Machine Learning Econometrics: Bayesian algorithms and methods

Dimitris Korobilis
University of Glasgow

Davide Pettenuzzo
Brandeis University

May 15, 2020

Summary

Bayesian inference in economics is primarily perceived as a methodology for cases where the data are short, that is, not informative enough in order to be able to obtain reliable econometric estimates of quantities of interest. In these cases, prior beliefs, such as the experience of the decision-maker or results from economic theory, can be explicitly incorporated to the econometric estimation problem and enhance the desired solution.

In contrast, in fields such as computing science and signal processing Bayesian inference and computation has long been used for tackling challenges associated with ultra high-dimensional data. Such fields have developed several novel Bayesian algorithms that have gradually been established in mainstream statistics, and they now have a prominent position in machine learning applications in numerous disciplines.

While traditional Bayesian algorithms are powerful enough in order to allow estimation of very complex problems (for instance, nonlinear dynamic stochastic general equilibrium models) they are not able to cope computationally with the demands of rapidly increasing economic datasets. Bayesian machine learning algorithms are able to provide rigorous and computationally feasible solutions to various high-dimensional econometric problems, thus, supporting modern decision-making in a timely manner.

JEL Classification: C11; C15; C49; C88

Keywords: MCMC; approximate inference; scalability; parallel computation

Contents

1. Introduction and background

The purpose of this review is two-fold. The first aim is for this to be an accessible reference of various algorithms that economists can use for inference problems in the Big Data era. The second aim is to introduce methods and algorithms developed outside economics (e.g. computing science, machine learning, engineering) and discuss how economists can benefit from this wealth of work done by other scientists. The primary focus is on Bayesian algorithms, even though in many cases the algorithms analyzed are appropriate for maximum likelihood inference. Bayesian methods have been traditionally used in econometric problems that either involve complex likelihood structures or a large number of variables relative to observations. Such examples are the class of dynamic stochastic general equilibrium (DSGE) models, panel data with fixed effects or cross-sectional data with many predictors (e.g. growth regressions). In particular, Monte Carlo methods have allowed to simplify even the toughest of inference problems. However, existing Monte Carlo techniques such as the Gibbs sampler or Metropolis-Hastings algorithms are inherently demanding and can quickly hit a computational bottleneck. Therefore, a major question that this review attempts to answer is the following: what other options are there for speeding up Bayesian inference when faced with high-dimensional models and data?

The starting point for Bayesian estimation and computation is Bayes rule

$$p(\theta|y) = \frac{p(y|\theta)p(\theta)}{\int p(y|\theta)p(\theta)d\theta}, \tag{1}$$

where θ represents the parameters of our chosen model we want to estimate, $p(y|\theta)$ is the likelihood function of the specified model, $p(\theta)$ is the function of parameters before seeing the data (prior), and $p(\theta|y)$ the distribution of the parameters after observing the data (posterior). The quantity $\int p(y|\theta)p(\theta)d\theta$ is called the marginal likelihood and is a constant that ensures that the posterior has a density that integrates to one.[1] The idea here is that parameters are random variables, despite the fact that Bayesian consistency requires that in the limit (infinite observations) θ should converge to the true point parameter θ_0.

Maximum likelihood (ML) inference would require us to work only with $p(y|\theta)$,

[1] When one wants to calculate the posterior analytically this integral needs to be evaluated numerically. Otherwise when sampling methods are used we might only need to know the kernel of the posterior, in which case we simply use the expression $p(\theta|y) \propto p(y|\theta)p(\theta)$.

however, maximizing complex functions (e.g. a high-dimensional, nonlinear likelihood) is not a computationally trivial task. Instead, as Angelino et al. (2016) observe, the Bayesian paradigm is about integration. The Bayesian needs integration in order to compute marginal and conditional posteriors, prior predictive distributions (marginal likelihoods) for model comparison and averaging, and posterior predictive distributions for making predictions.

Needless to stress that in high dimensions integration doesn't become computationally more desirable than maximization used in the ML approach! So what are the relevant Bayesian tools that a modern economist could and should have in her toolbox in order to perform Bayesian inference in high-dimensions? Some key estimation algorithms that econometricians and economists have been using already for decades, are reviewed in the next Section. Subsequently, Section 3 covers several algorithms developed in fields such as computer vision, signal processing, and compressive sensing, among other fields that rely on analysis of high-dimensional data. Finally, recommendations are provided on specific ways of speeding up Bayesian inference by simplifying an econometric model in such a way that one can get "more mileage" from Bayesian algorithms.

2. A review of Bayesian computation

2.1. Exact and approximate analytical results

2.1.1. Uniform and conjugate priors

There are only a handful of cases of prior distributions that, when multiplied by a likelihood function, allow for analytical derivation of the posterior distribution and all its moments. In standard linear regression settings, uniform and natural conjugate priors allow for working with posterior distributions that belong to well known classes (Normal, Gamma, Wishart). The uniform prior collapses to multiplying the likelihood by a constant, such that the posterior is proportional to the likelihood, and the posterior mode becomes identical to the maximum likelihood estimate. The natural conjugate prior for regression models with coefficients β and variance parameter σ^2 has the form

$$p\left(\theta\right) \equiv p\left(\beta, \sigma^2\right) = p(\beta|\sigma^2)p(\sigma^2). \tag{2}$$

The "unnatural" feature of the natural conjugate prior formulation is that we need specify our prior for σ independently, but our prior opinion about β is conditional on the values of σ. Nevertheless, such priors lead to an analytical expression for the parameter posteriors, that is, posterior means, variances, and other moments are readily available in closed form. This is the reason why such priors were widespread many decades ago, well before cheap and strong computing became available. Interestingly, recent econometric papers have revived interest in using such simple priors, by exploiting their simplicity in order to estimate effortlessly large vector autoregressions (VARs) with hundreds of thousands of coefficients; see the discussion in Korobilis and Pettenuzzo (2019).

2.1.2. Normal and Laplace approximations

In more complex settings where conjugate priors cannot be defined, the posterior can sometimes be approximated by a Normal distribution. According to the Bayesian central limit theorem, under certain conditions, the posterior distribution $p\left(\theta|y\right)$ is asymptotically Normal. The Bernstein-von Mises theorem states that the posterior distribution is asymptotically independent of the prior distribution, thus, giving further justification to a Normal approximation of the posterior distribution.

Laplace (1774) was the first to argue that for any continuous posterior that is smooth and well-peaked around its point of maxima (mode), a Normal approximation is a sensible choice. First, note that if $\theta^{\star} = \arg\max_{\theta\in\Theta} p\left(\theta|y\right)$ is the maximum of the posterior function, then this will also be the maximum of the log-posterior $h\left(\theta\right) = \log\left(p\left(\theta|y\right)\right)$. Then a second-order Taylor series expansion of the log-posterior around $\theta^{\star}$ gives

$$\begin{aligned} h\left(\theta\right) &\approx h\left(\theta^{\star}\right) + \dot{h}\left(\theta^{\star}\right)\left(\theta-\theta^{\star}\right) - \frac{1}{2}\left(\theta-\theta^{\star}\right)' \ddot{h}\left(\theta^{\star}\right)\left(\theta-\theta^{\star}\right), \\ &\approx const - \frac{1}{2}\left(\theta-\theta^{\star}\right)' \ddot{h}\left(\theta^{\star}\right)\left(\theta-\theta^{\star}\right), \end{aligned} \tag{3}$$

where $\dot{h}\left(\theta^{\star}\right)$ and $\ddot{h}\left(\theta^{\star}\right)$ are the first and second derivatives of the log-posterior function. Given that $\theta^{\star}$ is a maximum, it follows that $\dot{h}\left(\theta^{\star}\right) = 0$, which justifies the simplification in the second row of equation (3). Similarly, $\ddot{h}\left(\theta^{\star}\right)$ is positive definite, which implies that the log-posterior is proportional to a Normal kernel. Equivalently, by taking the exponential function on both sides of the expression in equation (3) we have

$$p\left(\theta|y\right) = \exp\left(h\left(\theta\right)\right) \sim N\left(\theta^{\star}, \ddot{h}\left(\theta^{\star}\right)\right), \tag{4}$$

which provides a justification for a Normal approximation to the posterior. Therefore, instead of integrating to find the posterior, the Bayesian inference problem becomes an optimization one: once we find $\theta^\star$ and $\ddot{h}(\theta^\star)$, we have everything we need in order to describe the (approximate) posterior analytically.[2] The approximation error of the Laplace approach is $O\left(N^{-1/2}\right)$. If the posterior is asymmetric or skewed, then including higher-order derivatives of $h(\theta)$ in the Taylor series expansion can improve the approximation (Lindley, 1980). However, evaluating numerically such terms for the log-posterior function can be impractical computationally. Laplace approximations are fast, accurate and easy to implement. Nevertheless, in high-dimensional problems it becomes difficult, if not impossible, to numerically evaluate the joint posterior mode because the posterior function could be too complex and multi-modal.

