta} \sim N(\mathbf{b}_0, \mathbf{B}_0)$ and $\boldsymbol{\gamma} \sim N(\mathbf{g}_0, \mathbf{G}_0)$, these can be updated in two separate Normal Gibbs steps by exploiting the marginal-conditional decomposition of the bivariate normal likelihood. For the update of $\boldsymbol{\beta}$ we express

$$f(y_i, s_i) = f(y_i|s_i)f(s_i)$$

where according to the properties of the bivariate normal the conditional distribution is given by

$$f(y_i|s_i) = N(x'_{yi}\boldsymbol{\beta} + \sigma_{12}\sigma_{22}^{-1}(s_i - x'_{si}\boldsymbol{\gamma}), \omega_{11})$$

where the conditional variance of y_i is $\omega_{11} = \sigma_{11} - \sigma_{12}\sigma_{22}^{-1}$. From here it follows that the full conditional distribution of $\boldsymbol{\beta}$, $\pi(\boldsymbol{\beta}|y, s, \boldsymbol{\gamma}, \mathbf{\Sigma}) = N(B, \mathbf{B})$ where

$$\mathbf{B} = \Big[\mathbf{B}_0^{-1} + \omega_{11}^{-1}\sum x_{yi}x'_{yi}\Big]^{-1}, \mathbf{b} = \mathbf{B}\Big[\mathbf{B}_0^{-1}\mathbf{b}_0 + \omega_{11}^{-1}\sum x_{yi}(y_i - \sigma_{12}\sigma_{22}^{-1}(s_i - x'_{si}\boldsymbol{\gamma}))\Big].$$

Similarly, assuming $\boldsymbol{\gamma} \sim N(\mathbf{g}^0, \mathbf{b}_0)$ and expression the joint likelihood as

$$f(y_i, s_i) = f(s_i|y_i)f(y_i)$$

where according to the properties of the bivariate normal the conditional distribution is given by

$$f(s_i|y_i) = N(x'_{si}\boldsymbol{\gamma} + \sigma_{12}\sigma_{11}^{-1}(y_i - x'_{yi}\boldsymbol{\beta}), \omega_{22})$$

where the conditional variance of s_i is $\omega_{22} = \sigma_{22} - \sigma_{12}\sigma_{11}^{-1}$. From here it follows that the full conditional distribution of $\boldsymbol{\gamma}$, $\pi(\boldsymbol{\gamma}|y, s, \boldsymbol{\beta}, \mathbf{\Sigma}) = N(\mathbf{g}, \mathbf{G})$ where

$$\mathbf{G} = \Big[\mathbf{G}_0^{-1} + \omega_{22}^{-1}\sum x_{si}x'_{si}\Big]^{-1}, \mathbf{g} = \mathbf{G}\Big[\mathbf{G}_0^{-1}\mathbf{g}_0 + \omega_{22}^{-1}\sum x_{si}(s_i - \sigma_{12}\sigma_{11}^{-1}(y_i - x'_{yi}\boldsymbol{\beta}))\Big].$$

We summarize the three step Gibbs' algorithm below

1. **Initialization**: The initial inputs of the algorithm:

   - $\mathbf{b}_0$, $\mathbf{B}_0$: the prior mean and covariance matrix of $\boldsymbol{\beta}$,
   - $\mathbf{g}_0$, $\mathbf{G}_0$, the prior mean and covariance matrix of $\boldsymbol{\gamma}$,
   - the prior parameters ν_0 and $\mathbb{R}_0$ of the Wishart prior for $\mathbf{\Sigma}^{-1}$,
   - the initial starting points of the algorithm $\mathbf{\Sigma}^{(0)} : \{\sigma_{11}^0, \sigma_{12}^0, \sigma_{22}^0\}$ (and $\omega_{11}^0 = \sigma_{11}^0 - \sigma_{12}^0\sigma_{22}^{0-1}$, $\omega_{22}^0 = \sigma_{22}^0 - \sigma_{12}^0\sigma_{11}^{0-1}$)

- and $\boldsymbol{\gamma}^0$,
- set counter at $g = 1$

2. **Update vector** $\boldsymbol{\beta}$: Draw $\boldsymbol{\beta}^{(g)}$ from $\pi(\boldsymbol{\beta}|Y, \boldsymbol{\gamma}^{(g-1)}, \boldsymbol{\Sigma}^{(g-1)},) = N_k(\mathbf{b}^{(g)}, \mathbf{B}^g)$ where

$$\mathbf{B}^{(g)} = \Big[\mathbf{B}_0^{-1} + \omega_{11}^{-1^{(g-1)}} \sum x_{yi} x'_{yi}\Big]^{-1},$$
$$\mathbf{b}^{(g)} = \mathbf{B}^{(g)}\Big[\mathbf{B}_0^{-1}\mathbf{b}_0 + \omega_{11}^{-1^{(g-1)}} \sum x_{yi}\Big(y_i - \sigma_{12}^{(g-1)}\sigma_{22}^{-1^{(g-1)}}(s_i - x'_{si}\boldsymbol{\gamma}^{(g-1)})\Big)\Big].$$

3. **Update vector** $\boldsymbol{\gamma}$: Draw $\boldsymbol{\gamma}^{(g)}$ from $\pi(\boldsymbol{\gamma}|Y, \boldsymbol{\beta}^{(g)}, \boldsymbol{\Sigma}^{(g-1)},) = N_k(\mathbf{g}^{(g)}, \mathbf{G}^g)$ where

$$\mathbf{G}^{(g)} = \Big[\mathbf{G}_0^{-1} + \omega_{22}^{-1^{(g-1)}} \sum x_{si} x'_{si}\Big]^{-1},$$
$$\mathbf{g}^{(g)} = \mathbf{G}^{(g)}\Big[\mathbf{G}_0^{-1}\mathbf{g}_0 + \omega_{22}^{-1^{(g-1)}} \sum x_{si}\Big(y_i - \sigma_{12}^{(g-1)}\sigma_{11}^{-1^{(g-1)}}(y_i - x'_{yi}\boldsymbol{\beta}^{(g)})\Big)\Big].$$