2.1.3. Bayesian Quadrature and Monte Carlo

Assuming that the parameter vector θ has K elements, Naylor and Smith (1982) note that the marginal posterior of θ_i, $i = 1, ..., K$, is of the form

$$p(\theta_i|y) = \int p(y|\theta)p(\theta)d\theta_{j\neq i}, \tag{5}$$

where $d\theta_{j\neq i}$ denotes integration over the $K-1$ terms θ_j, $j = 1, ..., K$ for $j \neq i$. As discussed later in this review, many modern Bayesian machine learning algorithms exploit this result and work with the marginal posterior distribution. This is because the K marginals $p(\theta_i|y)$ can be trivially processed in parallel using modern multi-core systems. Of course, this was not the initial intention of the early work of Naylor and Smith (1982). Rather their focus on the marginal posterior in equation (5) was driven by their desire to use iterative quadrature methods for estimating such integral. Naylor and Smith (1982) in particular suggest an adaptive Gauss-Hermite quadrature, while others have proposed Gaussian process (GP) priors[3] leading to the "Bayesian quadrature" algorithm. Alternatively, the integral in equation (5) can be evaluated using Monte Carlo Integration. Rasmussen and Ghahramani (2003) argue that classical Monte Carlo estimators violate the *Likelihood Principle* and instead propose a Bayesian Monte Carlo procedure.

[2] As with maximum likelihood or maximum a-posteriori (MAP) inference (see subsection 3.4) one can use a range of well-known numerical optimization routines to find the posterior mode $\theta^\star$ (e.g. quasi-Newton methods).

[3] GP priors are priors over functions and their values.

2.2. Importance sampling

A natural question is what should a Bayesian do if she derives an expression for the posterior distribution that is not in a form that she recognizes or can easily be sampled from (e.g. Normal, Bernoulli, Gamma or any other distribution that we can sample from easily). Under this scenario, importance sampling offers a very intuitive and simple solution: if you do not recognize $p(\theta|y)$, choose instead a "proposal distribution" $q(\theta)$ that is easy to sample from and convert its samples into samples from the desired density $p(\theta|y)$. Assume we collect n such draws, $\widehat{\theta}^{(1)}, ..., \widehat{\theta}^{(n)} \sim q$. Next, estimate weights $w^{(i)} = \frac{p(\widehat{\theta}^{(i)}|y)}{q(\widehat{\theta}^{(i)})}$,[4] for $i = 1, ..., n$, and use them to obtain the importance weighted estimator

$$\widetilde{\theta} = \frac{\sum_{i=1}^{n} w^{(i)} \widehat{\theta}^{(i)}}{\sum_{i=1}^{n} w^{(i)}}. \tag{6}$$

As long as the support of q contains the support of $p(\theta|y)$, it can be shown that $\widetilde{\theta}$ converges to $E(\theta|y)$; see Geweke (1989) for detailed results. Unfortunately, when θ is high-dimensional it can be very hard to find a q that meets this condition, hence importance sampling becomes harder to implement in very large models.

2.3. Metropolis-Hastings algorithm

Metropolis-Hastings is a class of Monte Carlo algorithms based on accept/reject sampling, that extends ideas in importance sampling. Assume we have obtained S samples from a proposal distribution q and further assume that the $(i-1)^{th}$ sample we generated, denoted by $\widehat{\theta}^{(i-1)}$, is indeed a sample from $p(\theta|y)$. Finally, denote with $\widehat{\theta}^{\star}$ the i-th candidate sample from q. Then $\widehat{\theta}^{\star}$ is accepted with probability

$$\alpha\left(\widehat{\theta}^{\star}, \widehat{\theta}^{(i-1)}\right) = min\left\{1, \frac{p(\widehat{\theta}^{\star}|y)q(\widehat{\theta}^{(i-1)}|\widehat{\theta}^{\star})}{p(\widehat{\theta}^{(i-1)}|y)q(\widehat{\theta}^{\star}|\widehat{\theta}^{(i-1)})}.\right\} \tag{7}$$

If the acceptance ratio α is larger than a random draw u from a $Uniform(0,1)$ then we accept the draw and set $\widehat{\theta}^{(i)} = \theta^{\star}$, otherwise we discard it and set $\widehat{\theta}^{(i)} = \widehat{\theta}^{(i-1)}$.

In order to guarantee that draws $\widehat{\theta}^{(i)}$ are samples from the target posterior $p(\theta|y)$ we

[4] It is important to note that, henceforth, $p(x)$ denotes a distribution function for random variable x, and $p(\widehat{x})$ denotes the same distribution p evaluated at the value $\widehat{x}$. The latter is going to be a number (probability), and the difference between the two expressions stems from the fact that once values $\widehat{x}$ are sampled (observed), these are not random variables any more.

aim to approximate, we need several desirable features for this chain such as irreducibility and aperiodicity. Chib and Greenberg (1995) offers an early, accessible reference to Metropolis-Hastings. Applications of the MH algorithm are numerous in economics, with most notably its use in nonlinear state-space formulations for the purpose of estimating dynamic stochastic general equilibrium (DSGE) models. As with importance sampling, the Metropolis-Hastings algorithm can become inefficient in very large dimensions, with low rates of acceptance, poor mixing of the chain and highly correlated draws.

2.4. Gibbs sampler

With the Gibbs sampler the aim is to sample from the conditional posterior, that is, the posterior of each parameter conditional on all other model parameters being fixed to a known value. Assume that θ has n elements or blocks, $\theta_1, ..., \theta_n$, e.g. in the most plain univariate regression with one regressor this would be $\theta_1 = \beta$ and $\theta_2 = \sigma^2$. Thanks to a straightforward application of Bayes Theorem, it holds that samples from the conditional posteriors are also samples from the joint parameter posterior

$$p(\theta_j|\theta_1, ..., \theta_{j-1}, \theta_{j+1}, ..., \theta_n, y) = \frac{p(\theta_1, ..., \theta_n|y)}{p(\theta_1, ..., \theta_{j-1}, \theta_{j+1}, ..., \theta_n|y)} \propto p(\theta_1, ..., \theta_n|y) \equiv p(\theta|y). \tag{8}$$

The conditional posterior for each θ_j is proportional to the joint posterior simply because the denominator is a constant (all θ_k for $k \neq j$ are conditioned upon and are known/fixed, hence, $p(\theta_1, ..., \theta_{j-1}, \theta_{j+1}, ..., \theta_n|y)$ is the value of the p.d.f.). The Gibbs sampler can be viewed as a special case of Metropolis-Hastings algorithms where every draw is accepted with probability one: if we assume that the conditional posterior $p(\theta_j|\theta_1, ..., \theta_{j-1}, \theta_{j+1}, ..., \theta_n, y)$ $\forall$ j is the proposal density q, then it is trivial to show via equation (7) that $\alpha = min\left\{1, 1\right\}$.

The Gibbs sampler is probably the most user-friendly among the class of MCMC algorithms. It simplifies computation of some complex econometric and statistical models that would otherwise be extremely hard to estimate with maximum likelihood. Deriving a conditional posterior involves an expression for a parameter θ_j by keeping other parameters fixed (to their last sampled values), an idea that is most useful in nonlinear and latent parameter models. For instance, consider the example a Markov switching autoregression (AR) for measuring business cycles: conditional on knowing the indicator variables indexing the Markov states, Gibbs sampler inference on the autoregressive coefficients and the variance parameter is identical to that of the standard

AR model. Furthermore, more complex nonlinear problems can be easily transformed to linear, Gaussian problems that can be approximated trivially by the Gibbs sampler; see, among others, the well-known estimator for stochastic volatility models of Kim et al. (1998).

3. Bayesian methods in the Big Data Era

As we adjust to the new reality of having larger amounts of data available, the Bayesian computation methods that we have briefly reviewed in the previous section also need to be adapted and improved. In particular, as the data size, number of features, size of the models, and model space all growth, it becomes computationally harder to evaluate the likelihood function, visiting all the parameters in the model, while at the same time using all the data. At the same time, the algorithms that we discussed previously also start to experience slower mixing rates. One may therefore be sceptical about the possibility of adapting Bayesian methods to keep up with this trend. However, it is worth noting that Bayesian methods features a number of important advantages that make them particularly appealing even in the Big Data Era. First and foremost, Bayesian methods offer the flexibility and adaptivity required to deal with a reality in which the volume of the data grows. The updating rule which is at the core of Bayesian methods is sequential in nature and suitable for dealing with constantly growing data streams.

The main complication of applying Bayesian methods to big data problems has to do with the computational bottlenecks that the previously described algorithms face, and for that reason existing literature has been hard at work developing new (approximate) methods to deal with this evolving reality. In this Section, these issues are discussed in more detail and some of the solutions that have been proposed in the literature to deal with the increasing amounts of data and the larger computational costs that researchers face when implementing Bayesian methods, are reviewed.

3.1. Speeding up MCMC

The first step towards making Bayesian computation feasible in a high-dimensional setting, is to use approximations that replace computationally intensive steps of MCMC algorithms. One solution proposed in the literature is to use approximate samplers that use data sub-samples (*minibatches*) rather than the full data set. Examples include subsampling Markov chain Monte Carlo (MCMC) implementations; see Bardenet, Doucet, and Holmes (2017) for an excellent review of these approaches. The basic idea is to to estimate the likelihood function for n observations from a random subset of m observations, where $m \ll n$. With conditionally independent observations, one can rewrite the log-likelihood $\ell\left(\theta\right) = \log p\left(\left.y\right|\theta\right)$ as follows

$$\ell\left(\theta\right) = \sum_{i=1}^{n} \ell_i\left(\theta\right) \tag{9}$$

where $\ell_i\left(\theta\right) = \log p\left(\left.y_i\right|\theta\right)$ denotes the log-likelihood contribution of the i-th observation in the sample.[5] As it turns out, estimating (9) using simple random sampling where any $\ell_i\left(\theta\right)$ is included with the same probability, generally results in a very large variance. This problem could be eliminated if one were to re-weight the draws using so called probability proportional-to-size sampling, but unfortunately computing these weights can be computationally very expensive. One way to sidestep this computational bottleneck is to make the $\left\{\ell_i\left(\theta\right)\right\}_{i=1}^{n}$ more homogeneous by using control variates so that the population elements are roughly of the same size. In this way, a simple random sampling would then expected to be efficient. This is the approach taken by Quiroz et al (2019), who use control variates to obtain a highly efficient unbiased estimator of the log-likelihood, with a total computing cost that is much smaller than that of the full log-likelihood in standard MCMC. They show that the asymptotic error of the resulting log-likelihood estimate is negligible even for a very small number of random samples m ($m \ll n$), and demonstrate that (i) sub-sampling MCMC is substantially more efficient than standard MCMC in terms of sampling efficiency; and (ii) their approach outperforms other subsampling methods for MCMC proposed in the literature, including those listed at the beginning of this section.

Sub-sampling has important implications for MCMC inference. For example, in the standard MH sampler we accept a proposal draw with probability $u \sim Uniform(0, 1)$ if and only if

$$\alpha = \frac{p(\theta^{\star}|y)q(\widehat{\theta}^{(i-1)}|\widehat{\theta}^{\star})}{p(\widehat{\theta}^{(i-1)}|y)q(\widehat{\theta}^{\star}|\widehat{\theta}^{(i-1)})} > u, \tag{10}$$

where we remind $\widehat{\theta}^{(i-1)}$ is the draw we have accepted in the previous iteration, and $\widehat{\theta}^{\star}$ the candidate draw in the current iteration, which will be accepted with probability α. Evaluating repeatedly (in a Monte Carlo fashion) the expression in equation (10) using high-dimensional posterior densities, is quite cumbersome. By rearranging terms in this equation, taking logarithms, and splitting the likelihood function over the N

[5] The assumption that the total log-likelihood can be decomposed into a sum of terms where each term depends on a unique piece of information is not overly restrictive. It applies to longitudinal problems but also to certain time series problems such as AR(p) processes.

observations in the data y we have

$$\begin{aligned}
\log\left\{\frac{p(y|\widehat{\theta}^{\star})}{p(y|\widehat{\theta}^{(i-1)})}\right\} &> \log\left\{u\frac{q(\widehat{\theta}^{\star}|\widehat{\theta}^{(i-1)})}{q(\widehat{\theta}^{(i-1)}|\widehat{\theta}^{\star})}\right\} \Rightarrow \qquad (11)\\
\frac{1}{N}\sum_{n=1}^{N}\log\left\{\frac{p(y_n|\widehat{\theta}^{\star})}{p(y_n|\widehat{\theta}^{(i-1)})}\right\} &> \frac{1}{N}\log\left\{u\frac{q(\widehat{\theta}^{\star}|\widehat{\theta}^{(i-1)})}{q(\widehat{\theta}^{(i-1)}|\widehat{\theta}^{\star})}\right\} \Rightarrow \qquad (12)\\
\frac{1}{N}\sum_{n=1}^{N}\lambda_n(\widehat{\theta}^{\star},\widehat{\theta}^{(i-1)}) &> c(u,\widehat{\theta}^{\star},\widehat{\theta}^{(i-1)}). \qquad (13)
\end{aligned}$$

Therefore, instead of sampling the full MH step in (10), one can subsample the log-likelihood ratio quantity $\lambda_n(\widehat{\theta}^{\star},\widehat{\theta}^{(i-1)})$ and subsequently perform the approximate test $\lambda_n^{\star}(\widehat{\theta}^{\star},\widehat{\theta}^{(i-1)}) > c(u,\theta^{\star},\widehat{\theta}^{(i-1)})$ in order to decide whether to accept $\widehat{\theta}^{\star}$ or not.

3.2. Hamiltonian Monte Carlo

Hamiltonian Monte Carlo (HMC) methods offer an alternative solution to the limitation of Metropolis-Hastings algorithm in exploring efficient high-dimensional posterior distributions. In particular, by carefully exploiting the differential structure of the target probability density, HMC provides an automatic procedure that yields a more efficient exploration of the probability space in such high dimensions. More specifically, HMC uses an approximate Hamiltonian dynamics simulation based on numerical integration which is then corrected by performing a Metropolis acceptance step.

In order to sample from the K-dimensional posterior distribution $p(\theta|y)$, HMC introduces an independent K-dimensional auxiliary variable δ with density $p(\delta|\theta)$, which leads to the joint density

$$p(\theta,\delta) = p(\delta|\theta)p(\theta) \qquad (14)$$

In most applications, including Stan, $p(\delta|\theta)$ is specified to be independent from the parameter vector θ, for example using a multivariate normal distribution, i.e. $\delta \sim N(0,M)$, which leads to

$$p(\theta,\delta) = p(\theta)N(0,M) \qquad (15)$$

Let $H(\theta,\delta)$ denote the *Hamiltonian* function, i.e. the negative joint log-probability,

$H(\theta, \delta) = -\log p(\theta, \delta)$, and similarly let $\mathscr{L}(\theta)$ denote the logarithm of the target density $p(\theta)$. It can be shown (see Girolami and Calderhead, 2011) that

$$H(\theta, \delta) = -\mathscr{L}(\theta) + \frac{1}{2} log \left\{ (2\pi)^K |M| \right\} + \frac{1}{2} \delta' M^{-1} \delta \tag{16}$$

In practice, given a candidate draw $\delta^{(i)}$ from the $N(0, M)$ auxiliary density and the current draw $\theta^{(i)}$, the derivatives of $H(\theta, \delta)$ with respect to θ and δ,

$$\begin{aligned} \frac{\partial H}{\partial \theta} &= -\mathscr{L}'(\theta) \\ \frac{\partial H}{\partial \delta} &= M^{-1} \delta \end{aligned} \tag{17}$$

give rise to the transition $\theta^{(i)} \rightarrow \theta^*$ and $\delta^{(i)} \rightarrow \delta^*$. Next, the proposed θ^* (and δ^*) are retained with probability

$$min \left\{ 1, \exp \left(H \left(\theta^{(i)}, \delta^{(i)} \right) - H \left(\theta^*, \delta^* \right) \right) \right\} \tag{18}$$

If the proposal is not accepted, the previous parameter value is returned for the next draw and used to initialize the next iteration.