4. **Update** $\boldsymbol{\Sigma}$: Draw $\boldsymbol{\Sigma}^{-1^{(g)}}$ from $\pi(\boldsymbol{\Sigma}^{-1}|Y, \boldsymbol{\beta}^{(g)}, \boldsymbol{\gamma}^{(g)}) = W(\nu_1^{(g)}, \mathbf{R}_1^{(g)})$where

$$\nu_1^{(g)} = \nu_0 + n \ , \ \mathbf{R}_1 = \Big[\mathbf{R}_0^{-1} + \sum(\mathbf{y}_i - X_i\boldsymbol{\delta}^{(g)})(\mathbf{y}_i - X_i\boldsymbol{\delta}^{(g)})'\Big]^{-1}$$

with $\boldsymbol{\delta}^{(g)} = (\boldsymbol{\beta}^{(g)}, \boldsymbol{\gamma}^{(g)})$

5. **Repeat** steps (2) to (4) for $B + G$ times, where B is the initial burn-in period required for the chain to converge which is discarded and G is the required simulation sample size.

References

[1] S. An, F. Schorfheide, Bayesian analysis of DSGE models, *Econometric reviews ,* 26.2-4, 113–172, (2007).

[2] Berger, J.O., Moreno, E., Pericchi, L.R., Bayarri, M.J., Bernardo, J.M., Cano, J.A., De la Horra, J., Martín, J., Ríos-Insúa, D., Betrò, B. and Dasgupta, A., 1994. An overview of robust Bayesian analysis. Test, 3(1), pp.5-124.

[3] J-H Chan, M.S. Joshi, Optimal Limit Methods for Computing Sensitivities of Discontinuous Integrals Including Triggerable Derivative Securities, *IIE Transactions*, Volume 47, Number 9, 978–997, (2015)

[4] J-H Chan, M.S. Joshi, D. Zhu, First and Second Order Greeks in the Heston Model,*Journal of Risk*, Vol 17, 4, 19–69, (2015)

[5] R.C.H. Cheng and G. M. Feast. "Some simple gamma variate generators." Applied Statistics (1979): 290-295.

[6] S. Chib, L. Jacobi, Calculating Causal Effects from Panel Data in Randomized Experiments with Partial Compliance. *Advances in Econometrics,* Volume 23 (eds S. Chib, W.E. Griffiths, G. Koop and D. Terrell), 2009, Jai Press, Elsevier Science, Amsterdam, 183–215.

[7] Chib, S., Jacobi, L., Bayesian Fuzzy Regression Discontinuity Analysis and Returns to Compulsory Schooling, to appear *Journal of Applied Econometrics*, accepted July 14th 2015.

[8] M. Giles, P. Glasserman. Smoking adjoints: Fast Monte Carlo greeks, *Risk Magazine,* 19.1 (2006), 88-92.

[9] P. Glasserman, Monte Carlo Methods in Finance, Springer 2004.

[10] P. Glasserman, Z Liu. Estimating Greeks in simulating Lévy-driven models. *The Journal of Computational Finance,* 14.2 (2010): 3.

[11] Edward Greenberg. Introduction to Bayesian econometrics. Cambridge University Press, 2012.

[12] Andreas Griewank, Andrea Walther, Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation, Second Edition, SIAM 2008

[13] C. Homescu, Adjoints and Automatic (Algorithmic) Differentiation in Computational Finance (2011). Available at SSRN: http://ssrn.com/abstract=1828503 or http://dx.doi.org/10.2139/ssrn.1828503

[14] M.S. Joshi, and D. Zhu, Optimal Partial Proxy Method for Computing Gammas of Financial Products with Discontinuous and Angular Payoffs, (2016) Applied Mathematical Finance, 23:1, 22–56.

[15] M.S. Joshi, D. Zhu, An Exact Method for the Sensitivity Analysis of Systems Simulated by Rejection Techniques, *European Journal of Operational Research*, 254 (2016), 875–888.

[16] G. Koop, (2003), Bayesian Econometrics, Wiley–Interscience

[17] Lopes, H. F., Tobias, J. L. (2011). Confronting prior convictions: On issues of prior sensitivity and likelihood robustness in Bayesian analysis. *Annu. Rev. Econ.,* 3(1), 107-131.

[18] O'hagan, A., B. R. Luce (2003). A primer on Bayesian statistics in health economics and outcomes research, Medtap International Inc

[19] Ulrich K. Müller. Measuring prior sensitivity and prior informativeness in large bayesian models. *Journal of Monetary Economics*, 59(6):581–597, 2012.

[20] S. Särkkä. Bayesian filtering and smoothing. Cambridge University Press., 2013.

[21] R. Suri, Implementation of sensitivity calculations on a Monte Carlo experiment. *Journal of Optimization Theory and Applications,* 40, 625–630, 1983.

[22] C.J. Perez, J. Martin, and M.J. Rufo. MCMC-based local parametric sensitivity estimations. *Computational Statistics & Data Analysis*, 51(2):823–835, 2006.

[23] Tanizaki, H. A simple gamma random number generator for arbitrary shape 944 parameters. *Economics Bulletin,* 3 (7), 1–10, (2008).