3.3. Parallelizing MCMC

MCMC methods are characterized by the Markov property, that is, the fact that we need to first assess the current sample $\widehat{\theta}^{(i)}$ in order to decide whether $\widehat{\theta}^{(i+1)}$ is a possible sample from the target posterior. Therefore, due to this sequential dependence between iterations, it seems an oxymoron to attempt to parallelize *across* MCMC iterations. As a consequence, a natural first step toward parallelization – assuming we have a high-dimensional parameter θ that can be split into r independent blocks θ_r, $r = 1, ..., R$ – would be to parallelize *within* each iteration. That way we can compute each $p(\theta_r|y)$ in a separate worker. Malewicz et al. (2010) demonstrate such an algorithm in what is known as Google Pregel. However, Scott et al. (2016) note that not only such algorithms have very bad convergence rates, they are also extremely inefficient once one factors in computing costs and the marginal reductions in computing times.[6] Similarly, Gonzalez et al. (2011) propose two parallel versions of the Gibbs sampler with good convergence

[6] For the Pregel environment in particular, a ten-fold increase in computing capacity only reduces computation time by a factor of two.

guarantees, namely the Chromatic sampler and the Splash sampler. However, such parallel samplers are limited by the fact that there must be frequent (i.e. at each MCMC iteration) communication between the workers.

Instead of breaking a high-dimensional vector of parameters θ into smaller subvectors, Scott et al. (2016) propose to break the data y into R smaller blocks that can be distributed to an equivalent number of workers. This means that the high-dimensional posterior can be written as

$$p(\theta|y) = \prod_{r=1}^{R} p(\theta|y_r) \propto \prod_{r=1}^{R} p(y_r|\theta)p(\theta)^{1/R}, \tag{19}$$

where the prior is broken into R independent components, $p(\theta) = \prod_R p(\theta)^{1/R}$ such that the total amount of prior information in the system is not affected by our decision to break y in R blocks.[7] Assuming for simplicity that all workers each produce S draws of θ, then the consensus posterior will comprise S draws that are weighted combinations of the $R \times S$ draws from all workers. Angelino et al. (2016, Section 4.2.1) provide citations to further studies that implement similar ideas towards the design of parallel MCMC.

An alternative way to exploit the idea of partitioning the data into R non-overlapping subsets y_r, $r = 1, ..., R$, is to use the Weierstrass transform. For a function $f(\theta)$ the Weierstrass transform is a convolution of a Gaussian density with standard deviation h and $f(\theta)$, and is of the form

$$W_h f(\theta) = \int_{-\infty}^{+\infty} \frac{1}{\sqrt{2\pi}h} \exp\left\{-\frac{(\theta-\mu)^2}{2h^2}\right\} f(\mu)d\mu. \tag{20}$$

$W_h f(\theta)$ can be thought of as a smooth approximation to $f(\theta)$.[8] Applying this transform

[7] One critique of this approach is that the prior may not provide enough regularization for each separate computation.

[8] When $h \to 0$, the transformation $W_h f(\theta)$ converges to $f(\theta)$.

to the posterior density, we get

$$\begin{aligned} p(\theta|y) &= \prod_{r=1}^{R} p(\theta|y_r) \approx \prod_{r=1}^{R} W_h p(\theta|y_r) && (21) \\ &= \prod_{r=1}^{R} \int_{-\infty}^{+\infty} \frac{1}{\sqrt{2\pi}h} \exp\left\{ -\frac{(\theta-\mu_r)^2}{2h^2} \right\} p(\mu_r|y_r) d\mu_r && (22) \\ &\propto \int_{-\infty}^{+\infty} \prod_{r=1}^{R} \exp\left\{ -\frac{(\theta-\mu_r)^2}{2h^2} \right\} p(\mu_r|y_r) d\mu_r. && (23) \end{aligned}$$

This last expression shows that after applying the Weirstrass transform, the posterior of θ can be viewed as the outcome of marginalizing latent parameters $\mu_1, ..., \mu_R$ from an augmented posterior $p(\theta, \mu_1, ..., \mu_R|y)$. This enables a subset-based Gibbs sampling, that is highly parallelizable, where we can first sample $\theta|\mu_r, y \sim N(\widehat{\mu}, h^2)$ and then $\mu_r|\theta, y \sim \frac{1}{\sqrt{2\pi}h} \exp\left\{ -\frac{(\theta-\mu_r)^2}{2h^2} \right\} p(\mu_r|y_r)$, see Wang and Dunson (2013) for more details on this scheme.

Other avenues of parallelizing MCMC do exist and their success depends on the inference problem at hand. For example, in problems with nonlinear coefficients whose posterior does not have a closed form expression, the Griddy Gibbs sampler of Ritter and Tanner (1992) can be used in order to evaluate such parameters in a grid (instead of sampling from their highly complex conditional posterior). The approximation in the Griddy-Gibbs sampler can be trivially parallelized, although the full algorithm itself can become very inefficient in high-dimensional models. Other examples include the Adaptive Griddy-Gibbs (AGG) and the Multiple-Try Metropolis (MTM) of Liu et al. (2000). Another related issue in MCMC methods is that of whether one needs to run a very long chain with as many iterations as (computationally) possible, or follow the advice of Gelman and Rubin (1992) and run several chains in parallel. Assuming random starting points, running chains in parallel allows to assess and speed up convergence by combining their output. Of course, when such chains run in parallel but are independent, the gains in efficiency are low. The Interchain Adaptive MCMC algorithm (Craiu et al., 2009) allows for parallel chains to interact within an adaptive Metropolis setting, such that substantial speed up in convergence is achieved. The adaptive element in this algorithm relies on the fact that each chain learns from its past, but also from the past iterations of other chains. Using this algorithm, Solonen et al. (2012) quote dramatic speed up in convergence by using only 10 chains in parallel.

The “Affine-Invariant” Ensemble MCMC sampler of Goodman and Weare (2010) also involves parallel processing of chains in batches with efficiency gains in high dimensions. However, such samplers processing non-independent chains in parallel are restricted by the fact that communication between workers in a cluster must be frequent. Therefore, such samplers are slower than respective single-core MCMC samplers *per iteration*, and computational gains from processing parallel chains only come from the fact that total convergence is achieved using a lower number of iterations.

3.4. Maximum a posteriori estimation and the EM algorithm

Despite the increased availability of methods for making MCMC faster, there are cases where sampling from the full posterior might not be feasible or desired. As long as parameter uncertainty is not important for a specific empirical problem, one can work with point estimates that summarize some important features of the posterior distribution. In order to proceed with a point estimate $\widehat{\theta}$ of the unknown parameters θ, we can introduce a cost function $\mathscr{C}$ that we aim to minimize. Therefore, the Bayesian equivalent of classical point estimation takes the following form

$$\arg\min_{\theta^*} \int \mathscr{C}(\theta - \theta^*) p(\theta|y) d\theta. \tag{24}$$

It is trivial to show that when using the quadratic cost function $\mathscr{C}(\theta - \theta^*) = (\theta - \theta^*)^2$, equation (24) is minimized for $\widehat{\theta} = \int \theta p(\theta|y) d\theta$, which is the posterior mean and is also known as the *minimum mean square error (MMSE)* estimator. Similarly, the absolute cost function $\mathscr{C}(\theta - \theta^*) = |\theta - \theta^*|$ leads to the posterior median as the optimal point estimate.

An alternative point estimate can be obtained using the hit-and-miss cost function of the form

$$\mathscr{C}(\theta - \theta^*) = \begin{cases} 1, & \text{if } |\theta - \theta^*| \geq \delta \\ 0, & \text{if } |\theta - \theta^*| < \delta \end{cases} \tag{25}$$

for δ very small. Inserting this cost function in equation (24) we obtain the solution

$$\widehat{\theta} = \arg\max_{\theta^*} p(\theta|y), \tag{26}$$

which is the posterior mode, also known as the *maximum a posteriori (MAP)* estimator. Given (from Bayes rule) that $p(\theta|y) = p(y|\theta)p(\theta)$, it becomes apparent that Maximum

Likelihood inference is a special case of MAP estimation with the uniform prior $p(\theta) \propto 1$.

MAP methods have been used in Bayesian inference for several decades. In a seminal paper, Tipping (2001) derives a MAP estimate of the parameters of a support vector machine model under a "sparse Bayesian learning (SBL)" prior. This prior for a parameter θ is a special case of a hierarchical Bayes structure where θ depends one some unknown hyperparameters ξ that are random variables and have their own prior. Such hierarchical priors are used extensively nowadays in Bayesian analysis as a means of imposing shrinkage and computation is typically tackled by means of the Gibbs sampler (see Korobilis, 2013, for more details). In high-dimensional settings, however, sampling is not always feasible and Tipping (2001) derives a MAP estimator for the SBL prior using type-II maximum likelihood methods.[9]

Of course there are numerous ways one can solve the convex optimization problem in equation (26), and we can't review all of them in such a short review. For example, Green et al. (2015) review proximal algorithms for obtaining the MAP estimate in high-dimensional settings; see also Parikh and Boyd (2013). Nevertheless, among all possible algorithms here we distinguish the EM algorithm. One reason for doing so is because the EM algorithm can be thought of as the optimization equivalent of the Gibbs sampler. Another important reason is that the EM algorithm is a unifying inference tool that nest several other approximating algorithms, such variational Bayes and message passing algorithms. There are several examples of high-dimensional MAP inference using the EM algorithm, and most notably we mention Rockova and George (2014).

3.5. Variational Bayes and Expectation Propagation

3.5.1. Variational Bayes

As in MAP inference, the main idea behind variational Bayes is to use optimization instead of sampling. First we introduce a family of densities $q(\theta)$ and subsequently we try to find a certain density $q^{\star}(\theta)$ that minimizes the Kullback-Leibler (KL) divergence

[9] The name "type-II maximum likelihood" is a bit deceiving, as this method finds the value of the hyperparameter ξ that maximizes the data marginal likelihood and not the likelihood function; see Berger (1985).

to the exact posterior $p(\theta|y)$. Mathematically we want to minimize the following function

$$\begin{aligned} q^\star(\theta) &= \arg\min_{q(\theta)} KL(q\|p) && (27) \\ &= \arg\min_{q(\theta)} \int q(\theta) \log \left\{ \frac{q(\theta)}{p(\theta|y)} \right\} d\theta, && (28) \end{aligned}$$

where it holds that $KL(q\|p) \geq 0$, with value equal zero only when $q(\theta)$ is identical to the true posterior $p(\theta|y)$. It can be shown that this minimization problem is equivalent to finding a $q(\theta)$ that maximizes the marginal likelihood. This is because for the logarithm of the marginal likelihood holds

$$\begin{aligned} \log(p(y)) &= \log(p(y)) \int q(\theta) d\theta = \int q(\theta) \log(p(y)) d\theta && (29) \\ &= \int q(\theta) \log \left\{ \frac{p(y,\theta)/q(\theta)}{p(\theta|y)/q(\theta)} \right\} d\theta && (30) \\ &= KL + \int q(\theta) \log \left\{ \frac{p(y,\theta)}{q(\theta)} \right\} d\theta, && (31) \end{aligned}$$

which, given that $KL \geq 0$, gives

$$p(y) \geq \exp\left(\int q(\theta) \log \left\{ \frac{p(y,\theta)}{q(\theta)} \right\} d\theta \right). \qquad (32)$$

Therefore, the VB optimization problem becomes that of maximizing the lower bound for the marginal likelihood. Note that this problem is different from MAP because here we are looking to optimize with respect to a function $q(\theta)$ and not just the random variable θ. For that specific reason, this optimization problem for the functional $q(\bullet)$ can be solved iteratively using calculus of variations. Before we do so, it is convenient to split θ into J independent blocks, i.e. $q(\theta) = \prod_{j=1}^{J} q(\theta_j)$.[10] Then we can show that $p(y)$ can be maximized by iterating sequentially through

$$\begin{aligned} q^\star(\theta_1) &\propto \exp\left(\int \log p(y,\theta) p(\theta_{(-1)}) d\theta_{(-1)} \right), && (33) \\ &\vdots \\ q^\star(\theta_J) &\propto \exp\left(\int \log p(y,\theta) p(\theta_{(-J)}) d\theta_{(-J)} \right), && (34) \end{aligned}$$

[10] This decomposition is called the *mean-field approximation*, a term originating from Physics.

where $\theta_{(-j)}$ denotes θ with its j^{th} element removed. It turns out that this iterative scheme is very similar to the EM algorithm. Each integral provides the expectation of the joint posterior with respect to the density $p(\theta_{(-j)})$ for all $j = 1, ..., J$. Loosely speaking, this scheme also resembles a Gibbs sampler. However, instead of sampling, we fix $\theta_{(-j)}$ to their posterior mean values.

3.5.2. Expectation Propagation

Expectation propagation (EP) is related to variational Bayes, but it can be considered as a separate class of algorithms. In contrast to VB, EP attempts to minimize the "reverse" KL divergence measure

$$KL = \int p(\theta|y) \log \left\{ \frac{p(\theta|y)}{q(\theta)} \right\} d\theta. \tag{35}$$

We showed previously that the variational Bayes optimization problem leads to calculating expectations with respect to the proposal density $q(\theta)$ (after we split θ into independent blocks). In contrast, the EP optimization problem shown in equation (35) can be thought of as requiring to take expectation with respect to the unknown posterior $p(\theta|y)$. For that reason the EP optimization approach is different to VB.

First, we assume that the joint distribution can be decomposed into N "factors" of the form

$$p(\theta, y) = \prod_{n=1}^{N} f_n(\theta). \tag{36}$$

Next we need to choose $q(\bullet)$ based on the exponential family of distributions, and assume that this is also decomposed into N factors of the form

$$q(\theta) = \frac{1}{Z} \prod_{n=1}^{N} \widetilde{f}_n(\theta), \tag{37}$$

where Z is a normalizing constant that makes the distribution integrate to one. The idea is to process the EP optimization problem for each of the N factors separately.[11] Each factor $\widetilde{f}_n(\theta)$ is refined *iteratively* by making $q^{\star}(\theta) \propto \widetilde{f}_n(\theta) \prod_{j=1, j\neq n}^{N} \widetilde{f}_j(\theta)$ a closer approximation to $p^{\star}(\theta) \propto f_n(\theta) \prod_{j=1, j\neq n}^{N} \widetilde{f}_j(\theta)$. Then $\widetilde{f}_n(\theta)$ is removed from the

[11] By splitting the problem into N factors/batches, it should become apparent that EP algorithms can be trivially parallelized.

approximating distribution by calculating $q^{\star\star}(\theta) = q(\theta)/\widetilde{f}_n(\theta)$ and we define $p^{\star\star}(\theta) = \frac{1}{Z^\star} f_n(\theta) q^{\star\star}(\theta)$. In the final step of the iterative scheme, the factor $\widetilde{f}_n(\theta)$ is updated such that the sufficient statistics of $q(\theta)$ match those of $p^{\star\star}(\theta)$.

While the description provided is very generic, implementations of expectation propagation can take several interesting forms depending on the application. The loopy belief propagation algorithm that is used to compute marginal posterior distributions in Bayesian networks is a special case of EP, as are other cases of the general class of *message passing algorithms*.[12] Such algorithms are at the forefront of statistical and machine learning research in the Big Data era.

3.6. Approximate Bayesian Computation

In high-dimensional applications, with high complexity and volume of available data, calculation of the likelihood or the posterior might be computationally intractable or closed-form expressions might not be available. There are also cases in fields such as image analysis or epidemiology where the normalizing constant of the likelihood is unknown. Approximate Bayesian Computation (ABC) is specifically appropriate for use in such cases. Therefore, the argument in favor of ABC is not only that it is more computationally efficient than MCMC methods, rather it can be used in many complex problem when application of MCMC is infeasible.

A basic version of ABC, that provides n samples of the parameter of interest θ, can be summarized with the following pseudo-algorithm

Basic ABC rejection sampler

```
for $i = 1 : n$
    repeat
        * Generate a $\widehat{\theta}^\star$ randomly from the prior $p(\theta)$
        * Generate randomly data $z$ using the specified econometric model, with $\theta$
          fixed to the generated value $\widehat{\theta}^\star$
    until $\rho(z, y) \leq \epsilon$
```

[12] In computing science message passing is the concept of depicting graphically, typically using graphical models, how the parameters and the factors (functionals) interact with each other. The resulting class of algorithms can be extremely powerful and trivially parallelizable; see Korobilis (2020) for more details.

```
        set $\widehat{\theta}^{(i)} = \widehat{\theta}^{\star}$
    end for
```

In this algorithm, $\rho\left(z, y\right)$ is a distance function (e.g. Euclidean) measuring how close the generated data z are relative to the observed data y, and $\epsilon \to 0$. In the case of high-dimensional data, the probability of generating a z that is close to y goes to zero. Therefore, in practice ABC algorithms evaluate the distance between summary statistics of z and y. In this case we would evaluate instead the distance function $\rho\left(\eta\left(z\right), \eta\left(y\right)\right) \leq \epsilon$, where $\eta(\bullet)$ is a function defining a statistic which most often is not sufficient. Using summary statistics may result in loss of accuracy, especially in cases where not many summary statistics of a dataset are available.

The above scheme samples θ from the approximate posterior

$$\begin{aligned} p_{\epsilon}\left(\theta|y\right) &= \int p_{\epsilon}\left(\theta, z|y\right) dz & (38)\\ &= \int \frac{p(\theta) \times p\left(z|\theta\right) \mathcal{I}\left(\rho\left(\eta\left(z\right), \eta\left(y\right)\right) \leq \epsilon\right)}{p(z)\mathcal{I}\left(\rho\left(\eta\left(z\right), \eta\left(y\right)\right) \leq \epsilon\right)} dz & (39)\\ &\approx \frac{p(\theta)p(y|\theta)}{p(y)} \equiv p(\theta|y), & (40)\end{aligned}$$

where $\mathcal{I}\left(A\right)$ is a function that takes the value one if expression A holds, and it is zero otherwise.

An obvious problem with this scheme is that it heavily relies on the choice of prior. Particularly in high-dimensional settings, using simulated values from the prior $p(\theta)$ is inefficient and results in proposals that are located in low probability regions of the true posterior we want to approximate. In this case we can define the following MCMC-ABC algorithm, which is a likelihood free MCMC sampler

MCMC-ABC algorithm

```
    for $i = 1 : n$
        repeat
          * Generate $\widehat{\theta}^{\star}$ from a proposal distribution $q\left(\theta|\theta^{(i-1)}\right)$
          * Generate $z$ from the likelihood $p\left(y|\widehat{\theta}^{\star}\right)$
        * Generate $u$ from $\mathcal{U}_{[0,1]}$ and compute the acceptance probability
            $\alpha\left(\widehat{\theta}^{\star}, \widehat{\theta}^{(i-1)}\right) = min\left\{1, \frac{p(\widehat{\theta}^{\star})q\left(\widehat{\theta}^{(i-1)}|\widehat{\theta}^{\star}\right)}{p(\widehat{\theta}^{(i-1)})q\left(\widehat{\theta}^{\star}|\widehat{\theta}^{(i-1)}\right)}\right\}$
        if
            $u \leq \alpha\left(\widehat{\theta}^{\star}, \widehat{\theta}^{(i-1)}\right)$ and $\rho\left(z, y\right) \leq \epsilon$, set $\widehat{\theta}^{(i)} = \widehat{\theta}^{\star}$
        else
            set $\widehat{\theta}^{(i)} = \widehat{\theta}^{(i-1)}$
        end if
    end for
```

The algorithm is not literally speaking "likelihood-free" as the likelihood is used in order to generate z. However, the likelihood is not used in order to calculate the acceptance probability $\alpha\left(\widehat{\theta}^{\star}, \widehat{\theta}^{(i-1)}\right)$.

ABC can be extended in several interesting ways, for example combined with sequential Monte Carlo, or they can incorporate model selection in a trivial way.[13] As with variational Bayes, ABC has experienced immense growth in mainstream statistics over the past two decades, and our prediction is that it will also soon be embraced by economists in order to solve complex problems.[14]

[13] The posterior probability of a given model can be approximated by the proportion of accepted simulations given the model.

[14] See for example Frazier, Maneesoonthorn, Martin and McCabe (2019) for an application of ABC algorithms in producing financial forecasts in computationally efficient ways.

4. Non-algorithmic ways of speeding up Bayesian inference

The purpose of this Section is to build further intuition by demonstrating various ways to approximate a high-dimensional inference problem simply by re-writing the likelihood and facilitating computation.[15] There are specific problems where just by simply re-writing the likelihood in an equivalent form we can gain a lot in computation – especially when Bayesian sampling methods are used to approximate the posterior (such as traditional MCMC methods). Of course there are numerous examples of such approaches in the literature, and we only selectively quote some tools we have favored ourselves while trying to develop new estimation algorithms. We provide a few examples from some popular classes of models in economics, namely regressions with many predictors and large vector autoregressions.

4.1. Random projection methods

Random projection methods have been used in fields such as machine learning and image recognition as a way of projecting the information in data sets with a huge number of variables into a much lower dimensional set of variables. To fix the basic ideas of random projections, let X be a $T \times k$ data matrix involving T observations on k variables where $k \gg T$. X_t is a $1 \times k$ vector denoting the t^{th} row of X. Define the projection matrix, Φ, which is $m \times k$ with $m \ll k$ and $\widetilde{X}_t' = \Phi X_t'$. Then $\widetilde{X}_t$ is the $1 \times m$ vector denoting the t^{th} row of the compressed data matrix, $\widetilde{X}$. Since $\widetilde{X}$ has m columns and X has k, the former is much smaller and is much easier to work with. To see how this works in a regression context, let y_t be a scalar dependent variable and consider the relationship:

$$y_t = X_t \beta + \varepsilon_t. \tag{41}$$

If $k \gg T$, then working directly with (41) is impossible with some statistical methods (e.g. maximum likelihood estimation) and computationally demanding with others (e.g. Bayesian approaches which require the use of MCMC methods). Some of the computational burden can arise simply due to the need to store in memory huge data matrices. For instance, calculation of the Bayesian posterior mean under a natural conjugate prior requires, among other manipulations, inversion of a $k \times k$ matrix involving

[15] Another factor that affects computation is the choice of programming language and the way one interacts with it. However, discussing such details is beyond the scope of our review.

the data. This can be difficult if k is huge. In order to deal with a large number of predictors, one can specify a compressed regression variant of (41)

$$y_t = \widetilde{X}_t \beta^c + \varepsilon_t. \tag{42}$$

Once the explanatory variables have been compressed (i.e. conditional on Φ), standard Bayesian regression methods can be used for the regression of y_t on $\widetilde{X}_t$. If a natural conjugate prior is used, then analytical formulae exist for the posterior, marginal likelihood, and predictive density, and computation is trivial.

Note that the model in (42) has the same structure as a reduced-rank regression, as the k explanatory variables in the original regression model are squeezed into a small number of explanatory variables given by the vector $\widetilde{X}_t' = \Phi X_t'$. The crucial assumption is that Φ is not estimated from the data, rather it is treated as a random matrix with its elements sampled using random number generation schemes.[16] The underlying motivation for random compression arises from the Johnson-Lindenstrauss lemma. This states that any k point subset of the Euclidean space can be embedded in $m = O\left(\log(k)/\epsilon^2\right)$ dimensions without distorting the distances between any pair of points by more than a factor of $1 \pm \epsilon$, where $0 < \epsilon < 1$. There are various ways to draw Φ; most obviously we can generate this matrix from N(0,1) or a Uniform(0,1) distributions. Alternatively we can draw Φ_{ij}, the ij^{th} element of Φ, (where $i = 1, .., m$ and $j = 1, .., k$) from the following scheme that generates a sparse random projection

$$\begin{array}{l} \Pr\left(\Phi_{ij} = \frac{1}{\sqrt{\varphi}}\right) = \varphi^2 \\ \Pr\left(\Phi_{ij} = 0\right) = 2\left(1-\varphi\right)\varphi \\ \Pr\left(\Phi_{ij} = -\frac{1}{\sqrt{\varphi}}\right) = \left(1-\varphi\right)^2 \end{array}, \tag{43}$$

where φ and m are unknown parameters.[17] While the remarkable properties of random compression hold even for a single, data oblivious, random draw of Φ, in practical situations (e.g. forecasting) we would like to ensure that we work with random projections that are optimal in a data-rigorous sense. As long as each compressed model projected with the matrix Φ can be estimated very quickly (e.g. using natural conjugate

[16] For that reason, random projection methods are referred to as *data oblivious*, since Φ is drawn without reference to the data.

[17] The Johnson-Lindenstrauss lemma suggests that Φ should be a random matrix whose columns have unit lengths and, hence, Gram-Schmidt orthonormalization is done on the rows of the matrix Φ.

priors), then one should be able to generate many random projections and estimate simultaneously many small models. Then goodness-of-fit measures can be used to assess which compressed models (corresponding to different random projections) fits the data better.

In summary, huge dimensional data matrices (that are too large to insert to standard econometric models) can be compressed quickly into a much lower dimension by generating random projections, without the cost of solving some computationally expensive optimization problem. The resulting compressed data matrix can then be used in a statistical model such as a regression or a vector autoregression, that can be estimated easily with traditional estimation tools. This very general approach has excellent potential applications in numerous problems in economics. For an application in large vector autoregressions and for further references, see Koop et al. (2019).

4.2. Variable elimination in regression

Variable elimination or marginalization is a machine learning procedure used in graphical models that, loosely speaking, allows (via certain rules) to break a high-dimensional inference problem into a series of smaller problems. We can use similar ideas in a standard regression setting in order to facilitate high-dimensional inference. Assume that we work again with a regression model setting with p predictors, but this time interest lies in the j-th predictor and its coefficient. We can rewrite the regression as

$$y = x_j \beta_j + x_{(-j)} \beta_{(-j)} + \varepsilon, \tag{44}$$

where y, x_j and ε are all $T \times 1$ vectors and $x_{(-j)}$ is a $T \times (p-1)$ predictor matrix with predictor j removed. It might be the case that we are interested only in parameter β_j because this is a policy parameter. A first useful result is the one of partitioned regression: defining the $T \times T$ annihilator matrix $M_j = I_T - x_j \left(x_j' x_j \right)^{-1} x_j'$, it is easy to show using the algebra of partitioned matrices that $\widehat{\beta}_j$, the OLS estimates of β_j can be obtained as the solution of

$$\widehat{\beta}_j = \left(x_j' x_j \right)^{-1} x_j' \left(y - x_{(-j)} \widehat{\boldsymbol{\beta}}_{(-j)} \right) \tag{45}$$

where the sub-vector $\widehat{\beta}_{(-j)}$ is the solution of the following regression

$$\widehat{\beta}_{(-j)} = \left(x^{\dagger\prime}_{(-j)} x^{\dagger}_{(-j)} \right)^{-1} x^{\dagger\prime}_{(-j)} y^{\dagger} \tag{46}$$

with $x^{\dagger}_{(-j)} = M_j x_{(-j)}$ and $y^{\dagger} = M_j y$ denoting the projections of $x_{(-j)}$ and y on a space that is orthogonal to x_j.

This result provides very useful intuition about the relationships between our variables and coefficients in the OLS regression. Most importantly they can be generalized to efficient procedures for high-dimensional inference. Consider for example combining partitioned regression results with a penalized estimator instead of OLS. To demonstrate this point, we consider an alternative partition of the regression. Define the $T \times 1$ vector $q_j = x_j / \|x_j\|$, and generate randomly a matrix Q_j that is normalized as $Q_j Q_j' = I - q_j q_j'$. This means that the matrix $Q = [q_j, Q_j]$ is orthogonal, such that multiplying both sides of (44) by Q' gives

$$\begin{aligned} Q'y &= Q'x_j \beta_j + Q'x_{(-j)} \beta_{(-j)} + Q'\varepsilon \Rightarrow & (47) \\ \begin{bmatrix} q_j' y \\ Q_j' y \end{bmatrix} &= \begin{bmatrix} q_j' x_j \\ Q_j' x_j \end{bmatrix} \beta_j + \begin{bmatrix} q_j' x_{(-j)} \\ Q_j' x_{(-j)} \end{bmatrix} \beta_{(-j)} + Q'\varepsilon \Rightarrow & (48) \\ \begin{bmatrix} y^* \\ y^+ \end{bmatrix} &= \begin{bmatrix} \|x_j\| \\ 0 \end{bmatrix} \beta_j + \begin{bmatrix} x^*_{(-j)} \\ x^+_{(-j)} \end{bmatrix} \beta_{(-j)} + \widetilde{\varepsilon}, & (49) \end{aligned}$$

where $y^* = q_j' y$, $y^+ = Q_j' y$, $x^*_{(-j)} = q_j' x_{(-j)}$, $x^+_{(-j)} = Q_j' x_{(-j)}$ and $\widetilde{\varepsilon} = Q'\varepsilon$. In this derivation we have used the fact that $Q_j' x_j = Q_j' q_j \|x_j\| = 0$ because Q_j and q_j are orthogonal. Additionally, $var(\widetilde{\varepsilon}) = \sigma^2 Q'Q = \sigma^2 = var(\varepsilon)$ because by construction $Q'Q = I$. The likelihood of the transformed regression model in equation (49) is multivariate Normal, which means we can use standard results for conditional Normal distributions to show that we can first estimate $\beta_{(-j)}, \sigma^2$ by regressing y^+ to $x^+_{(-j)}$, and then at a second stage obtain β_j by regressing y^* on $\|x_j\|$ conditional on $\beta_{(-j)}, \sigma^2$ being known. This is a very useful result since now, conditional on obtaining in a first step some estimates of $\beta_{(-j)}, \sigma^2$, we can estimate β_j in a regression with known variance.[18] Korobilis and Pettenuzzo (2019) apply these ideas to a high-dimensional VARs under a wider class of hierarchical shrinkage priors. Considering that the exact way of calculating marginal posteriors would involve solving numerically a $p-1$-dimensional integral for

[18] Most importantly, we can do so in parallel for all predictors $j = 1, ..., p$.

each j, doing a rotation of the form shown above, and deriving the marginal posteriors analytically, means large gains in computation can be achieved.

4.3. Multivariate estimation equation-by-equation

Some of the most important quantitative exercises that policy-makers are interested in, involve the vector autoregressive (VAR) model and its variants. Economic theories can be tested reliably only in a multivariate econometric setting, and the same holds to a large degree for measuring the impact of shocks to the wider economy. While a large part of empirical analysis is done using VARs of say three or five variables, there is an expanding literature that acknowledges the benefits of large VARs. In particular, small structural VARs might not be invertible meaning that their residuals will not span the same space as the structural shocks that macroeconomists want to identify. Therefore, it comes to no surprise that there is an expanding and lively literature on methods for estimating large VARs.

A vector autoregression for an $1 \times n$ vector of variables of interest y_t can be written in the following form

$$y_t = B_0 + \sum_{i=1}^{p} y_{t-i} B_i + \varepsilon_t, \tag{50}$$

but we can write it in familiar multivariate regression form as

$$y_t = X_t B + \varepsilon_t, \tag{51}$$

where $X_t = (1, y_{t-1}, ..., y_{t-p})$, $A = [B_0, B_1, ..., B_p]$ and $\varepsilon_t \sim N(0, \Sigma)$ with Σ and $n \times n$ covariance matrix. Accumulation of parameters in VARs is quite different compared to univariate models. A VAR with $n = 3$ variables, intercept terms and $p = 1$ lag has 18 parameters. The same VAR with $n = 50$ variables has 3825 parameters. The last VAR with $p = 12$ has 31,325 parameters. This gives an idea of the polynomial rate at which the number of parameters increases as n and/or p increase. The problem with VARs proliferates if we want to use independent priors on the coefficients B_i that would allow to shrink each of their elements independently. Doing so implies that we need to write the VAR in seemingly unrelated regression (SUR) form, where in this form the right hand side matrix of predictors is $Z = I_n \otimes X$. For large VARs this $T \times n(np+1)$ matrix becomes so large that handling it eventually becomes computationally infeasible, despite the fact that it is sparse and one can rely on more efficient sparse matrix calculations.

Nevertheless, there are still simple ways to use independent priors. Koop et al. (2019) in the context of developing random projection algorithms for large VARs, proposed to break the VAR into a collection of n univariate equations. Using ideas from estimation of simultaneous equation models we can transform the VAR in triangular form. Consider the Cholesky-like decomposition of the covariance matrix, $\Sigma = A^{-1}D\left(A^{-1}\right)'$ where D is a diagonal matrix for variances, and A^{-1} is a uni-triangular matrix of the form

$$\boldsymbol{A}^{-1} = \begin{bmatrix} 1 & 0 & ... & 0 & 0 \\ \alpha_{2,1} & 1 & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & 0 & 0 \\ \alpha_{n-1,1} & \cdots & \alpha_{n-1,n-2} & 1 & 0 \\ \alpha_{n,1} & \cdots & \alpha_{n,n-2} & \alpha_{n,n-1} & 1 \end{bmatrix}. \tag{52}$$

Under this decomposition we can rewrite the VAR in equation (51) as

$$\begin{aligned} y_t &= X_t B + u_t \left(A^{-1}D^{\frac{1}{2}}\right)' \Rightarrow & (53) \\ y_t A &= X_t B A + u_t, D^{\frac{1}{2}} \Rightarrow & (54) \\ y_t + y_t \widetilde{A} &= X_t \Gamma + u_t, D^{\frac{1}{2}} \Rightarrow & (55) \\ y_t &= X_t \Gamma - y_t \widetilde{A} + u_t, D^{\frac{1}{2}}, & (56) \end{aligned}$$

where $u_t \sim N(0, I)$, $\Gamma = B \times A$ and $\widetilde{A} = A - I$ is a lower diagonal matrix created from A after we remove its unit diagonal elements. This is a so-called triangular VAR system due to the fact that $\widetilde{A}$ has a lower triangular structure. It cannot be estimated as a multivariate regression using standard linear estimators because y_t shows up both on the left-hand side and the right-hand side of the equation. However, due to the lower triangular structure of $\widetilde{A}$ and the fact that D is diagonal the system can be estimated equation-by-equation using simple OLS. This means that in high dimensions we can essentially write the VAR in this form and apply any univariate regression estimator and algorithm we like.[19] More importantly, note that the last equation shows that all contemporaneous covariances among the n VAR equations can be written as RHS predictors $-y_t$. This is an important implication because it shows that $\widetilde{A}$ can be treated as a regression parameter and (given that we can estimate these equations recursively)

[19] Of course, note that this flexibility comes at the cost of shrinkage or variable selection being dependent on the ordering of the variables in the VAR; see Koop et al. (2019) for a discussion.

we can readily apply methods of the previous section to impose shrinkage also on the VAR covariance matrix.

Finally, we can derive a similar triangular VAR that has slightly different representation and implications for estimation. Begin with equation (51) but now rewrite it in the form

$$\begin{aligned}
y_t &= X_t B + u_t \left(A^{-1} D^{\frac{1}{2}} \right)' \Rightarrow & (57)\\
y_t &= X_t B + u_t \left(\left(\widetilde{A}^{-1} + I \right) D^{\frac{1}{2}} \right)' \Rightarrow & (58)\\
y_t &= X_t B + u_t \widetilde{A}^{-1} D^{\frac{1}{2}} + u_t D^{\frac{1}{2}} \Rightarrow & (59)\\
y_t &= X_t B + v_t \widetilde{A}^{-1} + v_t, & (60)
\end{aligned}$$

where $v_t \sim N(0, D)$ and $\widetilde{A}^{-1} = A^{-1} - I$ is a triangular matrix created by removing the identity diagonal of A^{-1}. This system can also be estimated equation by equation, where in equation i we use residuals from the previous $i-1$ equations. This form has different implications for designing estimation algorithms compared to the one in (56), even though they are observationally equivalent. Equation (60) allows direct estimation of the VAR matrices B and A^{-1}, while equation (56) estimates functions of those, i.e. Γ and A. Such examples show that high-dimensional inference can be approximated by efficient transformations of the VAR model that allow to readily apply univariate estimators which are simpler and possibly algorithmically faster.

5. Conclusions

We have attempted to provide a wide review of algorithms and methods for speeding Bayesian inference to cope with high-dimensional data and models. Our review is very high-level and should be seen as a first-step introduction to the various tools that a modern econometricians need to have in their toolbox. As always, there are several pros and cons with the various algorithms, and the choice of the "*right*" algorithm is application specific. There are some excellent and in-depth recent reviews of some of these algorithms that demonstrate their use in various interesting contexts. For example, Angelino et al. (2016) and Green et al. (2015) provide some excellent detailed reviews of various algorithms. Blei et al. (2010) provide an accessible introduction to variational Bayes methods. Sisson et al. (2018) provide a recent review and references of Approximate Bayesian Computation (ABC) methods. The review paper by Zhu et al. (2017) focuses on scalability and distributed computation of Monte Carlo methods, as well as regularized Bayesian inference. Bayesian machine learning is a very lively literature, as is the case with non-Bayesian machine learning approaches that are also expanding rapidly. We have tried to provide a gentle introduction to this literature and bridge the gap between the expanding computing needs of economists and computational advances proposed in various other literatures such as comprehensive sensing, computer vision and AI.

References

[1] Angelino, E., Johnson, M. J. and R. P. Adams (2016). Patterns of scalable Bayesian inference. *Foundations and Trends® in Machine Learning*, 9(2-3), 119-247.

[2] Bardenet, R., Doucet, A. and Holmes, C. (2017). On Markov chain Monte Carlo methods for tall data. *Journal of Machine Learning Research*, 18, 1-43.

[3] Berger, J. O. (1985). *Statistical decision theory and Bayesian analysis.* Second Edition, Springer-Verlag: New York.

[4] Blei, D. M., Kucukelbir, A. and McAuliffe, J. D. (2017). Variational Inference: A Review for Statisticians *Journal of the American Statistical Association*, 112(518), 859-877,

[5] Chib, S. and Greenberg, E. (1995). Understanding the Metropolis-Hastings Algorithm. *The American Statistician*, 49(4), 327-335.

[6] Craiu, R., Rosenthal, J. and Yang, C. (2009). Learn from thy neighbor: Parallel-chain and regional adaptive MCMC. *Journal of the American Statistical Association*, 104(488), 1454-1466.

[7] Frazier, D.T., Maneesoonthorn, W., Martin, G.M. and McCabe, B.P.M. (2019). Approximate Bayesian forecasting. *International Journal of Forecasting*, 35, 521-539.

[8] Gelman, A. and Rubin, D. (1992). Inference from iterative simulation using multiple sequences. *Statistical Science*, 7(4), 457-472.

[9] Geweke, J. (1989). Bayesian Inference in Econometric Models Using Monte Carlo Integration. *Econometrica*, 57(6), 1317-1339.

[10] Girolami, M. and Calderhead, B. (2011). Riemann manifold Langevin and Hamiltonian Monte Carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73, 123-214.

[11] Goodman, J. and Weare, J. (2010). Ensemble samplers with affine invariance. *Communications in Applied Mathematics and Computational Science*, 5(1), 65-80.

[12] Green, P. J., Łatuszyński, K., Pereyra, M. and Robert, C. P. (2015). Bayesian computation: a summary of the current state, and samples backwards and forwards. *Statistics and Computing* 25(4), 835-862.

[13] Kim, S., Shephard, N. and Chib, S. (1998). Stochastic volatility: Likelihood inference and comparison with ARCH models. *The Review of Economic Studies* 65(3), 361-393.

[14] Koop, G., Korobilis, D. and Pettenuzzo, D. (2019). Bayesian compressed vector autoregressions. *Journal of Econometrics* 210, 135-154.

[15] Korobilis, D. (2020). High-dimensional macroeconomic forecasting using message passing algorithms. *Journal of Business and Economic Statistics*, forthcoming.

[16] Korobilis, D. and Pettenuzzo, D. (2019). Adaptive hierarchical priors for high-dimensional vector autoregressions. *Journal of Econometrics*, 212, 241-271.

[17] Laplace P. - S. (1774). Memoire sur la Probabilite des Causes par les Evenements. *l'Academie Royale des Sciences*, 6, 621-656. English translation by S.M. Stigler in 1986 as "Memoir on the Probability of the Causes of Events" in *Statistical Science*, 1(3), 359-378.

[18] Lindley, D. V. (1980). Approximate Bayesian methods. *Trabajos de Estadistica Y de Investigacion Operativa*, 31, 223-245.

[19] Liu, J. S., Liang, F. and Wong, W. H. (2000). The multiple-try method and local optimization in Metropolis sampling. *Journal of the American Statistical Association*, 95(449), 121-134.

[20] Malewicz, G., Austern, Matthew H. Bik, A. J. C., Dehnert, J. C., Horn, I., Leiser, N., and Czajkowski, G. (2010). Pregel: A system for large-scale graph processing. In SIGMOD'10, 135-145.

[21] Naylor, J. and Smith, A. (1982). Applications of a Method for the Efficient Computation of Posterior Distributions. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 31(3), 214-225.

[22] Rasmussen, C. E. and Ghahramani, Z. (2002). Bayesian Monte Carlo. In Proceedings of the 15th International Conference on Neural Information Processing Systems (NIPS'02). MIT Press, Cambridge, MA, USA, 505-512.

[23] Ritter, C. and Tanner, M. (1992). Facilitating the Gibbs Sampler: the Gibbs Stopper and the Griddy-Gibbs sampler. *Journal of the American Statistical Association*, 87, 861-868.

[24] Rockova, V. and George, E. (2014). EMVS: The EM approach to Bayesian variable selection. *Journal of the American Statistical Association* 109(506), 828-846.

[25] Scott, S. L., Blocker, A. W., Bonassi, F. W., Chipman, H. A., George, E. I. and McCulloch, R. E. (2016). Bayes and Big Data: The consensus Monte Carlo algorithm. *International Journal of Management Science and Engineering Management* 11, 78-88.

[26] Sisson, S. A., Fan, Y. and Beaumont, M. A. (2018). Overview of approximate Bayesian computation. arXiv: 1802.09720v1.

[27] Solonen, A., Ollinaho, P., Laine, M., Haario, H., Tamminen, J. and Jarvinen, H. (2012). Efficient MCMC for climate model parameter estimation: Parallel adaptive chains and early rejection. *Bayesian Analysis*, 7(2), 1-22.

[28] Tipping, M. (2001). Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research* 1, 211-244.

[29] Wang, X. and Dunson, D. B. (2013). Parallel MCMC via Weierstrass sampler. ArXiv preprint, arXiv:1312.4605, 2013.

[30] Zhu, J, Chen, J. Hu, W., and Zhang, B. (2017). Big learning with Bayesian methods. arXiv:1411.6370v2.